Embedded Wireless Sensor Network for Aircraft/Automobile Tire Structural Health Monitoring

Farrukh M. Gondal

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science
In
Electrical & Computer Engineering

Dr. Amitabh Mishra, Chairman
Dr. Rakesh K. Kapania, Co-Chairman
Dr. Ronald D. Moffitt, Committee Member

April 27, 2007
Blacksburg, Virginia

Keywords: Structural Health Monitoring (SHM), Tire Strain, Wireless Sensor Network (WSN), Resistive Strain Gauge, TinyOS, NesC

Embedded Wireless Sensor Network for Aircraft/Automobile Tire Structural Health Monitoring

Farrukh M. Gondal

(ABSTRACT)

Structural Health Monitoring (SHM) of automobile tires has been an active area of research in the last few years. Within this area, the monitoring of strain on tires using wireless devices and networks is gaining prominence because these techniques do not require any wired connections. Various tire manufacturers are looking into SHM of automobile tires due to the Transportation Recall Enhancement, Accountability and Documentation (TREAD) act which demands installation of tire pressure monitoring devices within the tire. Besides measuring tire pressures, tire manufactures are also examining ways to measure strain and temperature as well to enhance overall safety of an automobile. In other words, tire manufacturers are looking to develop a sensor system that can monitor the health of an automobile tire.

A sensor system that can measure the overall strain of a tire is known as a centralized strain sensing system. However, a centralized strain sensing system cannot find the location and severity of the damage on the tire, which is a basic requirement. Various sensors such as acceleration and optical sensors have also been proposed to be used together to get more local damage information on the tire but a tire equipped with multiple sensors is neither practical nor cost effective. In this thesis we have developed a strain sensing system that performs accurate, local strain measurements on the tire and transmits them to a console inside the vehicle wirelessly. Our sensing system utilizes a new sensing material called Metal Rubber^{TM} which is shown to be conductive like metal, and flexible as rubber. In addition, we have also developed a reliable and an energy efficient geographic routing protocol for transporting strain data wirelessly from a tire surface to the driver of the automobile.

Acknowledgments

I would like to take the opportunity to thank people who guided and supported me during the research. This research would not have been possible without their contribution. I would like to thank my professor, Dr. Amitabh Mishra, for being a great advisor. Dr. Mishra has given me support with timely encouragement for a continuous and constructive thought process. I thank him for his categorical comments on the thesis; it has helped me present my research in a much better way. I would like to thank Dr. Kapania and Dr. Moffitt for guiding me through out my research. I am grateful to my committee co-chairs for their time in serving on my committee, and suggesting technical modifications and corrections.

I would like to thank two graduate students, Scott Bland and Mohammed Rabius Sunny, and two undergraduate students, Arya Afrashteh and Ryan Wilson, for their contribution early in this project. I would like to show my appreciation to my parents and family for their continuous support during all these years. This work would not have been possible without their support. Therefore, I would like to thank them for their love, support, and encouragement.

This work was supported by funding from NASA under contract NNLO5AA29G. Their support is gratefully acknowledged. Also, I would like to thank MIL3® for providing me with an academic license for the OPNET® Modeler software. I would not be able to complete this research without their support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Introduction and Motivation

Recently, various tire manufacturers are looking into intelligent tires due to the Transportation Recall Enhancement, Accountability and Documentation (TREAD) act which demands installation of tire pressure monitoring devices within the tire. Tires that can measure strain within themselves are known as intelligent tires. A vehicle equipped with intelligent tires is considered far more reliable than the one which has regular tires. Besides measuring tire pressures, tire manufactures are also examining ways to measure strain and temperature as well to enhance overall safety of a vehicle. In other words, tire manufacturers are looking to create a sensing system that can monitor the health of an automobile tire. Tire manufacturers have not conducted any research in this area since this is the first time they are required to monitor the health of a tire due to the TREAD act. Therefore, we have focused our research into developing a strain sensing system to monitor the health inside the tire.

This chapter will discuss the challenges faced during the research work such as developing a reliable and energy efficient strain sensing system of an automobile tire. Moreover, this chapter will list the approaches taken in a short passage to overcome the challenges. It will also summarize the key contributions made by this research work. Also, this chapter will present an outline of the whole thesis in the end.

1.1 Approaches to Solve the Problem

Two important tasks must be undertaken for a successful SHM of an automobile tire. First, we must measure strain off an automobile tire accurately. Second, we must transmit strain information reliably to the driver. In order to accomplish first task, we have found approaches to get local strain measurements on the tire using a new sensing material. The new sensing material is conductive like metal, and is flexible as rubber. In order to send strain information to the user, a wireless system is much more practical than a wired system since tire is continuously moving. But wireless transmission does not guarantee successful delivery of data because radio signals experience fading due to the environment. Needless to mention, we are dealing with a harsh wireless environment due to the multi-path fading in tire structure made up of metal and rubber.

In other words, the range of wireless transmission becomes limited in an automobile tire. We

can solve this problem by planting intermediate wireless devices as relays between the tire and the user which will extend the range of the wireless transmission. This system of placing intermediate nodes between the source and destination to measure ambient conditions in a harsh environment is called Wireless Sensor Networks (WSNs). WSNs have a structure of an ad hoc network which means sensor nodes are usually scattered in a sensor field. Sensor nodes need to coordinate among themselves to create a working network to send useful information either to other sensors or to a fixed or mobile base station (BS) [1].

We have developed a WSN to monitor strain on a tire using commercial off-the-shelf hardware and software components such as Crossbow motes [2], and TinyOS [3]. In most of WSNs applications, sensor nodes are constrained in energy supply and communication bandwidth. For this reason, extensive research has been conducted in every layer of networking protocol stack to create energy efficient WSNs. For example, at the networking layer, it is extremely important to find the most efficient route for data transfer that can maximize the lifetime of sensor nodes. Therefore, we are focusing on developing reliable and energy efficient routing for this particular application that may find use in other areas in WSNs.

There have been various routing protocol suggested for WSNs which are intended to be energy efficient, scalable, and reliable [4]. One of the techniques used in these routing protocols is to use geographic location of a sensor to make the routing protocol scalable. Geographic routing protocols perform poorly for mobile and sparse networks as stated earlier. The sensors embedded in the tire surface are considered mobile nodes due to the rotation of the tire. In this thesis, we have developed few techniques that build on existing geographic routing protocols to deal with mobile networks. One of the techniques is to predict location of mobile nodes based on the past movement named Location Prediction of Mobile Nodes (LPMN). This information will be made available to the node pick the next hop node. The node will be able to pick a next hop node which is more cost effective in terms of its geographic location. We have tested our protocol in terms of energy usage and packet drop rate.

1.2 Research Challenges

During the course of the research, we have experienced many road-blocks such as finding a suitable sensor and increasing the lifetime of a WSN. These road blocks steered the research into different directions to look for alternative approaches. In the next few sections, explain the

research challenges faced and the approaches taken to overcome them.

1.2.1 Energy Consumption of a Sensor Node

Sensor nodes have limited energy supplied by the batteries and methods to recharge the batteries in the field are expensive and complex. It is highly inefficient to replace dead batteries from a transceiver placed on the tire rim. Therefore, it is necessary that sensor nodes use energy in an efficient manner. The main tasks that consume energy are data sensing, communication, and the processes in the controller. Energy can be saved by designing energy aware protocols at application, link, and network layers. Power management in WSN is a very important issue because radio communication dominates the power consumption in a sensor node. There are various protocols that save power by limiting radio communication as much as possible. For example, sensor nodes can be put to sleep for extended periods of time when no event of interest is detected. But it's desirable that the sensor nodes should be able to wake up to monitor critical events when they occur. Scheduling of sleep-wakeup patterns for sensor nodes is an important area of current research in WSN.

1.2.2 Reliable Sensor Network

In scenarios where the sensors are operating in remote or dangerous territory, it may be impossible to retrieve the nodes in order to recharge batteries. Therefore, the network should be considered to have a certain lifetime during which nodes have energy and can gather, process, and transmit information. This means that all aspects of the node, from the sensor module to the hardware and protocols must be designed to be extremely energy-efficient. In addition to reducing energy dissipation, protocols should be robust to mobility in order to achieve high throughput for mobile and sparse networks. The sensors embedded in the tire surface are considered mobile nodes due to the rotation of the tire. Most of the existing routing protocols in WSN perform poorly for mobile nodes in terms of packet delivery ratio. For this reason, we developed LPMN to build on existing protocols to create a reliable WSN.

1.2.3 Suitable Strain Sensor

Intelligent tires that can measure strain of a tire are likely to be highly in demanded due to the TREAD act. The commercial strain gauges are not found to be suitable for this application due to the possibility of de-bonding of the sensor from tire rubber. For this application, we needed a strain sensor that can measure strain on a flexible surface such as rubber and can be as conductive as metal. We have found a strain sensor that has some of the properties mentioned

above. A local company, NanoSonic Inc., has developed this strain sensor called Metal Rubber™ [5]. However, we have faced many difficulties such as hysteresis in using Metal Rubber™ as a strain sensor in our application.

1.2.4 Hysteresis in Metal Rubber™

Metal Rubber™ exhibits hysteresis in strain measurements due to its material properties which makes it difficult to be used as a strain sensor. An approach is needed that can model the hysteresis exhibited by the metal rubber while subjected to various degrees of strain so that one can accurately map the measured strains to true strains. Fractional derivate approach [6] has been shown to model the hysteresis reasonably well [6]. This approach is discussed in detail in later chapters.

1.3 Contributions of this Research

This thesis develops a wireless sensor network architecture to measure strain of an automobile/aircraft tire. The main accomplishments of this work:

1. Developed an application architecture running on hardware components such as Crossbow motes under TinyOS operating system that allows sensing of strain on the rubber surface and transmits the measurements to a fixed base station using wireless communication.

2. Investigated several strain sensing options that can measure large strains on the rubberized surfaces. Identified a localized strain sensing system using a new sensor material called Metal Rubber™. Metal Rubber™ has shown to possess all desirable properties to act as a possible strain sensor except that it exhibits hysteresis.

3. Developed a geographic routing protocol suitable for cylindrical network topologies such as a tire and evaluated its performance. This protocol has shown to have a higher packet delivery ratio than other geographic protocols such as GAF.

The following publications have resulted/planned from the research presented in this thesis:

1. D. Afrashteh, S. Bland, F. M. Gondal, R. K. Kapania, A. Mishra, R. D. Moffitt, and R. R. Wilson, "**Embedded Wireless Sensors for Aircraft/Automobile Tire Structural Health Monitoring**," in *IEEE SECON*, Reston, VA, 2006.

2. F. M. Gondal, R. K. Kapania, A. Mishra, and R. D. Moffitt, "**An Energy Efficient Geographic Routing Algorithm for Mobile & Sparse Wireless Sensor Networks**," submitted to *MILCOM 2007* Orlando, Florida, 2007.
3. F. M. Gondal, R. K. Kapania, A. Mishra, R. D. Moffitt, and M. R. Sunny, "**A Wireless Sensor Network Architecture for Structural Health Monitoring of Aircraft/Automobile Tires**," submitted to *MILCOM 2007* Orlando, Florida, 2007.
4. F. M. Gondal, R. K. Kapania, A. Mishra, R. D. Moffitt, and M. R. Sunny, "**A Wireless Sensor Network Architecture for Automobile Tires**," *IEEE Trans. On Mobile Computing,* 2007.

1.4 Thesis Organization

This thesis explains the work done on the SHM of an aircraft tire. The thesis has two main components which are: (1) measuring strain on an aircraft tire accurately and (2) sending the strain information wirelessly to the pilot in the aircraft. In Chapter 2, we explain the related background and available technologies on strain sensors. In Chapter 3 and Chapter 4, we discuss the architecture adopted to measure the strain using Crossbow® wireless motes. In Chapter 5, the validity of the strain results acquired by Crossbow® motes is confirmed by comparing these results to theoretical strain on a metal bar. Also in Chapter 5, we test suitability of an elastic sensor called Metal Rubber™ for accurately measuring strain of an aircraft tire. This chapter discusses few shortcomings with Metal Rubber™ and proposes some solutions to overcome these.

In Chapter 6, we discuss the related background and present a literature survey of available reliable and energy efficient routing schemes for Wireless Sensor Networks (WSN). In Chapter 7, we discuss our proposed geographic routing protocol which is evaluated in Chapter 8. We discuss our results and their analyses in Chapter 9. The thesis ends with conclusions and a discussion of future directions in Chapter 10.

Chapter 2
Background of Strain Sensor Technology

Strain sensing is a well studied area with applications such as monitoring of strain on buildings, bridges, and machines. In this chapter, we will review some of the most common strain measuring techniques. These techniques will cover most of the strain sensing technology from commercially available to the solutions proposed in research articles. We will cover some of the basic definitions and techniques of strain sensing to set the stage for discussing methods to measure strain. In section 2.2, we will discuss the details involved in setting up a strain sensing system using a resistance strain gauge. This section will be followed by the capacitance method to measure strain in section 2.3. In the last three sections of the chapter, we will briefly discuss other methods such as an optical, a sonic, and a polyvinylidene difluoride (PVDF) film sensors.

2.1 Strain and displacement

Figure 2.1 shows a bar that can be elongated to length L_1 from an initial length L_o when it is stretched with the horizontal force F from both sides. The relative change in the bar's length is known as strain (ε). Strain (ε) experienced by this bar can be calculated with the following equation:

$$\varepsilon = \frac{L_1 - L_o}{L_o} \tag{2.1}$$

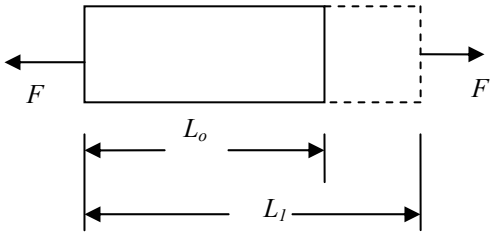

Figure 2.1: Deformation in a bar

The above scenario only describes strain along one dimension. For a multidimensional case,

we need to find strain for each dimension. For example, a two dimensional object will have separate values for horizontal and vertical strain given by ε_x and ε_y respectively as well as sheer strain [7]. Any device that is used to measure surface deformation is called a strain gauge. These strain gauges are divided into four groups namely mechanical, optical, electrical, and acoustical. Out of these four groups, the electrical resistance strain gauges are most commonly used to measure strain.

2.2 Resistance Strain Gauge

The resistance strain gauge shown in Figure 2.2 is a self-temperature compensated strain measurement tool [8]. It is a low cost sensor which is widely used in measuring strain of various structures. It is composed of a fine resistance wire grid which can be bonded to a surface to measure its strain. The strain is measured by acquiring the change in resistance caused by the fine wire grid when it is deformed. The grid pattern shown in the figure is an electric conductive material. The strain sensitivity is a function of deformation which takes place when the grid is deformed causing electrical changes in the basic resistivity of the material. [9]. The resistance change in the strain gauge due to the deformation is very minute and can not be measured directly with an ordinary ohmmeter. A resistive strain gauge is usually a part of a larger electric circuit in most practical strain measurement system. This electrical circuit is usually a bridge where strain gauges are attached to one or more of its arms.

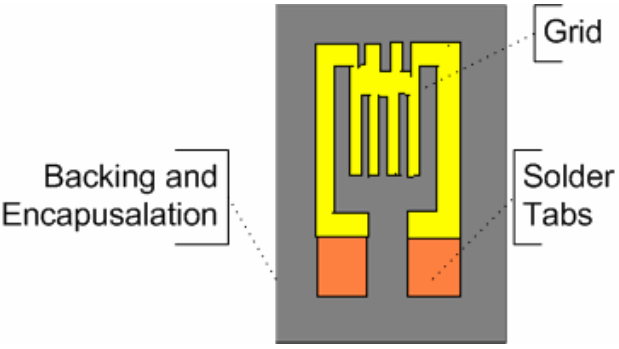

Figure 2.2: A Typical Foil Strain Gauge

The bridge circuit is known as a Wheatstone bridge. Figure 2.3 shows a Wheatstone bridge circuit diagram. The main function of a Wheatstone bridge is to get an amplified voltage response proportional to the resistance change in one of the strain gauges [7]. Wheatstone bridge is usually made up of two voltage dividers connected to form a bridge. In Figure 2.3, resistors R_1 and R_2 forms one voltage divider, and R_3 and R_4 forms the other voltage divider. Each of the

voltage dividers has an overall voltage of E divided between two parts of the divider. The Wheatstone bridge operating principals are discussed in the following sections.

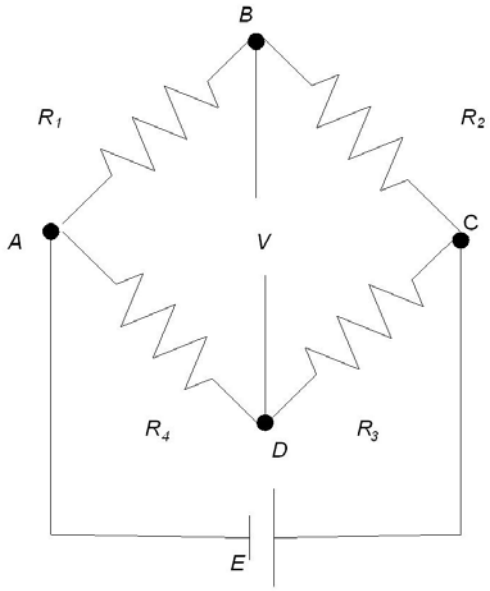

Figure 2.3: Constant-Voltage Wheatstone bridge

Using the voltage divider equations, the voltage drop across R_1 and R_4, denoted by V_{AB} and V_{AD}, respectively are given by the following equations:

$$V_{AB} = \frac{R_1}{R_1 + R_2} E \quad V_{AD} = \frac{R_4}{R_3 + R_4} E \tag{2.2}$$

where, E is the applied voltage across the bridge. The voltage output of the bridge V, which is just the voltage change between nodes B and D in Figure 2.3, is given by:

$$V = V_{AB} - V_{AD} = \frac{R_1 R_3 - R_2 R_4}{(R_1 + R_2)(R_3 + R_4)} E \tag{2.3}$$

It is clear that the output voltage of the bridge is zero if we satisfy the following equation

$$R_1 R_3 = R_2 R_4 \tag{2.4}$$

Wheatstone bridge is said to be balanced when the output voltage of the bridge is zero. Equation 2.4 illustrates a very important concept. Any resistance changes in one of the four arms of the bridge will give a nonzero voltage output of the bridge. Therefore, we can replace any of the arms of the bridge with a strain gauge and balance the bridge by satisfying equation 2.4.

Then, we will be able to measure strain using the strain gauge as the bridge output voltage will deviate from zero with respect to the strain applied on the strain gauge. This change in voltage output of the bridge V is caused by a change in the resistance of any of the four arms of the bridge which is given by:

$$\Delta V = \frac{(R_1 + \Delta R_1)(R_3 + \Delta R_3) - (R_2 + \Delta R_2)(R_4 + \Delta R_4)}{(R_1 + \Delta R_1 + R_2 + \Delta R_2)(R_3 + \Delta R_3 + R_4 + \Delta R_4)} E \tag{2.5}$$

One can choose to have one, two, or four of the arms of the bridge replaced by strain gauges. These bridge arrangements are called quarter, half, and full bridge arrangements respectively. The following sections will explain such bridge arrangements in further detail.

2.2.1 Quarter Bridge

The arrangement shown in Figure 2.4 is called a quarter bridge since one of the four bridge arms is replaced by an active strain gauge. In this arrangement, R_g is the active strain gauge which can change the voltage output of the bridge with respect to the resistance change of the strain gauge.

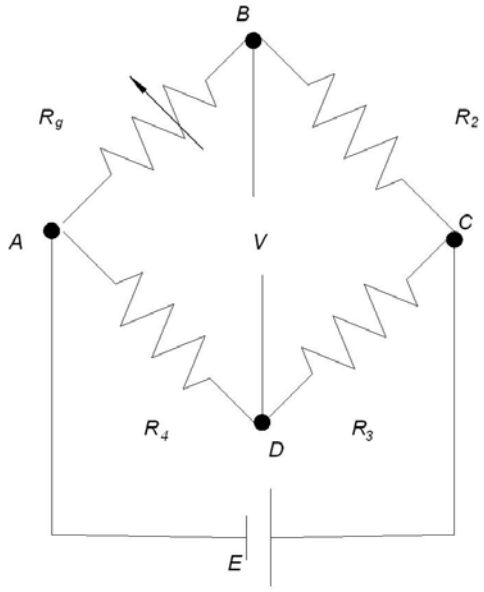

Figure 2.4: Quarter Bridge Arrangement

The bridge's output voltage is relevant to the change in resistance of the active strain gauge. We can calculate the strain measured by using the bridge's output voltage in the following equation:

$$\frac{V}{E} = \frac{G_f \varepsilon \times 10^{-3}}{4 + 2G_f \varepsilon \times 10^{-6}} mV/V \tag{2.6}$$

where, G_f is the active strain gauge's gauge factor, and E is the bridge's excitation voltage. Gauge factor is defined as the fractional change in resistance in respect to the fractional change in length along a given axis of the strain gauge. It is a dimensionless quantity that applies to the changes in the strain gauge as a whole. Typical gauge factors are close to 2.0. Gauge factor is usually known beforehand for a specific strain gauge [10]. A strain gauge's gauge factor can be calculated with the following equation:

$$G_f = \frac{\Delta R / R}{\Delta L / L} \tag{2.7}$$

2.2.2 Half Bridge

The half bridge completion as shown in Figure 2.5 has two active strain gauges which makes it more accurate than a quarter bridge completion. If we position two strain gauges in a way that both respond equally to the strain but with opposite directions, then resistance changes due to temperature variation will be cancelled. Therefore, half bridge completion compensates for temperature variation much better than quarter bridge completion hence acquiring more accurate strain readings.

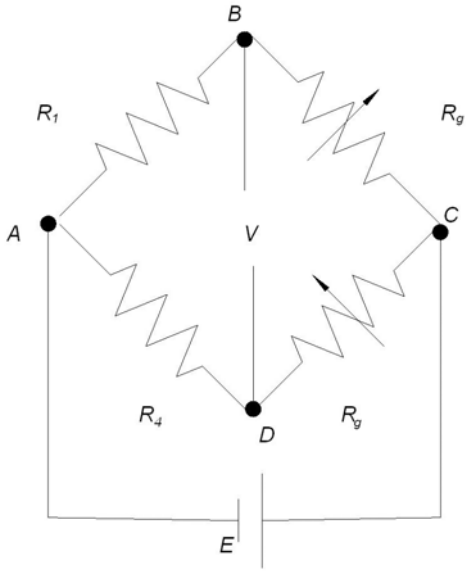

Figure 2.5: Half Bridge Arrangement

Strain can be measured using the bridge's output voltage on the following equation:

$$\frac{V}{E} = \frac{G_f \varepsilon (1+\nu) \times 10^{-3}}{4 + 2G_f \varepsilon (1+\nu) \times 10^{-6}} mV/V \tag{2.8}$$

where, ν is the Poisson ratio of the strained material. The Poisson ratio is the ratio of the contraction strain normal to the applied load divided by the extension strain in the direction of the applied load [10]. If the strain in the direction of the force is our x-component strain measurement in

Figure 2.1, then the strain measurement in the y-component will be the strain measurement in the x-component times the Poisson ratio. The strain measurements become more accurate using a full bridge arrangement.

2.2.3 Full Bridge

The full bridge completion is composed of one strain gauge per arm allowing up to four strain gauges per bridge completion. The highest strain sensitivity and temperature compensation can be achieved using four active strain gauges creating a full bridge completion as shown in Figure 2.6. Equation 2.7 is still valid for a full bridge completion with known Poisson ratio.

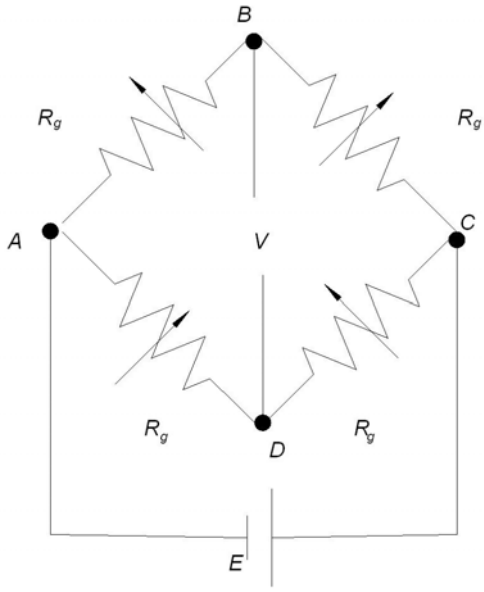

Figure 2.6: Full Bridge Arrangement

2.3 Capacitance Method

This method can be used to measure strain of a structure which has a conductive material, such as steel, embedded in a dielectric material, such as rubber. An automobile tire closely resembles

the structure defined above. An automobile tire has two steel wires to support the structure surrounded by rubber. This structure resembles an electric condenser with steel wires serving as electrodes. The spacing between the steel wires is a function of the capacitance of the condenser. Capacitance increases as the steel wire layer shrinks and vice versa. We can measure the strain by keeping track of the change in the capacitance [11].

The steel wires embedded in a dielectric material as shown in Figure 2.7 are given a charge of q and $-q$ Coulombs, respectively. The electric field through the centers of the steel wire is given by Gauss Law as shown below:

$$E_r = \frac{q}{2\pi\varepsilon r} + \frac{q}{2\pi\varepsilon(d-r)} \tag{2.9}$$

where, ε is the dielectric constant of the rubber. The difference in potential between the steel wires is given by:

$$V = -\int_{d-a}^{a} E_r dr = \frac{q}{\pi\varepsilon}\ln\left(\frac{d-a}{a}\right) \tag{2.10}$$

The capacitance of the condenser per unit length is given by:

$$C = \frac{q}{V} = \frac{\pi\varepsilon}{\ln\left(\frac{d-a}{a}\right)} \tag{2.11}$$

The capacitance decreases as the spacing between the electrodes is enlarged. Therefore, the deformation of the material will change the capacitance creating a strain sensor as shown in Figure 2.7 [12].

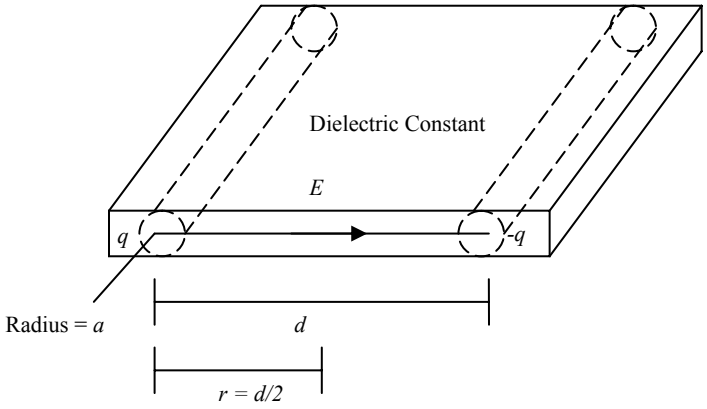

Figure 2.7: Condenser Model

2.4 Optical Method

The optical method that can be used to measure strain is called photogrammetry. Photogrammetry is accomplished by taking an image of the test piece in the unloaded state and compared to the image taken in the loaded state by means of a digital camera. The test piece has referenced dots which are used to measure strain. We can measure strain by measuring the displacement of the referenced dots after the deformation of the material. These stored images are processed using an algorithm developed in programming software to read these images as data files and determine the coordinates of the referenced dots. Once the coordinates are known for the unloaded and loaded dot positions, the displacements and strains can be determined. Figure 2.8 shows the photogrammetry method, where ΔL is displacement or strain in the direction of force being applied.

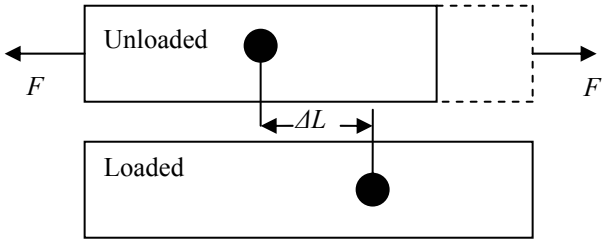

Figure 2.8: Optical Method

2.5 Sonic Sensor

Sonic sensors can detect damage on a surface by measuring the time of flight of sound pulses. Sound pulses can be formed by a sonic transducer which are reflected by the target, and detected

by the sonic transducer as shown in Figure 2.9.

