ROBUST COMMUNICATION FOR LOCATION-AWARE MOBILE ROBOTS USING MOTES

by

BRIAN WISE MULANDA

B.S., Washburn University, 2005

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2008

Approved by:

Major Professor
David Gustafson

Copyright

Abstract

The best mode of communication for a team of mobile robots deployed to cooperatively perform a particular task is through exchange of messages. To facilitate such exchange, a communication network is required. When successful execution of the task hinges on communication, the network needs to be robust - sufficiently reliable and secure. The absence of a fixed network infrastructure defeats the use of traditional wire-based communication strategies or an 802.11-based wireless network that would require an access point. In such a case, only an ad hoc wireless network is practical.

This thesis presents a robust wireless communication solution for mobile robots using motes. Motes, sometimes referred to as smart dust, are small, low-cost, low-power computing devices equipped with wireless communication capability that uses Radio Frequency (RF). Motes have been applied widely in wireless sensing networks and are typically connected to sensors and used to gather information about their environment. Communication in a mote network is inherently unreliable due to message loss, exposed to attacks, and supports very low bandwidth. Additional mechanisms are therefore required in order to achieve robust communication.

Multi-hop routing must be used to overcome short signal transmission range. The ability of a mobile robot to determine its present location can be exploited in building an appropriate routing protocol. When present, information about a mobile robot’s future location can aid further the routing process. To guarantee message delivery, a transport protocol is necessary. Optimal packet sizes should be chosen for best network throughput. To protect the wireless network from attacks, an efficient security protocol can be used.

This thesis describes the hardware setup, software configuration, and a network protocol for a team of mobile robots that use motes for robust wireless communication. The thesis also presents results of experiments performed.

Acknowledgements

I would like to acknowledge Dr. David Gustafson, my major professor, for the opportunity to work on this thesis under his supervision and the constant support and guidance he has provided me during my master's studies. I would also like to acknowledge the two other members of my committee: Dr. Gurdip Singh, Dr. Xinming (Simon) Ou. They have been instrumental in providing technical guidance on the material I needed to successfully complete this thesis. Special thanks go to my brother, Allan B. Liavoga, whose optimism motivated me to pursue higher education here in the United States. I acknowledge the support of many others who have contributed directly and indirectly towards my thesis. They include: Dr. Dan Andresen, Dr. Rodney Howell, Herve Oyenan, Chris Zhong, Jaidev Manghat, Jorge Valenzuela, Aaron Chleborad, Justing Dugger, Vivian Onyango, Miriam Kindle, Dr. Jepkoech Tarus, and my sisters: Brilliant Kuboka, Caroline Idaya, and Lily Kidayu. There are many more that I am not aware of and whose names do not appear above, that have supported me in ways that I might never discover. I acknowledge them.

Dedication

I would like to dedicate this thesis to my mother, Mary, who consistently prayed for us.

TABLE OF CONTENTS

List of Figures

CHAPTER 1 - Introduction

1.1 Interaction in a robot team

For a team of robots to successfully execute a particular task in a cooperative fashion, the robots must be able to interact [or communicate].

One way to achieve this is through sensors and actuators on the robot. For example, a robot with vision capabilities may be programmed to visually identify when it has been assigned a task. The presence of a bright red object in the view of the robot could signify that the robot should turn in a particular direction and move a specified distance to wait for further instruction. To provide feedback, for example about completion of task, a robot could perform a predefined action or sequence of actions. To correctly interpret this feedback, a receiver would need to be appropriately equipped with sensors and be available to sense the feedback in time. Such communication techniques that require the robots to have appropriate actuators and sensors could limit the range of "messages" that the robot can send or receive. Furthermore, communication in this case may not be precise or even reliable due to factors that could distort the actuating or sensing processes.

A better way to interact is through exchange of messages – actual bytes. For example, a robot, designated as the team leader, could send a message (single byte or a stream of bytes) to another robot instructing the receiving robot to turn in a particular direction and move a specified distance to wait for further instruction. To provide feedback about completion of the task, the instructed robot could send a message back to the lead robot. Since these messages are predefined and their meaning known to both communicating parties, they can be interpreted precisely and correctly. A wider range of messages can be defined in this case compared to when relying purely on sensing and actuating that does not involve exchange of actual bytes.

Therefore, communication through exchange of messages remains the best way to facilitate interaction in a team of mobile robots deployed to cooperatively perform a particular task.

1.2 The Problem

A team of mobile robots is employed to cooperatively perform a particular task. The task involves the robots moving to designated areas as they gather information about their environment, and communicating feedback to a lead robot [or control center]. The lead robot assigns each robot an area to patrol and responds to feedback it receives from the robots. Each robot maintains status information about all other robots in the team. Status information includes the robot's last known location. A robot, therefore, needs to be aware of at least the location of other robots in the team.

A robot first advertises its existence by sending out a message seeking to know the identity of the lead robot. A non-lead robot in the team simply ignores this message. The only lead robot, if present, responds back with a registration confirmation message. Once registered at the lead robot, a robot is eligible for task assignment. The robot control structure in this case is primarily centralized, but could be restructured to allow decentralized behavior. Task assignment is achieved using an algorithm executed by the lead robot. In some cases, it is necessary for a team to learn about the existence of a new robot. Additionally, the lead robot needs to know when a robot is no longer part of the team so that it can re-distribute tasks among the remaining robots.

Each robot can send or receive an arbitrary number of messages. There is no restriction on the size of a message exchanged between robots. The robots follow a communication protocol when exchanging command or feedback information while cooperatively performing a task. Each message is crucial to the successful completion of the task since it can impact the behavior or action of a robot.

Clearly, such a team of mobile robots which must cooperate to perform the task of patrolling designated areas require a communication network to facilitate exchange of messages. Since the messages can impact the behavior or action of a robot, and therefore the successful completion of or execution of a set of actions for, a task, the communication network must be robust – sufficiently reliable and secure.

1.3 The Communication Network

A robust communication network is required for successful interaction through exchange of messages by a team of mobile robots deployed to cooperatively perform a particular task. The absence of a fixed network infrastructure defeats the use of traditional wire-based communication strategies as well as an 802.11-based wireless network that relies on the existence of an access point. An ad hoc wireless network is therefore most practical for this purpose.

A wireless communication network that utilizes the ad hoc mode of the 802.11 standard could suffice for such a team of mobile robots. Robots operating on batteries, however, would run out of power sooner as they send and receive messages through their 802.11-based wireless interfaces. A communication device, such as a mote, that requires less energy could be used. This could offer the advantage of increasing the operating duration of a robot

This thesis presents a robust wireless communication solution for mobile robots using motes. Motes, sometimes referred to as smart dust, are small, low-cost, low-power computing devices equipped with wireless communication capability that uses Radio Frequency (RF). Motes have been applied widely in wireless sensing networks and are typically connected to sensors and used to gather information about their environment. Communication in a mote network is inherently unreliable due to message loss, exposed to attacks, and supports very low bandwidth. Additional mechanisms are therefore required in order to achieve robust communication.

Multi-hop routing must be used to overcome short signal transmission range. The ability of a mobile robot to determine its present location can be exploited in building an appropriate routing protocol. When present, information about a mobile robot’s future location can aid further the routing process. To guarantee message delivery, a transport protocol is necessary. Optimal packet sizes should be chosen for best network throughput. To protect the wireless network from attacks, an efficient security protocol can be used.

1.4 Communication in a Mote Network

Motes can be used to provide an ad hoc communication network for exchange of mission critical messages by a team of mobile robots deployed to cooperative perform a task. A mote network is however inherently unreliable due to the possibility of packet loss. Motes communicate using Radio Frequency (RF) which is a broadcast medium. This exposes the network to attacks, some of which could disrupt communication. Additional mechanisms are therefore necessary to ensure packet delivery, mote authenticity, and confidentiality in a mote network.

1.4.1 Causes of packet loss in a mote network

In a mote network, packet loss is inevitable. A packet sent by one mote to another may not arrive due to link-layer contention, signal attenuation, channel errors, or congestion.

1.4.1.1 Link-layer Contention

Neighboring motes contend for the shared wireless channel before transmitting. Key problems that arise due to channel contention include: hidden terminal problem, exposed terminal problem, and unfairness [1].

Hidden Terminal problem

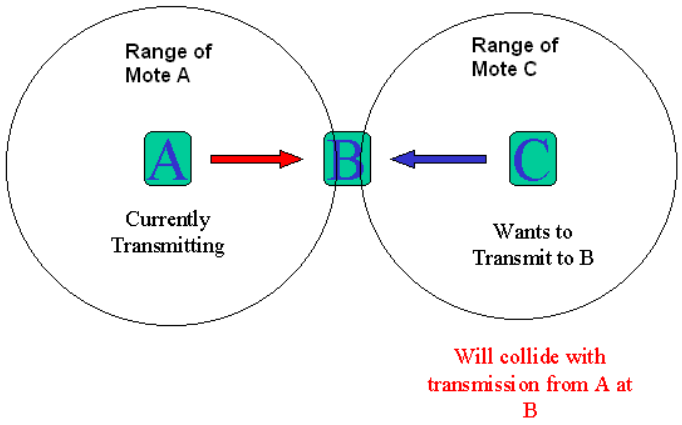

Figure 1.1 A hidden terminal problem is within the interfering range of the intended receiver, but outside the sensing range of the transmitter.

A hidden terminal C is within the interfering range of the intended receiver B, but outside the sensing range of the transmitter A. Collision from the hidden terminal C may hinder correct receipt of the packet by the receiver B.

Exposed Terminal problem

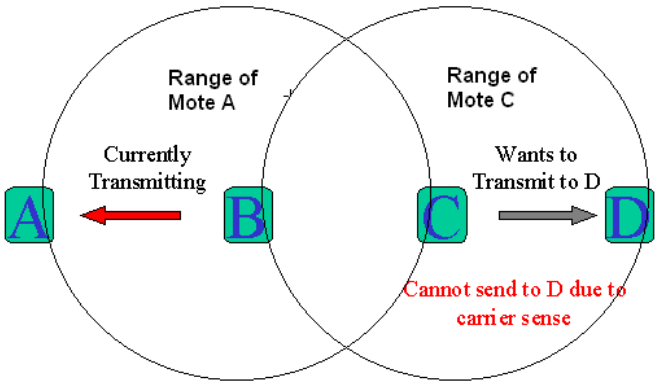

Figure 1.2 An exposed terminal is one that is within the sensing range of the transmitter, but outside the interfering range of the receiver.

In this case, terminal B is waiting to transmit to mote A. So terminal B listens to the shared channel and senses another terminal C, the exposed terminal, also listening to the channel. Even though the transmission of terminal C may not interfere with reception at terminal A, terminal B cannot start transmitting because it senses a busy medium.

Unfairness

Unfairness may occur depending on the Medium Access Control (MAC) protocol in use. A MAC protocol attempts to prevent collisions, which can occur when multiple motes within each other transmission range transmit at the same time, by allowing a mote to first listen to the channel for a clear signal before transmitting. If the channel is not clear, the listening mote will back-off from transmitting. For example, a MAC protocol that utilizes the binary exponential back-off scheme will always favor the latest successful transmitter.

1.4.1.2 Signal attenuation

The strength of a signal from a mote reduces as the signal travels. The longer the distance a signal travels, the weaker it gets. This leads to corruption, and eventual loss, of a packet.

1.4.1.6 Channel error

Channel errors in a mote network can be generated by transmission channel noise due to existence of other networks or devices in the environment. For example, packet error rates in a mote network increase under 802.11b interference [27].