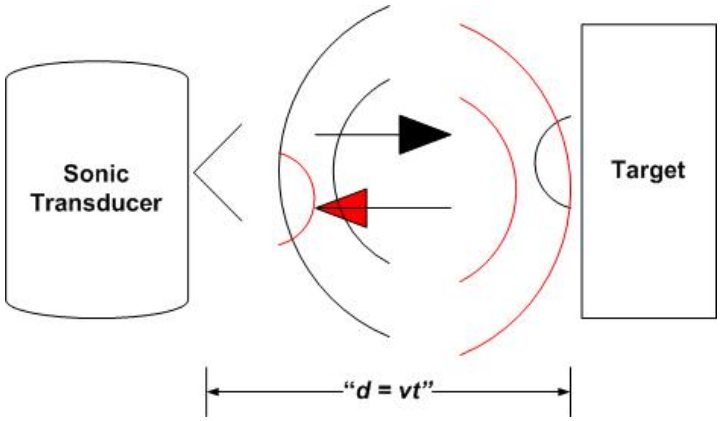

Figure 2.9: Sonic Sensor

Where, d is the distance between the target and the transducer, v is the speed of the sound, and t is time taken by the pulses to travel between the target and the transducer. Sonic sensors can measure high strain with a relative ease. However, there are few disadvantages with sonic sensors which are as follows:

- The sensitivity of the sensor depends on the temperature.
- The speed of the sound depends on the transmission medium which affects the measurements.
- The acoustic reflectivity of the rubber material is difficult to sense since rubber absorbs the sound.
- The size and shape of the target affects the measurements.

2.6 Polyvinylidene Difluoride (PVDF) Film Sensor

PVDF film is a special plastic material that can be used as a strain sensor because a voltage (V) is produced with the film is stretched as shown by the following equation:

$$V = k\frac{L}{\Delta L} \tag{2.12}$$

Where, k is the constant that depends on the type of PVDF film, L is the original length of the film, and ΔL is the change in length. Voltage measured is independent of the film area and impedance of the film is inversely proportional to the film area.

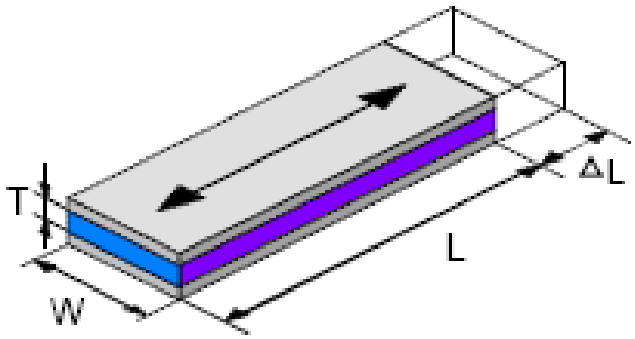

Figure 2.10: PVDF Film Sensor

PVDF film is thin and flexible so it can be used easily with tire rubber. It also has high mechanical strength and impact resistance so it can measure large strain. However, it is highly temperature dependent which requires compensating for temperature in strain measurements.

2.7 Conclusion about Strain Sensing Technology

This chapter outlined few approaches to measure strain on an automobile tire such as resistance strain gauge, capacitance method, optical sensor, sonic sensor, and PVDF film sensor. We discussed the advantages and shortcomings of all the methods. Strain on an automobile tire can be measured by implementing any of the methods described in this chapter. However, all these techniques need to be able to send measurements wirelessly since tire rotation makes wired connections unfeasible. We will discuss the wireless network architecture for the strain sensing system of an automobile tire in the next chapter.

Chapter 3
Strain Measurements using Wireless based System

Strain measurement techniques have undergone some changes recently depending upon the underlying technology being used to measure strain. Recent advances in wireless networking and micro-fabrication of microprocessors has given rise to a new field called WSNs. WSNs can be used to predict system failures which can be used to avoid disasters such as an explosion in a reactor or the collapsing of a bridge. This field is not far away from creating an automated sensing network in every part of our life; from automobile monitoring to human heart monitoring.

WSNs have several applications in various fields. However, this thesis proposes a WSN solution for monitoring strain of an automobile tire. The following sections will discuss the hardware components, operating system, and network architecture of the wireless strain measuring system that was developed as part of this thesis. Specifically, section 3.1 and 3.2 discuss sensor node hardware such as tiny processing units that can retrieve strain information and send this information using radio transceiver to the user. In section 3.3 we present the details of software being run on top of these processing units.

3.1 Sensor Node Hardware

Sensor node hardware used in the strain measurement system can be grouped into following two categories:

3.1.1 General Purpose Computers

This category includes low power PCs, embedded PCs, and various personal digital assistants (PDA). This will be considered the display system on the dashboard of an automobile. These nodes typically use operating systems such as Windows or Linux with wireless communication protocols such as Bluetooth or IEEE 802.11. They can accommodate various sensors because of their high processing power [13].

3.1.2 Embedded Sensor Nodes

This category includes the Berkeley mote family, the UCLA Medusa family, Ember nodes, and MIT μamp. These nodes have small form factor, low power processing and communication, and simple sensor interfaces. These nodes have access to only one programming language, such as C, due to memory constraint [13].

Figure 3.1: Wireless, Multi-hop, Intelligent Radio Modules

3.2 Wireless Sensor Node

3.2.1 Single-node Architecture

A sensor is a device that can measure a physical quantity in the environment which can be understood by the controller of the sensor. When such a sensor can send this information wirelessly to other sensors or to an end user with a PC, we have a wireless sensor node. A node in a wireless sensor network (WSN) should be able to compute, store data, communicate with other nodes, and sense the needed events. These nodes needs to be equipped with right sensors and communication hardware which can allow for reliable and energy efficient sensing of the needed events.

Sensor nodes can be deployed over various geographical topographies where these nodes can form a sensor network. The hardware components of a wireless sensor node solely depend on the application of the sensor. However, the generalized components of a wireless sensor node are a controller, a memory device, sensors and actuators, and a communication device as shown in Figure 3.2 [1].

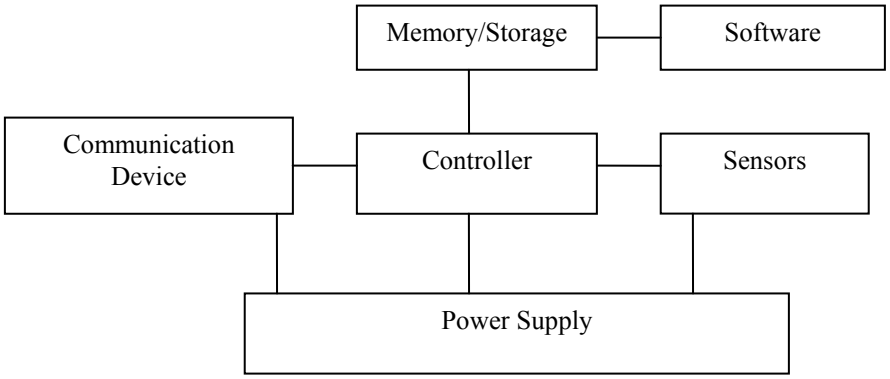

Figure 3.2: Hardware Components of a sensor node

A qualitative description of these components is as follows:

- Controller: A controller is a central part of a wireless sensor node. It processes all the data it receives from memory, sensor, or communication device. It is also capable of executing arbitrary code from the memory. In other words, a controller is a Central Processing Unit (CPU) of a node.

- Memory: This is the main resource for storing programs and intermediate data coming from the sensors or the communication device. The size of the memory depends on the application of the sensor.

- Sensors: Sensors are the interface to the physical world as they monitor the critical events in the environment and report the data to the controller. In our case, a sensor is a strain gauge which can accurately measure strain on a tire surface and send this data to the controller for processing.

- Communication Device: The communication device is used to send data to other nodes to form a sensor network. We are using wireless communication in our case which is more interesting than wired communication. We will be using radio frequency (RF) as a transmission medium to send data wirelessly. The specific frequency band for wireless transmission will be discussed in later chapters. The communication device is equipped with a transceiver which can take data from a controller to send it over the air and vice versa.

- Power Supply: A sensor node gets the energy to perform the computation and

communication from a power supply. A power supply is usually some form of batteries with limited energy. Sometimes, we can harvest power from the environment such as solar cells or kinetic energy to recharge the batteries.

3.3 Node-Level Software Platforms

The sensor nodes can be described as little computers designed to do small task such as measuring strain. Therefore, these nodes need an operating system to measure and send data back to the base station. The operating system of an embedded system only provides the service required by the nodes due to limited resources.

3.3.1 Operating Systems

TinyOS is the operating system used by the radio module shown in Figure 3.1. This survey will cover TinyOS in detail as this tool is used with the Berkeley Motes in measuring strain of a tire.

3.3.1.1 TinyOS

TinyOS is an open-source, event based operating system for embedded network sensor devices. It has many advanced features such as multi-hop support, efficient routing algorithms, wireless reprogramming, wireless database queries, and a custom C style language called NesC which allows for rapid application development. TinyOS allows a programmer to develop and load an application onto a mote that collects data from any specific sensor used and transmits that data back to the base station. TinyOS has no file system. It supports only static memory allocation, implements a simple task model, and provides minimal device and network abstractions to ensure that application code is small. TinyOS takes a language based approach which makes the needed part of the operating system to be compiled with the application. An application developed by NesC wires these components together [13].

3.3.1.2 NesC

NesC is an extension of C which is used to develop TinyOS applications. The programming paradigm used in NesC is different from the typical procedural paradigm used in most languages such as Java and C/C++. NesC separates construction and composition; programs are built out of components which are assembled to form whole programs. Components can be sub-divided into two sections, one for specification and one for implementation. Interfaces are meant to

represent the functionality of the components being provided to the user. The component behavior could be defined in terms of the interfaces it provides. Interfaces are also bi-directional meaning they can be used by the provider using a command type function or the user of the interface using an event type function. This is important because commands in TinyOS are non-blocking as their completion is signaled by an event.

3.4 Networking with Sensors

Networking is a key part of WSNs. The radio communication is the most expensive operation in terms of energy usage. Therefore, a node must find efficient ways for communication to save energy. The wireless nodes are deployed in an ad hoc manner which creates issues such as node failures, unstable links, and network disconnections. Efficient protocols, routing algorithms, load balancing and energy awareness can counter most of the issues stated above. This section will discuss the issues stated above with a generic sensor network for strain measurements as shown in Figure 3.3. Figure 3.3 has two transceivers for every tire which sends data to a base station which will be located on a car dashboard. The following sections will describe Media Access Control (MAC) schemes, topology, and networking of a generic sensor network for the tire shown in Figure 3.3.

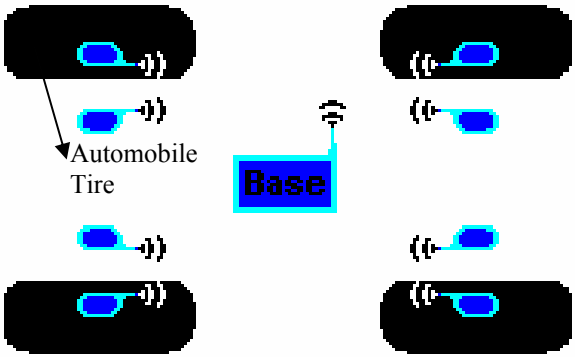

Figure 3.3: Sensor Network for Strain Measurement of an automobile with four tires

3.4.1 MAC Protocol

Sensor network is a distributed and a collaborative system with a main goal to improve overall application performance. TinyOS uses the 802.15.4 CSMA/CA MAC protocol for wireless networks. CSMA/CA allows for sensing of the channel used by the motes to see if anyone is currently sending data. If there is data being currently sent on the channel, the mote which intends to transmit will wait for a random amount of time before attempting to transmit known as back-off period. If the mote attempts to send data while another mote is simultaneously sending

data there will be a collision. The network stack used by TinyOS performs the other necessary data link layer processing such as preamble detection, time synchronization, and Cycle Redundancy Check (CRC) calculations. The portion of the network stack responsible for CSMA/CA as well as preamble detection is the ChannelMonC component. While it is detecting if the link is in use, it is also simultaneously searching for the preamble. The preamble is a set of bits that indicates that a packet is being sent. Once a preamble has been detected, it begins looking for the start symbol which indicates where the beginning of the packet is. The next 2 bytes after the preamble are the synchronization bytes that are used to synchronize timing between two motes. The next quantity to be calculated or checked is the CRC code. This 2 byte code checks to see if the bits set in the payload are accurate and were not manipulated by the noise. The current version of TinyOS is 1.1.10 which does not implement reliable data transfer such as NACK/ACK for packets. This is, however, planned to be implemented in later versions of TinyOS.

It is important to know that a sender can send a packet to the base station through another node if its transmissions cannot reach the base station. These intermediate nodes act as routers in forwarding the packet to the next-hop neighbor. A routing algorithm, therefore, is necessary to efficiently send the data between two end-points.

3.4.2 Routing

Wireless strain networks require a reactive routing protocol to keep up with the rapid changes in the topology of the network, due to status change in nodes or links. Therefore, methods such as geographic and energy aware routing can be used. A combination of these techniques can be used to establish power minimizing high quality links to the base station [13]. The link quality estimators can be any performance parameters such as throughput or end-to-end delay or any thing else. The estimator used in our protocol is reliability of the link. Reliability is defined as the packet being correctly decoded at the receiver. For example, the sender will always try to send packets directly to the base station first. If the packets cannot be transmitted successfully, then the sender will do a single hop with another node which has a direct reliable link to the base station. This process will continuously be checking for reliable transmission from direct connection to 4 hops link in the network specified in Figure 3.3 until a reliable path is found. Therefore, the topology of the network will be constantly changing to get the best reliability for all the nodes.

3.4.2.1 Multi-Hop Routing Algorithm

Sensor nodes which are deployed in the field are required to conserve the energy in all the operations that they undertake which are active listening, data reception and transmission. In general, the largest amount of energy is required in the transmit operation and the transmit power further increases if the distance between a sender and a receiver increases. To conserve energy and hence transmit power sending node transmits to its neighbors which are short distance away. Neighbors transmit the packet to their neighbors and in this manner the packet ultimately reaches the destination. Multi-hop routing under TinyOS relies on creating a mesh among all the nodes.

The data movement and route decision engines in TinyOS are split into separate components with a single interface between them to allow other route-decision schemes to be easily incorporated for future traffic. Use of the multi-hop library does not affect other applications [14].

Figure 3.4 shows an example of multi-hop algorithm with Crossbow®. In Figure 3.4 (a), all the nodes have the direct connection to the base station as the most reliable link. However, Figure 3.4 (b) shows that when node 3 loses the direct connection to the base station, it creates a one hop link to the base station through node 2. In Figure 3.4 (c), node 3 makes one hop link to base station through node 1 since node 2 is inactive due to failure. The routing and topology of the network in Crossbow® reacts to unstable links and node failures by using multi-hop routing which also helps if quality of service (QoS) is important.

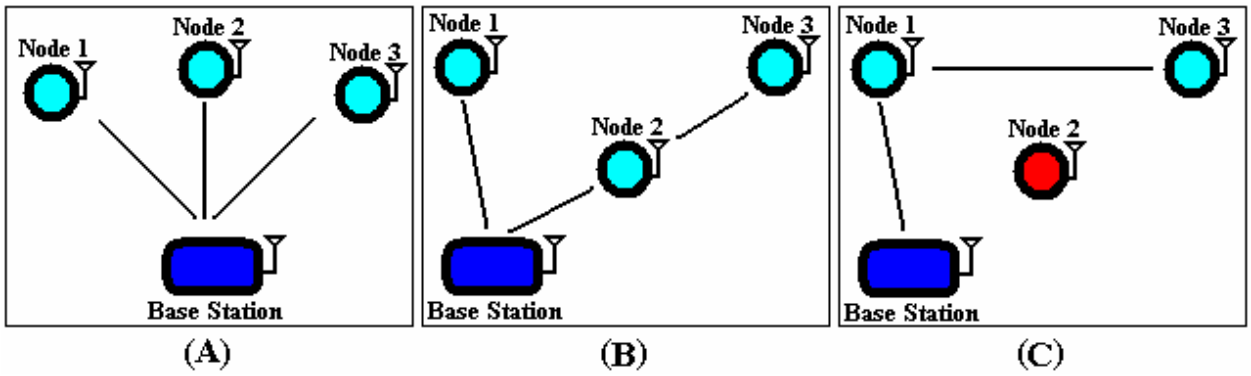

Figure 3.4: Multi-Hop with Crossbow®

3.4.3 Transport Protocol

In this section, we discuss issues of multi-hop routing, and data reliability and ways in which these properties are achieved in a WSN transport protocol. Transport protocols can reduce

congestion in a network and packet loss in a wireless medium, and to guarantee data reliability [15]. The transport protocols currently being used for internet cannot be used for WSN due to the ad-hoc nature of WSN. Data reliability is an important part of any WSN and currently there are no reliable transport protocols for wireless sensor networks as a result packets are easily lost. We need a dynamic data rate control and congestion avoidance transport protocol to create a reliable and energy efficient WSN. Our focus in this research has been mainly on routing layer of a WSN. In future, one can look into developing a reliable transport protocol.

3.5 Conclusions

In this chapter, we presented some of the key components required to create a strain sensing wireless system such as processing units that measure strain and transmit it to other node including the user. This information leads the way into some of the approaches to measure strain of an automobile which are discussed in the next chapter.

Chapter 4
Strain Measurements of Tires

Tire manufacturers are looking to develop advanced tire sensor systems to monitor strain, temperature, and air pressure of a tire. A few of the possible strain measuring solutions that utilize a wireless sensor system in a tire are considered in the following sections. In this chapter, we discuss in section 4.1 a centralized approach to measure strain using the capacitance method. However, a centralized system cannot predict localized damage on the tire which requires an alternate solution which is presented in section 4.2.

4.1 Global Strain Sensing System

A sensor sensing system that can measure the overall strain of a tire without localizing the damage on the tire will be considered a global strain sensing system. The capacitance method described below is an example of a global strain sensing system.

4.1.1 Capacitance Method

As stated in section 2.3, capacitance can measure strain of a structure which closely resembles a dielectric condenser such as an automobile tire. A simple tuning circuit can be used to transmit tire capacitance wirelessly to an external receiver. This system is composed of an external transmitter, a strain sensor, and an external receiver.

The transmitter emits white noise produced by a function generator that can be picked up by a tuning circuit antenna located in the tire. The tuning circuit is composed of the conductance of a coil (L in Henries), the capacitance of a tire (C_x in Faradays), and a resistance in Ohms. The tuning circuit is just an LC circuit which has the following impedance (Z):

$$Z = \frac{1}{j\left(\omega C_x - \frac{1}{\omega L}\right)} \tag{4.1}$$

The denominator in equation 4.1 is set to zero to find the tuning frequency (f_t) which is a function of the tire capacitance.

$$f_t = \frac{1}{2\pi\sqrt{LC_x}} \tag{4.2}$$

The external receiver can pick up the tuning frequency. This experiment can be done to find the relationship between strain and capacitance using strain values known beforehand. This method is a passive wireless system therefore its life is not limited by a battery. Also, this method does not require a suitable strain gauge that can measure strain accurately of a tire rubber. However, this method cannot find the exact location of the damage on the tire. Additionally, this method may not be able to provide enough resolution of strain signal to differentiate between a normal strain and a catastrophic failure [11].

4.2 Local Strain Sensing System

A strain sensing system that can measure the strain on a specific location on a tire using a strain gauge is called a local strain sensing system.

4.2.1 Wireless Strain Gauge

Localized strain sensing system requires a strain gauge, a transmitter, a receiver, and a data display device which can be a PC. Here, we study two possible schemes using MicroStrain® V-Link® and Crossbow® motes.

4.2.2 MicroStrain® V-Link®

MicroStrain® V-Link® is a wireless transmitter which works in the 2.4 GHz frequency range. V-Link® measures and transmits data wirelessly for a specified period of time. V-Link® can also log data to onboard memory which can be wirelessly downloaded at any later time. Multiple V-Link's® can log data simultaneously, however, only one device is allowed to talk over the airspace to the base station at a given time. In the case where simultaneous communications at a given time are desired, one base station per communicating node is required.

Each node can sample data for a specific time interval and store it in its own memory. The base station will cycle through each node retrieving the logged data one by one. This process can be performed fast enough to seem like all nodes are simultaneously streaming their data to the base station. A picture of a V-Link® wireless transceiver is shown below:

Figure 4.1: V-Link® Wireless Transceiver

V-Link® has four differential inputs with a built-in hardware amplifier and an offset to measure signals that are in the millivolt range such as strain sensors. V-Link® comes with a default full-bridge completion. V-Link® can work with a half-bridge or a quarter-bridge completion, but that circuit must be completed outside the node before connecting to the differential input. The node also provides the excitation voltage needed for any sensor. The schematic of the MicroStrain® V-Link® strain sensing system is shown below:

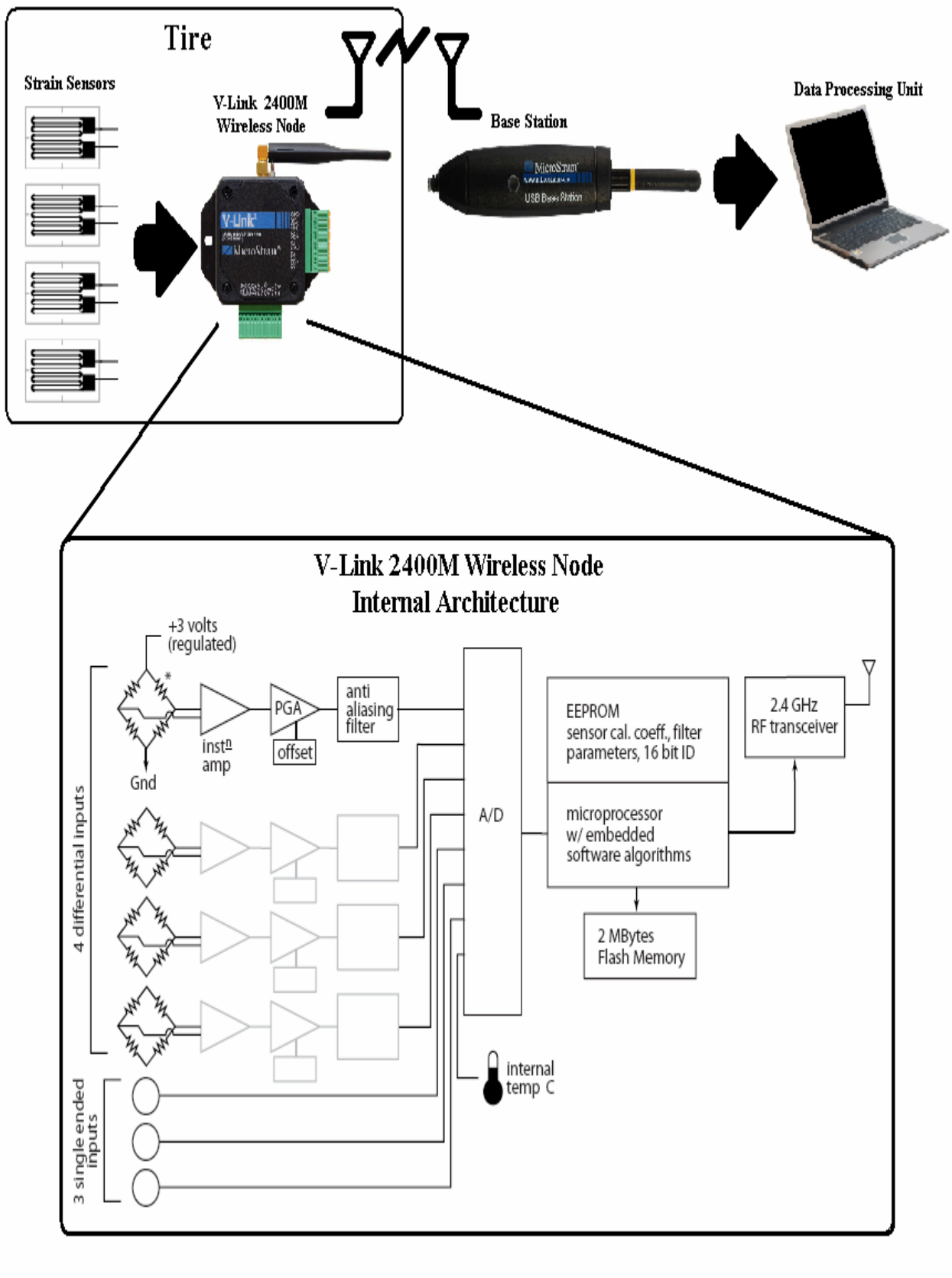

Figure 4.2: Schematic of a V-Link® Strain Sensing System

Figure 4.2 shows that a bridge output voltage goes through a Programmable Gain Amplifier (PGA). Amplification of the signal is needed to get more resolution since the change in the bridge output voltage is usually very small. An anti-aliasing filter is used to make sure that data is not corrupted by noise outside the frequency band. The data is then sent through an Analog to Digital Converter (A/D) to change the voltage signal into bits that can be sent wirelessly or saved in the 2 Megabytes of flash memory. This system uses 12 bit A/D converter which has the following relationship between volts and bits:

$$Outpit\ Bits = Output Volts \frac{4.096[Bits]}{3.00[Volts]} \tag{4.3}$$

The onboard offset and amplification values can be saved to the V-Link's® EEPROM for the programmable gain amplifier and offset components to use. The sampled voltage signal from the strain gauge will be amplified and offset by the given values and then converted to bits by the A/D converter. The converted sampled value will then be sent wirelessly by the 2.4 GHz transceiver to the base station where the measured strain will be calculated. The base station is hooked up with a USB cable to a PC to display the results for the user. Figure 4.3 shows the strain measurements which are the two peaks in the graph of micro-strain with respect to time. The resistive strain gauge of 350 ohms with a gauge factor of 2.075 and a quarter bridge arrangement used to measure strain with V-Link® is shown in Figure 4.3.

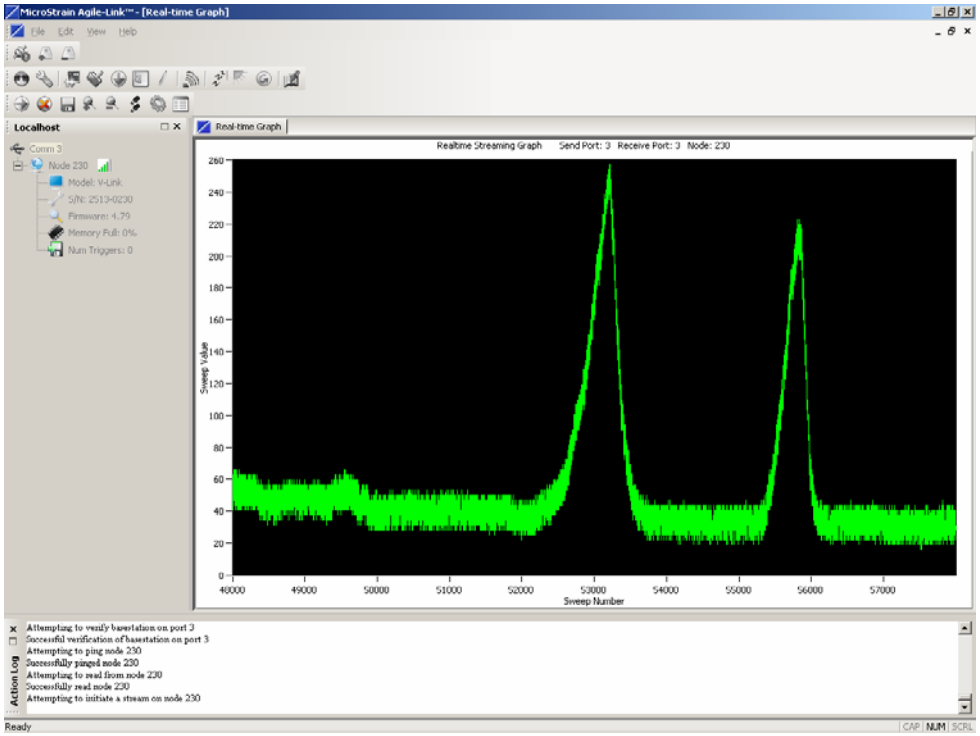

Figure 4.3: Strain Measurement with MicroStrain® V-Link®

4.2.3 Crossbow® Motes

This setup requires a transceiver (MPR410CB), a data logger (MDA300), and a base station (MIB510) as shown in Figure 4.4. The strain gauges are connected to the data logger. A transceiver is attached to the data logger and the base station to allow radio communication between them. Applications written for these motes use the TinyOS operating system and the NesC language. The application specifies a sample interval time which is used by the unit to pace its data readings. A radio packet is created with the sampled data and sent wirelessly once all the samples have been obtained from the data acquisition board. The base station then receives the packet and converts its data into meaningful units such as micro-strain and displays them to the user. The schematic of the setup is shown in Figure 4.5.

Figure 4.4: Data logger (MDA300) (left), Base station (MIB510) (middle), and Transceiver (MPR410CB) (right)

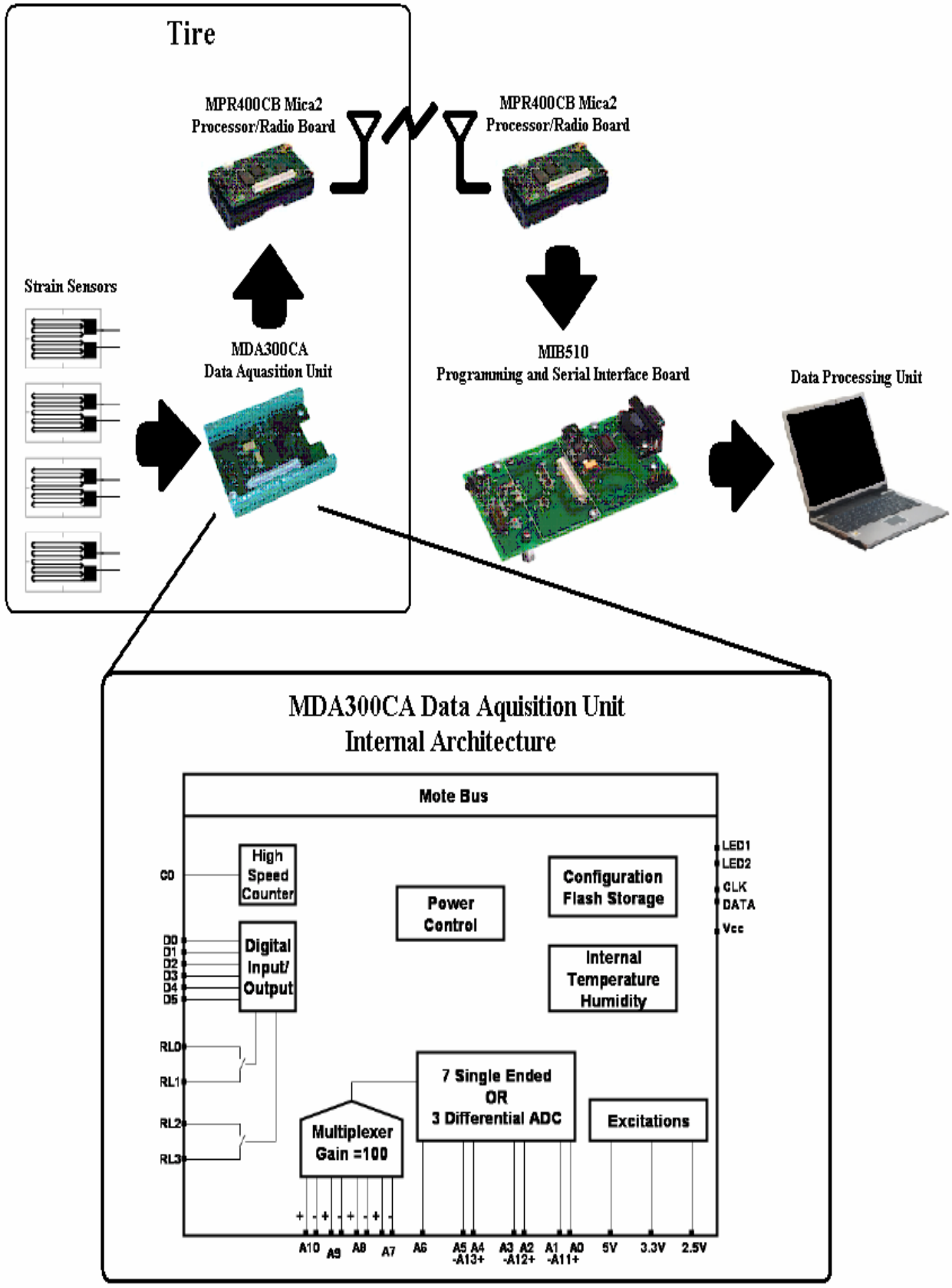

Figure 4.5: Schematic of a Crossbow® motes strain sensing system

This setup shown in Figure 4.5 can support up to four strain gauges at a time. The strain data is logged into the data logger which can be sent to the base station at a later time. The mica2 transceivers are capable of working on multiple frequencies including 315 MHz, 433 MHz, 915 MHz, or 2.4 GHz ISM band. The mica2 has excitation voltages for exciting external sensors at 5.0V, 3.3V, and 2.5V. These excitation voltage pins can be used for turning on active external sensors or for an external bridge completion circuit's excitation voltage. Channels A7 to A10 are the differential precision analog to digital converters. These channels use an internal amplifier with a gain of 100. These channels have a dynamic range of ±12.5mV. The gain is used to increase the resolution of the signal since strain signals are usually very small. The voltage is converted into bits using A/D converter. The signal in bits is sent wirelessly to the base station using the transceivers. At the base station, signals can be converted back into voltage by applying the following formula to the ADC value read in from the channel:

$$Voltage = 12.5[mV]\left(\frac{ADC - READING[Bits]}{2048[Bits]} - 1\right) \tag{4.4}$$

The voltage signal can be converted into micro-strain units using equations for a specific bridge completion. The calibration of these strain measurement needs to be done with known load and strain values. This step is needed to prove that we are measuring the strain accurately.

4.3 Conclusions

In this chapter, we have discussed a centralized and a few localized approaches to measure strain of an automobile tire. Both of theses approaches show some potential to be used as viable strain sensing systems for an automobile tire. We have successfully acquired some promising results using the localized approaches which are discussed in detail in the next chapter.

Chapter 5
Discussion of Results for Strain Sensors

This chapter deals with the strain measurement validation, which is described in section 5.1. In section 5.2, we discuss the Graphical User Interface (GUI) that was built for facilitating testing result visualization. We also present the challenges and issues with finding a suitable sensor for measuring strain of a tire in section 5.3.

5.1 Verification of Strain Results

This section describes how strain was created on a sample and how we took physical strain measurements to verify the strain measurement reported by the Crossbow® motes. The first part of this section explains the setup required to predict strain on a steel bar using theoretical procedure. The later parts of this section discuss the mathematical equations needed to calculate theoretical strain on a steel bar. In the end, the experimental results are compared with theoretical measurements to confirm the validity of the results. This step provided us the confidence that we are measuring strain accurately using Crossbow® motes.

5.1.1 Theoretical Strain

Theoretical methods can be used to predict strain with reasonable accuracy for a simple structure such as a beam. However, theoretical methods are not very reliable for real world complex structures such as a wing of an aircraft or a robotic arm of a space probe. For complex structures, finite element or finite difference models are used for predicting the strain. However, simple analytical methods do provide accurate strain computations for a simple structure used in this experiment.

5.1.1.1 Apparatus

The apparatus used in this experiment consists of a simple beam which is a steel bar, a frame in which that beam can be mounted and its deformation measured, and a loading fixture that allows weights to be attached to the beam.

The beam, shown in Figure 5.1, is constructed from aluminum alloy and has a rectangular cross-section with dimensions of about 1/4 by 1 1/2-inches. We measured the actual cross-sectional dimensions with the dial caliper provided, and used those actual dimensions in the calculations. The two electrical strain gages connected to the beam are used to measure strain values which should be close to theoretical values.

Figure 5.1: Aluminum Alloy Test Beam

The loading fixture and one of the weights are shown in Figure 5.2. The two Allen bolts in the loading fixture can be loosened so that it can be slid onto the beam. The weights are machined from aluminum bronze rod stock of 5-inch nominal diameter. They are nominally 2, 3, 5, 10, and 15 pounds, and each has a hook threaded into a tapped hole on one side for hanging them from the eye screw of the load fixture. A digital weighing machine is used to measure the exact values of these weights.

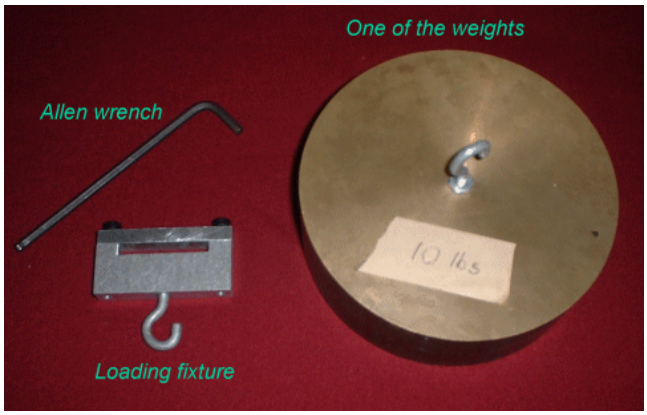

Figure 5.2: Loading Fixture and Example Weights

The loading frame is shown in Figure 5.3. The beam, loading fixture and weight are assembled in one possible configuration.

The frame includes a vertical beam support that can each be moved to any one of seven positions along the bottom of the frame which are 2.5 inches apart. The beam can also be fixed using the slot cut into the left hand side of the frame, and the clamp. The slot holds the beam at the same height as the vertical supports. This device is very flexible which allows for many beam

configurations. Therefore, it is ideal for conducting simple tests of strain measurements of simple structures [16].

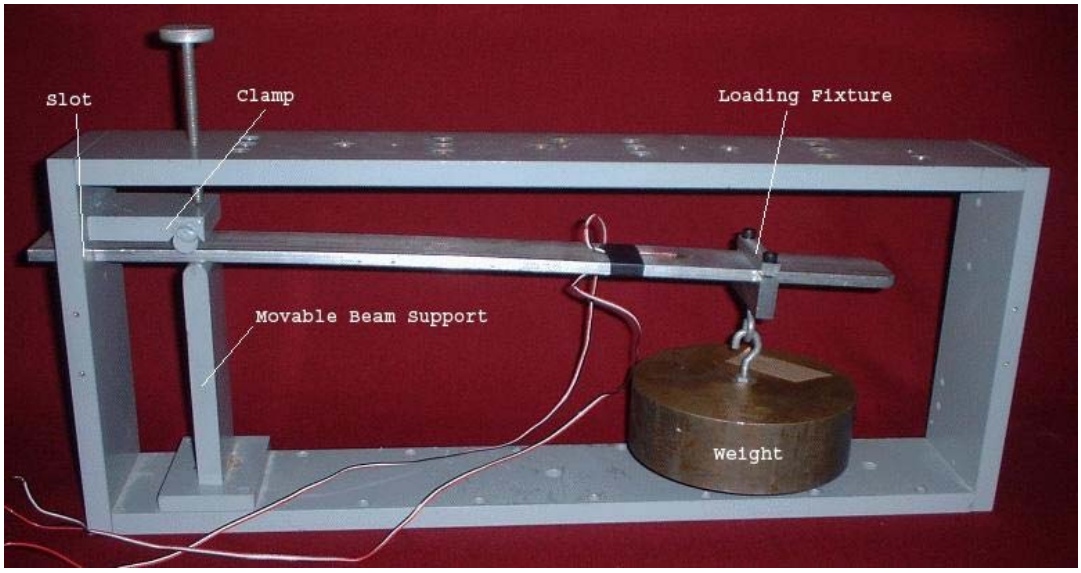

Figure 5.3: Cantilever Beam Arrangement

5.1.1.2 Strain

Figure 5.3 shows the experimental setup for measuring the strain of a steel beam. The displacement in the beam is proportional to the force applied downward by the weight. The force applied downward is proportional to the bending moment of the beam. Therefore, one can measure strain applied on the bar by calculating the displacement in the beam for a given weight. The following sections provide equations for calculating bending moment.

5.1.1.2.1 Theoretical Definition of Strain

The axial strain in the beam, ε_x, is proportional to the bending moment and is given by:

$$\varepsilon_x = \frac{My}{EI} \tag{5.1}$$

where, M is the bending moment at the strain gage, E is the Young's Modulus of elasticity of the material, y is the distance from the neutral axis of the beam to the surface, and I is the second area moment of the cross section of the beam.

The bending moment of the beam, M, is given by the following equation:

$$M = P(L - x) \tag{5.2}$$

where, P is the force applied downward due to the weight, L is the length of the beam from clamped edge to the load point, and x is the distance from the clamped edge to the strain gage location. These variables are labeled clearly in Figure 5.4 with a weight hanging on the loading fixture.

The second area moment of the cross section of the beam, I, is given by the following equation:

$$I = \frac{1}{12} bh^3 \tag{5.3}$$

where b is the width of the beam and h is the thickness of the beam. Again, these dimensions are clearly labeled in Figure 5.4.

For small loads that are used in this experiment, the distance from the neutral axis of the beam to the surface, y, can be approximated with the following equation [16]:

$$y = \frac{h}{2} \tag{5.4}$$

If we substitute for M, I, and y in equation 5.1, we get the following equation for the axial strain:

$$\varepsilon_x = \frac{6P(L-x)}{Ebh^2} \tag{5.5}$$

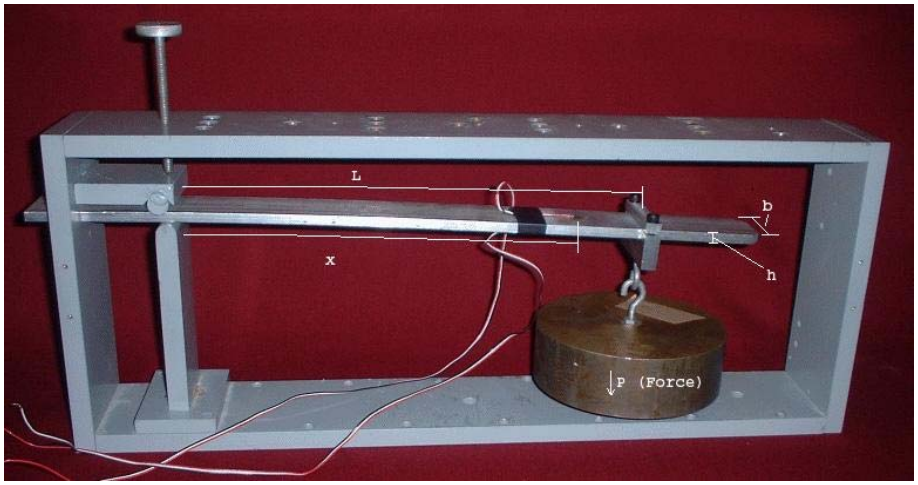

Figure 5.4: Experimental Setup with labeled dimensions

5.1.1.3 Strain Calculations for given Weights

Using a caliper and measuring tape, the dimension of the beam that was used in the testing are:

$$L = 352.4\,mm, x = 270.5\,mm, b = 38.4\,mm, h = 6.45\,mm, E = 70x10^9\ N/m^2$$

The following table shows the axial strain calculated using equation 5.5 for three different loads:

Load (kg)	P (N) = Load *(9.8 m/s^2)	Micro-strain (με)
0.888	8.71	36.9
1.376	13.49	57.18
4.782	46.91	198

Table 5.1: Strain Calculations using Theoretical Method

5.1.2 Vishay Micro-Measurement Strain Indicator

We also used Model P-3500 Strain Indicator by Vishay Micro-Measurement to get strain measurements on a steel bar. The Model P-3500 Strain Indicator is a portable, battery-powered instrument with unique features for use in stress analysis testing, and for use with strain gage based transducers. This step provides another set of data to compare with which increases the validity of strain readings obtained by the Crossbow® units [16]. This setup requires connecting the strain gage to a quarter bridge configuration to a P3500 Strain Indicator and SB10 Switch and Balance unit as shown in Figure 5.5.

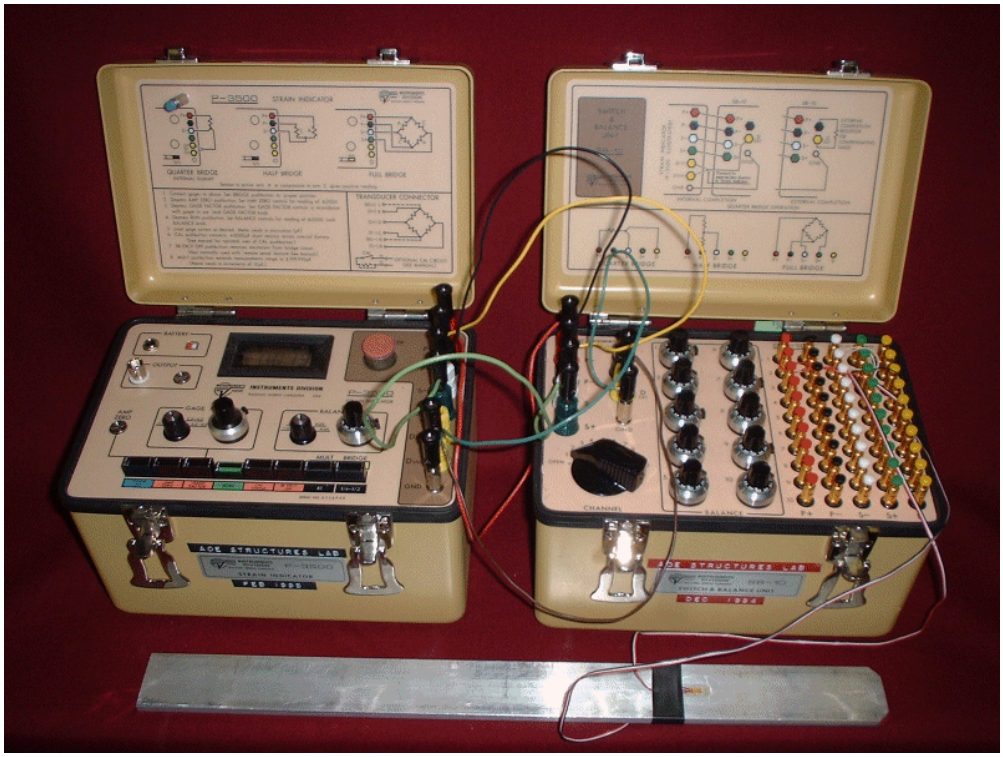

Figure 5.5: P3500 Strain Indicator (left) and SB10 Switch and Balance Unit (right)

5.1.3 Crossbow® Strain Indicator

The bridge output voltage of the quarter bridge is the input to the differential channel of the Crossbow® data acquisition mote. The data acquisition will convert the voltage difference into an ADC reading. This ADC reading can be converted back to voltage in the software using the following equation.

$$Voltage = 12.5[mV]\left(\frac{ADC - READING[Bits]}{2048[Bits]} - 1\right) \tag{5.6}$$

Strain can be calculated as micro-strain units from the voltage using equation 5.7 with excitation voltage of 2.5 volts and gage factor of 2.145.

$$\frac{V}{E} = \frac{G_f \varepsilon \times 10^{-3}}{4 + 2G_f \varepsilon \times 10^{-6}} mV/V \tag{5.7}$$

where, G_f is the active strain gage's gage factor, and E is the bridge's excitation voltage.

5.1.4 Discussion of Results

This following table summarizes the strain readings obtained by theoretical method, P3500 Strain Indicator, and Crossbow® readings:

Load (kg)	Theoretical Strain (με)	P3500 Strain Reading (με)	Crossbow® Reading Median (με)	Crossbow® Reading Error from Theoretical (%)	Crossbow® Reading Error from P3500 (%)
0.888	36.9	41	37.665	2.073	8.134
1.376	57.18	64	67.716	18.426	5.806
4.782	198	206	202.814	2.431	1.547

Table 5.2: Strain Measurements

The strain readings obtained from the Crossbow® nodes are very close to the theoretical and P3500 readings which is also shown by the following plots for the three weights:

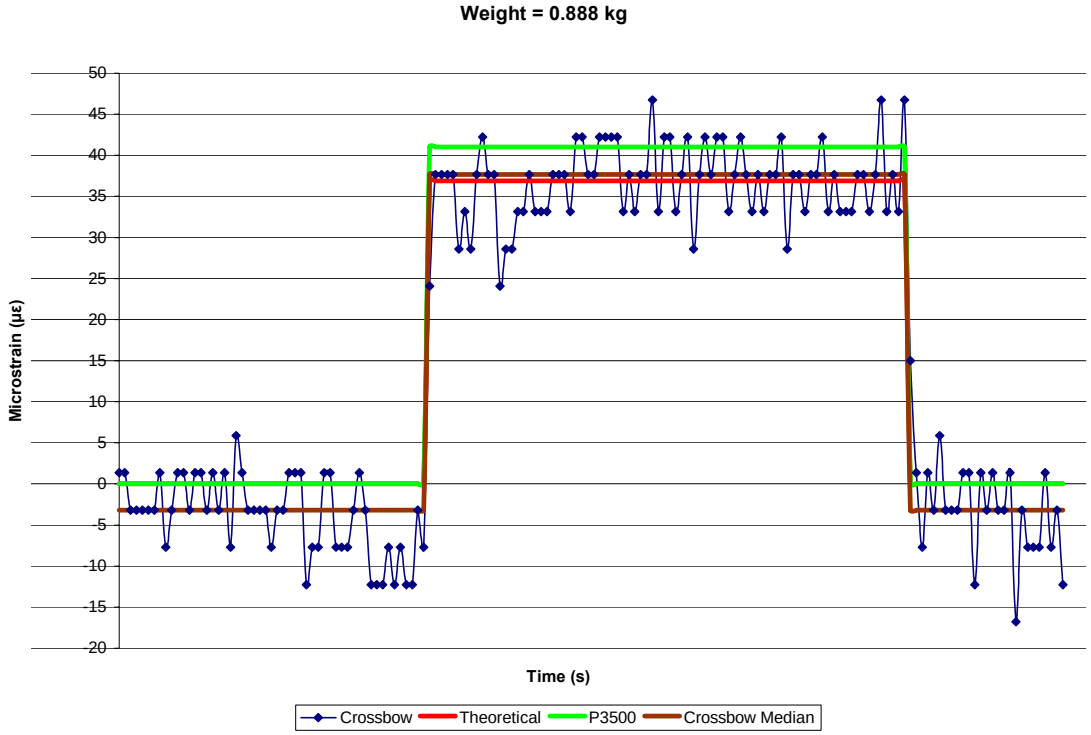

Figure 5.6: Strain Readings for Weight = 0.888 kg

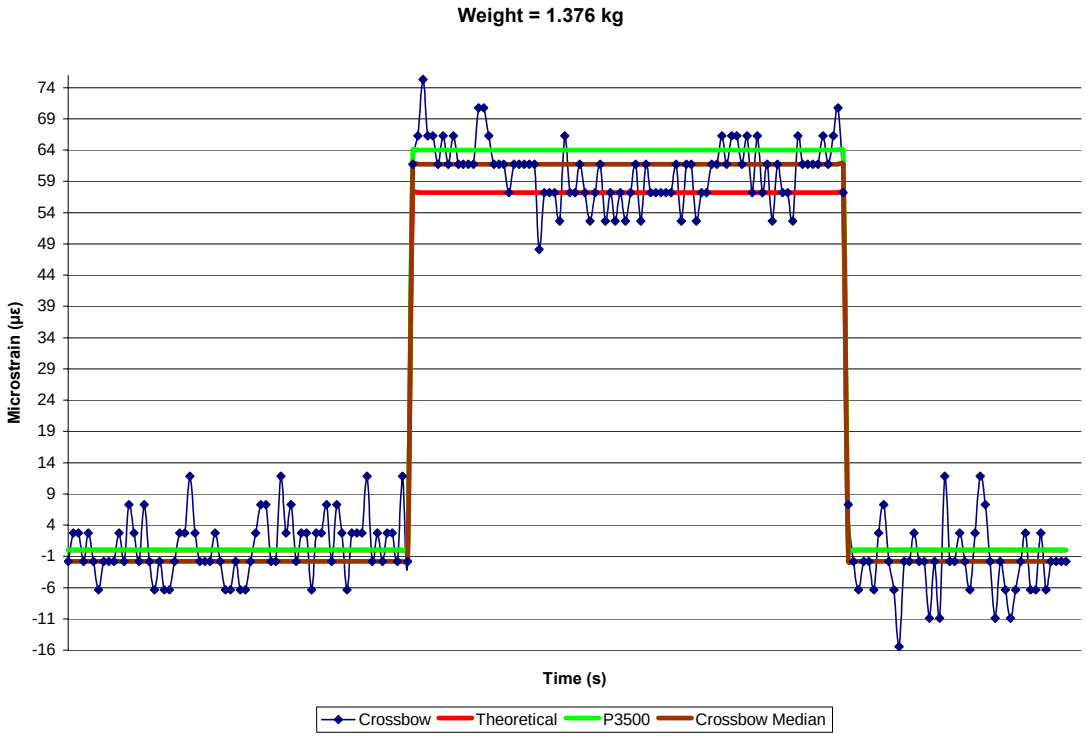

Figure 5.7: Strain Readings for Weight = 1.376 kg

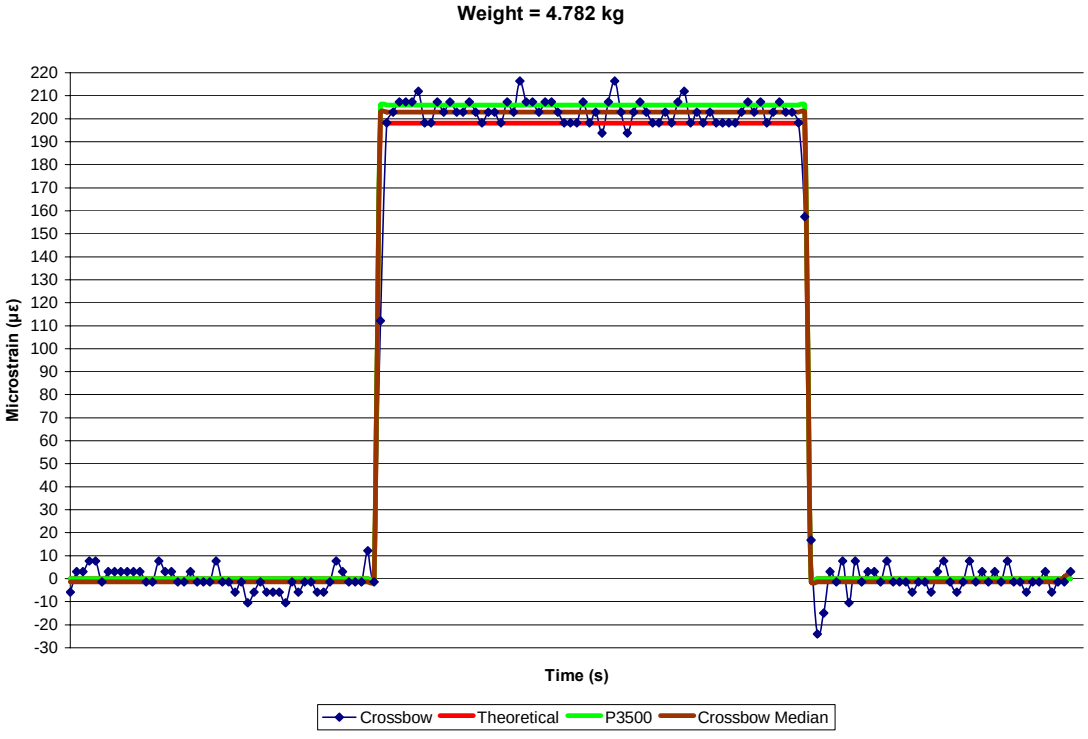

Figure 5.8: Strain Readings for Weight = 4.782 kg

5.2 Mote Monitoring

For testing purposes a console based application was implemented which would listen for messages sent by the base station to a computer's communications port. These messages contain information about the WSN such as each mote's battery power, temperature reading, and strain readings. The purpose of this application was to display the information held within these messages to trouble shoot and test the WSN as well as the sensing.

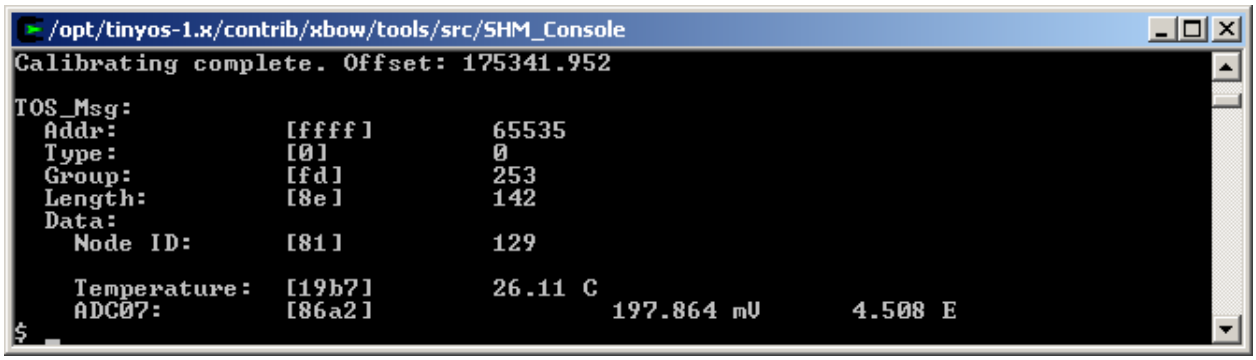

Figure 5.9: The console based mote monitoring application

The console based application was not as easy to read or decipher because it did not show the results in a nice plot. We agreed upon implementing a visually appealing application which can plot these values onto a graph. Such an application would significantly increase our performance and speed up the testing process.