1.4.1.7 Congestion

Congestion occurs when a link or node is carrying so much data that its quality deteriorates. In a mote network that has limited bandwidth, sending large messages can lead to network overload. Even though sufficiently large buffers can accommodate enormous amount of data, when this data has to be transmitted, it could lead to collisions or even denial of service - a security threat discussed in the section that follows.

1.4.2 Security threats

Motes are vulnerable to attacks due to the fact that they use RF which is a broadcast medium. Some of the threats that a mote network is exposed to include: snooping, unauthorized modification, service delay, and denial of service.

1.4.2.1 Snooping

Unauthorized interception of information, commonly known as snooping, can occur if a mote is present within range of other motes. It is not possible to prevent snooping because motes communicate via radio frequency. Any mote within range of a radio signal from a mote, and listening on the same frequency of the robot mote is capable of eavesdropping on the communication between robot motes.

Reasoning messages between robots can reveal detailed information about the workings of the robot teams. Part of this information includes: assignments to particular robots to search designated areas, re-assignments to an area where there is evidence of suspicious devices, other robot re-organization information. A listening mote can gain, through spoofing, detailed knowledge about the workings of the robot teams and team mission goals. This, however, does not necessarily prevent the intended destination robot motes from receiving the reasoning messages.

1.4.2.2 Unauthorized modification

A more potentially damaging threat is unauthorized change of information going across the network. As we have realized, it is possible for a listening mote to receive the same information intended for a robot mote in the network. In the case where a receiving robot mote is within communication range of a sending robot mote, the receiver is able to receive information directly from the sender.

However, in the likely case that the receiver is not within direct communication range, such that information from the sender has to be routed to the receiver by an intermediate mote, the receiver relies on the routing mote to deliver the information. Since we are dealing with a purely broadcast medium, a listening mote within direct range of the receiver and the sender could potentially pose as a router. If this mote is able to provide a shortest path or least cost route to the receiver, routed information from the mote could get to the receiver quicker than information could leave a legitimate mote providing routing service for the sender.

The listening mote could potentially modify the information it receives from the sender and further route it to the intended destination revealing no information about itself. This information could be such that the receiver relies on it to determine what action to take. The receiver could also be relying on the correctness of the information which may need to be released to another robot. Threats such as interruption or prevention of correct operation of the robot or unauthorized control of part of the robot system could arise.

A listening mote can impersonate a mote existing in the network, or a non-existent robot mote. By sending a message with information indicating that the message originates from a known mote, a mote could receive an assignment to perform some operation. It now becomes part of the team of robots trusted to perform certain tasks and to produce and possibly communicate results. This can hurt team performance. The listening mote can easily mislead the rest of the team especially by posing as the leader of the team, capable of making assignments and even disabling other robot motes.

1.4.2.3 Service delay

It is possible for an attacking mote to temporarily inhibit services within the network by delaying message delivery or performance of some other task. Due to routing requirements in the network, for example, a robot mote can be selected to provide a route to another robot mote not within range of a sender of information. If a listening mote can masquerade as part of the team and position itself conveniently such that it is a router candidate, the mote can be chosen to propagate signals to the intended final receiver of the message. The listening mote, on receiving the information, and knowing its responsibility to route the information to an robot mote, can hold on to the message longer than it should with the sole purpose of delaying message delivery. This could further result in delayed execution of some tasks.

1.4.2.4 Denial of service

Other than a temporary delay in service delivery, a long-term service delay popularly referred to as "denial of service" can also occur. A listening mote can prevent the leading robot from sending out information regarding team assignments. The listening mote can achieve this by, for example, sending too many messages to the lead robot to overload it in such a way as to prevent it from being able to perform other tasks. This can be even more damaging in a network environment where the robot motes give a high priority to message processing which could be very critical in the assignment of tasks and achievement of team goals.

1.4.3 Examples of specific attacks

1.4.3.1 Physical node capture

The mobile robots in this application are designed to be deployed into open areas that may or may not be monitored for physical safety. The robots are semi-autonomous, and not tele-operated even though they do have the capability to allow for total human control. One of the likely areas for deployment includes battle zones. A node can be destroyed or picked up and taken away by an enemy. Another possible deployment location is a disaster area where all sorts of rescue operations are being carried out. It

could be due to, for example, fire, earthquake, or some other phenomena. A node can be destroyed in the process of contributing to rescue efforts.

1.4.3.2 Flooding attack

Network flooding is a well know problem [14], and could easily occur in this robot network. It can be achieved through exploits, some of which are available at Metasploit (www.metasploit.com). An attacking node could transmit too many messages and consequently overload the network which already has limited bandwidth. This is a form of Denial of service attack [13].

1.4.3.3 Network protocol invasion

This kind of attack requires some knowledge about the workings of the network communication protocol. In the current implementation, which uses sliding window protocol with selective repeats, a receiving node sends a block acknowledgement to a sending node [16]. The acknowledgement contains a bitmap which provides information about which packets have already been received and which ones have not yet been received. Also included in the acknowledgement is the sequence number of the first expected packet. An attacking node can fake this acknowledgement, for example, by providing a sequence number that is within the window and a bitmap which would cause the sender to retransmit all of the packets previously sent. If an attacking node can send several of these in a short period of time, it can introduce congestion or cause collisions to occur, and thereby disrupting the operation of the protocol.

The workings of the network protocol could be disrupted also if the attacking node sends packets which a receiver of the packet would use to determine what action to take. A reset packet, which I defined as one to send to a node to request it resets the information it is keeping about another node, is an example. If a false reset packet is received by a node, it will reset state information about other nodes and this can cause other nodes to get confused about information they receive from the compromised node. For example, resetting the sequence number information could cause a node's data packets to be rejected because they might no longer fall within the window expected by a receiver.

1.4.3.4 Denial of service using high-energy signals

A special form of denial of service attack in this robot network can be achieved if an adversary broadcasts high-energy signal [13]. A powerful enough transmission could jam the entire mote network. Such an attack can also occur at the medium access control (MAC) layer. By continuously requesting channel access with a request-to-send signal, an attacking node can prohibit other nodes within its immediate range from transmitting, leading to denial of service.

1.4.3.5 Electronic node capture

Unlike physical node capture, in electronic node capture, an adversary does not need to take away a mote [or robot]. The adversary can instead gain control of the mote by injecting code into it. The motes we decided to use for this robot network allow for in-network programming. It is possible to reprogram, thus overwriting the program via radio. Malicious code can be ported onto the mote in this manner. Regardless of what the code has been written to accomplish, it is very likely that it will disrupt the communication capability of the robot. The communication capability happens to be the most important capability for the robots, especially when a robot is within communication range of at least one other robot.

1.4.3.6 Information theft

A mote is primarily a broadcast medium. It transmits signals at a particular frequency. Any mote within communication range of a transmitting mote and listening on the same frequency as the transmitting mote can receive and process the signal. I will note though that for a listening mote to form good packets out of the signals it picks up, it has to have detailed knowledge about the packet format. The current implementation uses a format whose specification is publicly available. Therefore, it is possible for an adversary to introduce a mote in the robot network that will be able to decipher the signals and reform the messages transmitted over the network.

1.5 Related Work

Significant research has been done in the area of reliable communication in Mobile Ad Hoc Networks (MANETs) [1,3,9]. Chonggang W. et. al [8] surveys transport control protocol for wireless sensor networks (WSNs). Energy-conservation, congestion control reliable data dissemination, security, and management of WSNs are problems to be overcome in WSNs. Attention has however turned to transport control protocols, which are important for data dissemination and energy-conservation for WSNs. Ahmad Al Hanbali et. al [9] discusses the challenges of implementing TCP over MANETs. The performance of TCP degrades in MANETs environment because TCP has been optimized for running over wired networks. Christian L. et. al, [3] surveys the key problem of congestion control in mobile ad-hoc networks. A summary of common design patterns in congestion control mechanisms for MANETs is available at the end of the survey. Many of the papers have pointed out that TCP performance in MANETs is not optimal.

Still, design, implementation, and testing of TCP over MANETs have focused mainly on Wireless Sensor Networks which has unique characteristics in terms of communication patterns. Techniques developed as a result of this research therefore may not suit a wireless network of a team of mobile robots with high-reliability communication requirements.

Security in Wireless Sensor Networks has also been studied [13,14]. Many of the techniques used in securing wired networks are not feasible for WSNs primarily due to the fact that devices used in WSNs suffer from lack of processing, memory, and battery power.

TinySec [11] has however emerged as a promising approach to securing wireless networks of resource constrained devices like motes. TinySec is the first fully-implemented link layer security architecture for wireless sensor networks. It guarantees message authenticity, integrity, and confidentiality. TinySec uses Message Authentication Codes (MACs) and supports authenticated encryption and authentication only security options. Unfortunately, implementations of TinySec are available only for Mica, Mica2, and Mica2Dot platforms. There is no known implementation of TinySec for MicaZ platform.

CHAPTER 2 - The Network Protocol

The wireless network protocol for use by the team of mobile robots is distributed between the robot and the mote. Part of the protocol runs on the robot and the other part on the mote. Its architecture follows a layered model similar to OSI, but not exactly.

On the mote, only three layers are present. The mote does not keep state information about logical links to other motes in its transmission range. Thus, it does not need a part - Logical Link Control (LLC) - of the Data Link layer. Furthermore, the application on the mote interacts directly with the Medium Access Control (MAC) layer when sending packets over radio and does not offer network, transport, session, or presentation services as in OSI.

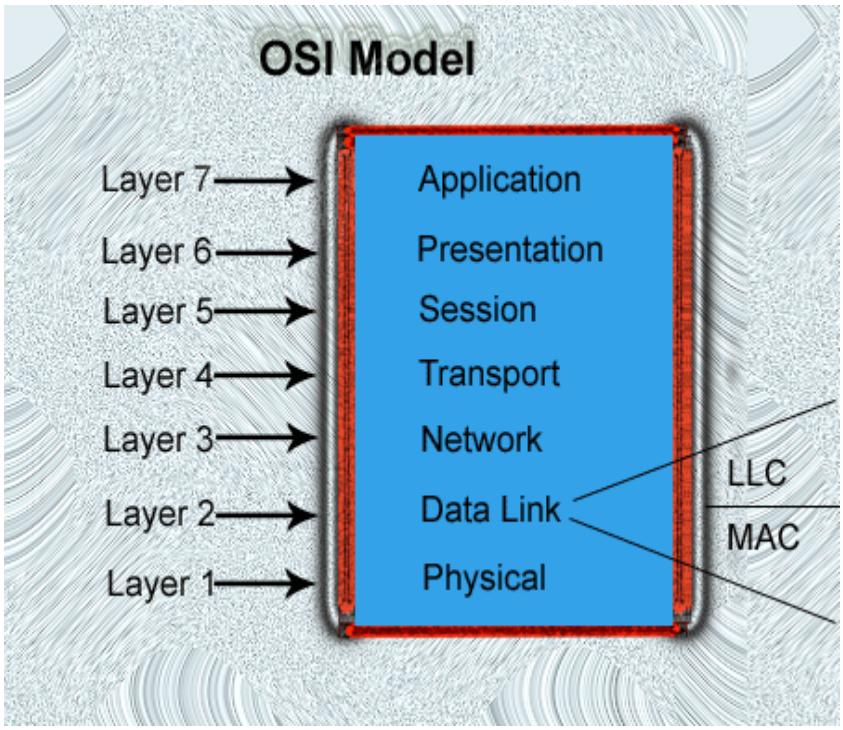

Figure 2.1 Open Systems Interconnectivity Model

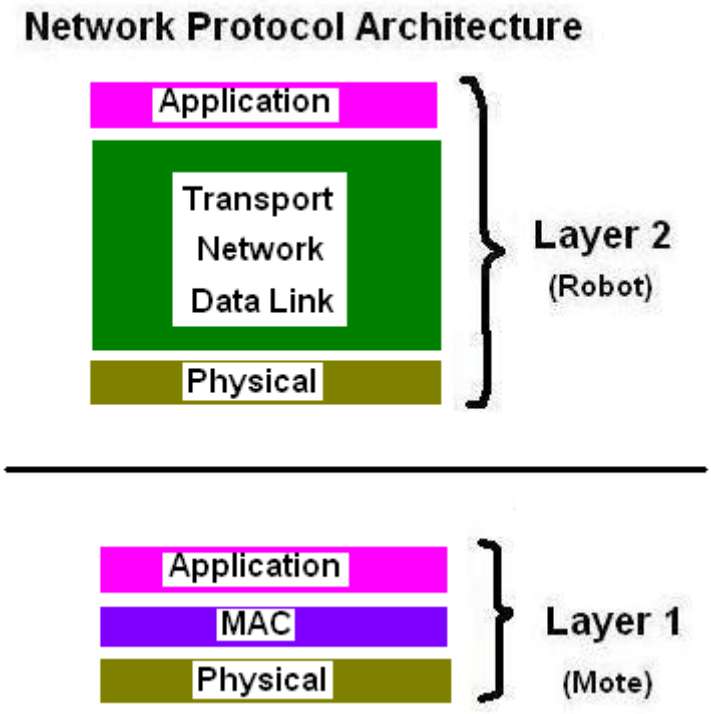

Figure 2.2 Network Protocol Architecture showing Layer 1 and Layer 2

The protocol portion on the robot uses a cross-layered architecture. Here, three layers (transport, network, and data link - LLC) are bundled into one. Information between the layers is shared. There is no clear separation of these layers. The robot does not perform any Medium Access Control because it is connected to a mote by a non-shared dedicated physical medium. An application, like the one described in chapter 1, can be written on the robot to use the protocol.

2.1 The Hardware

The hardware comprises of robots, USB cables, motes, and programming boards.

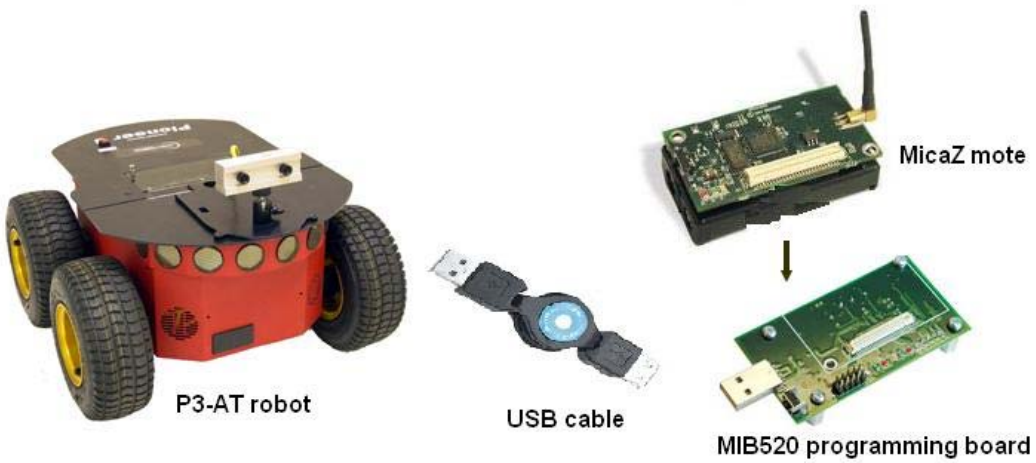

Figure 2.3 Hardware setup: A MicaZ mote, attached to a MIB520 programming board is connected to each P3-AT robot using a USB cable.

2.1.1 The robots

Each robot is a Pioneer P3-AT robot. The Pioneer P3-AT (or just P3-AT) robot is a high performance, highly versatile, all-terrain robot used for research and a variety of prototyping applications [23]. It is equipped with an on-board computer and runs on three batteries. The robot can run on fully charged batteries for 3-6 hours. However, experimental results indicate the battery voltage depleting after 30-45 minutes of continuous operation with fully mobility. The robot provides both serial and USB interfaces. On flat ground, it can move up to speeds of 700cm per second. The particular robots used in experiments associated with this thesis contain: 1.6Ghz processor, 512MB RAM, two USB interfaces, and other hardware components. The Application Programming Interface (API) used is the Advanced Robotics Interface for Applications (ARIA) 2.5.1 running on Windows XP Professional operating system.

2.1.2 The motes

Each mote is a MicaZ [19]. The MicaZ mote provides radio frequency (RF) communication through transceivers. It connects to a MIB520 board through a UART connector. The MIB520 board comes with a USB interface through which it can be connected to another device; in this case, a P3-AT robot. When attached to a MIB520 programming board, the mote should not be set to use its batteries. The board, which is powered by the device it is connected to, automatically powers the mote. Setting the mote [using an on/off switch on the opposite side from the antenna] in this case can damage the mote.

Figure 2.4 MicaZ mote

Figure 2.5 The MicaZ specification (August 2008)

Processor/Radio Board	**MPR2400CA**	**Remarks**
Processor Performance		
Program Flash Memory	128K bytes	
Measurement (Serial) flash	512K bytes	> 100,000 Measurements
Configuration EEPROM	4K bytes	
Serial Communications	UART	0-3V transmission levels
Analog to Digital Converter	10 bit ADC	8 channel, 0-3V input
Other Interfaces	Digital I/O, 12C, SPI	
Current Draw	8 mA	Active mode
	< 15 µA	Sleep mode
RF Transceiver		
Frequency band	2400 MHz to 2483.5 MHz	ISM band, programmable in 1MHz steps
Transmit (TX) data rate	250 kbs	
RF power	-24 dBm to 0 dBm	
Receive Sensitivity	-90 dBm (min), -94 dBm (typ)	
Adjacent channel rejection	47 dB	+5 MHz channel spacing
	38 dB	-5 MHz channel spacing
Outdoor range	75m to 100m	½ wave dipole antenna, LOS
Indoor Range	20m to 30m	½ wave dipole antenna
Current Draw	19.7 mA	Receive mode
	11 mA	TX, -10 dBm
	14 mA	TX, -5 dBm
	17.4 mA	TX, 0 dBm
	20 µA	Idle mode, voltage regulator on
	1 µA	Sleep mode, voltage regulator off
Electromechanical		
Battery	2X AA batteries	Attached pack
External Power	2.7V – 3.3 V	Molex connector provided
User interface	3 LEDs	Red, green and yellow
Size (in)	2.25 x 1.25 x 0.25	Excluding battery pack
(mm)	58 x 32 x 7	Excluding battery pack
Weight (oz , grams)	0.7 , 18	Excluding batteries
Expansion Connector	51-pin	All major I/O signals

2.2 The Software

2.2.1 Layer 1 (on the mote)

A micaz mote comes installed with the TinyOS operating system. On it runs the entire layer 1 protocol stack. For the motes to be able to communication with each other via radio, they are programmed to use share a channel.

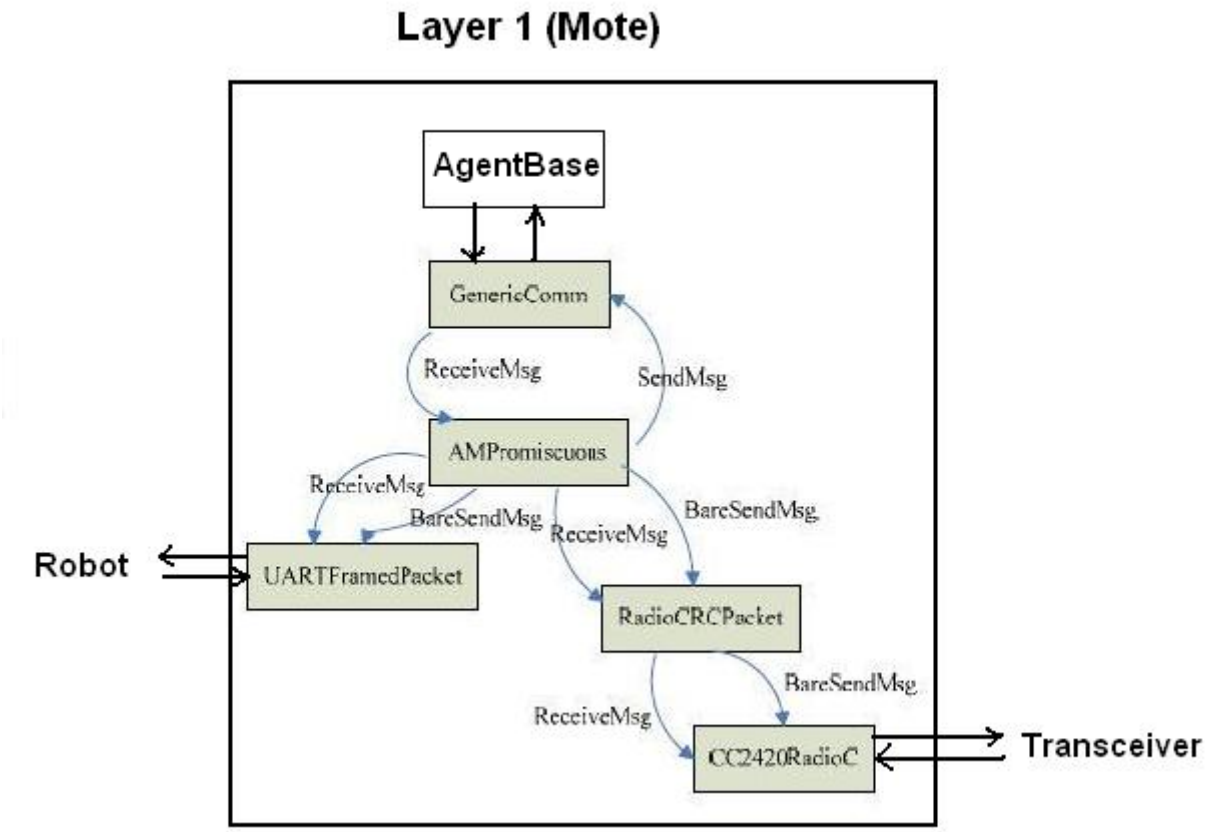

Figure 2.6 Layer 1 of the Network Protocol on the mote

2.2.1.1 Physical layer

The entire physical layer is implemented in the mote hardware. It controls modulation (TX) and demodulation (RX) of signal when sending and receiving packets respectively.