We are currently developing the Graphical User Interface (GUI) application which will plot the mote's sensing readings. This application is called the SHM Mote Monitor.

The SHM Mote Monitor should be able to show the motes communicating with the WSN, plot graphs for the sensing readings, and record important events that can not be shown graphically.

5.2.1 Application Layout

The SHM Mote Monitor application was divided into three regions. Figure 5.10 shows these three regions as the Event Log, Network View, and the Graphing Workbench. The Event Log and Network View regions can be hidden to give the user a larger workspace for the graphs. As implied by their names, the Event Log will record important events concerning the application's status and the messages received from the WSN while the Graphing Workbench will contain the graphs that are currently opened. The Network View region will hold a tree like structure of the motes that are within the WSN. The motes displayed under the Network View region will

contain tasks relevant to their purpose within the WSN. For instance, a sensing mote will contain a task which enables its strain readings to be plotted onto a graph. The following sections will discuss these regions in more detail.

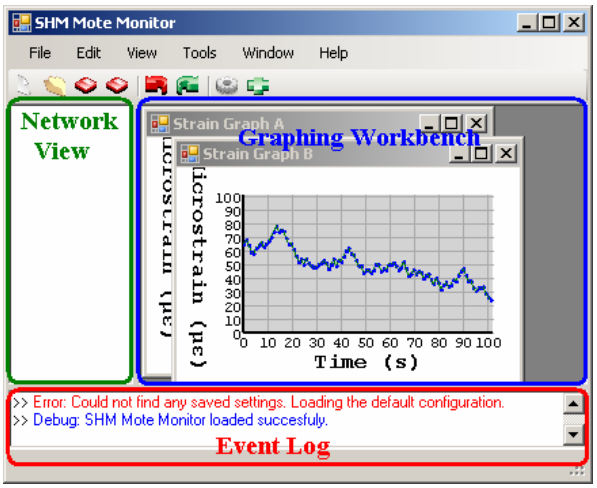

Figure 5.10: SHM Mote Monitor region layout. The red region contains the Event Log, blue contains the Graphing Workbench, and green contains the Network View

5.2.1.1 Event Log

The Event Log region acts as a text based terminal that can keep the user updated with important events concerning the status of the application and its data flow. These events are partitioned into categories which can be shown or hidden as well as color coded to easily distinguish the different event categories from each other. Currently there are three categories included by default; Error, Debug, and Packets. As their names imply, the Error event category will display error messages while the Debug event category will show debugging messages. The Packets event category formats the sent and received network messages into their Raw, Parsed, and Converted data representations. The Raw data representation simply displays the message as it was received; a series of hexadecimal values. The Parsed data representation will parse the message into its corresponding values and label them. For example, a sensing mote's message can be parsed into the mote's ID, battery power, and strain readings. On the other hand, the Packet's Converted data representation will take an extra step and not only parse the packets; it will convert the hexadecimal values into their meaningful engineering unit representation.

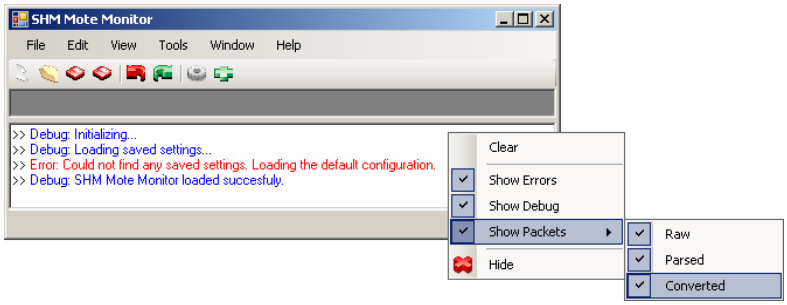

Figure 5.11: The SHM Mote Monitor's Event Log region with the menu strip expanded to show its functionality

5.2.1.2 Graphing Workbench

The main purpose of the SHM Mote Monitor applications is to graphically represent the status of the WSN. The Graphing Workbench region holds graphing windows such as strain graphs or network topology graphs. These graphing windows act as child windows for a Multiple-Document Interface (MDI) application. In other words, the graphing windows act as child windows to the Graphing Workbench region where they can be moved, maximized, minimized, tiled, and cascaded without overlapping the other regions. There is no limit to the number of graphing windows that can be shown on a screen at a given time.

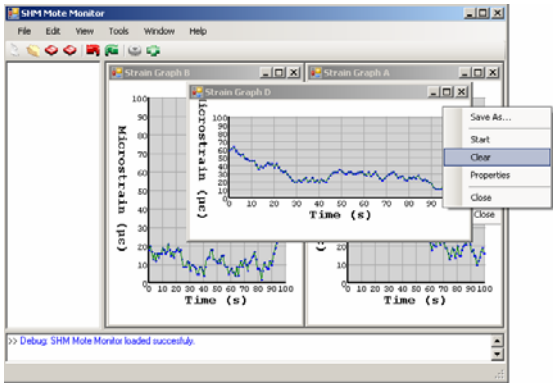

Figure 5.12: SHM Mote Monitor's Graphing Workbench region with three strain graph windows opened

5.2.1.3 Network View

The Network View region shows the WSNs that are currently connected to the application. It will arrange the motes that are connected to the base station in a tree view diagram which can be expanded or compressed. The base station and the motes with which it is communicating with will have their own menu strip for additional tasks. These tasks vary according to whether the mote is acting as the base station, a relay, or a sensing mote. For instance, a sensing mote may

have a “create strain graph” menu strip task to plot its strain readings. While the base station mote will have a “view topology” task to graphically represent how the motes are interconnected.

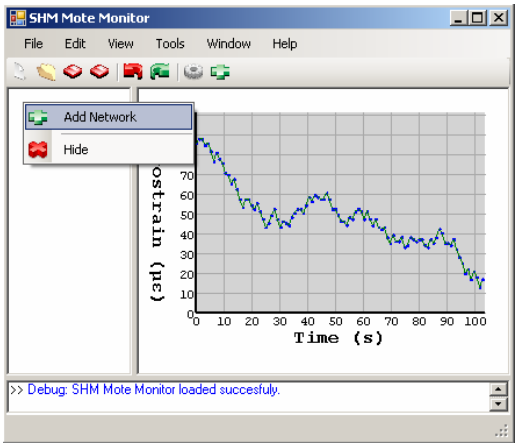

Figure 5.13: SHM Mote Monitor’s Network View region with its menu strip expanded

5.2.2 Network Communication

The SHM Mote Monitor will need to be able to communicate with the WSN’s base station. A Communications class will be required in order to do so. The Communications class will handle the application’s networking tasks by sending and receiving messages via sockets for all the open WSN connections.

The incoming messages would be parsed, analyzed, and their data will then be sent to its relevant destination of the application. In other words, if we receive a sensing mote’s message, the Communications class would parse the message and find out what mote and mote type sent it. The mote type will notify that this mote is a sensing mote and the message may include sensor readings. Then it would send the sensor readings to its corresponding Graphing Window to be plotted.

5.2.3 Application Design

A design had to be constructed before we got started with the application’s implementation. Figure 5.14 shows the SHM Mote Monitor’s class diagram. Each box symbolizes a different class or module for the system.

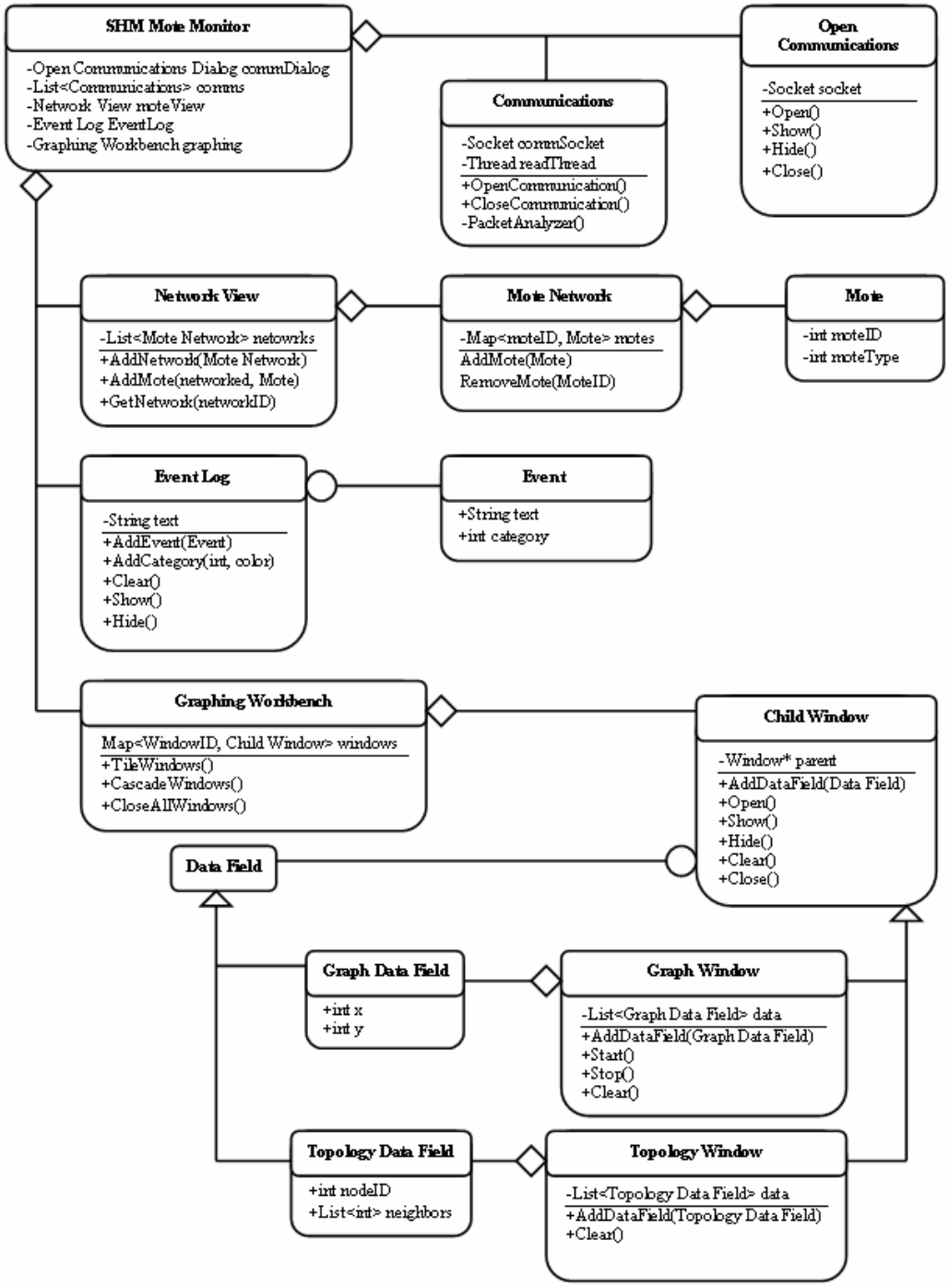

Figure 5.14: Class diagram for the SHM Mote Monitor application

5.2.4 Future Work on GUI Development of Mote Monitoring

Currently we are working on the socket based network communication class. Once the Communications class is complete we can move on to the Network View region and create a tree structure for the motes that are communicating with the open networks. These two components are the main segments of the application that are yet to be completed.

Some additional features have yet to be implemented as well. Some features include the MDI window which can graphically represent the networks topology, the Topology Window class, as well as a properties dialog to edit the graph window's settings. These settings will allow the user to change the graph's color scheme, label names, and show or hide the grid, lines, labels, values, etc…

5.3 Sensors

A new strain gage needs to be discovered that can measure strain on a rubber surface accurately. This sensor is referred as "Rubber Sensor" in this thesis. Rubber sensor is shown in Figure 5.15. Currently, resistive strain gages can read strain up to 10% linearly. Tires may undergo much higher amounts of strain.

Figure 5.15: Rubber Sensor

5.3.1 Metal Rubber™

NanoSonic Inc created a revolutionary material called Metal Rubber™ which is shown in Figure 5.16. This material is conductive like metal, and is flexible as rubber [5]. Such material properties are ideal for a possible sensor that can measure strain on a tire. Basically, the resistive strain gage will be replaced by Metal Rubber™ sensors which will have their specific conductivity. The conductivity of Metal Rubber™ sensors changes by the strain which shows a possibility of a feasible strain sensor. In order to test the change in conductivity by strain, we tested the conductivity of Metal Rubber™. The experimental setup and the conductivity results are discussed in the following sections.

Figure 5.16: Metal Rubber™

5.3.1.1 Experimental Setup

We used a stretching machine developed by another student, Thomas McQuigg, at Virginia Tech to stretch the Metal Rubber™ as shown in Figure 5.17. Force applied on the testing material can be measured by the scale provided on the machine. The applied force is useful information but we were more interested in measuring the displacement in the Metal Rubber™. We measured the displacement manually using a measuring scale. Manual measurement adds uncertainty to the results due to the human error. However, we were just trying to get a sense of this material conductivity reaction to displacement. This experiment is performed with an improved setup by another graduate student, Mohammed Rabius Sunny. The data obtained by this simple experiment helped Sunny to set up his experiment which is discussed elsewhere.

Metal Rubber™ is secured in the stretching machines using the clamps on the machine. A non conductive material is put between the Metal Rubber™ and the clamps to ensure that metal clamps cannot affect the conductivity of Metal Rubber™. The conductivity of the material is measure using flat clips as shown in Figure 5.17 which are connected to a multi-meter which can measure the resistance. This simple setup provided some interesting results on the behavior of Metal Rubber™ which are discussed in the next section.

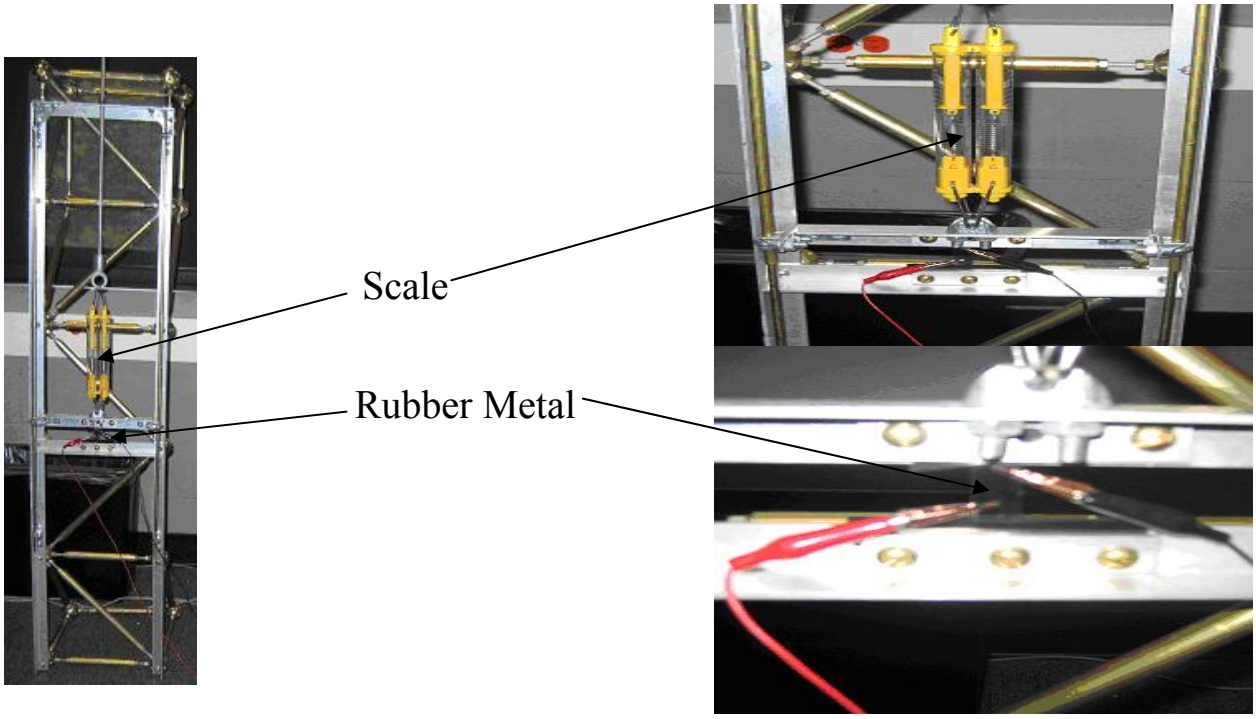

Figure 5.17: Experimental Setup

5.3.1.2 Conductivity Results

In our experiment, we put Metal Rubber™ under cyclic strain of 0%, 8.33%, 16.67%, and 25% for four cycles. In other words, the experiment started at time = 0, with no strain (0%) on the Metal Rubber. At the beginning of the experiment, we found the initial resistance of Metal Rubber™. After 1 minute from the start of the experiment, strain of 8.33% has been applied the material and resistance of the material is recorded. After 2 minutes from the start of the experiment, strain of 16.67% has been applied to material and resistance is measured. After 3 minutes from the beginning of the experiment, 25% strain has been applied to the material and resistance is measured again. After one minute since the material was stretched to 25 %, the strain has been reduced to 16.67% and resistance is measured. The resistance is measured in this way for four cycles in fixed time increments.

The plot of the measured resistance values with respect to cyclic strain is shown in Figure 5.18 and Figure 5.19. These plots clearly show that resistance of the material for a given strain changes for every cycle. Ideally, if a material has a resistance of 180 ohms for 8.33% strain in

one cycle, then the resistance should be close to 180 ohms for 8.33% in a different cycle. The plot shows that resistance of the material has some hysteresis built into it.

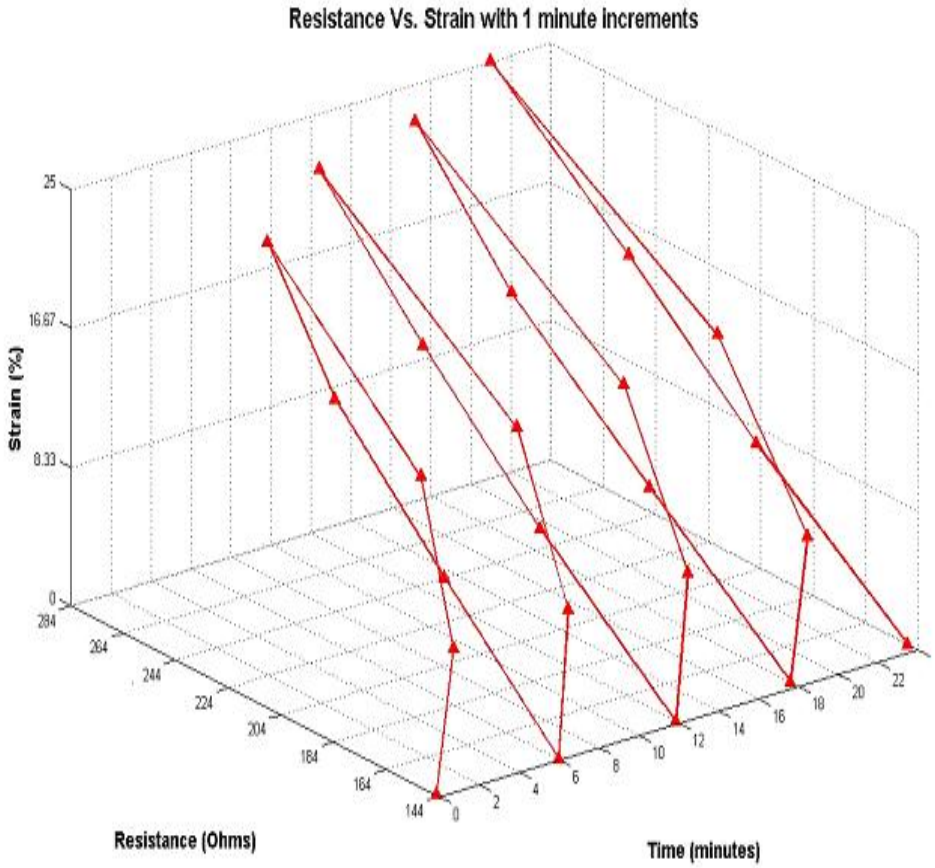

Figure 5.18: Resistance vs. Strain with 1 minute increments

Figure 5.19 shows the similar plot of the resistance of the material with respect to resistance in a two dimensions. We can see the similar hysteresis effect in this plot for the resistance of the material. An ideal sensor should have a straight line for the resistance of the material for a cyclic strain.

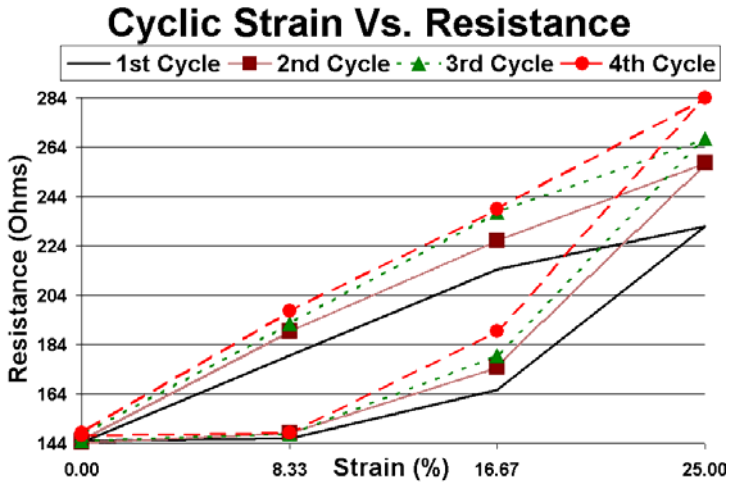

Figure 5.19: Cyclic Strain vs. Resistance

The same resistance of the material with respect to time only is shown in Figure 5.20. An ideal sensor should have shown a nice sine wave for the resistance of the material with respect to time. We can see that resistance keeps on changing for a given strain for every cycle.

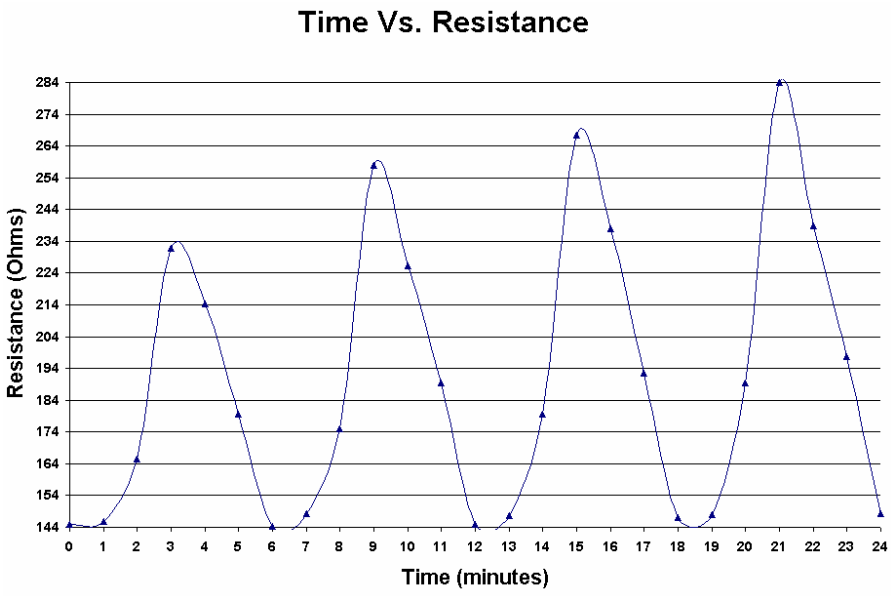

Figure 5.20: Cyclic Strain Resistance with respect to Time

5.3.1.3 Hysteresis in Metal Rubber™

The experiment discussed in last section showed the hysteresis effect with the conductivity of the Metal Rubber™. We need an approach that can deal with the hysteresis effect and measure strain accurately. A fractional derivate approach to minimize the error caused in the

measurements by hysteresis has been shown to be appropriate. This approach is discussed briefly in the following section.

5.3.1.4 Approaches to Address Hysteresis

The change of resistance in Metal Rubber™ can be viewed as a combination of applied strain and a separate function of strain. This relationship is shown in the following equation:

$$R(t) = C_0 + C_1\varepsilon(t) + \sum_{l=2}^{M} C_l D^{\alpha_l}(\varepsilon(t)) + \sum_{k=M+1}^{N} C_k \int_0^t \varepsilon(\tau)^{2*(k-M)} d\tau \tag{5.8}$$

where, $R(t)$ is the resistance as a function of time, $\varepsilon(t)$ is the strain as a function of time, $C_i(i = 0,..,N)$ and $\alpha_j(j = 2,..,M)$ are material parameters. Also, the term that causes strain to change with time is $\sum_{k=M+1}^{N} C_k \int_0^t \varepsilon(\tau)^{2*(k-M)} d\tau$. This term is used to model the hysteresis behavior of Metal Rubber™ in a mathematical form. In short, the mathematical model of hysteresis can be used to model a clear strain versus resistance relationship. This work is still being pursued by another other graduate student in the group.

5.3.2 Other Sensors that Measure High Strain

Hysteresis in Metal Rubber™ limits its effectiveness as a sensor as mentioned in earlier sections. Therefore, we looked into alternative sensors that can measure high strain accurately. The three main properties of a sensor required for measuring strain on an automobile tire rubber are sensitivity, precision, and range. SHM of automobile tire does not require high sensitivity and precision as we are mainly concerned with large strains. However, we require strain sensors with large strain range that can measure strain accurately on automobile tire surface. Some of the commonly used displacement sensors and their properties are listed in Table 5.3. These sensors have been described in detail in Chapter 2.

Sensor	**Sensitivity**	**Precision**	**Range**
Variable Resistor	High	Moderate	Large
Foil Strain Gage	Low	Moderate	Small
Parallel Plate Capacitor	Moderate	Moderate	Moderate
Sonic/Ultra-Sonic	High	High	Large
Optical	High	Moderate	Large

Table 5.3: Types of Sensors and their properties

5.4 Conclusion

In this chapter, we have discussed some of the initial strain results acquired through by using Crossbow® motes on a metal bar. Also, we have discussed the GUI developed to show strain results in a logical manner to the user. In addition, we have looked into a new sensor, Metal Rubber™, to measure strain accurately on a tire surface. However, we have faced few challenges with Metal Rubber™ such as hysteresis due to its material properties. Also, we have examined few other strain sensors which are capable of measuring large strains. Besides measuring strain of a tire, the strain information needs to be transmitted reliably in a wireless medium to the user. For that, we have developed a reliable and energy efficient WSN for this application which is discussed in the coming chapters.

Chapter 6
Routing in Wireless Sensor Networks

Wireless networks have seen a tremendous growth over the past few years. Some of the current devices featuring wireless networks are cell phones, global positioning system (GPS), and wireless internet access. The next advance in wireless technologies will be development of large sensor networks. Sensor networks are becoming economically feasible because of the low cost of manufacturing small sensors. These sensor networks will measure ambient conditions in the environment and transform these conditions into signals that can be analyzed to predict events such as level of oxygen in a building on fire.

WSNs are inherently different from current wireless networks such as the cellular network. One of the main differences is that WSNs lack an infrastructure which is present in current cellular network. In other words, a node in a WSN may not have base stations which are present in the cell phone network. Therefore, WSNs are an ad hoc network in which sensor nodes are usually scattered in a field. Sensor nodes need to coordinate among themselves to create a working network to send useful information either to other sensors or to a fixed or mobile sink which we refer as base station (BS). In most of WSNs applications, sensor nodes are constrained in energy supply and communication bandwidth. For this reason, extensive research has been conducted in every layer of networking protocol stack to create energy efficient WSNs. For example, at the networking layer, it is extremely important to find the most efficient route for data transfer that can maximize the lifetime of sensor nodes.

Routing in WSNs is inherently different from traditional mobile ad hoc networks (MANET) or cellular networks. First, WSNs do not have a global addressing scheme for large number of nodes like traditional wireless networks because the cost of ID maintenance is high. The main difference between WSNs and traditional wireless networks is the tight constraints on energy, processing, and storage capacities. Also, WSNs are application dependent as the design requirements change with the application. Due to these differences, many new routing

algorithms are being developed for WSNs. In this chapter, we delve into some of the common routing schemes in WSNs. In Chapter 7, we have developed Location Prediction of Mobile Nodes (LPMN) that builds on the routing schemes proposed in the literature and tailored it to unique needs of strain monitoring of an automobile tire. Also, we have analyzed the performance of the LPMN in terms of packet drop rate, energy usage, and end to end delay. In this chapter, we first discuss some of the challenges faced by all the routing protocols in WSNs in section 6.1. After careful review of these challenges, we discuss some of the common routing protocols in section 6.2.

6.1 Routing Challenges in WSNs

WSNs need to overcome numerous challenges such as limited power supply, limited communication bandwidth, and limited computing power. The design of routing protocols in WSNs is influenced by the constraints stated above. In this section, we will investigate some of these challenges to get a deeper understanding of routing protocols for WSNs.

The initial topology of the sensor network depends on the deployment of the nodes. Nodes can be deployed either in a fixed manner or scattered randomly on the field. If latter, then these nodes need to create an ad hoc network over multiple wireless hops to the base station. Nodes deployed in the field are equipped with limited power supply which requires an energy efficient routing protocol to run on these nodes.

Energy conserving forms of computation and communication is essential to prolong the lifetime of a WSN. In a multi-hop environment, each node may originate new data and may also forward data from other nodes. Therefore, power failure of few nodes can cause severe degradation in the performance of a WSN. Therefore, it is important to design a routing protocol that can conserve energy.

Data reporting in WSNs can be time driven, event driven, query driven or hybrid of all these methods. The time driven method is suitable for an application that requires data to be sent to BS on regular time intervals. The event or query driven methods work well for an application where only certain events need to be reported back to the base station. In our case, hybrid of these methods will work well as we need to monitor catastrophic events such as tire burst as well as regular data reporting to test the connectivity of the network.

In many WSNs applications, all nodes can be homogenous in terms of available power and communication bandwidth. However, in many cases, nodes can have different roles or capabilities depending on the applications.

Some sensor nodes may fail due to power failure or physical damage. The node failures should not affect the connectivity of the WSN. The routing protocol should be designed in a way to adjust for node failures by rerouting the data or adjusting the power of nodes. The routing scheme should be able to accommodate number of sensor nodes from hundreds to thousands; therefore, routing protocol should be scalable as well.

In our case, sensor nodes are assumed fixed on an automobile tire. However, sensor nodes are moving with respect to the base station (BS) due to the tire rotation. The BS is on the fixed part of the car. The routing protocol needs to sustain a working topology even in presence of mobile nodes. The traditional problems that arise during a wireless communication such as multi-path fading and presence of high bit error rates are also present in WSN. The routing protocol should be designed to keep reliable communication links in a challenging wireless medium. Also, the routing scheme needs to be highly connected meaning sensor nodes should have multiple routes to the sink (BS).

6.2 Routing Protocols in WSNs

In this section, we will present some of the developed routing protocols in WSN. It turns out most of the routing protocols are based on the network structure.

6.2.1 Protocols based on Network Structure

The network structure plays a crucial role in the operation of routing protocols in WSNs. We will cover some of the basic routing protocols that belong to this category.

6.2.1.1 Flat Network Routing

In flat networks, all the nodes play the same role and they collaborate together to perform sensing tasks. It is not practical to give a unique identifier to each node due to large number of nodes. For this reason, data is requested through queries from the sensor nodes. Another name for this type of routing is data centric routing [4]. Sensor Protocols for Information via Negotiation (SPIN) and Directed Diffusion are key examples of the flat network routing protocols.

6.2.1.1.1 Sensor Protocols for Information via Negotiation (SPIN)

The idea behind SPIN is to specify the data using high level descriptors. Before transmission, high level descriptors are exchanged among sensors via a data advertisement mechanism, which is the key feature of SPIN. Each node upon receiving new data, advertises it to its neighbors and interested neighbors, i.e. those who do not have the data, retrieve the data by sending a request message. SPIN's negotiation solves the classic problems of flooding such as redundant information passing, overlapping of sensing areas and resource blindness thus, achieving a lot of energy efficiency [4].

6.2.1.1.2 Directed Diffusion

Directed Diffusion is another data centric routing protocol where all the data generated by sensor nodes is named by attribute-value pairs. Sensors measure events and create gradients of information in their respective neighborhoods. BS queries the network by broadcasting interest. Gradients are formed along the route based on the interests. The goal is to find a good aggregation tree that gets the data from source nodes to the BS [4].

6.2.1.2 Hierarchical Network Routing

In hierarchical routing, nodes are classified among various levels based on the job given to them. For example, higher energy nodes can be used to process and send the information, while low energy nodes can be used to perform the sensing. This protocol uses cluster based sensing which improves scalability and energy usage.

6.2.1.3 Geographic Routing

In location based routing, sensor nodes are addressed by means of their locations. The distance between neighboring nodes can be estimated using the incoming signal strength. Energy is saved by putting inactive nodes to sleep. Scheduling sleep period for each node in a localized manner is an active research issue. Most of the geographic routing protocols use greedy algorithms to forward the packet to the destination.

- **Greedy Protocol**: A node picks the next hop neighbor with the least distance to the destination. In Figure 6.1, source has two nodes within its radio range and the source picks the node closer to the destination (shown by the arrow) using greedy routing

protocol. The term greedy comes from the fact that the protocol is greedy toward picking the path with the least distance to the destination.

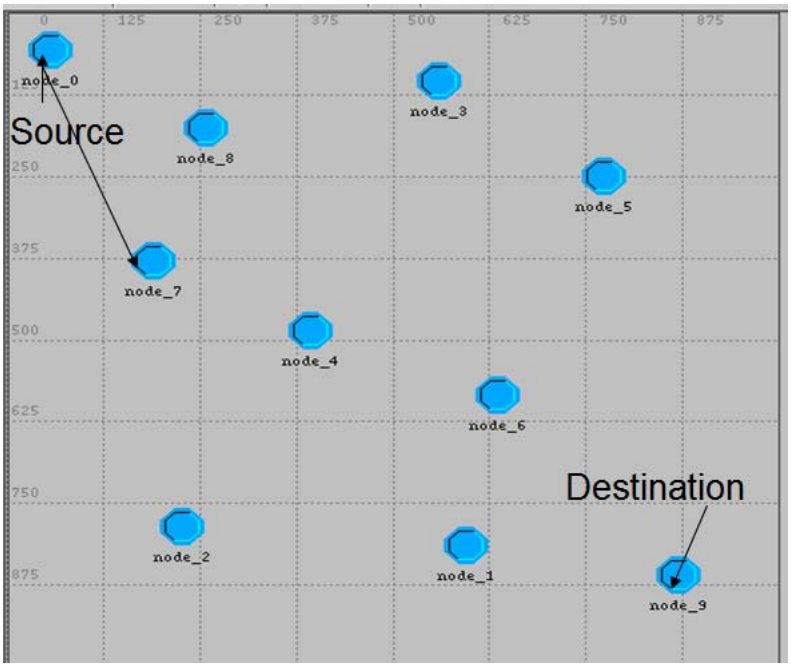

Figure 6.1: Network with Greedy Routing Protocol

Most of the location based routing protocols are based on the fundamental concept of greedy protocol. However, these greedy protocols differ in their approach to counter communication holes. The easiest way to navigate around the holes is to flood the network. Some of the early protocols used restrictive flooding to counter communication holes. Flooding is effective but an inefficient way of finding routes as it requires unnecessary computations and transmissions. Karp et al [17] avoided the holes using the technique of planar graphs which are computed out of the original network. The packet follows the perimeter of the planar graph to go around the hole. However, most of the traffic concentrates around the nodes on perimeter that tend to use up their energy faster than the other nodes. Also, all the nodes are always in listening mode which consumes the battery.

WSN nodes have limited storage and constrained energy resources. Therefore, it is vital that the routing in WSN to use minimum storage and conserve energy resources to prolong the lifetime of the network. Energy of a node can be saved by using geographical routing which sends data to a targeted area and hence decreases flooding in a network. The geographical information of a node can by obtained by using techniques such as strength of received signal.

There are three main advantages of using geographic routing in WSN which are as follows:

- WSN application can target a specific area using geographic routing.
- Data aggregation techniques can use geographic information to find redundant data from a specific location.
- Geographic location of a node can be used as an identifier instead of an IP address.

Wireless ad hoc networks have been using geographic routing to create a reliable network for more than a decade now. Location Aided Routing (LAR) is one of the geographic routing protocols used in ad hoc networks which uses a node's location to deliver packet in pre-defined area. LAR used flooding method whenever it could not find a path to the destination. Due to the success of geographic routing in wireless ad hoc networks, similar ideas are being implemented for WSNs. Greedy Perimeter Stateless Routing (GPSR) is one of the first geographic routing developed for WSNs [17]. GPSR forwards the packet toward the destination by using greedy method which chooses immediate neighbors based on the distance to the destination. If greedy forwarding fails to find a path to the destination, then GPSR creates a planar sub-graph and goes around the communication hole. This technique of perimeter routing puts a heavy burden on the nodes on the perimeter which makes the nodes on the perimeter fail earlier than other nodes. LAR and GPSR can be used for WSN, but they are not the most effective geographic routing protocols for WSN. LAR and GPSR do not consider energy in making routing decision which is a scarce resource in WSNs. Geographic and Energy-Aware Routing (GEAR) is one of the first geographic routing protocols to consider energy in making routing decisions [18]. GEAR uses both nodes' locations and nodes' residual energy to make routing decisions. However, in GEAR, the nodes with lower overhead bear much more load than other nodes which make them use-up their energy relatively quickly. Also, in GEAR, nodes are constantly awake thereby draining their energy. Therefore, another routing protocol named Geographical Adaptive Fidelity (GAF) proposed to put the idle nodes to sleep which resulted in energy savings [19]. However, GAF does not make any routing decisions and must be integrated with other routing algorithms. Most of the geographic routing algorithms perform routing by flooding or use perimeter routing when greedy routing fails. To date, only few geographic routing algorithms have been proposed that are energy efficient.

After carefully analyzing the present geographic routing protocols, we discovered the

following drawbacks:

1. Greedy forwarding can find a route to the destination if most of the links are reliable. However, greedy forwarding does not balance energy consumption even in ideal conditions.

2. Perimeter forwarding requires a sub planar graph which requires more overhead on the nodes for transmitting information about sub planar graph. Also, the nodes on the boundaries use their energy faster due to more packets being routed through them. Subsequently, there are more holes in the network as the nodes on the perimeter start dying thereby leading to a network failure.

3. Most of the geographic routing protocols do not differentiate energy consumption of a node while in different states namely idle, transmitting, and receiving.

Therefore, a better geographic routing algorithm needs to be envisioned which uses geographic locations to route the packet and uses energy efficiently among the nodes at the same time. The desired features of this geographic routing protocol are as follows:

- The new routing protocol must use geographic information to direct the packet toward the destination and in the process should avoid flooding.

- The new routing protocol must pick the node with the most residual energy to route the data which will prolong the life of the network.

- The new routing protocol must save energy by putting idle nodes to sleep as long as there is path available from the source to destination.

We will be discussing some of the developed geographic routing protocols which use energy efficiently in a WSN in the next few sections. However, we need to be familiar with some of the most common assumptions made by these protocols before we can discuss these protocols in detail. Therefore, we discuss some of the common assumptions made by most of the geographic routing protocols in the following section.

6.2.1.3.1 Assumptions made by the Geographic Routing Protocols

Geographic routing protocols assume that the WSN consists of large number of sensing nodes.

Also, sensing nodes send their data from different geographic locations to the sink or base station (BS). In addition to the assumptions stated above, there are few additional assumptions that the protocol considers are given in the following:

- Each node in the network knows its location and the locations of its neighbors. Any location based routing scheme requires a localization technique to determine either relative or absolute position of nodes. One of the most common techniques for localization is Global Positioning System (GPS) devices. However, a GPS device is not a feasible solution for our project due to energy consumption. There are few other localization techniques that can determine relative position of the nodes namely multilateration and signal strength ranging. Multilateration, also known as hyperbolic positioning, is the process of locating an object by accurately computing the time difference of arrival (TDOA) of a signal emitted from the object to three or more receivers. It also refers to the case of locating a receiver by measuring the TDOA of a signal transmitted from three or more synchronized transmitters. Incoming signal strength can also be used to determine the distance between the node and its neighbors.

- The destination or sink node is fixed, where rest of the nodes can be mobile.

- Each mobile node knows its speed and the direction it is moving. This information is also communicated to the neighboring nodes.

- Each node knows its residual energy and also can attain residual energy of its direct neighbors.

- The transmission radius of every node remains same which can be attained by transmitting signal at a specific power out of an Omni-directional antenna.

- Each packet has the geographic information which shows the user the location of sensing target.

- All the links are bidirectional which means that if a node can transmit data to a neighbor, then it can also receive data from that neighbor.

- Each node in the network has five basic components namely memory, controller, communication device, sensor, and power supply as shown in Figure 3.2.

Now, we discuss some of the most common WSN geographic routing protocols being used in more detail in the following sections.

6.2.1.3.2 Geographic Adaptive Fidelity (GAF)

GAF is an energy aware location based routing protocol in which network area is divided in fixed zones that form a virtual grid. In each zone, node elects one node to stay awake for a certain period of time, and then the rest can go to sleep. This one node is responsible for reporting data to the BS on behalf of rest of the nodes. Therefore, GAF can conserve energy without changing routing topology among virtual grids [19]. The sensing virtual grids are described in detail below:

6.2.1.3.2.1 Sensing Grid States

GAF forms equal area grids between source and destination. Also, the nodes should be able to communicate with the nodes in adjacent grids. One example of a grid construction with three virtual grids is shown in the Figure 6.2.

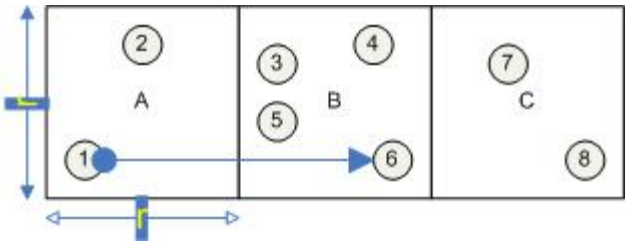

Figure 6.2: Virtual Grids in a Network

In the virtual grids shown in Figure 6.2, node 1 can communicate with nodes 3, 4, 5, and 6 because they are in the adjacent gird. However, node 1 can not communicate with node 7 and 8. Therefore, node 1 can pick any of the nodes for packet forwarding from node 3 to 6. In this example, node 1 picks node 6 for forwarding. Therefore, nodes 3 to 5 can go to sleep and conserve energy.

The dimensions of the grid will be determined by the transmission radius of the nodes, *R*. Each grid is a square with *r* units long. In order to guarantee that any node in grid A should be able to communicate with any node in grid B, the distance between two farthest node in adjacent grids must not be larger than *R*. In other words, the length of the diagonal connecting two adjacent nodes must be smaller than R. By using Pythagoras theorem, we can find the following

relationship for grid dimension:

$$r^2 + (2r)^2 \le R^2,$$
$$r \le \frac{R}{\sqrt{5}} \tag{6.1}$$

Besides sensing grids, GAF also have unique states for the nodes in the network which are described below.

6.2.1.3.2.2 Nodes' States

A node in a network consumes energy in three states namely idle, transmitting, and receiving. A node spends more energy during transmitting or receiving, but energy consumed in idle state cannot be ignored. The idle: receive: transmit ratio is measured differently in various papers such as 1:1.05:1.4, 1:2:2.5, and 1:1.2:1.7 [20].

It is shown in GAF that a node can be in three states namely sleeping, active, and discovery. The state transition diagram of a node is shown in Figure 6.3.

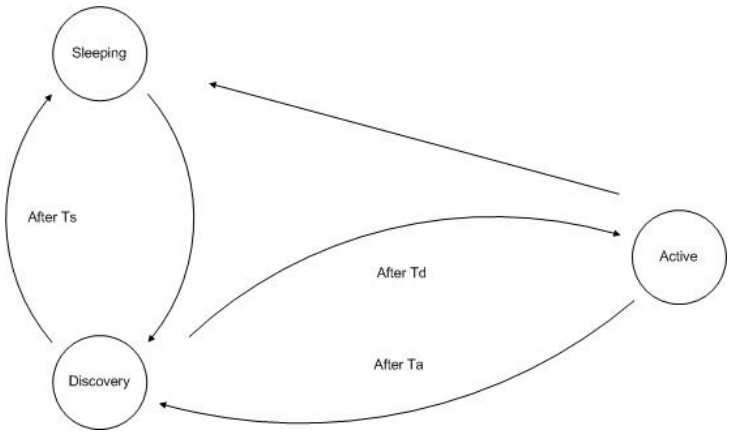

Figure 6.3: State Transition of a Node

It is shown in the state transition diagram that a node goes into discovery state from active after *Ta*. Once the node is in discovery state, then the states of the neighboring nodes in the grid determines the next state of the node in the discovery state which are sleeping or active. This decision is based on many factors such as geographic location, residual energy, and the time a node has stayed active. The node in the discovery state needs to determine which of its neighboring nodes will have more residual energy or have been active shorter than the node in discovery state, then one of the neighboring nodes is picked to be active and the node in

discovery node goes to sleep. The goal of the state transition among nodes is to keep nodes away from depleting all of their energy.

One of the main concerns with GAF is the large dependency on the node densities to keep the connected network. Also, the data rates can slow down because of the congestion on the one active node in the virtual grid. GAF also assumes that all the nodes in the network are equally important which may not be the case in some scenarios. Also, the energy savings of GAF start diminishing as the nodes become more mobile. Additionally, GAF relies on the ad hoc routing protocol it is running on to quickly reroute the traffic if the active node transmitting packets is put to sleep. GAF governs the energy savings through sleep cycles and depends on the routing protocol to route the packet.

6.2.1.3.3 Most Forward within Radius (MFR), Distance Routing (DIR) and Geographic Distance Routing (GEDIR)

MFR selects the best neighbor A that will minimize the dot product $\overline{DA}.\overline{DS}$, where, S, D are the source and destination nodes respectively. $\overline{DS}$ is the Euclidean distance between the nodes D and S. MFR is a loop free routing protocol. DIR selects the best neighbor based on the minimum angular distance from the imaginary line joining the current node and the destination. GEDIR is a greedy algorithm in which a node selects the best neighbor of the current vertex whose distance to the destination is minimized. These protocols can become loop free by memorizing past traffic or enforcing time stamping [21]. A study performed in [21] shows that in 99 percent of the cases these greedy protocols create the same path.

6.2.1.3.4 Span

Span selects coordinators among the nodes based on their positions. “Coordinators” is a node that can receive and forward packets to other coordinator nodes. Network coordinators forward message to other coordinators to reach the destination while non-coordinators sleep. All the nodes periodically wake up and check if they should become coordinators. Span ensures enough coordinators so every node has a coordinator in its radio range. Span rotates the coordinators among all the nodes to ensure that all the nodes share the task equally. Span attempts to increase lifetime of the network by minimizing the number of coordinators while making sure that enough coordinators are present to keep a connected path between the source and the destination [22].

6.2.1.3.5 The Greedy Other Adaptive Face Routing (GOAFR)

GOAFR is a combination of a greedy and face routing protocols. The greedy algorithm always picks the next neighbor which is closest node to the destination. However, if the closest node to the destination is the current node, then adaptive face routing can utilize the face structure of planar graphs such that the message is routed from node *s* to node *t* by traversing a series of face boundaries. The goal of face routing is to find the best node on the boundary by using geometric planes. GOAFR can achieve both worst case optimality and average case efficient by using combination of greedy and face routing protocols [23].

6.2.1.3.6 Geographic and Energy Aware Routing (GEAR)

GEAR proposed by Yu et al. uses a combination of geographic and energy aware informed neighbor selection process to route the data [18]. This protocol saves energy and increase throughput by only considering a certain region rather than sending the packet to the whole network. Each node keeps an estimated cost of reaching the destination which is a combination of residual energy of the node and the distance to the destination. Upon receiving a packet, node find the next destination out of its neighbors depending on the lowest estimated cost to the destination. If more than one node exists with similar estimated cost, then GEAR turns into a greedy geographic routing protocol which picks the neighboring node closest to the destination. One of the main concerns with GEAR is that each node must keep a table of its neighbor's energy level and location. This process requires some kind of message transmission all the time which will use the energy of the nodes. Also, nodes have limited memory available to them which makes it impossible to have long tables for all the nodes. Also, GEAR assumes that the link is bidirectional among all the neighbors which is not a guarantee with wireless environment. The nodes need to compute the estimated cost of transmission to neighbors which requires energy consumption.

6.2.1.3.7 Energy Efficient Geographic Routing (EEGR)

GAF proposed that if the whole network can be split into equal grids, then redundant idle nodes in the grids can be put to sleep to conserve energy. There have been other geographic routing protocols suggested which forms the grids like GAF such as Energy Efficient Geographic Routing (EEGR) algorithm. EEGR divides the area between the source and destination in equal grids instead of the whole network [24]. EEGR also tries to balance energy

consumption among the nodes by using residual energy of a node as metric in making routing decisions. EEGR works really well for static and dense network, however it performs poorly for mobile and sparse networks. Most of applications of WSN are not static such as a soldier carrying an image sensor can be considered a mobile node in the network. Therefore, it is vital to design a geographic routing protocol which performs well in terms of reliability and energy efficiency for mobile WSNs.

In EEGR, the cost function will be a combination of distance between the node and the destination and the residual energy at the node [24]. The cost function is computed by the following equation:

$$C(N_i, D) = d(N_i, D) + \frac{\beta}{H}\left(E_{ni} - e(N_i)\right) \tag{6.2}$$

Where, $C(N_i,D)$ is the cost of neighboring node N_i picked as next hop, $d(N_i,D)$ is the distance between the node N_i and the destination area (D) normalized by such distance among all neighbors N, β is the tunable weight for the contribution of residual energy in the cost function, H is the number of hops traversed by the packet, E_{ni} is the maximum energy of a node, and $e(N_i)$ is the consumed energy of a node N_i normalized by the consumed energy of all the neighbors N. All the information required in computing the cost such as distance, consumed energy, and the number of hops traversed can be known among neighbors by a simple hello message. This process will continue until the packet reaches the destination while balancing the energy among the nodes. The motivation behind this routing protocol is shown in the following points:

- When all the nodes have equal energy, then the node which is closest to the destination is picked as the next hop because the part of the cost function with residual energy will become zero.

- When all the nodes are equidistant to the destination, then the node with maximum residual energy is picked as the next hop because the part of the cost function with distance to the destination will become zero.

Also, as the number of hops traversed by the packet becomes bigger, then this algorithm turns into greedy forwarding by picking the node closest to the destination.

6.2.1.3.8 Location Fault Tolerant Geographic Routing Scheme for Wireless Ad Hoc Networks

Location fault tolerant geographic routing scheme combines the position based clustering with geographic routing to improve the performance of WSNs that has location inaccuracies due to mobility [25]. This scheme forms clusters of nodes and the shape and size of the clusters depends on the nodes' radio range. This scheme assigned a cluster head to the least mobile node. Therefore, every routing decision is made through the cluster head which keeps track of nodes entering or leaving the nodes. This scheme shows improved results over other geographic routing schemes for a mobile network with location inaccuracies.

6.2.1.3.9 Yet Another Greedy Routing (YAGR)

YAGR extends ideas from potential based routing. Potential based routing avoids the problem of communication hole in geographic routing by creating a potential field across the network. In other words, a node with a packet to forward knows the potential of its neighbors. And, the packet is forwarded to neighbor with highest potential. The potential of a node is mainly a function of the node' distance to the destination. The readers are encouraged to look into [26] for more details.

6.3 Conclusion

In this chapter, we have discussed the common routing protocols developed in WSNs. We presented the advantages as well as pitfalls of all the routing protocols. Some of the routing protocols are more energy aware than others which makes them unique to other protocols. Also, few of the routing protocols are more reliable than others in terms of packet delivery. However, all the routing protocols discussed in this chapter demonstrate unreliable packet delivery results for a mobile and sparse network. Therefore, we have developed a method called Location Prediction for Mobile Nodes (LPMN) which improves the results of routing protocols for mobile and sparse networks. LPMN is discussed in detail in the next chapter.

Chapter 7
Routing Protocol for Mobile Networks: Location Prediction for Mobile Nodes (LPMN)

Wireless sensor networks (WSNs) envision to sense and monitor our physical world by a dense network of wireless transceiver equipped with various sensors. There has been a lot of research being done in WSNs to create a sensing world around us [27]. Most of the research being done in this field is geared toward static and fixed networks. But, WSNs are considered mobile in many scenarios. For example, in military applications, sensors equipped on soldiers running in a battle field create a mobile WSN. We have encountered a sparse WSN in our work developing a sensing system for structural health monitoring (SHM) of an automobile or aircraft tires where small number of sensors equipped with wireless transceivers embedded in a tire creates a sparse WSN. Therefore, we tried to find a routing protocol which can deal with mobile and sparse WSNs. Unfortunately, we have found that most of the routing protocols are designed for flat and static networks and they are inadequate for mobile and sparse WSNs. Mobile and sparse WSNs are much more difficult to deal with than static and dense WSNs. In mobile and sparse WSNs, wireless transmission suffers from multipath channel fading, shadowing, limited routing paths, node failures, and interference much more than in static WSNs. Therefore, routing path between the source and destination is more likely to fail in a mobile WSN. It is vital to have a reliable backup path when one of the routes fails to avoid packet loss and large end to end delay. Various routing protocols have been proposed for mobile ad-hoc network (MANET). But they do not perform well for WSNs due to the tight energy constraints on sensor nodes. Therefore, reliable packet delivery, energy efficiency and small end to end delay are highly preferred features for routing protocols in mobile and sparse WSNs. We present mobility aware routing protocol called the Location Prediction for Mobile Nodes (LPMN) which provides reliable packet delivery against route failures in SHM of an automobile tire as well as other mobile applications of WSNs. Each node with LPMN predicts its new location based on its past movements and communicates this information among its neighbors. LPMN works on top of the

existing geographic routing protocols. Light overhead is incurred during LPMN because mobility information can be added to control packet which is already a part of most geographic routing protocols. The best next hop node is self-elected on a per-packet basis after considering mobility of all the neighbors which makes LPMN highly adaptable to change in nodes' mobility. Through cooperation among neighboring nodes, the packet delivery ratio and end to end delay is improved since links which are least likely to fail are chosen to relay.

7.1 Related Work

There has been some work done on the physical layer to deal with mobile WSNs such as mitigating multipath fading and efficient forward error correction. Also, there have been some approaches which deal with mobility by improving the MAC layer of WSN such as overcoming the effect of Doppler shift by using adaptive frame size in the MAC layer [28]. These methods in physical and MAC layer improve the wireless transmission, but they do not address the scenario when a wireless node simply moves out of radio range during the transmission. It is necessary to work with the routing layer in mobile WSNs to deal with the route failure because of node mobility. There have been some approaches which recommend inserting a multi-radio mobile agent layer between the sink layer and sensor layer such as Multi-Radio Enabled Mobile Wireless Sensor Network (MEMOSEN) to deal with mobile sensors [29]. But this approach does not deal with mobile sensors directly and recommends adding more wireless radio between the sensors and the sink layer which is not cost effective. There have been many protocols suggested in mobile WSNs and MANETs which recommends having many backup paths if the main path fails [30], [31]. Most of these protocols require a network wide re-routing which is harder to implement in a mobile network. Our approach does not require a network wide communication and the next relay node is picked on per packet basis based on the neighbors mobility. One of the main features of our approach is that LPMN runs on top of existing geographic routing protocols and LPMN adapts well to the mobility of the WSNs.

After having looked at the need to deal with mobility in the routing algorithms, let us now take a look at how mobility can be characterized and how the various mobility patterns have evolved out of this characterization.

7.2 Mobility Patterns

Considering the importance of node mobility in designing an effective routing algorithm for

mobile ad hoc networks, we have focused on designing LPMN to improve the performance of mobile networks. It has been found that there exists a correlation between future and past movement speeds and direction of sensor nodes. For example, consider the subway trains for a major metropolitan city. The route of each train is not completely random; else the whole purpose of the subway would be defeated. Therefore, it is possible to predict the time of a train arrival at a given station by looking at the speed, the route of the train, and the number of stops. On the other hand, we can also have a situation where the nodes move in a completely random fashion such as the children running around on a playground. Hence to cover all the scenarios of mobility patterns from completely random to highly predictable random motions, we categorize mobility into three patterns as deterministic, semi-deterministic, and random**.**

7.3 Deterministic Mobility Model

Deterministic model is the most simple mobility model. For example, if the nodes are moving in a straight line, then every next move will be exactly similar to the last move. A sample scenario will be cars moving in a straight line with the same speed.