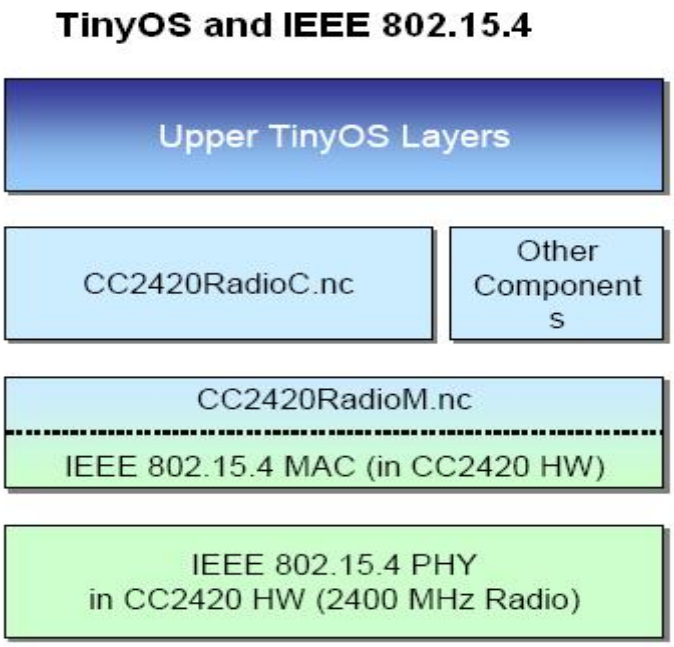

Figure 2.7 TinyOS and IEEE 802.15.4

2.2.1.2 MAC layer

Part of the Medium Access Control (MAC) layer is implemented in hardware. The software portion of this layer is open source and can be modified as desired, compiled and ported onto the mote. The Medium Access Control (MAC) layer works with the CC2420 radio stack [24] and is based on IEEE 802.15.4 standard.

CC2420 offers 16 channels based on IEEE 802.15.4, even though, in this case, all motes share the same channel. It uses O-QPSK(offset quadraturephase shift keying) with half sine pulse shaping. MicaZ mote uses the Back-off MAC (B-MAC) protocol [10] at this layer to control access to the shared RF channel.

When sending a packet, the MAC layer performs a clear channel assessment with an initial random back-off of 1-15 jiffys (66686 ticks per jiffy). If the channel is busy, a back-off is triggered. Otherwise, the packet is transmitted. Optionally, the MAC layer can wait for an acknowledgement when the packet is received. A “send done” signal is issued to the mote application.

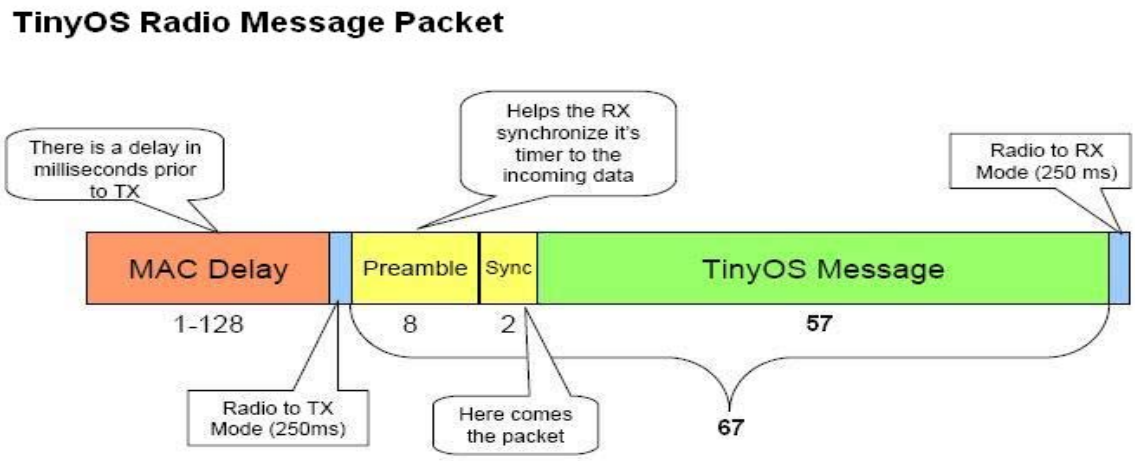

Figure 2.8 TinyOS Radio Message Packet

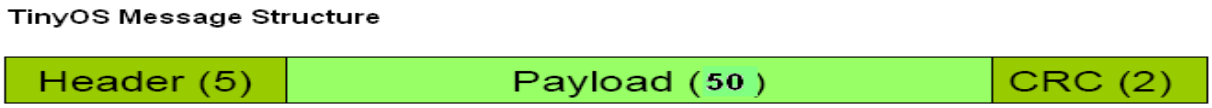

Figure 2.9 TinyOS Message Structure

When a MAC header is detected, a radio interrupt is issued to signal an incoming packet. Packets are demodulated (RX) on a first in first out (FIFO) fashion. The mote application is then signaled with the packet.

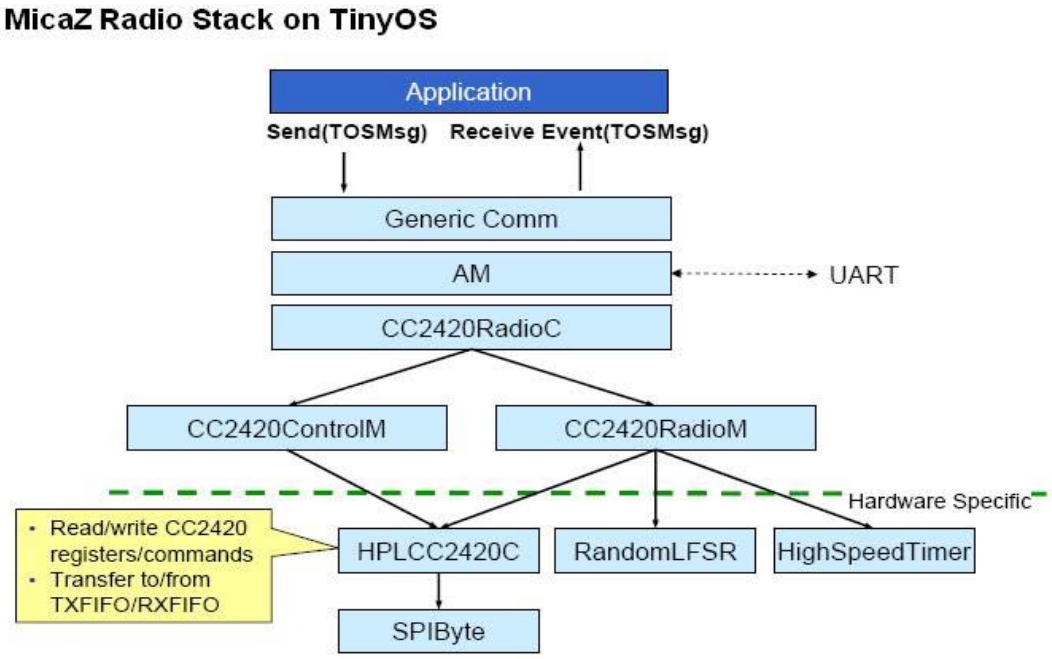

Figure 2.10 MicaZ Radio Stack on TinyOS

2.2.1.3 Application layer (on mote)

The nesC application at this layer performs two functions: processes application-level command packets and forwards all other packets over radio.

A framed packet from the robot is sent to the MIB520 programming board through an RS-232 serial interface. This may be a serial cable connected to a serial port or a USB cable on a USB port which the operating system [on the robot] associates with a virtual serial port. In both cases, the framed packet is a sequence of bytes serialized to the programming board. The mote receives a packet from the board through a universal asynchronous receiver/transmitter (UART) connector. The packet header is checked for corruption and a corrupt free packet is pushed up the protocol stack to an application on the mote.

Packets exchanged through the serial interface are framed according to the RFC 1662 standard where a SYNC byte is used at the beginning and end of the frame.

Packets received over radio are also checked for corruption before being passed up to the application.

Commands are not broadcasted over the radio. Depending on the command, a packet may be sent back through the UART to the robot. All other packets are immediately queued for [broadcast] transmission over radio. All incoming packets from radio are forwarded to the UART for transmission to the robot.

Code structure for the AgentBase application on the mote

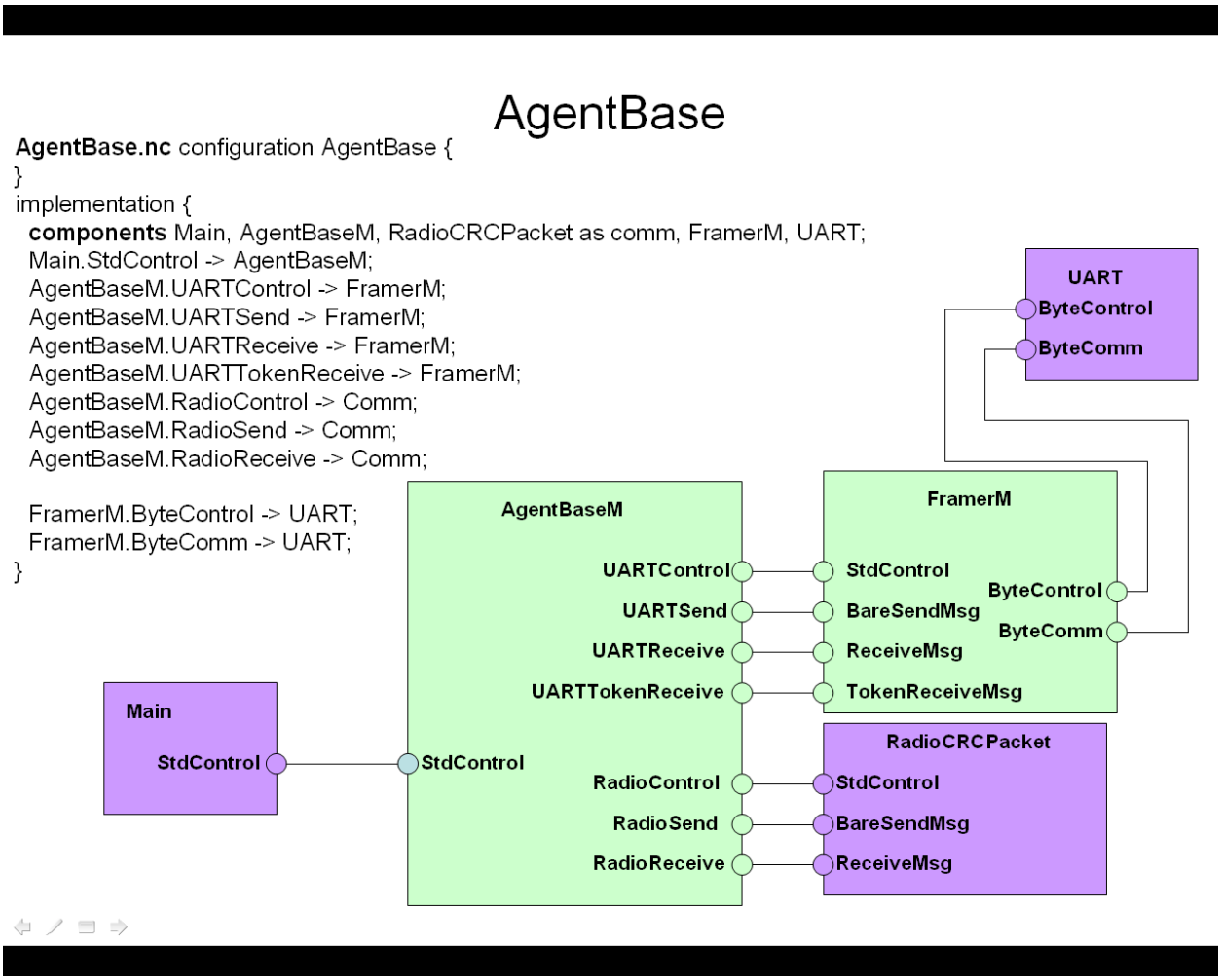

Figure 2.11 Code structure of protocol on mote

2.2.2 Layer 2 (on robot)

The protocol portion on the robot uses a cross-layered architecture. Here, three layers (transport, network, and data link - LLC) are bundled into one - TND. Information between the layers is shared. There is no clear separation of these layers. The robot does not perform any Medium Access Control because it is connected to a mote by a non-shared dedicated physical medium.