7.4 Semi-Deterministic Mobility Model

After looking at a deterministic model, let us look at a model in which movement is somewhat predictable. For example, the ants moving in a general direction can be considered a semi-deterministic model as shown in Figure 7.1.

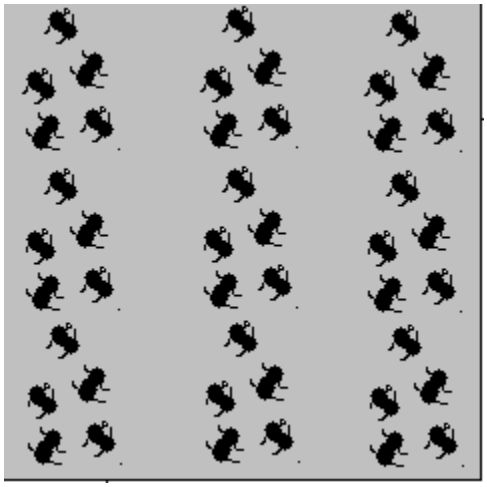

Figure 7.1: Movement of Ants (Semi-Deterministic Mobility Model)

In this case, the movement is not completely deterministic, but the ants are moving in one

general direction with subtle direction changes which create a semi-deterministic mobility model. In other words, we can predict the movement of the node with semi-deterministic mobility model by observing the last few movement vectors.

7.5 Random Mobility Model

We will now consider the random mobility mode as shown in Figure 7.2. The movement in this model is very hard to predict as every move is completely independent of the last move. The accuracy of prediction in this mobility model completely depends on the degree of randomness.

Figure 7.2: Random Motion

7.6 Location Prediction

After having seen how the classification of various mobility patterns, we now take a look at how LPMN can use this mobility information to enhance the efficiency of routing algorithms. For example, in Figure 7.3 (a), node 1 wants to send data to node 6, but node 6 is out of transmission range of node 1 shown by the circle. Therefore, node 1 needs to pick one of the neighbors within the transmission range as a next hop node which can send data to node 6. If node 1 does not have any information about its neighboring nodes future position, then node 1 will pick node 4 as the next hop as shown in Figure 7.3 (b). Unfortunately, node 4 moves out of the transmission range of node 1 and this packet is likely to be dropped. However, if all the nodes are running LPMN and node 1 can predict its' neighbors movement, then node 2 will be picked as the next hop router as shown in Figure 7.3 (c) and the packet is forwarded toward the destination.

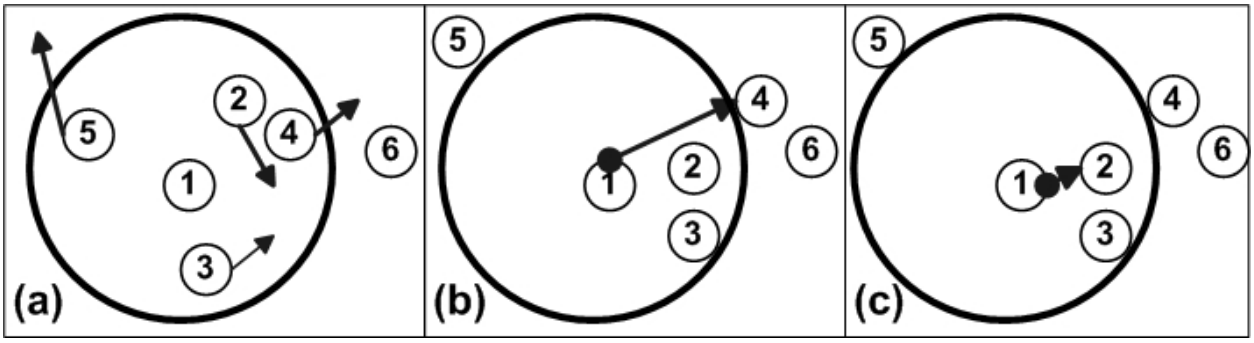

Figure 7.3: (a) Snapshot of Network (b) Snapshot of the Network without LPMN (c) Snapshot of the Network with LPMN

7.7 Predicting Location of Mobile Nodes using Moving Average

Until now, we have discussed how mobility affects the efficiency of the routing algorithms in WSNs and characterized mobility into different patterns. We also discussed how this information could further be used in designing efficient routing algorithms with location prediction of mobile nodes. And in doing so, we assumed that the routing algorithm knows about the underlying mobility pattern being followed by the nodes, which is not the case as the existing geographic routing algorithm does not have any idea about the underlying mobility pattern followed by the nodes. Thus we have to figure a way by which the routing algorithm can learn about the underlying mobility pattern and then use this information accordingly. In this section, we explain as to how the routing algorithm learns about mobility. This extra information about mobility, that is, the mobility pattern being followed by the nodes in the routing algorithm we assumed to be using, does not come free. The cost associated with it is the extra communication among the nodes about the mobility pattern. To explain this, consider Figure 7.4, where the nodes in the WSNs are placed as shown in the first snapshot at time = t_1, and after certain time and a number of moves, assume the nodes are placed as shown in the second snapshot at time = t_k, where *k* is some integer value. Each node here is associated with a tuple, that is, (x(t), y(t)), which is the location co-ordinate of the node at time t.

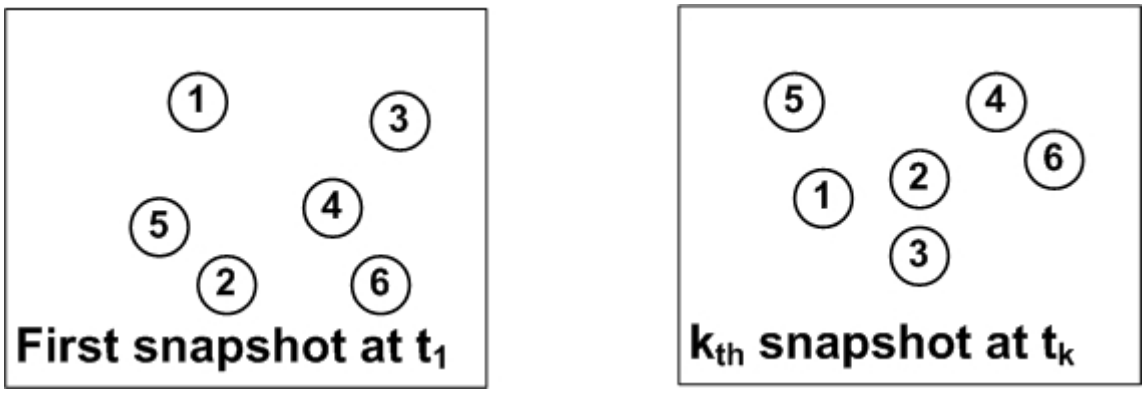

Figure 7.4: A sample WSN depicting the different snapshots of the location of the nodes taken time instant t_1 and t_k

Now consider node 1, assuming that it was at position $(x(t_1), y(t_1))$ at instant t_1 and then it moved through $(x(t_2), y(t_2))$ at t_2, $(x(t_3), y(t_3))$ at t_3 before arriving at $(x(t_k), y(t_k))$ at t_k. And if this information about the past locations of node 1 can be saved at node 1, then node 1 would have a history about its past locations. This method will work fine for smaller values of *k*, but it will not be efficient to keep track of past locations for large values of *k*. But, if we can keep a moving average of last *k* values of node 1 location, then we do not have to save a large table for node 1 past locations. Therefore, we keep a moving average of node past movement in LPMN. Next, we explain how we can calculate moving average of nodes' past locations.

We can predict the future position of a node based on its past movement. If the moving average of past movement of a node at discrete steps is known, then a prediction of next location of the node can be made. If this information can be communicated among neighboring nodes, then a better decision can be made by picking a node with least distance to the destination. Also, the nodes' location information will contain the information about speed and direction. In other words, the nodes' location is a function of nodes' speed and direction. The exponential moving average of a node movement can be calculated by the following equations:

$$\begin{aligned} \Delta x_{avg}(n) &= \Delta x_{avg}(n-1) + \alpha\left(\Delta x(n) - \Delta x_{avg}(n-1)\right) \\ \Delta y_{avg}(n) &= \Delta y_{avg}(n-1) + \alpha\left(\Delta y(n) - \Delta y_{avg}(n-1)\right) \end{aligned} \tag{7.1}$$

where, $\Delta x(n)$ is a movement node made in its x-coordinate at the *nth* discrete step, $\Delta x(n)_{avg}$ is the average movement in the x-coordinate at the *nth* discrete step, and α is the weighting factor which can be anything between 0 and 1. We have set α to 0.5 which means about 50% importance has been put on the most recent movement by the node. A similar equation is used

for the y-coordinate. After computing the moving average of nodes' past coordinates, a prediction about the coordinates of a node in the next discrete step can be found by the following equations:

$$x(n+1) = \Delta x_{avg}(n) + x(n)$$
$$y(n+1) = \Delta y_{avg}(n) + y(n) \tag{7.2}$$

This procedure is described in a diagram in Figure 7.5. The black diamonds in the figure are the *x* and *y* coordinates of a mobile node. The gray square shown in the figure is calculated based on the past movement of the node (black diamonds). In Figure 7.5, the node starts out with coordinates (0.0,0.0) and moves to a new position (0.5,1.0) few seconds later. We can observe the node movement to last known value of (-2.0, 1.0) in Figure 7.5. Based on the movement of the node from (0.0,0.0) to (-2.0,1.0), we can make a prediction that node may be at a new coordinates (-2.4,0.3) in the next time instant as shown in the figure. If this prediction can be communicated with nodes' neighbors, then a better decision will be made in choosing the next hop node.

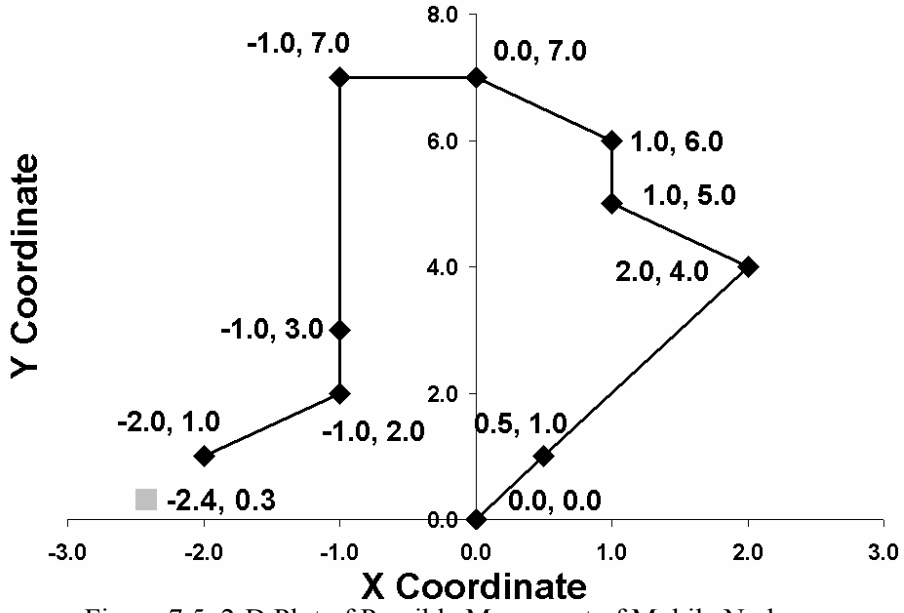

Figure 7.5: 2-D Plot of Possible Movement of Mobile Node

7.7.1 Algorithmic Description of LPMN

As mentioned earlier, the knowledge of location is important for successful routing of packets in a WSN. And in order to make geographic routing protocols location aware, we need some means by which the nodes can determine their present physical location. Thus in making geographic routing protocols location aware we assume that each node is equipped with a GPS,

which constantly updates the mobile host with its latest location information, with a fair degree of accuracy.

In the geographic routing protocols with LPMN, we have an extra piece of information that is the location information, which needs to be handled. Thus, in addition to the regular routing information which each node maintains locally, it also maintains a table with future position of its neighbors. Every node needs to compute the moving average of its past movement to predict its future movement. We present here pseudo code of LPMN that every node in a network runs on top of the geographic routing protocol. For example, every node is continuously updating its movement in both coordinates by running the following code:

```
  If (Self Interrupt is caused by the function for updating average position
of a node)
  {
    Deviation in x = New value of x – Old value of x
    Deviation in y = New value of y – Old value of y

    Old value of x = New value of x
    Old value of y = New value of y

    Moving average of x = Moving average of x + Soothing Factor*(Deviation in
    x - Moving average of x)
    Moving average of y = Moving average of y + Soothing Factor*(Deviation in
    y - Moving average of y)

    Next self interrupt for updating average position will be caused in 0.1
    seconds. We have picked the update interval to be 0.1 seconds; this
    parameter can be changed according to the mobility of the network. For
    example, the update interval can be lower to 1 seconds for a low mobility
    network.
  }
```

A node has updated information of its past movement by continuously updating its moving average of the change in both coordinates. A node in a network shares this information with all of its neighbors which helps them in making better decisions to forward packets.

7.8 Packet Forwarding

In LPMN, when an active node has a packet to forward, it queries all of its neighbors for information such as if the node's geographic location, and residual energy of the node. Then, it forwards the packet to the neighboring node which is closer to the destination, and requires least amount of energy. For example, in Figure 7.6, node 1 has a packet which needs to be sent to node 4. Thus, node 1 sends a hello packet to all of its neighbors informing them that it has a packet for node 4. We can see that node 4 is out of transmission range, but node 2 and 3 hears node 1 transmission. At this point, node 2 and node 3 sends their possible future location calculated through moving average back to node 1. By sending this possible future location,

node 1 has the ability to calculate the direction both of its neighbors are moving and their distance to the destination as shown in Figure 7.6. At this point, node 1 starts creating a table of possible next hop nodes with their location information as shown in Figure 7.6. The first check node 1 runs if the neighboring node will remain within its transmission range by looking at possible future locations of its neighbors. Node 1 can see that node 3 is moving out of its transmission range even though node 3 will be closer to the destination. Therefore, node 1 picks node 2 as the next hop node in this scenario. The same process will be run by node 2 to forward this packet to next node until it reaches the destination. We can see that calculating moving average of nodes' movement enables us to observe the mobility pattern of the nodes which can help in making better routing decisions.

Node #	Future Movement	Distance to the Destination	With in Transmission Range
2	(x, y)	A	Yes
3	(x, y)	B	No

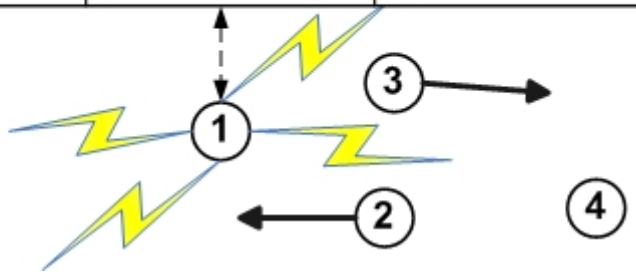

Figure 7.6: A Sample WSN showing the impact of LPMN on Routing Decisions

In short, node 1 can now use the location information available locally to it from the intermediate nodes, and node 1 can use this information to predict possible future locations of these nodes and then find out the best possible options for the route to node 4. The pseudo code for packet forwarding is as follows:

```
If (a node has a packet to forward)
  {
    Send a hello packet using its transmitter
    While(Time elapsed since the transmission of hello packet <= Total
    Transmission Time of the packet + Total Propagation Delay)
    {
      Continuously listen to receive transmission from the neighbors
    }
    Node picks one neighboring node out of all the neighbors using their
    average movement, geographic position, and residual energy. The cost
    depends on the geographic routing protocol such Greedy or GEAR.
}
```

LPMN runs on top of other geographic routing protocols such as GEAR and EEGR. We will discuss how these geographic routing protocols are modified due to LPMN.

7.9 LPMN Interactions with Geographic Routing Protocols

In principle, LPMN will run over any geographic routing protocol because it only adds mobility information in the communication between direct neighbors which is already a part of the geographic routing protocols. In later sections, we evaluate LPMN combined with Greedy Routing, GEAR and EEGR against these protocols without LPMN. We will discuss few modifications made to these routing protocols to accommodate LPMN in the next few sections.

7.9.1 LPMN Modifications to Greedy Routing and GEAR

The greedy routing and GEAR work almost similar as described in Chapter 6 beside few modifications. These modifications make these protocols location aware to improve their performance in a mobile network. With LPMN, greedy routing and GEAR use the location of the nodes computed by LPMN instead of actual geographical location of the nodes. We will show in Chapter 9 that LPMN improves the performance of greedy routing and GEAR in terms of packet delivery ratio for a mobile network.

In the modified Greedy and GEAR protocols, we have an extra piece of information that is the location information, which needs to be handled. Thus, in addition to the regular routing updates which each node communicates locally, it also communicates a new location information as shown in Figure 7.7, where (x, y) denotes the location co-ordinates of neighboring nodes of node 1. This table is maintained at each node for every other neighboring node in the network. And this table is populated progressively, as the nodes receive routing updates from neighboring nodes and, accordingly, the newer information is stored while the older information gets flushed out.

Node #	Future Movement	Distance to the Destination	With in Transmission Range
2	(x, y)	A	Yes
3	(x, y)	B	No
4	(x, y)	C	Yes
5	(x, y)	D	No

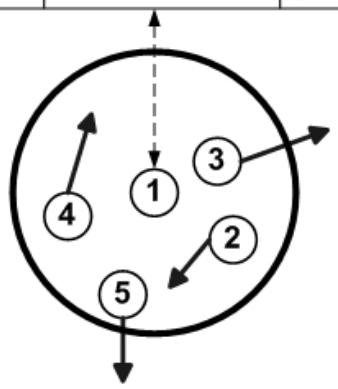

Figure 7.7: Table Maintained at Node 1 for all the Neighboring Nodes

In order to obtain location information of other nodes in the network, the location information needs to be shared among the nodes in the network. One way to achieve this is by piggy backing the location information to the regular routing updates.

7.9.2 LPMN Modifications to EEGR

Unlike Greedy and GEAR which simply pick the next hop node with the least cost, EEGR divides the area between the source and destination in equal grids and then forward the packet toward the destination by picking nodes from the grids. In other words, EEGR is more complicated than Greedy and EEGR because of the formation of virtual grids between the source and the destination. For this reason, we explain the modifications done to EEGR to make it location aware in a separate section. The overall idea of LPMN is still the similar to the one being used with Greedy and GEAR which is a node running LPMN finds the neighboring nodes that are likely to stay in the grid by using geographic position provided by LPMN, and then the node will pick the next hop with the least cost among the selected neighbors. For example, the expected movement vectors of nodes are shown in Figure 7.8. LPMN provides the expected movement vectors of the neighboring nodes to the node with a packet to forward. For example, node 1 in Figure 7.8 can use the movement information of the nodes in grid B to pick the node which is more likely to stay in the grid. As explained earlier in Figure 7.7, each node will maintain a table with location information of its neighbors.

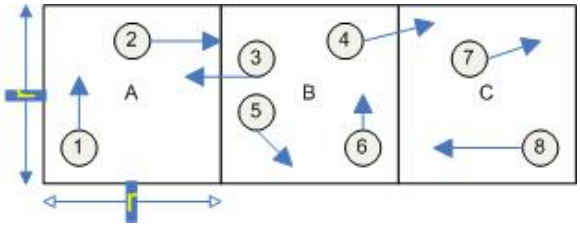

Figure 7.8: Nodes in Sensing Grids with their Mobility Vectors

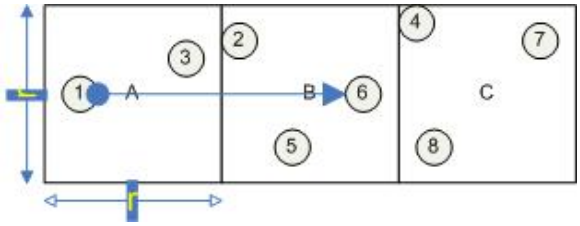

Figure 7.9: Node 6 is picked as the next router

Node 1 in Figure 7.8 can determine that node 5 and 6 are likely to stay in their grid by using the mobility information. Then, node 1 can pick between 5 and 6 based on the cost of each node which is defined in detail in section 6.2.1.3.7. It is shown in Figure 7.9 that node 6 will be picked as the next hop node if the cost is only the function of distance to the destination.

7.10 Tuning LPMN

LPMN leaves choice of parameter α, weighting factor, to applications. In this section, we discuss the effects and use of this parameter. Applications may wish to optimize the network performance by tuning α.

Weighting factor, α, can be anything between 0 and 1. If α is picked to be zero, then the most recent movement made by the node does not have any effect on the prediction of the node location. On the other hand, the prediction of the node location can be highly dependent on the most recent movement if α is picked to be 1. The value for α is highly dependent on the mobility of a particular application. For example, if we are dealing with a highly mobile network, then only the very recent location of the node can be used to predict location of the node or a high value of α is appropriate for this case. It is encouraged to tune α to get the maximum performance out of the mobile WSN. We have set α to 0.5 for the simulation done in our work. Also, the beauty of running a moving average of the coordinates of moving node is that the direction and speed becomes part of the algorithm. In other words, position of the node is a function of the speed and the direction of the node movement. Therefore, we can account for

speed and direction by observing node position.

7.11 Frequency of Routing Updates

LPMN, in its approach towards being proactive, heavily relies on periodic updates, to gain information about the topology changes occurring in its neighborhood. This leads to higher frequency of periodic updates, which in turn increases the routing overhead of the network, as periodic updates are mostly a full dump of the routing table. In the LPMN, however, we suggest suppressing the periodic updates for a longer period of time and encourage triggered updates, which would be much less compared to the number of periodic updates sent by all the nodes in the network. In the LPMN a triggered update can occur when a node has a packet to forward.

7.12 Adapting to High Mobility

LPMN helps the node with a packet to forward by picking the next hop node which is most likely to stay within the transmission range. The worst scenario would be the next hop node moves out of transmission range and the packet is dropped as shown in section 7.7.1. In scenarios with high mobility, this problem can greatly increase packet drop rate.

We can accommodate for high mobility by making each node estimate its future location and communicate this information with its neighbors. This system works well for the mobility model we use (semi-deterministic mobility model); it will be more difficult to estimate the node future location where movement is less predictable (random mobility model).

In order to compare the effect of the mobility on the network, we use simulation to compare the performance of mobile network with and without LPMN. We will see that the network behaves similar with and without LPMN when the mobility is low. However, the network without LPMN has higher packet drop rate for high mobile networks.

7.13 Conclusion

This chapter has discussed LPMN which runs on top of the existing geographic routing protocols and helps improve their results for mobile and sparse networks. This chapter has outlined the algorithmic description of LPMN by using pseudo code and diagrams. Next, in order to test the effectiveness of LPMN on geographic routing protocols, we designed a simulation model which is discussed in the next chapter.

Chapter 8
Design Model and Routing Descriptions

This chapter explains the simulation environment as well as the methodology used to carry out the simulations. We outline the simulation methodology used to create the geographic routing protocols with LPMN built on top of them. As we have mentioned earlier, LPMN will run over any geographic routing protocol because it only adds mobility information in the communication between direct neighbors which is already a part of the geographic routing protocols. Therefore, we will design three geographic routing protocols namely Greedy, GEAR, and EEGR in the simulation. Then, we will make these protocols mobility aware by adding LPMN on top of them. These protocols will be tested on a flat grid network where sensor nodes will originate data destined for other sensor nodes in the network. Therefore, we need to design a sensor node in the simulation which can originate data and route data to other nodes. These sensor nodes will be moving with semi-deterministic mobility which is described earlier. The semi-deterministic mobility needs to be designed in a way that a sensor node can move with subtle changes in directions as long it is advancing in one general direction. We have picked semi-deterministic mobility model in our simulations because we will able to predict nodes' movement which is impossible with a completely random mobility pattern. This network will be designed with a help of simulation software which can design WSN and test their effectiveness which is discussed in the next section.

8.1 OPNET® Modeler

We have chosen the OPNET® Modeler [32] as the simulator for this research because it realistically models arbitrary node mobility as well as physical radio propagation effects such as signal strength, interference, capture effect, and wireless propagation delay. OPNET® Modeler is the industry's leading simulation software for network modeling and simulation. It is an event driven simulator based on a series of hierarchical editors that directly parallel the structure of

real networks, equipment, and protocols. Also, OPNET® Modeler is built as layered software with three main layers namely network model, node model, and process model as shown in Figure 8.1. The layered structure helps in building each part of the simulation separately such as the network layer, routing layer, or the data link layer. In our work, the OPNET® Modeler is used for creating a test network for the simulations. We have implemented geographic routing protocols using OPNET® Modeler.

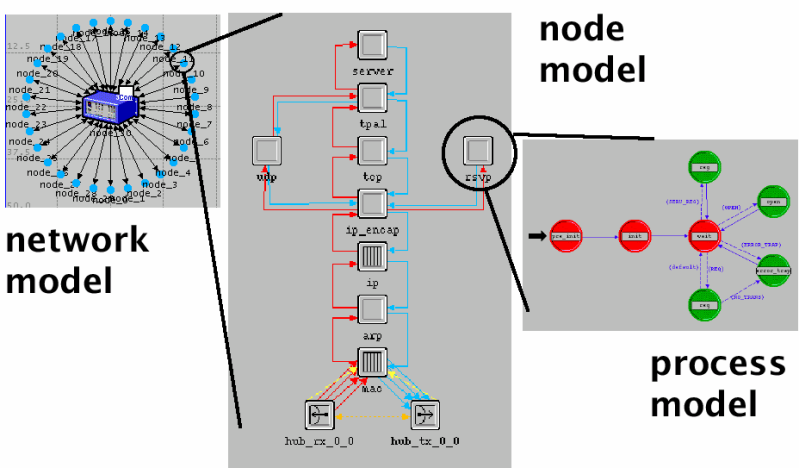

Figure 8.1: OPNET® Modeler Layout

8.2 Simulation Methodology

Having previewed the simulation environment, we will put forward our simulation methodology now. In an effort to stick to our goals while at the same time, conduct comparisons between the static and mobile networks, we chose our network size to be fixed through out the simulations. We also keep other variable parameters such as the amount of data traffic, size of the data packets and the link capacity constant through out the simulations. Thus the different test scenarios arise by varying the number of nodes in the network and the underlying mobility pattern along with the nodes to see how the standard geographic routing protocols and the mobility pattern aware geographic routing protocols perform in various scenarios.

In the rest of the section we present the topography, mobility model, traffic model, protocol implementation and the metrics used for performance evaluation.

8.3 Simulation Topography

The network simulations carried out for our study are based on flat grid topography. We have

two different topographies to two different scenarios. One of the topography is 0.1524 m x 1.117 m which represents the topography for an automobile tire. A standard automobile tire has a width of about 0.1524 m and a diameter of 0.356 m. So, the circumference of the tire surface comes up to be 1.117 m. In our study, we have assumed that the nodes are dispersed on the rectangular tire surface which makes up the topography of the network. Other than the automobile tire scenario, we also wanted to test the effectiveness of our algorithm for larger network. Therefore, we created another scenario with flat grid topography of 1000 m x 1000 m. The reason for selecting a larger topography is that, it results in a larger number of hop counts between source and destination. Since an evaluation of a routing protocol for WSN must test its forwarding ability along with its ability to effectively deal with link breakage and route repair scenarios. Thus larger topography seemed to a right choice for our simulations which provides a different set of results to show the effectiveness of LPMN in other mobile applications with large networks.

8.4 Traffic Model

For all our simulations, we have used the same traffic model that is a bursty traffic which creates traffic 10% of the time and does not generate any traffic 90% of the time. The packet inter-arrival time is classified by exponential distribution with mean inter-arrival time as 10 seconds (Reference Appendix II). The size of each data packet is same in all the simulations so we can compare the performance of different scenarios for similar traffic. We made sure that no node acts as both source and sink for data packets. The advantage of having a similar traffic model for all the test case scenarios is that the performance metrics can be compared directly among different scenarios which in turn is a measure of how well the routing algorithm had performed under various scenarios with different mobility patterns and different routing algorithms.

8.5 Mobility Pattern

We have considered the semi-deterministic mobility model for our simulations as explained in section 7.4. Here the mobility pattern refers to the movement pattern of the nodes in the WSN, where individual nodes follow a somewhat deterministic mobility model.

The semi-deterministic mobility patterns were generated using trajectory utility provided by the OPNET Modeler®. Here the nodes move in a same direction overall with small deviations for

each move. And for each simulation, the initial layout of the nodes on the topography was in a random fashion, thus giving us spatial diversity for each simulation. We have picked semi-deterministic mobility model because LPMN can improve the performance of the network when the mobility is somewhat predictable.

8.6 Energy Model

In order to quantify how various geographic routing protocols utilize energy of the sensor nodes in the network, we need to design an energy consumption model which is common for all the simulations. We can then test how different geographic routing protocols utilize energy with mobile and static networks by using a common energy consumption model. Our energy consumption model is similar to the one used in the original papers proposing GEAR [18] and EEGR [24] with static networks. We used the similar mode as the one shown in other papers to give us a baseline to compare the performance of these protocols in a mobile network. The idle: receive: transmit ratio is set as 1:1.05:1.4 in our simulations. Basically, every node in the simulation starts with a unit of energy and all the nodes spend 0.1 percent of energy to transmit a data packet which is the same model used in the original papers about GEAR and EEGR [18, 24]. The main reason to use the similar energy model as original papers is to make a valid comparison between the performance of static and mobile networks.

8.7 Sensor Node Design in OPNET®

To implement geographic routing protocols with LPMN, we have added several features to OPNET® Modeler [32], an event-driven network simulator with extensive support for simulation of wireless network protocols. OPNET® Modeler has a graphical user interface which makes it easier to construct a wireless network. Network topologies can be easily described using the primitive Nodes, Links, and Processes. Where Nodes represent end-hosts in the network, Links are the connectors through which Nodes communicate, and Processes are used to implement different network protocols and are the points where packets are created and consumed. Once the topology has been created, simulations can be run by defining various simulation scenarios. In addition to the support for wired networks, OPNET® Modeler has extensive support for wireless networks such as mobile nodes, media access control (MAC) protocols, radio transceivers, antennas, and channel propagation models. Figure 8.2 shows our implementation of a sensor node.

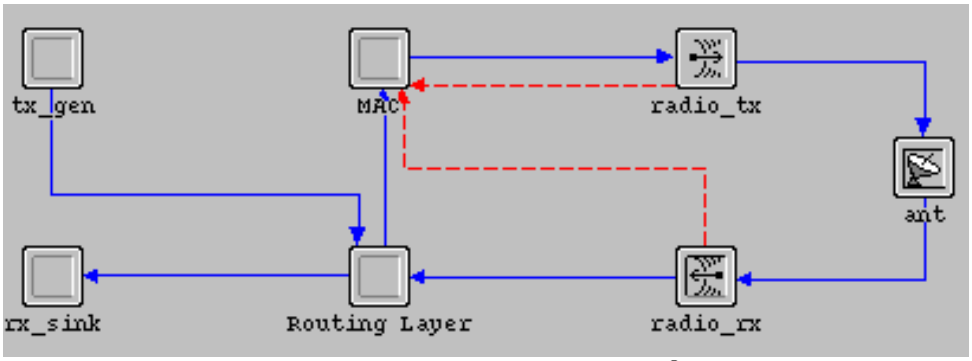

Figure 8.2: Sensor Node Design in OPNET® Modeler

Traffic generator (tx_gen) creates data packets which are destined for other sensor nodes. The blue arrow leading from the traffic generator to the Routing Layer is a packet stream which forwards the packet in the direction shown by the arrows. Traffic generator creates “data packets" that are sent to the Routing Layer that performs the transport and network-layer functions of the protocol stack. Routing Layer is the block which has the functionalities of geographic routing protocols built into it. The Routing Layer sends packets to the MAC layer, where MAC protocols are run. The MAC layer reduces the number of collisions by sensing the medium and only sends data if the medium is free. The MAC layer is modeled to be like CSMA/CA which is common for wireless transmissions. The effects of MAC layer on the results can be ignored since the same MAC layer is used for all the simulations. Finally, the packet is sent to the Transmitter, where the correct transmission power and frequency are added to the packet and the packet is sent through the radio channel. The red dotted arrows originating from the transmitter and the receiver going into MAC layers send a signal to the MAC layer when the channel is busy. These arrows help the MAC layer in sensing the channel before transmitting to avoid collisions. The main attributes of the transmitter are shown in Table 8.1:

Attribute	Value
Data Rate	38400 bits/second
Channel Bandwidth	2600 KHz
Frequency	900 MHz
Power	5.232 µWatts
Modulation	Non-Coherent Frequency Shift Keying

Table 8.1: Wireless Channel Attributes

The transmitter attributes are similar to attributes of Crossbow® transceiver (MPR410CB). The

power of the sensor radio transmitter is set so that any node within a 140 meter radius is within communication range and is called a neighbor of the node for the simulations with network of size 1000 m x 1000 m. The radio used in the nodes can communicate within a circular radius of 140 m (radio range), has data rate of 3.84 kbps and employs CSMA type modified MAC protocol. The Channel sends a copy of the packet to each node connected to the channel. The packets are received by each node's Receiver and then passed up through the MAC, Routing Layer, and Traffic Sink. The Routing Layer de-packetizes the data and update all the required statistics.

8.8 Incorporating Routing Functionality

The process model for Routing Layer works differently for various routing protocols such as geographic Greedy routing, GEAR, and EEGR. However, the overall structure of the Routing Layer is same for all the protocols. In other words, the Routing Layer sits in the idle condition waiting for the packet arrival for all the protocols. After the packet arrival, the Routing Layer extracts the information from the packet to determine the type of the packet and act accordingly. For example, if the packet received by the routing layer is a HELLO packet from a neighboring node, then the routing layer responds by sending the response, HELLO message, back to the neighboring node. In the case of Greedy Routing, the response to HELLO message has the geographic coordinates of the neighboring nodes which help the node with a packet to choose the neighbor with the least distance to the destination. In the case of GEAR, the response to HELLO message has an extra piece of information beside the coordinates of the node which is the residual energy of the neighboring nodes. The node with a packet picks the next hop node based on the cost which is a function of the location and residual energy of the neighboring nodes as described earlier. In both Greedy Routing and GEAR, the overall structure of the Routing Layer is same with a little difference which is the residual energy of nodes being sent in the response to HELLO messages in GEAR.

In the case of EEGR, the same information is being communicated among nodes as GEAR. But, we have to incorporate formation of virtual grids between the source and the destination in EEGR. The virtual grids can be formed easily by using the location of source and destination and the radio range of the nodes. In order to create virtual grids, the inclination angle is found between the line segment between the source and the destination by using simple trigonometry identities. Using the inclination angle, we can find the center point of virtual grids between the

source and the destination. After the formation of the virtual grids, the packet forwarding continues similar to Greedy and GEAR. The main difference in designing these different geographic routing protocols is the math involved such as the cost function and the virtual grids. Therefore, the overall process model of these geographic routing protocols is same for all the simulations. Figure 8.3 shows the design of the process model of the Routing Layer. The main three states in the routing layer are idle, arrival, and transmit data packet. The routing layer sits in the idle state until a new packet arrives or the timeout expired for the HELLO packet sent by the node. This model is only designed for the static case with no mobile nodes.

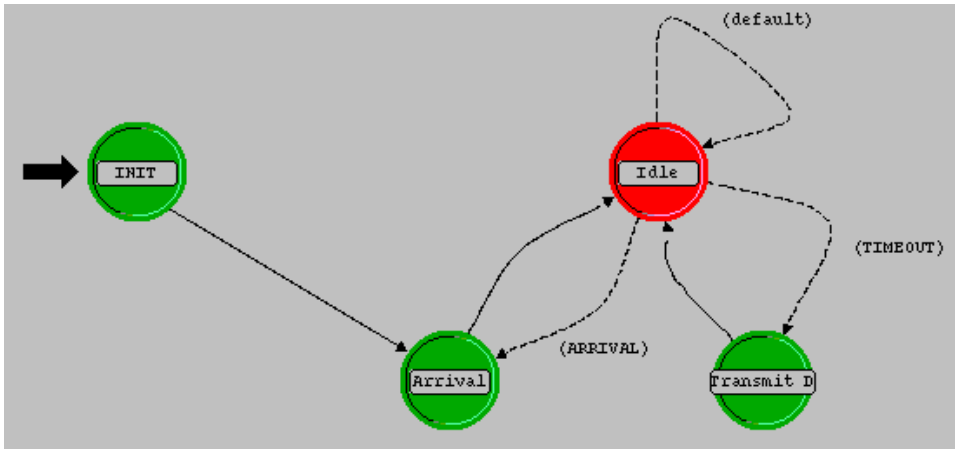

Figure 8.3: Routing Process Model

8.9 Routing Process Model with LPMN

For the mobile case with LPMN, another state called Update_Avg is added to the Routing Layer as shown in Figure 8.4. Update_Avg state continuously takes the moving average of the past movement of the mobile node which can be used to make better routing decision as discussed in Chapter 7. As discussed earlier, this new mobility information can be communicated among neighbors by piggybacking it to the regular routing updates. This will avoid adding extra routing overhead and it makes it easier to implement LPMN with any geographic routing protocol.

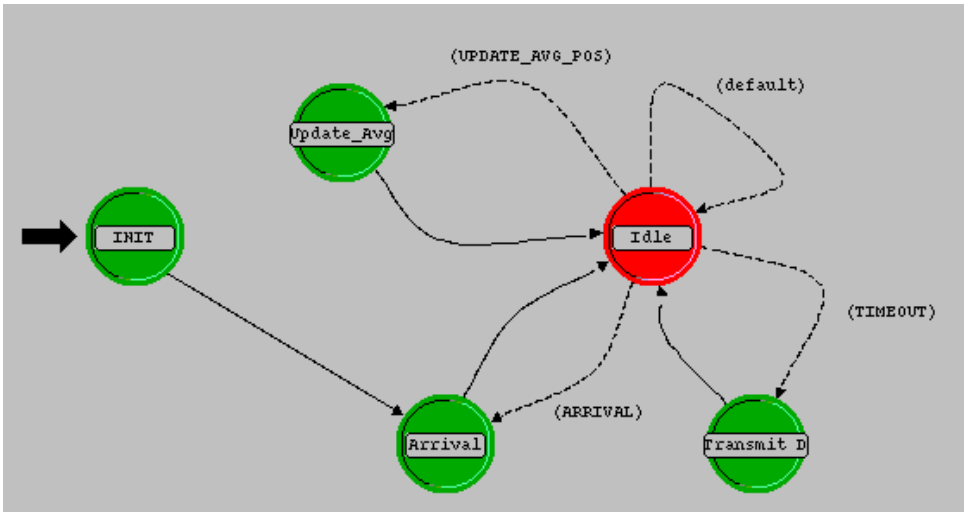

Figure 8.4: Routing Process Model with Mobility

8.10 Conclusion

In this chapter, the simulation model of the routing protocols with LPMN was discussed. Using this simulation model, we have obtained important results which are discussed in detail in next chapter. These results will show the improvement made by LPMN in mobile and sparse networks.

Chapter 9
Discussion of Results

In this chapter, we use the simulation settings discussed in Chapter 8 to study the performance of the routing protocols namely Greedy Routing, GEAR, and EEGR protocol. These protocols have been analyzed for static networks in detail in the original papers [18, 24]. But, these protocols have not been analyzed for mobile and sparse networks in the original papers. Therefore, we have studied these protocols in two different mobile and sparse networks to study their effectiveness. Also, we are looking to find a reliable geographic routing protocol for the network of sensor monitoring the health of the tires which is a sparse network. At the same time, we want to study the improvements LPMN can make on these routing protocols in a large mobile network. Therefore, the first network we have analyzed is a large mobile network to test the effectiveness of geographic routing protocols for large mobile network applications such as the one created by the soldiers equipped with wireless sensors in a battle field. We have picked the large network scenario to test the effectiveness of geographic routing protocols with and without LPMN. The second network resembles an automobile tire equipped with wireless sensors to monitor the health of the tire. We have chosen to test the second scenario to find a geographic routing protocol that performs best for our application of SHM of an automobile tire.

9.1 Performance Parameters

For evaluating the geographic routing protocols with LPMN, we have chosen four parameters which are commonly used in the literature to evaluate the performance of various geographic routing protocols. They are:

- Packet Delivery Ratio: It is the ratio of number of packets received at the destination to the number of packets sent by the source. This ratio ranges from 0 to 1 where a ratio of 1 means that all the data has been successfully received at the destination. Packet delivery ratio is an important metric as it describes the loss rate that will be seen by the transport protocols which run on top of the network layer. Thus packet delivery ratio in

turn reflects the maximum throughput that the network can support. For all our simulations we have kept the same traffic model, so the packet delivery ratio will give us a fair comparison as to how efficient the underlying routing algorithm is under similar traffic load.

- Network Lifetime: Network lifetime is the time during the simulation when all sources are partitioned from their respective target regions. In other words, it is the time during the simulation when a network cannot forward any packets because all the routes have been broken because of node failures. Where, a node failure is mainly due to the nodes running out of battery life in the simulation.

- End to End Delay: End to End Delay is defined as the time spent by a packet between its origination at the source and successful delivery at the destination. End to end delay is an important parameter as strain information may not mean much if it is received later than the expected deadline.

- Average Energy Consumption per node: It is the average of consumed energy among all the nodes in the network. We assume that all the nodes in the simulation model starts out with a unit of energy. The nodes will be considered depleted of energy when energy is zero.

9.2 Verification

To verify the routing algorithms, we started with verification of the existing geographic routing protocols for specific features. In order to verify the geographic routing protocols, we first considered a network of two nodes. The simulation starts with the nodes placed at a distance more than the transmission range of the nodes and then the nodes move towards each other as the simulation proceeds. On examining the trace files, we could see that initially when the distance between the nodes was more than the transmission range, the packets were dropped but as the nodes moved within transmission range of each other, packets were exchanged.

Next to verify the effectiveness of LPMN, we kept the same topology as well as the movement scenario. Here also initially when the nodes are out of transmission range of each other, the packets were dropped, but as they are about to move within the transmission range of each other, there were exchange of packets. So far, we have seen the similar performance from both

scenarios. But, few moments before the mobile neighbor was moving out of the transmission range, the source node stop sending the packets because it knows that the mobile neighbor is about to move out of transmission range due to the mobility awareness and any packets sent to its neighbor will be dropped. Therefore, LPMN helped the source node from deferring transmission of packets which helps in reducing the packet drop rate. This small test showed us that an improvement can be made by LPMN which makes the nodes mobility aware. In order to observe the overall improvements made by LPMN, we will present performance of various scenarios under different conditions.

9.3 Results and Analysis

In this section we present our simulation results for the performance comparison between standard geographic routing protocols and mobility pattern aware geographic routing protocols. As described earlier, for all the simulations the traffic pattern, the duration of simulation, the data packet size, the transmission range of the nodes and the link capacity are kept uniform. The variable parameters for every simulation are number of mobile nodes, the routing algorithm, the underlying mobility pattern and the initial layout of nodes on the rectangular grid.

9.3.1 Large Network Case

The first network is a large network of size 1000 m x 1000 m. There are three different scenarios in the large network. The first scenario is described as static case because all the nodes are static. The second scenario is described as semi-deterministic mobility case because all the nodes are following semi-deterministic mobility pattern. And, the last scenario is called mobility aware semi-deterministic case because all the nodes are following semi-deterministic mobility pattern while running LPMN on top of the routing protocols which make this case mobility aware. These three scenarios help us in determining how the mobility affects the performance of routing protocols and the improvements made by the LPMN to a mobile network.

9.3.1.1 Packet Delivery Ratio

As we discussed when defining the simulation metrics, packet delivery ratio is calculated by dividing the total number of data packets received at all the nodes, by the total number of data packets sent out by the source. Packet delivery ratio forms an important metric for performance evaluation of a geographic routing protocol because, given similar scenarios, the number of data packets successfully delivered at the destination depends mainly on path availability, which in

turn depends on how effective the underlying geographic routing algorithm is in a mobile scenario.

In this section, we show the packet delivery ratio for three geographic routing protocols namely Greedy, GEAR, and EEGR. In Figure 9.1, we plot the packet delivery ratios of the geographic routing protocols at different number of nodes in the network to see how the packet delivery ratio varies at various numbers of nodes. Here for each set of nodes different movement scenarios were generated using the semi-deterministic mobility model.

As can be seen from the graph, packet delivery ratio is poor for the mobile scenario without LPMN i.e. nodes are picking routes with no awareness of the mobility pattern and hence the packets were forwarded over a broken link which were eventually dropped. But the performance of the mobile network improves with LPMN indicating that nodes are making better routes because they can predict their neighbors' mobility patter due to LPMN.

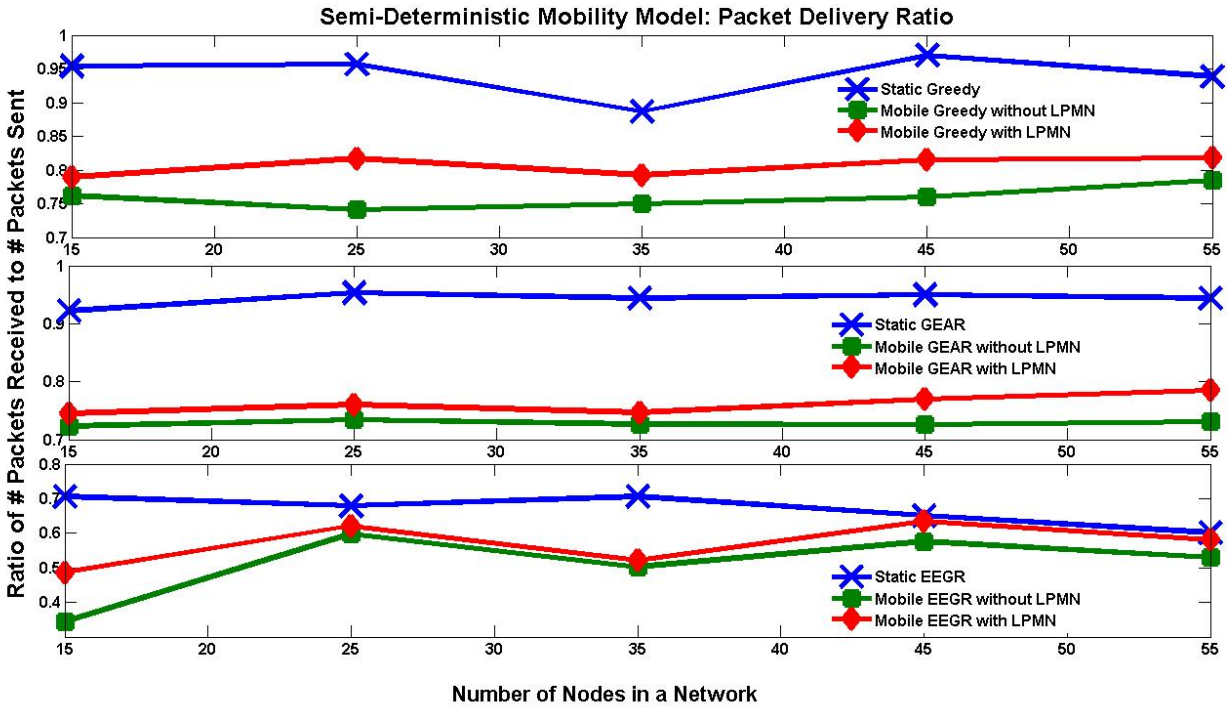

Figure 9.1: Large Network: Packet Delivery Ratio Curves

The plot of packet delivery ratio for Greedy Routing is shown in the top plot of Figure 9.1. Greedy Routing Protocol picks the next hop neighbor with minimum distance to the destination among the neighbors. The plot shows that the Greedy routing protocol has packet delivery ratio of about 0.941 on average for the static case. The plot also shows that packet delivery ratio is decreased by 19.2 % to 0.759 on average for the mobile network without LPMN. However, the packet delivery ratio is increased by 6.2 % to 0.806 on average for the mobile network with LPMN.

The plot of packet delivery ratio for GEAR is shown in the middle plot of Figure 9.1. GEAR picks the next hop neighbor with minimum cost to send the packet among the neighbors. The cost is a combination of residual energy of the node and distance to the destination as described in Chapter 6. The plot shows that the GEAR has packet delivery ratio of about 0.943 on average for the static case. The plot also shows that packet delivery ratio is decreased by 22.8 % to 0.728 on average for the mobile network without LPMN. But due to the mobility awareness, the packet delivery ratio is increased by 4.6 % to 0.761 on average for the mobile network with LPMN. We can see that GEAR does not perform as well as pure Greedy Routing Protocol in terms of packet delivery ratio because nodes take residual energy into consideration while picking the next hop neighbor in GEAR.

The plot of packet delivery ratio for EEGR is shown in the bottom plot of Figure 9.1. EEGR protocol is described in detail in Chapter 6. The plot shows that the EEGR has packet delivery ratio of about 0.669 on average for the static case. The plot also shows that packet delivery ratio is decreased by 23.7 % to 0.51 for the mobile network without LPMN. However, we can see the improvement as the packet delivery ratio is increased by 11.5 % to 0.569 for the mobile network with LPMN. The curves show that EEGR does not perform as well as Greedy and GEAR in terms of packet delivery ratio because EEGR performs poorly for mobile and sparse network [24].

9.3.1.2 End to End Delay

While we have shown the LPMN improves packet delivery ratio of mobile network, we still have to show how LPMN affects end to end delay. End to End delay is the time a data packet take in traversing from the time it is sent by the source node till the point it is received at the destination node. This metric is a measure of how efficient the underlying routing algorithm is,

because primarily the delay depends upon optimality of path chosen, the delay experienced at the interface queues and delay caused by the retransmissions at the physical layer due to collisions. The curves in Figure 9.2 show plots of end to end delay between the source and the destination.

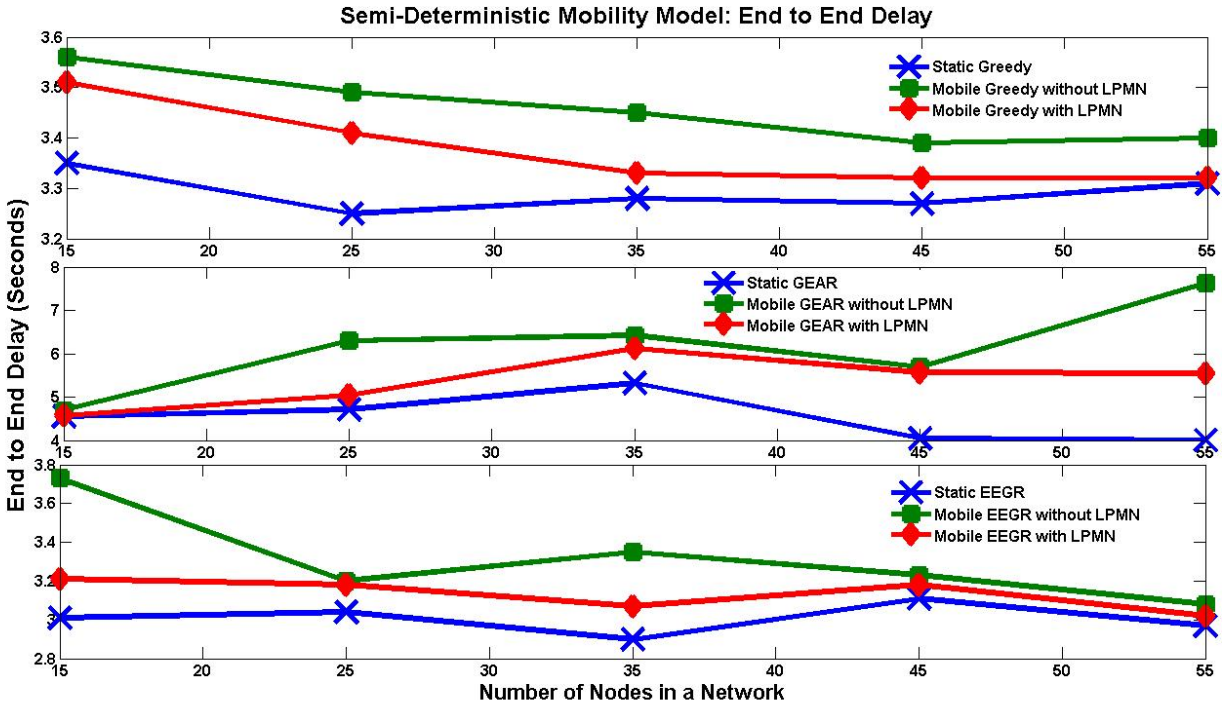

Figure 9.2: Large Network: End to End Delay Curves

The plot of end to end delay for Greedy Routing is shown in the top plot of Figure 9.2. The plot shows that the Greedy routing protocol has an end to end delay of about 3.29 seconds on average for the static case. The plot also shows that end to end delay is increased by 0.17 seconds to 3.46 seconds on average for the mobile network without LPMN. But due to the mobility awareness, the end to end delay is decreased by about 0.1 seconds to 3.36 seconds for the mobile network with LPMN.

The plot of end to end delay for GEAR is also shown in the middle plot of Figure 9.2. The plot shows that the GEAR has an end to end delay of about 4.53 seconds on average for the static network. GEAR has higher end to end delay than Greedy routing protocol because GEAR picks longer paths to the destination. The plot also shows that end to end delay is increased by 1.62 seconds to 6.14 seconds on average for the mobile network without LPMN. After adding LPMN to the mobile network, the end to end delay is decreased by 0.77 seconds to 5.37 seconds on average.

The plot of end to end delay for EEGR is also shown in the bottom plot of Figure 9.2. The plot shows that the EEGR has end to end delay of about 3.01 seconds on average for the static case. The plot also shows that end to end delay is increased by 0.31 seconds to 3.32 seconds on average for the mobile network without LPMN. But, the end to end delay is decreased by 0.19 seconds to 3.13 seconds on average for the mobile network with LPMN.

9.3.1.3 Energy Analysis

We have examined the effects of LPMN on the energy consumption of the network. We measure the energy consumption of the network as average consumed energy per node for varying number of nodes. Figure 9.3 show plots of average consumed energy per node. The one common theme in the energy curves in Figure 9.3 is that the mobile network has lower average consumed energy per node than the static network. Mobile network consumes less energy because nodes in a mobile network are less likely to be used as a router if they move out of transmission range of their neighbors. But, the average energy consumption becomes larger for the case when the nodes are mobility pattern aware because more mobile nodes are being picked as router to forward the data which leads to higher energy consumption on average.

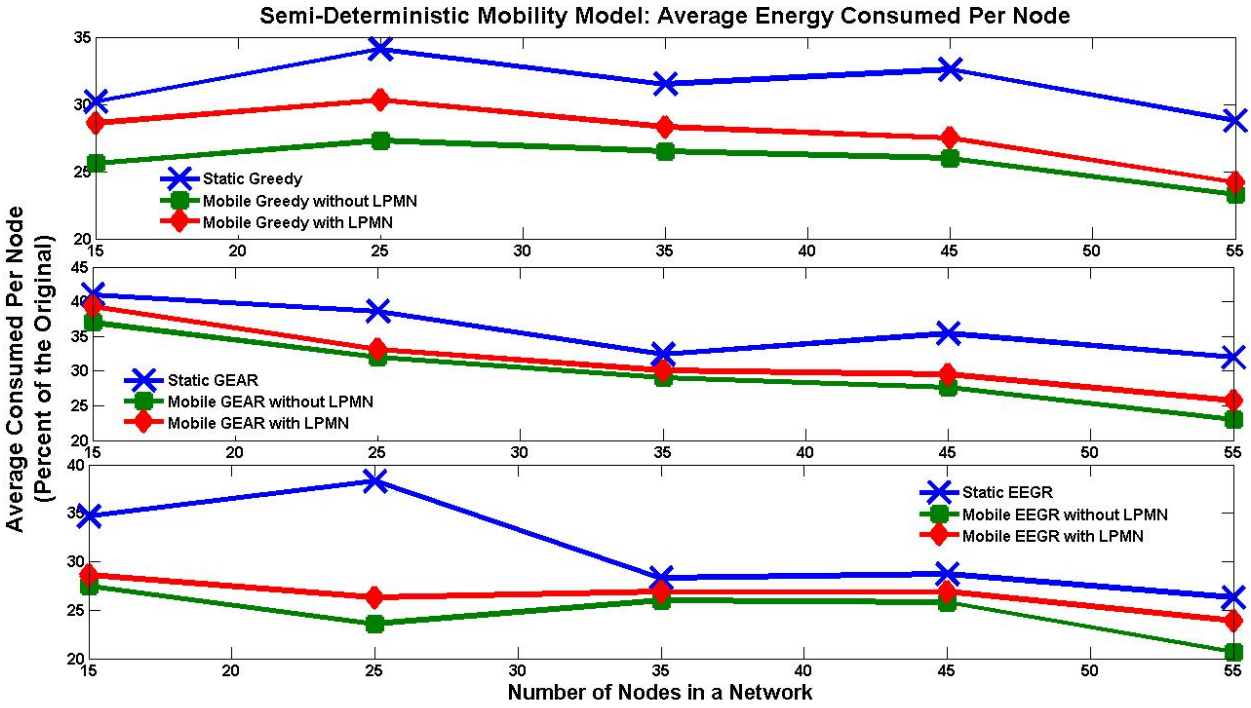

Figure 9.3: Large Network: Residual Energy Curves

In this section, we have analyzed the effectiveness of LPMN which is self learning about the underlying mobility pattern. We have learned that LPMN can improve the performance of geographic routing protocol in a mobile network if the nodes are moving in a semi-deterministic pattern. We had mentioned earlier that the performance of geographic routing protocols can be improved if it had knowledge about the underlying mobility pattern of the nodes.