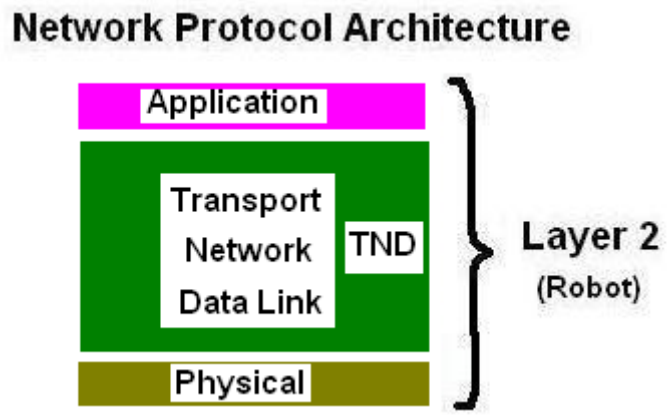

Figure 2.12 Wireless Network Protocol Architecture

Traditional-wired networks commonly apply the well known OSI layer model. In this model, the protocol is divided into layers.

In OSI, the transport layer is provides end-to-end reliability and therefore guarantees message delivery. The network layer, just below the transport layer, provides network functions including routing of individual network packets. The data link layer, right below the network layer, performs functions similar to the transport layer. It provides point-to-point reliability of packets. The physical layer sits below the data link layer; it is responsible for actual transmission or reception of packets over the network physical cable medium.

On the robot, a modified version of this model seems more appropriate. The layers remain unchanged and in the same order.

2.2.2.1 Physical layer

The robot interacts with the mote's MIB520 programming board through a serial interface based on RS-232. At this layer bytes of information are streamed over the physical medium. Even though the robot interfaces with the MIB520 board using a USB cable, the operating system creates and communications with the board through virtual serial ports.

2.2.2.2 Transport, Network, Data link (TND) layer

This forms the heart of the protocol portion on the robot. It provides the following services: out-going message fragmentation into data packets, incoming data packet re-assembly into messages, end-to-end message reliability, message re-ordering, packet routing, and packet error-checking.

Message fragmentation and re-assembly

All data sent over the mote network is encapsulated in a TinyOS message packet. The TinyOS packet by default supports a maximum payload of 29, an arbitrary value, which seemed sufficient for the kind of applications that utilize motes, set by the developers of the low-level TinyOS programs. It contains a header of 5 bytes plus a 16-bit CRC value. The maximum packet size supported is 127bytes.

However, experiments with a packet size of 100 bytes suffered too much corruption to be detected using the 16-bit CRC value. Using the maximum packet size would therefore not yield good results. The implementation used for this thesis assumes a maximum TinyOS packet size of 57 bytes, with 50 bytes for the payload. The TinyOS packet header contains an 8-bit value representing the actual size of the payload. Only this portion of the payload is transmitted and therefore providing a variable-size packet behavior.

To support messages of larger sizes, the TND layer performs message fragmentation. Out-going messages are fragmented into small enough data packets. The data packets are 50 bytes in size – 20-byte header and 30-byte payload. Effectively, an application-level message is fragmented into 30-byte packets. The CRC is computed over the entire TinyOS 50-byte payload.

Incoming data packets, from the mote, are re-assembled back into messages. The order in which the packets are received from the mote is arbitrary. The packets are re-ordered when necessary to ensure correct re-assembly back into an application-level message.

Since there is no limit on the size of an application message, there is also no limit on the number of out-going or incoming data packets.

Packet routing

The TND layer handles routing of packets. According to specification, the MicaZ mote has a maximum transmission range of 100m. Outdoor experiments have shown the effective range to be 6.3-40m (about 19-122ft) depending on the elevation of the mote from the ground (0-3ft). Interference from other signals also contributes to this reduced range. For the robots to be able to communicate over a wider (more than the effective transmission range) area, the robots must act as relaying stations for packets. Routing is required. The TND layer uses a geographical routing algorithm based on shortest path. The layer contains location information of each robot in the network. When a packet needs to be routed towards its destination, a neighboring robot, closest to the destination is chosen as the next hop for the packet. Neighboring robots are ones within the effective transmission range of the sending robot. This routing continues until the packet arrives at its destination. When a robot sending packets moves out of range of all other robots, the packets cannot be sent. The current implementation retries indefinitely. This implementation can be improved by incorporating a timeout.

End-to-end message reliability

Packet loss in a mote network is inevitable. The TND layer provides a mechanism, based on TCP's sliding window protocol with selective repeats [16], to guarantee packet delivery. There is both a sender and receiver timer which fire at fixed intervals of 3 seconds. This interval can be modified. The window has a size of 40, allowing up to 40 packets to be sent out before it waits for an acknowledgment. The acknowledgement contains a bitmap which provides information about which packets have already been received and which ones have not yet been received. Also included in the acknowledgement is the sequence number of the first expected packet.

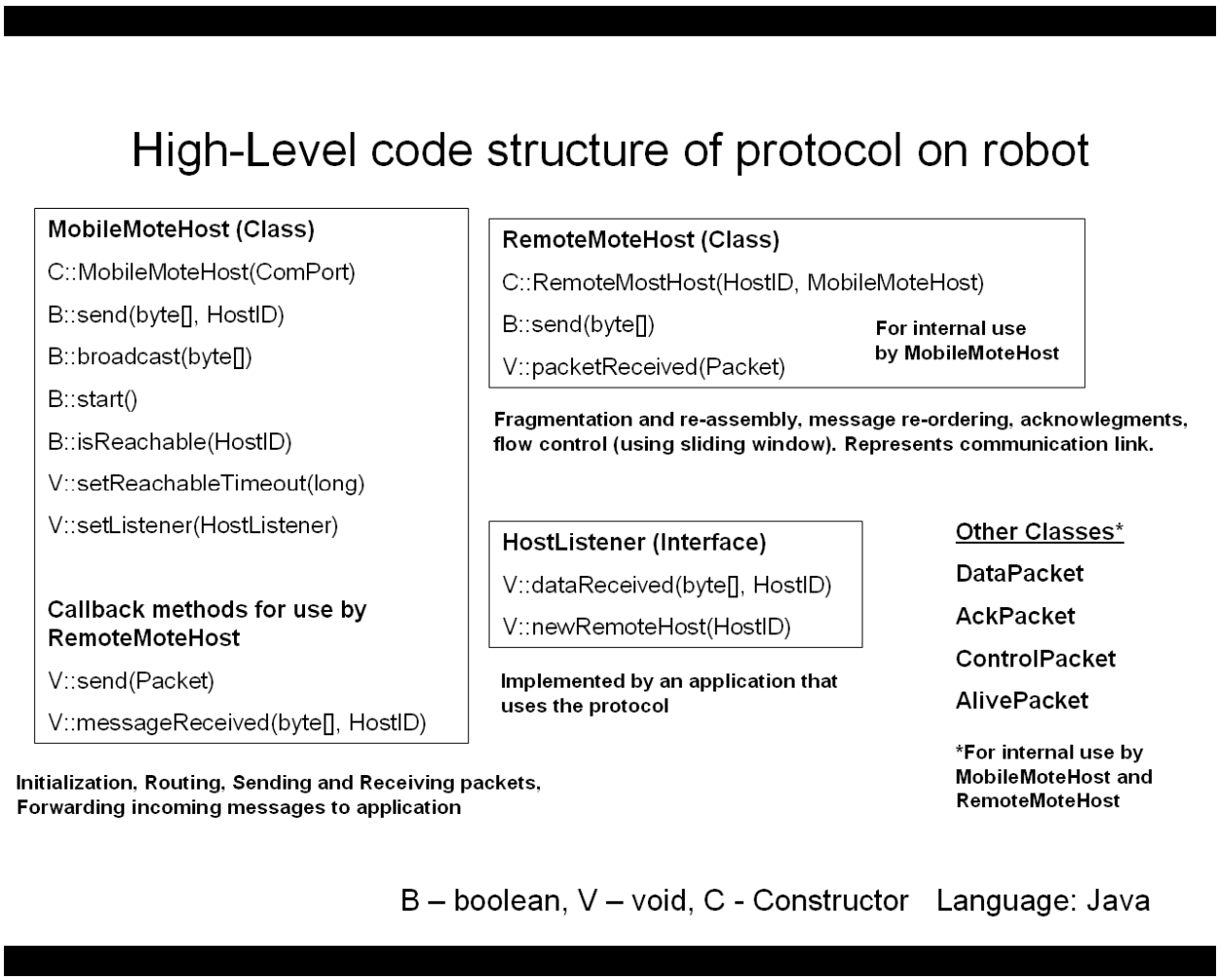

Figure 2.13 Code structure of protocol on robot

2.2.2.3 Application layer

The application layer represents the origin and final destination of all application-level messages. It receives corrupt-free messages from a single source in the order in which they were sent. If multiple sources send a message to a single mote, the order in which these messages are received is arbitrary.

2.3 Security – attack prevention, detection, and recovery mechanisms

The threats and attacks described in Chapter 1 all attempt to violate at least one of the three requirements or aspects of security: Confidentiality, Availability, and Integrity. Node capture, both electronic and physical, and network protocol invasion could lead to violation of all three

aspects. Denial of service as well as flooding, on the other hand, affects most, the availability of the system. Information theft violates confidentiality.

Rather than looking at specific remedies for each of these example attacks, I will discuss general mechanisms for achieving security and mention the specific aspect of security that the mechanism applies.

2.3.1 Configuration Management

In this system of multi-robots, configuration management could include ensuring that the motes are all running the same version of software. The motes are typically loaded with a runtime version of TinyOS operating system. In addition to the operating system, one needs to install the program that would run on the mote as soon as it is supplied with power. Ensuring that all motes are running up to date programs minimize the chance of a node capture incase of knowledge by the adversary of prior versions of the mote software. This can contribute towards availability of the mote for communication tasks.

Several tools, such as Moteworks, are available to assist a developer of such programs to port the software on to the motes. Version control software, like CVS or Subversion, can help ensure a development team has access to most recent versions of programs.

2.3.2 Encryption

In a robot network using motes as the communication medium, encryption can provide a means to conceal information from unauthorized receivers. This can be useful especially because the mote network is a broadcast medium, which as we understand, means that any mote within range of another transmitting mote can effortlessly receive transmitted data, as long as the two motes are operating in the same radio frequency. In a network that is physically exposed, it is easy for an adversary to introduce motes to listen to the network traffic. Encryption will ensure that only authorized motes can decipher the packets being transmitted over the network. This ensures confidentiality.

The strength of encryption, however, is limited due to the limited computing resources [19] available on the mote. A little encryption is obviously better than no encryption. Other factors could influence the need for more robust mechanisms to ensure data confidentiality.

Using cryptography entails a performance cost for extra computation that more often than not increases packet size [13]. How much more space is required to accommodate this could pose a restriction on the use of cryptographic techniques.

2.3.3 Authentication

Authentication, in this application, can be viewed as a mechanism designed to provide integrity. More specifically, we would like to be able to determine the trustworthiness of data or resources. As we have seen, it is possible for an adversary to masquerade as an authentic mote. It is also possible for an adversary to receive information in an unauthorized manner, modify it, and further retransmit it. When this information is received, and thought to be from a trusted source, it can cause damage if carefully designed to do so. Like the case of an acknowledgment that triggers resend of data packets which later hog the network by causing congestion and collisions which lead to packet loss and potentially further release of old data packets into the network.

2.3.4 Kerberos

Kerberos has been widely applied as a means of providing authentication services in a computer network [20]. Details of how it works have also been provided [20]. Even though it works well for computer networks, Kerberos may not be suitable for a mote network primarily due to the limited computing resources of such a network. There is also need to establish trust among the motes in the network for Kerberos to be option.

2.3.5 Public-key cryptography

Public-key cryptography is also not the most appropriate mechanism again due to limited computing resources. The bandwidth on a mote network is restricted at 250kbs [19]. RAM on the micaz mote is only 4K. This would not be sufficient to produce or check digital signature based on public and private keys.

2.3.6 One-time pass code

One-time pass codes [20] may suffice for initial transmissions. To provide authentication with every data packet or a block of data packets, a mote would have to generate multiple pass codes per communication session.

2.3.7 Location-Based Authentication

In the robot networks, the robots have capability to determine their geographical locations. As the move around, their locations change. However, from the teams' assignments, robots can know the approximate whereabouts of other robots. Therefore, any robot transmitting data could include their geographical information. Using robot knowledge about the work the sending robot was assigned and the area where this work was to be carried out, the receiving robot could approximate the expected geographical location. Heuristics can be applied to determine how far off this estimate is from the actual location values. Thresholds can then be used to determine if to accept or reject received information. It is important that the geographical information being transmitted be encrypted to ensure confidentiality. The possibility of replay attacks would necessitate the location-based authentication even in the case where the motes already use encryption/decryption keys for communication. This form of authentication relies on the assumption that the robots possess knowledge about robot teams' formation and assignments.

2.3.8 Secure software development

Routing implementation can be run through special checkers that try to analyze the security of software. One can use well known techniques for verifying correctness of software. Interfaces can be checked to determine if information within the router is flowing as specified [21].

One can use a programming language like Spark [22], which allows for annotations that help specify variable access and updates. Using this, an implementation of a routing protocol can be checked for security loop-holes.

Checking the routing protocol for correctness can ensure that even in the event of corrupt information, for example, invalid acknowledgment sequence numbers, that the routing device will not initiate an operation that disrupts communication over the mote network.

2.3.9 Battery-based intrusion detection

Monitoring the power levels in the motes can help indicate the possibility of an ongoing network attack [14]. By correlating attack activities with the mote power consumption patterns, we can tell whether the motes are using way more power than they should or whether they are using far less power than they should. Typically, the most power consuming operation for a mote is sending a message. Receiving a message does not consume as much power. Computations on the mote also require use power. A drained mote could indicate that the mote is transmitting too many messages, or possibly receiving and processing more messages than it typically is required to. This could be a sign of flooding, for example, or denial of service attack, or, equally possible, node capture.

2.3.10 Congestion control/detection

Congestion in the network can be a sign of too many packets in transit [16]. If the communication system is designed to monitor congestion in the network, thresholds could be used to determine the possibility of an ongoing attack. Most likely, if the congestion is due to an attack, it will be a flooding attack or denial of service.

2.3.11 Secure group management

Secure group management has been suggested as promising approach to decentralized intrusion-detection [13]. Network activities can be performed by a group of robots. The outcome of these activities can then be transmitted to the lead robot. This would require secure protocols for group management.

The low-speed, low bandwidth, ad-hoc, mobile network that is currently used as the primary communication medium for a team of semi-autonomous robots faces security challenges. Most of the attacks that can be launched on other traditional networks and even modern sensor networks can also be launched on this mote network. The network is susceptible to more specific attacks depending on how the communication protocol has been designed and implemented.

Due to limited computing resources, applying common techniques to ensuring confidentiality, availability, and integrity in the network may not be practical. Still, we can achieve some level of security using these techniques. How much Security is required depends on the threat model.

CHAPTER 3 - Experiments and Results

3.1 Experiment 1 – Packet Loss Rate

3.1.1 Description

Two robots (Robot1 and Robot2) are placed 46ft apart. Robot1 sends a burst of packets to Robot2. Robot2 takes note of how many packets, in one burst, are lost. Packet size is 30 bytes.

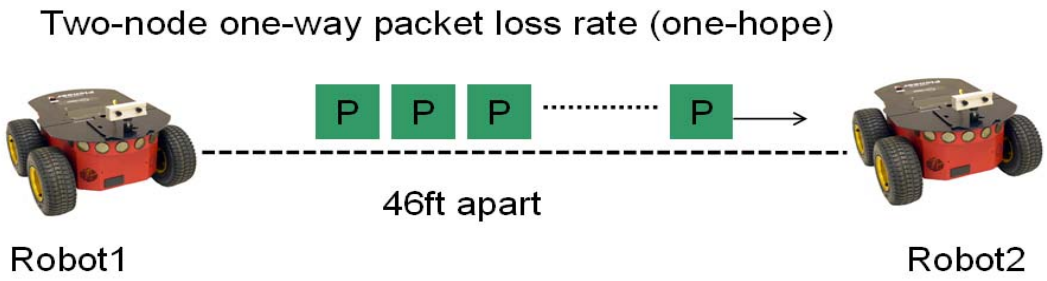

Figure 3.1 Experiment 1 - Setup

3.1.2 Results

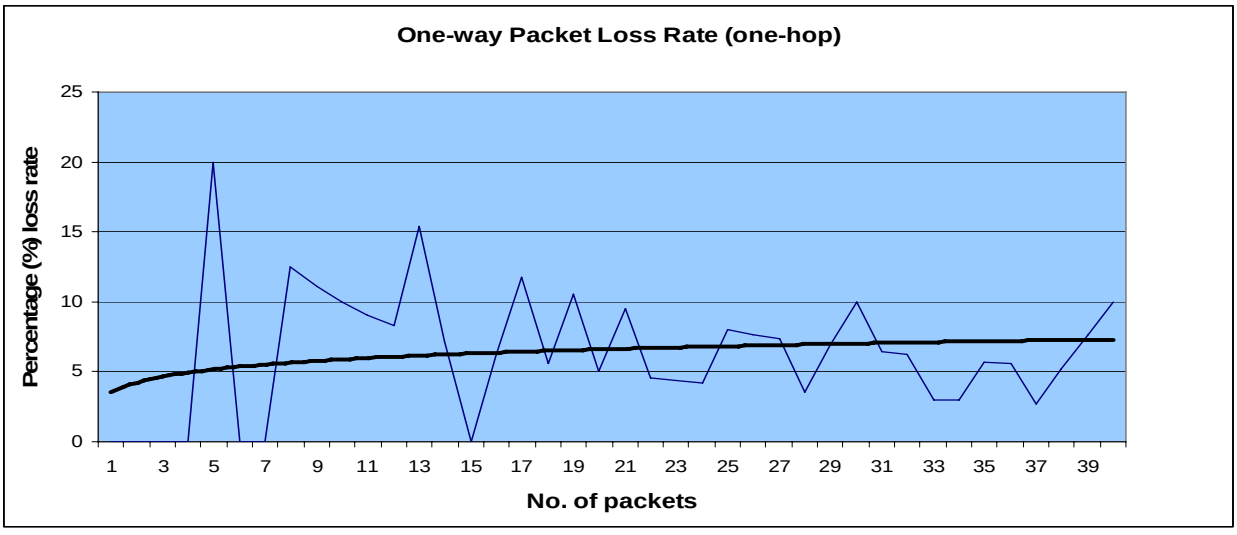

Figure 3.2 Experiment 1 – Graphed results

Since only one mote is sending packets, there is no link-layer contention. The distance is right at the circumference of the transmission range. The low loss-rate percentage could suggest channel error due to interference from the nearby 802.11 network or something else in the environment.

3.2 Experiment 2

3.2.1 Description

Two robots (Robot1 and Robot2), representing two network nodes are positioned outdoors 46ft (about 15.3m) apart and within communication range. The ad hoc wireless communication network using motes and the protocol described in chapter 2 is first initialized on both robots. Robot1 takes note of the current time (start time) then immediately sends a message of size 40 bytes to Robot2. On receiving the message, Robot2 sends that exact message back to Robot1. Robot1, again, takes note of the current time (end time) when it receives back the message it sent. The difference between the two times is computed ("end time" – "start time") and recorded.

Robot1 repeats the process adding 40 bytes each time. This continues until an arbitrary message size of 1,160 bytes. Packet size is 30 bytes.

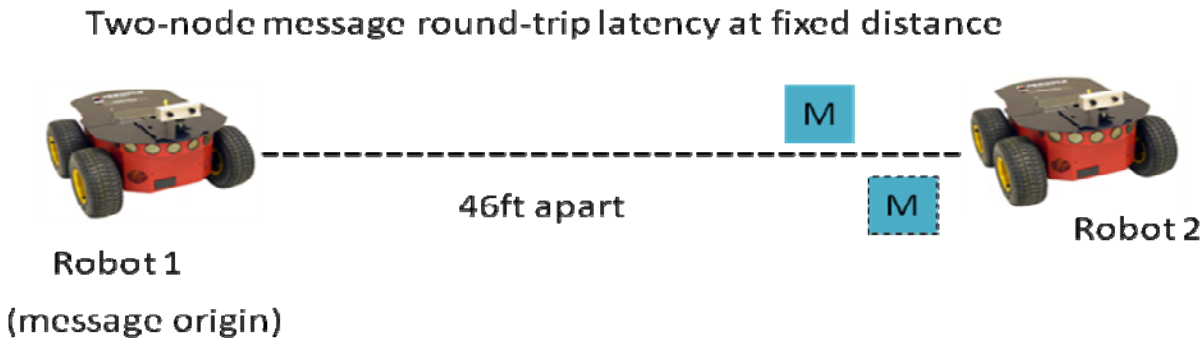

Figure 3.3 Experiment 2 - Setup

A 40-byte message takes a round-trip time of 1second. A message of size 400 bytes (ten times as big) takes about 2 seconds. Another message of size 800 bytes takes about 4-4.5 seconds. It takes 10 seconds to for a message of size 1160 bytes to be received back by robot 1.