True to our expectations, LPMN does improve the efficiency of the geographic routing protocols and this is confirmed by our results. Depending upon the somewhat predictable mobility pattern, nodes can learn their neighbors' movements by communicating history of their past movements among each other. And as the nodes forwards the packet over links that are less likely to broken due to mobility awareness, the network shows higher packet delivery ratio as well as less delay in packet delivery.

This gain in performance however is not free of cost. And the cost associated with it is the amount of processing needed to keep an average of past movements at each node as well as transmitting the location information along with the routing updates which increase the size of the routing updates. With smaller network sizes, this might not be an issue but as the network size increases more and more, the storage overhead could be large.

9.3.2 Analysis of Geographic Routing Protocols for SHM of an Automobile Tire

We have studied the performance of geographic routing protocols in a mobile network with and without LPMN for large networks. But, we still have not test the effectiveness of geographic routing protocols for a smaller and sparse network created by the wireless sensors embedded in the tire. During the research, we have realized that the nodes that are embedded on the tire do not move relative to each other when the tire is stationary and even when tire is moving. Only movement is that sink on the tire is moving relative to the base station which is on the car. But since the communication between the sink and the base station is wireless it does not matter if sink is moving or stationary. In essence, the sensor network inside the tire is flat and stationary. Therefore, we have not incorporated any mobility pattern while simulating the second network which resembles the tire sensor network. Still, the tire sensor network is considered a sparse network due to small number of nodes. For this reason, we have studied the performance of the geographic routing protocols in a simulated tire network to find the most reliable geographic routing protocol for tire sensor network.

The second network will be a smaller network representing a rectangular flat grid of a tire surface. The dimensions of the network are 0.1524 m x 1.117 m which is taken from for a standard automobile tire diameter and width. The tire network will have two scenarios which are:

- Straight Line: The first scenario will have nodes assembled in a straight line i.e. the grid is one dimensional on a long flat surface of the tire.
- Random Fashion: In the second scenario, grid is two dimensional with nodes dispersed randomly on a long flat surface of the tire.

These two scenarios help us in determine if the location of the nodes on the tire surface is important in picking the geographic routing protocol which performs best for SHM of an automobile tire. In all of the simulations of the network, the network size and the transmission range is kept unchanged. We vary the number of the nodes on a long flat surface network ranging form 5 to 30 nodes. The intention of changing the density of the network is to find the routing protocol which performs well for various level of density in a network.

We have analyzed the effectiveness of Greedy, GEAR, and EEGR in terms of network lifetime, packet delivery ratio, and end to end delay between the source and the destination.

9.3.2.1 Network Lifetime

Network lifetime is the time during the simulation when all sources are partitioned from their respective target regions. In other words, it is the time during the simulation when a network cannot forward any packets because all the routes have been broken because of node failures. Where, a node failure is mainly due to the nodes running out of battery life in the simulation. The plots for network lifetime are shown in Figure 9.4. The plot shows that the network lifetime increases with the number of nodes. This result is expected because network has more alternate routes to the main route with increasing number of nodes. Also, the Greedy routing protocol performs better than GEAR and EEGR on the whole. The main reason for this behavior is Greedy protocol always find the most efficient and reliable routes in terms of number of hops than GEAR and EEGR.

Also, EEGR performs the worst out of the three protocols which is expected because EEGR does not perform well for sparse networks. Also, the network lifetime is about same for the

straight line network and the scenario where the nodes are dispersed in random fashion which shows that the exact placement of nodes in not very important.

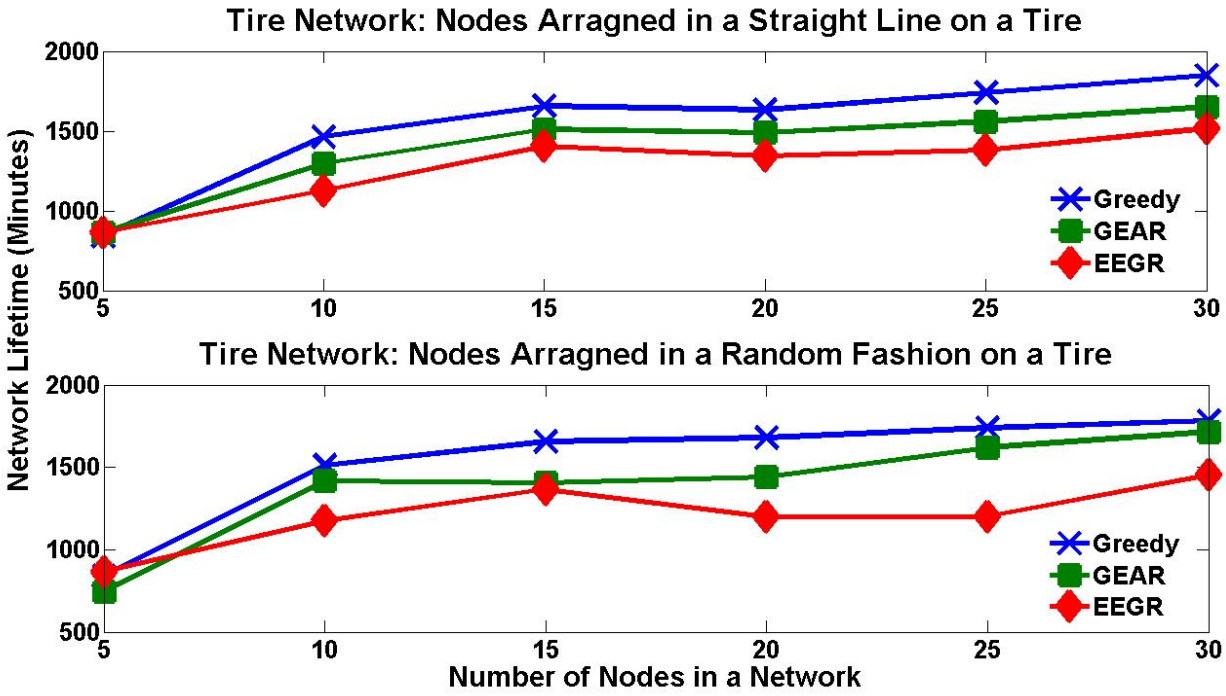

Figure 9.4: Tire Network: Network Lifetime

9.3.2.2 Packet Delivery Ratio

Packet delivery ratio is the ratio of number of packets received at the destination to the number of packets sent by the source. The plots for packet delivery ratio are shown in Figure 9.5. Again, we have observed that the packet delivery ratio is about same for the straight line network and the scenario where the nodes are dispersed in random fashion. Also, Greedy protocol has the best performance because it finds the most efficient route in terms of number of hops. GEAR does not perform as well as Greedy because nodes take residual energy into consideration while picking the next hop neighbor in GEAR. Therefore, GEAR does not pick the route with least number of hops compared to Greedy Protocol. Again, EEGR has the worst performance because we are dealing with a sparse network.

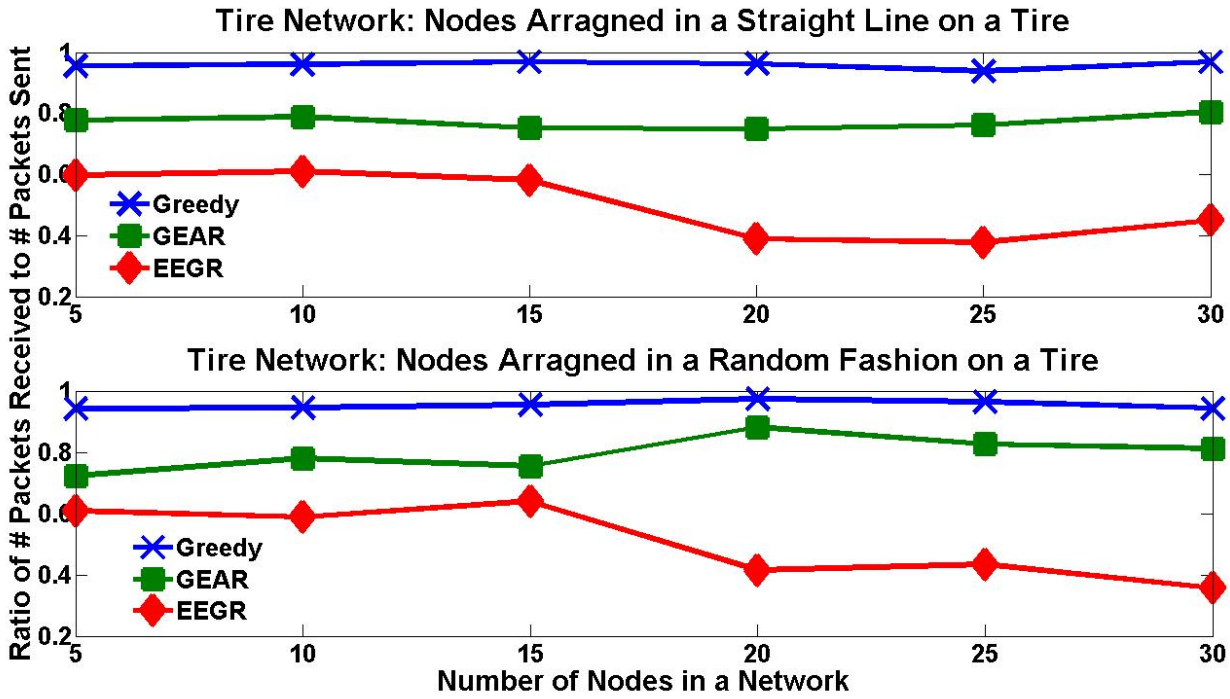

Figure 9.5: Tire Network: Packet Delivery Ratio

9.3.2.3 End to End Delay

End to end delay is the time spent by a packet between its origination at the source and successful delivery at the destination. The plots for end to end delay are shown in Figure 9.6. Again, the plots show that the results are almost similar for the straight line network and the scenario where the nodes are dispersed in random fashion. We can see in the plots that EEGR has the smallest end to end delay out of the three routing protocols. The main reason for this behavior is EEGR creates virtual grid between the source and the destination and actively pick nodes from the grids geographically to forward data to the destination. In other words, EEGR excludes the nodes that are not in the vicinity of the area between the source and the destination. Also, we can see that Greedy performs almost as well as EEGR because Greedy picks the route with the least number of hops. And, GEAR does not perform as well as other protocols because it does not find the route with least number of hops due to the consideration of residual energy.

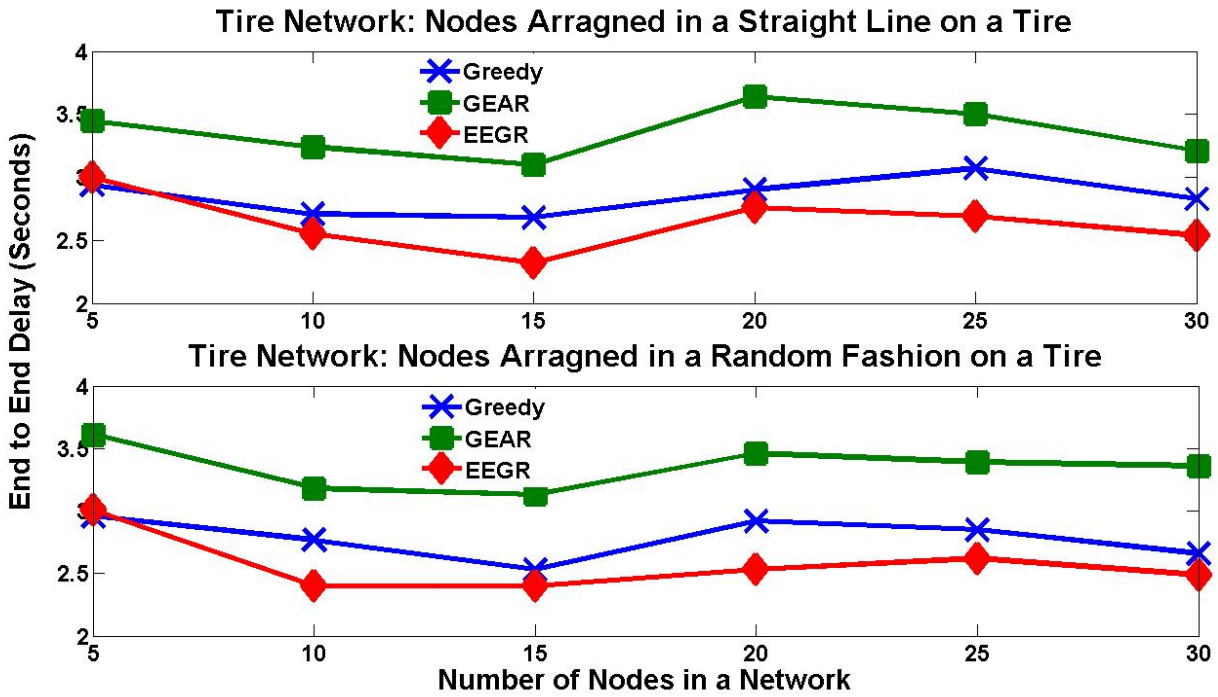

Figure 9.6: Tire Network: End to End Delay

We can conclude from the simulation results that a simple Greedy protocol is sufficient for the WSN to monitor the health of an automobile tire. We can achieve a high packet delivery ratio of about 0.96 with a reasonable end to end delay of about 2.9 seconds with Greedy protocol.

9.4 Conclusion

In this chapter, we have outlined the energy consumption and packet delivery ratio results of geographic routing protocols namely Greedy routing, GEAR, and EEGR with and without LPMN. We have demonstrated our approach of LPMN built on top of geographic routing protocols for mobile networks. Geographic routing protocols have excellent performance for static networks, but perform poorly for mobile and sparse networks.

LPMN adapts to the mobility of the network by making all the nodes predict their future location and communicate this information among direct neighbors. We have seen that LPMN improves the results of all the above protocols for mobile and sparse networks in terms of packet delivery ratio and end to end delay.

Although we have studied LPMN with Greedy Routing, GEAR, and EEGR for a semi-deterministic mobility model in a large network, LPMN can be evaluated with mobility models other than the two mentioned earlier. LPMN can also be tested for other traffic models. Finally,

experimentation is needed to validate these results with physical hardware (Crossbow) in actual scenario. At this time, we have discussed all the details of the research being done for SHM of an automobile tire. And, the next chapter discusses our achievements and future work that can be done for SHM of an automobile tire.

Chapter 10
Ongoing and Future Work

We conclude this thesis by exploring ideas for future research in SHM of an automobile/aircraft tire. We have proposed a position prediction technique that improves the performance of geographic routing protocols in terms of energy consumption and packet delivery ratio.

Overall, this thesis has developed network architecture to measure strain of an automobile/aircraft tire and to wirelessly transmit strain information to the base station. The main accomplishments of this work are:

1. We have designed an application architecture consisting of hardware components such as Crossbow motes, operating system such as TinyOS, and strain sensing system needed to measure strain on an automobile tire that can transmit data wirelessly.

2. We have developed a localized strain sensing system using a new sensor called Metal Rubber^{TM}. Metal Rubber^{TM} has ideal properties for a possible sensor that can measure strain on a tire. Unfortunately, Metal Rubber^{TM} has a hysteresis built in the material which limited its ability to be used as a sensor.

3. We have developed a technique to improve performance of existing geographic routing protocols in a mobile network by predicting future position of mobile nodes. Based on the simulations that we have carried out to support LPMN, we have observed that the packet delivery ratio is higher for a mobile network with position prediction.

This section describes the future research work which is mainly related with hardware, and sensor setup. This chapter discusses the transceiver placement on the tire which will be important factor in terms of performance of the strains sensing system. Also, the batteries on the transceiver have limited power which will run out eventually. Besides an energy efficient WSN,

an efficient power conservation technique can increase the lifetime of the batteries which is discussed in section 10.2.

10.1 Hardware Setup

A transceiver and data acquisition unit need to be placed within the tire with a power source to keep them running. This raises the issues of where the units should be placed and will the unit's power source outlast the tire's life span.

10.1.1 Transceiver Placement

The placement of the transceiver and/or data acquisition unit within the tire becomes an issue since there are many factors to consider such as safety of units as well as the strength of transceiver's signal.

10.1.1.1 Outside the Tire

The first transceiver placement, shown in Figure 10.1(a), depicts a transceiver on the outer portion of the tire's rim. The advantages of this option are: (a) Transceiver will have line-of-sight to the BS which ensures optimal signal strength, diminishing packet loss, and allows transceivers to operate under weaker signal strength which conserves power. (b) The equipment will be easily accessible for recharging, replacement, or repair. Some disadvantages involve the unit being exposed to hazardous debris which can damage it. This problem can be solved by making a specialized compartment within the tire's rim to protect the units.

10.1.1.2 Inside the Tire

Another place to put the transceivers could be on the inner portion of the tire's rim or within the tire's inner tread lining as show on Figure 10.1(b) and 17(c).

One advantage of this option is that shorter sensor wires will be needed which will provide more accurate strain readings. However there are a few disadvantages associated with this option such as (1) The transceiver needs to send signals through the tire's rubber and metal mesh, and the signals will be subject to multi-path fading due to the high density of metallic components surrounding the transceiver. (2) The equipment will be hard to reach for recharging, replacing, or repairing. (3) Also, if placed within the tire's inner tread lining the units will undergo most, if not all, the stress a tire is subjected to. The possible solutions being considered are: (A) Make the antenna weaved through the tire's rubber and metal mesh so the signal could bypass such

obstacles. (B) Sticking the antenna out of the tire's valve or small hole on the rim. (C) Applying a coating to the inner rim's surface to prevent signals from bouncing off as easily and (D) A specialized container can be made to protect the unit from temperature and shock variations.

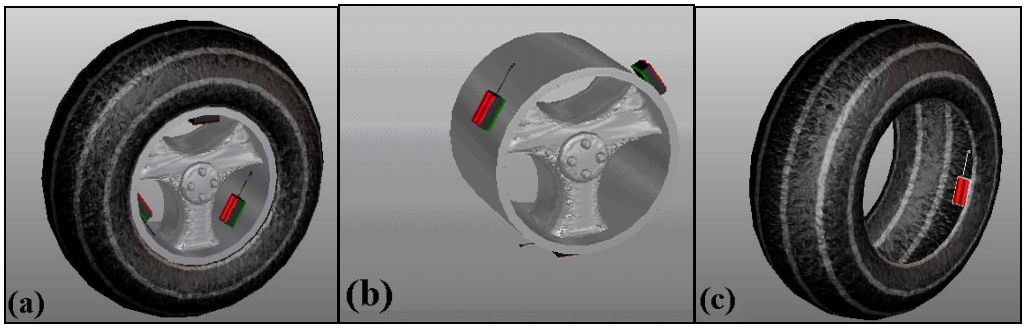

Figure 10.1: (a) Transceiver placed on the outer rim. (b) Transceiver placed on the inner rim. (c) Transceiver placed within the tire.

10.2 Power Conservation

Alternate solutions can be implemented if the transceivers' life span is significantly shorter than the tires'. A simple solution is to hook a battery charger up to the transceivers' batteries and place a plug on the tire's rim where the user can hook up a power source to the charger.

There are many options being considered to improve power conservation within the transceiver itself. An event driven algorithm will hold back on data transmission until a critical event is detected. Also, strain data acquisition may be suspended when the automobile/aircraft is not in motion as well as adapting the sampling rates according to the vehicle's velocity, transceiver's battery power, events raised, etc.

10.3 Conclusion

SHM of an automobile tire is a growing field due to the TREAD act. We have made great progress to design a strain sensing system that can measure strain accurately of a tire surface and transmits this information reliably to the driver. However, there is still work that needs to be done as mentioned in this chapter before an automobile or an aircraft can monitor health their tires.

Chapter 11
Appendix

11.1 Appendix I – Current Sensor Hardware

11.1.1 UC Berkeley Smart Dust Motes

These nodes are designed by University of California, Berkeley and are called smart dust. The mote is shown in Figure 11.1. This is a tiny node containing an MCU (ATMEL 90LS8535) with 8-bit Harvard architecture and 16-bit addresses. This controller provides 32 8-bit general purpose registers and runs at 4 MHz and 3.0 V. The system memory comprises 8KB flash memory and 512-byte SRAM as data memory. The radio is an asynchronous input/output device with hard real-time constraints. It consists of an RF Monolithic 916:50 MHz transceiver, antenna and collection of physical-layer components to configure the physical layer characteristics such as signal strength and sensitivity. It comes with a temperature sensor with an option to mount custom-selected sensors on the sensor board. The nodes run TinyOS [3] operating system which fits in 178 bytes of memory supporting two-level scheduling and allows for high concurrency to be handled in a very small amount of space.

Figure 11.1: UC Berkeley Motes

11.2 Appendix II – Average Inter-Arrival Time of the System

An exponentially distributed random variable X ~ exp (λ) has the following probability density function:

$$f_X(x) = \lambda e^{-\lambda x}, x > 0 \qquad (11.1)$$

The mean of X can be calculated as shown in the following equation:

$$E[X] = \int_0^\infty x f_X(x) dx = \frac{1}{\lambda} \tag{11.2}$$

Where, E[X] is the mean of random variable X. The probability that X is greater than α is given by the following equation:

$$P[X > \alpha] = \int_\alpha^\infty x f_X(x) dx = e^{-\lambda\alpha} \tag{11.3}$$

References

[1] H. Karl and A. Willig, *Protocols and Architecture for Wireless Sensor Networks*. West Sussex, England: Wiley, 2005.

[2] "Crossbow® Technology Inc. Global Leader in Sensory Systems." vol. 2006, 2005.

[3] "TinyOS." vol. 2006, 2004.

[4] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *Wireless Communications, IEEE [see also IEEE Personal Communications],* vol. 11, pp. 6-28, 2004.

[5] NanoSonic, "Metal Rubber™ Sensors and Electrodes Product #: MR-01-D5- and MR-01-S5."

[6] D. H. Everett, *Transactions of the Faraday Society,* vol. 51, p. 1551, 1955.

[7] A. S. Khan and X. Wang, *Strain Measurements and Stress Analysis*. Upper Saddle River, NJ: Prentice Hall, 2001.

[8] SensorLand, "Strain Gauge," 2006.

[9] A. L. Window, *Strain gauge technology*, 2nd ed. London ; New York: Elsevier Applied Science, 1992.

[10] "Bridge Output and Nonlinearity Errors For Various Bridge/Strain Arrangements in a Uniaxial Stress Field ": Vishay.

[11] R. Matsuzaki and A. Todoroki, "Passive wireless strain monitoring of tyres using capacitance and tuning frequency changes," *Smart Materials and Structures,* vol. 14, p. 561, 2005.

[12] R. Matsuzaki and A. Todoroki, "Passive wireless strain monitoring of actual tire using capacitance-resistance change and multiple spectral features," *Sensors and Actuators A: Physical,* vol. 126, pp. 277-286, 2006.

[13] F. Zhao and L. J. Guibas, *Wireless sensor networks : an information processing approach*. San Francisco: Morgan Kaufmann, 2004.

[14] "Wireless Sensor Netowrks Getting Started Guide," San Jose, CA: Crossbow®, 2005.

[15] W. Chonggang, K. Sohraby, L. Bo, M. Daneshmand, and H. Yueming, "A survey of transport protocols for wireless sensor networks," *Network, IEEE,* vol. 20, pp. 34-40, 2006.

[16] E. R. Johnson and W. J. Devenport, "Static Response of a Beam," 2006.

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

[18] Y. Yu, R. Govindan, and D. Estrin, "Geographical and Energy Aware Routing: a recursive data dissemination protocol for wireless sensor networks," UCLA, 2001, p. 11.

[19] Y. Xu, J. Heidemann, and D. Estrin, "Geography-Informed Energy Conservation for Ad-hoc Routing," in *Proc. 7th Annual ACM/IEEE Int'l. Conf. Mobile Comp. and Net.*, 2001, pp. 70-84.

[20] M. Stemm and R. H. Katz, "Measuring and Reducing Energy Consumption in Network Interfaces of Handheld Devices," *IEICE Transactions on Communications,* vol. E80-B, pp. 1125-1131, 1997.

[21] I. Stojmenovic and L. Xu, "Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 12, pp. 1023-1032, 2001.

[22] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Wireless Networks,* vol. 8, pp. 481-494, 2002.

[23] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-Case Optimal and Average-Case Efficient Geometric Ad Hoc Routing," in *Proc. 4th ACM Int'l. Conf. Mobile Comp. and Net.*: ACM, 2003, pp. 267-278.

[24] Y. Xiaozong, L. Renfa, and L. Yanyan, "An energy-efficient geographic routing algorithm for wireless sensor networks," 2005, pp. 694-699.

[25] L. Junlong and K. Geng-Sheng, "A Novel Location-fault-tolerant Geographic Routing Scheme for Wireless Ad Hoc Networks," 2006, pp. 1092-1096.

[26] N. Jongkeun, S. Danny, and K. Chong-kwon, "Greedy Geographic Routing using Dynamic Potential Field for Wireless Ad Hoc Networks," *Communications Letters, IEEE,* vol. 11, pp. 243-245, 2007.
[27] V. Rajaravivarma, Y. Yi, and Y. Teng, "An overview of Wireless Sensor Network and applications," 2003, pp. 432-436.
[28] P. Raviraj, H. Sharif, M. Hempel, C. Song, H. H. Ali, and J. Youn, "A mobility based link layer approach for mobile wireless sensor networks," 2005, p. 6 pp.
[29] C. Canfeng and M. Jian, "MEMOSEN: multi-radio enabled mobile wireless sensor network," 2006, p. 5 pp.
[30] G. Song, O. Yang, and S. Yantai, "Improving source routing reliability in mobile ad hoc networks," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 16, pp. 362-373, 2005.
[31] Y. YuHua, C. HuiMin, and J. Min, "An Optimized Ad-hoc On-demand Multipath Distance Vector (AOMDV) Routing Protocol," 2005, pp. 569-573.
[32] OPNET, "OPNET: Making Networks and Applications Perform." vol. 2006 Bethesda, MD, 2007.

Vita

Farrukh Mehmood Gondal was born in Sukkur, Pakistan in 1982. His interests cover a diverse area in Electrical and Computer Engineering ranging from Networking, Wireless Communications, to Signal Processing. He graduated from University of Maryland, College Park in 2005 with a Bachelor of Science degree in Electrical Engineering. As an undergraduate engineering student, he (as a part of a team of four) designed a Pocket PC application named, The Pocket Interactive Chords (PIC), which is written in Microsoft eMbedded Visual C++ and Microsoft Visual C++ that serves as a portable guitar tuning and chord learning application.

Mr. Gondal joined Virginia Tech in Fall 2005, as a graduate student, in the Bradley Department of Electrical Engineering. He has been working on energy-efficiency at the routing layer since Spring 2006 and has proposed a novel routing protocol to deal with mobile and sparse nodes as a part of his thesis work. A few current publications of his include:

[1] A. D. Afrashteh, S. Bland, F. M. Gondal, R. K. Kapania, A. Mishra, R. D. Moffitt, and R. R. Wilson, "**Embedded Wireless Sensors for Aircraft/Automobile Tire Structural Health Monitoring,**" in *IEEE SECON*, Reston, VA, 2006.

We are also in process to submit three different papers dealing with WSN architecture for Structural Health Monitoring of aircraft/automobile tire and an energy efficient geographic routing algorithm for mobile and sparse WSNs. These papers are listed below:

[2] F. M. Gondal, R. K. Kapania, A. Mishra, and R. D. Moffitt, "**An Energy Efficient Geographic Routing Algorithm for Mobile & Sparse Wireless Sensor Networks**," in *MILCOM 2007* Orlando, Florida, 2007.

[3] F. M. Gondal, R. K. Kapania, A. Mishra, R. D. Moffitt, and M. R. Sunny, "**A Wireless Sensor Network Architecture for Structural Health Monitoring of Aircraft/Automobile Tires**," in *MILCOM 2007* Orlando, Florida, 2007.

[1] F. M. Gondal, R. K. Kapania, A. Mishra, R. D. Moffitt, and M. R. Sunny, "**A Wireless Sensor Network Architecture for Applications with Mobile Nodes**," *IEEE Communications,* 2007.

Mr. Gondal successfully defended his Master's thesis on April 27th, 2007 and is working with Raytheon Missile System©, a government contractor in Tucson, Arizona, USA as a Multi-Disciplined Engineer II. He can be contacted at gondal@vt.edu.