3.2.2 Results

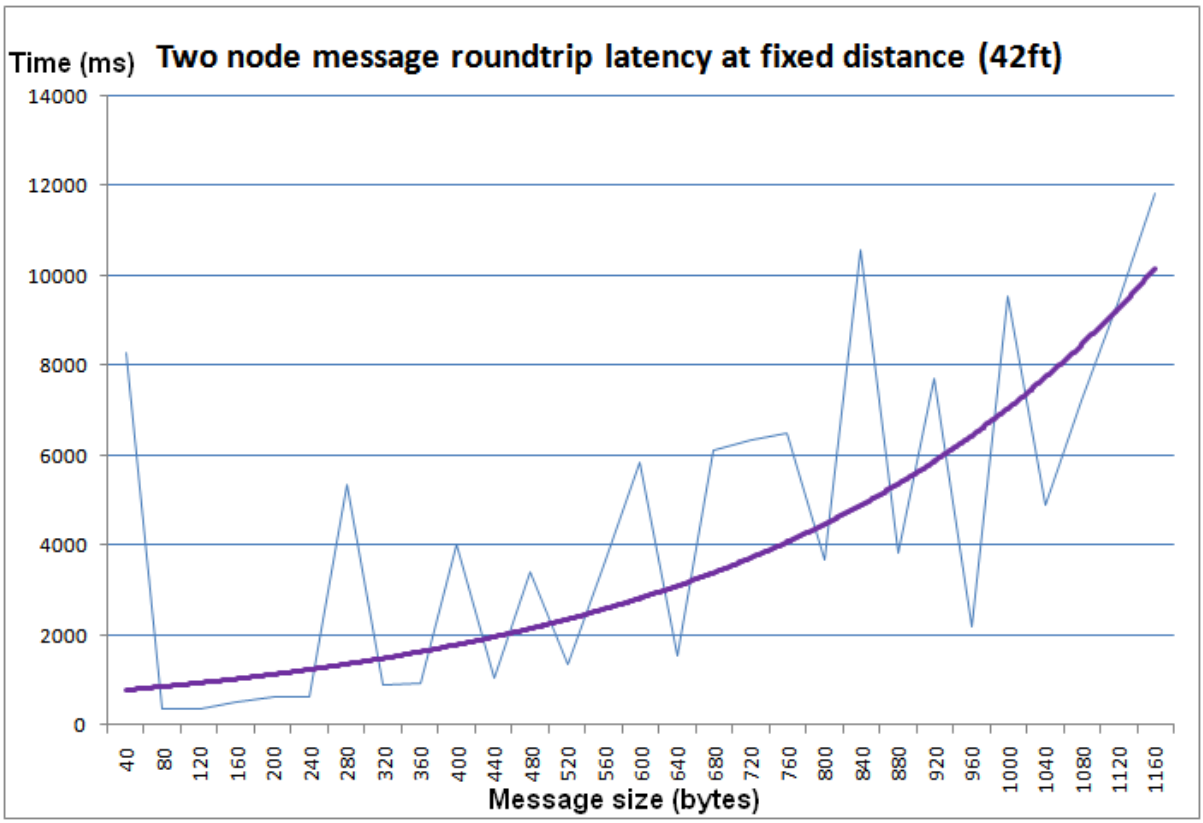

Figure 3.4 Experiment 2 – Graphed results (bytes)

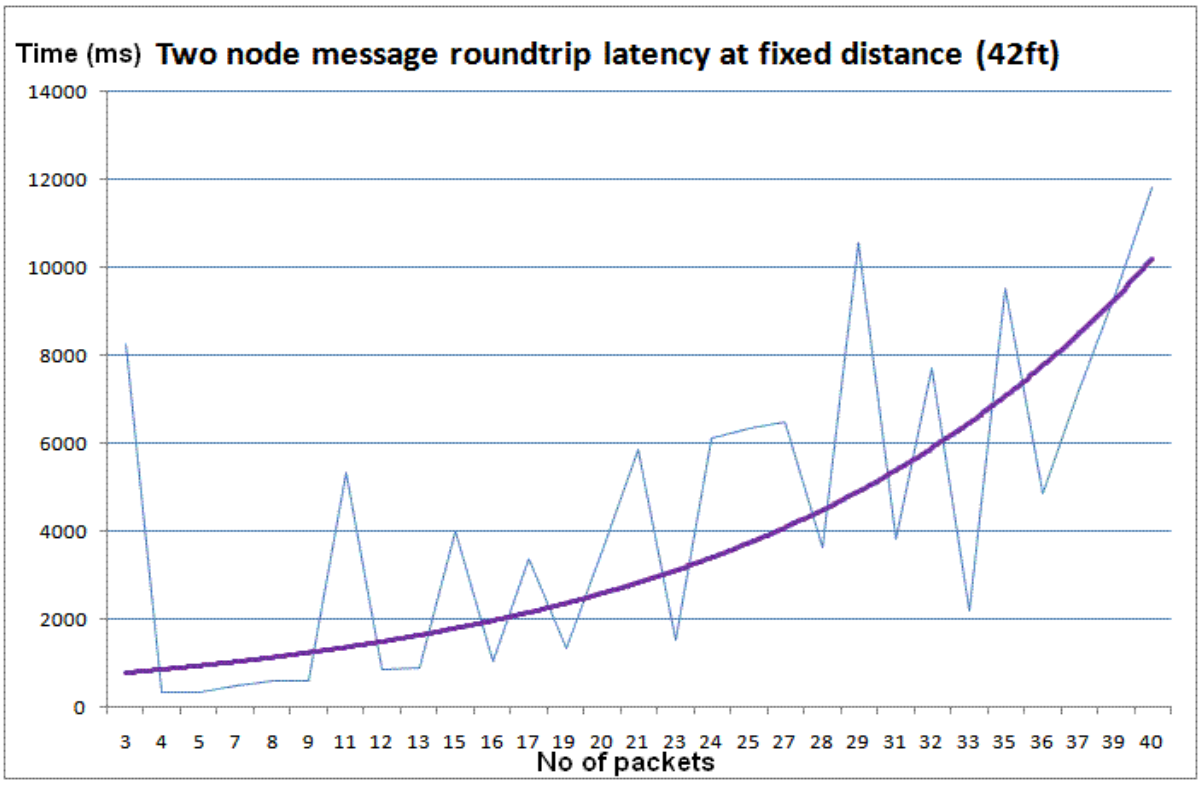

Figure 3.5 Experiment 2 – Graphed results (no. of packets)

3.2.3 Analysis

There is a linear relationship between the message size and the time overhead of fragmenting and re-assembling message packets. As the message size increases, this time overhead increases. Furthermore, packet loss-rate tends to increase with the number of packets sent by the mote. By increasing the message size gradually from 3 packets to 15 packets, the increase in round-trip latency is within a second. The additional time of sending 13 more packets is only 1 second. This implies a significant [time] overhead in sending additional packets. However, to send/ receive an additional 15 packets, from 13, it takes 2 seconds. Therefore, average round-trip latency per packet increases.

This trend implies that larger message sizes not only require more processing (fragmentation, re-assembly) time, but they also suffer higher packet loss rates.

3.3 Experiment 3

3.3.1 Description

Two robots (Robot1 and Robot3), representing two network nodes are positioned outdoors 92ft (about 30.6m) apart, and outside each other's communication range. Another robot (Robot2) is placed mid-way between Robot1 and Robot3 and within the communication range of both robots. The ad hoc wireless communication network using motes and the protocol described in chapter 2 is first initialized on all three robots. Robot1 takes note of the current time (start time) then immediately sends a message of size 40 bytes destined for Robot3. On receiving the message, Robot2 routes the message towards Robot3. Robot3, as soon as it receives the message, sends it back to Robot1. The message is routed by Robot2 on its way back. Robot1, again, takes note of the current time (end time) when it receives back the message it sent. The difference between the two times is computed ("end time" – "start time") and recorded.

This process repeats up to an arbitrary message size of 1,160 bytes. Packet size is 30 bytes.

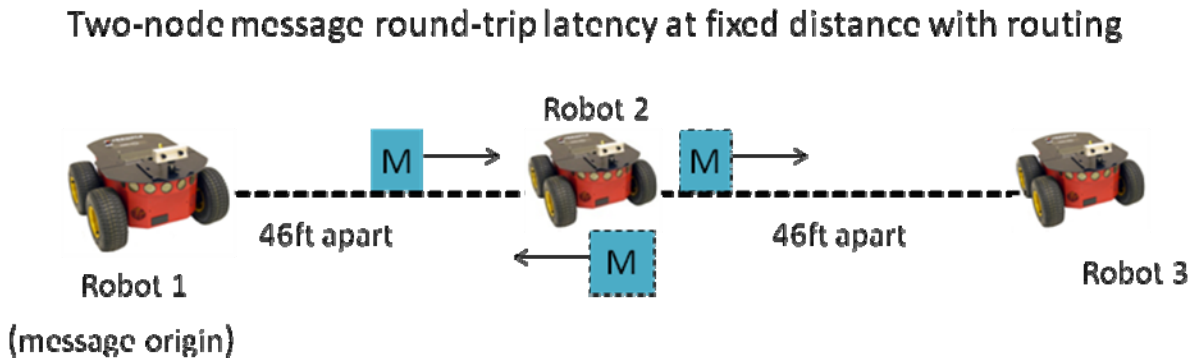

Figure 3.6 Experiment 3 - Setup

A 40-byte message takes a round-trip time of 1second. A message of size 400 bytes (ten times as big) takes about 2 seconds. Another message of size 800 bytes takes about 6-6.5 seconds. It takes 16 seconds to for a message of size 1160 bytes to be received back by robot 1.

3.3.2 Results

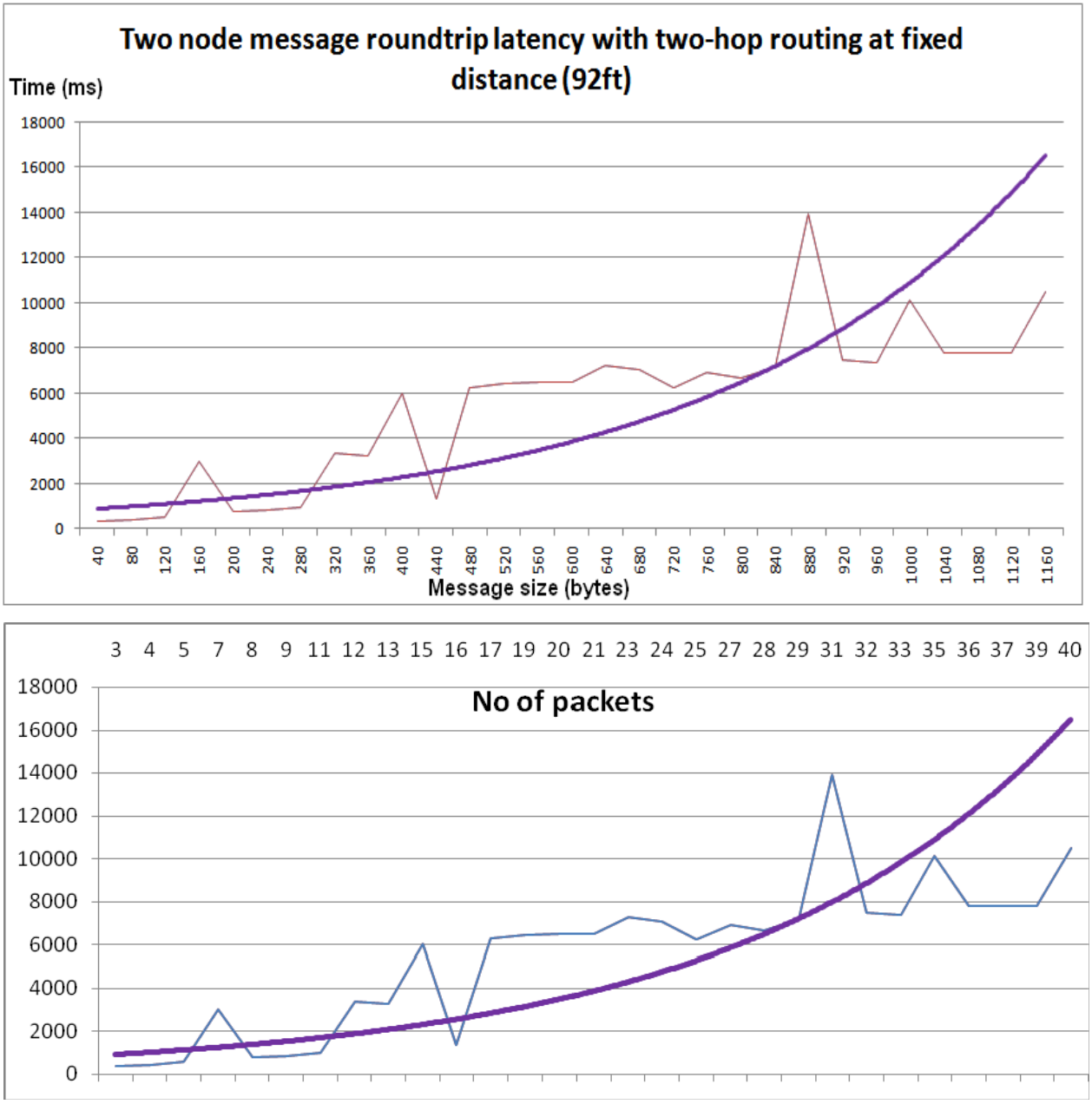

Figure 3.7 Experiment 3 – Graphed results (bytes vs packets)

3.3.3 Analysis

A similar trend, as in experiment 2 (one-hop, no routing), is seen here. Again, increased message size adds to the [time] overhead of message fragmentation and re-assembly. The latency is increased due to two-hop transmission, and a time overhead is introduced due to packet routing.

There is a higher chance of packet loss due to collisions at the routing node since packets are sent out immediately as others a received at the same node. Packet loss increases message latency.

3.4 Comparison of results from experiments 2 and 3

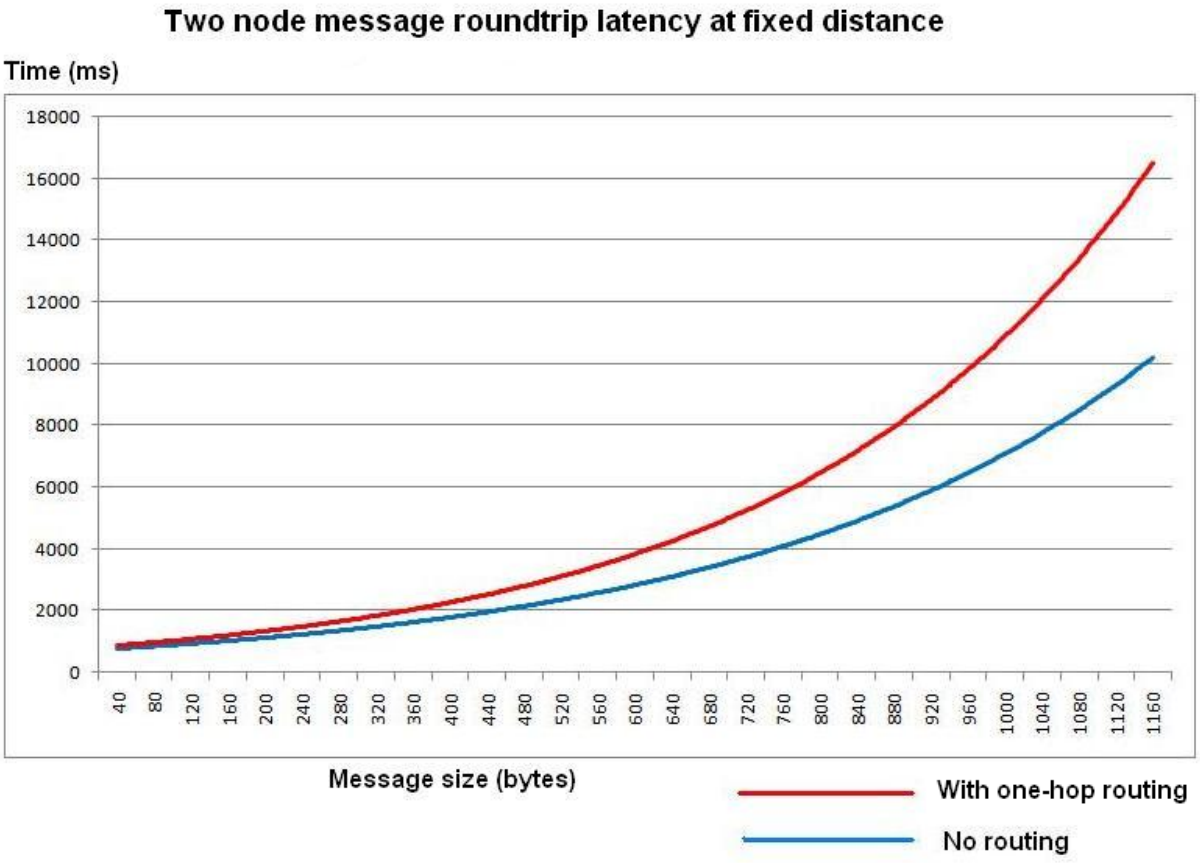

Figure 3.8 Comparison of results of experiments 2 and 3

3.4.1 Analysis

It takes longer for the message to be echoed back when packet routing is introduced. However, the rate of increase is also higher. There seems to be a linear relationship between message size and packet transmission with routing. Since there is a higher loss-rate with increased message size, there is higher number of packets re-transmissions. The retransmitted packets too incur routing overhead as well as increased latency due to the two-hop travel.

3.5 Experiment 4

3.5.1 Description

Two robots (Robot1 and Robot2), representing two network nodes are positioned outdoors starting at 4ft (about 1.3m) apart and within communication range. The ad hoc wireless communication network using motes and the protocol described in chapter 2 is first initialized on both robots. Robot1 takes note of the current time (start time) then immediately sends a message of size 800 bytes to Robot2. On receiving the message, Robot2 sends that exact message back to Robot1. Robot1, again, takes note of the current time (end time) when it receives back the message it sent. The difference between the two times is computed ("end time" – "start time") and recorded.

Robot2 then moves a foot further forward. Robot1 then sends another message of the same [800 bytes] size. When it receives it back, it again records the message round-trip latency. This is repeated until the robots are outside each other's communication range.

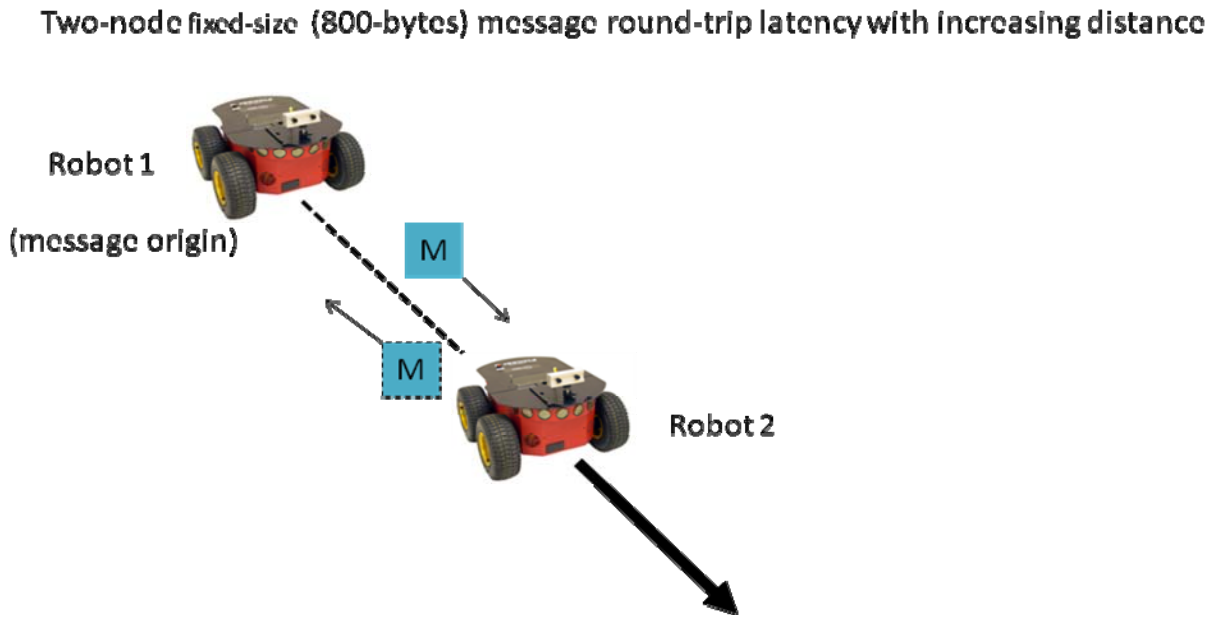

Figure 3.9 Experiment 4 - Setup

3.5.2 Results

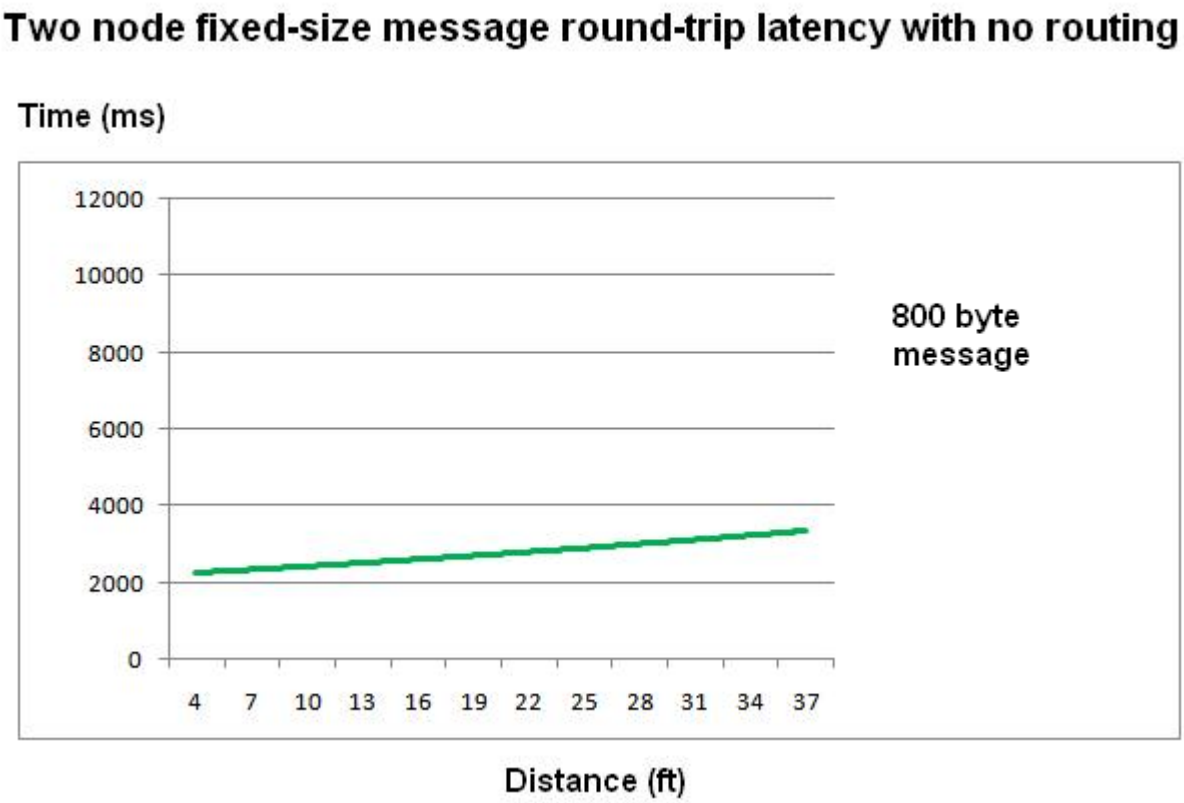

Figure 3.10 Experiment 4 – Graphed results

3.5.3 Analysis

The round-trip latency for a fixed-size message increases as the distance between the two communicating nodes increases. Increased distance can impact signal strength; the signal weakens as it travels. Therefore, increased distance can increase packet loss rates due to signal attenuation and consequently increasing packet re-transmission. This increases the time it takes for the entire message to be received at its origin. Higher throughput is therefore expected at shorter distances.

References

[1] G. Anastasi, E. Ancillotti, M. Conti, and A. Passarella. TPA: a Transport Protocol for Ad Hoc Networks. In *10th IEEE Symposium on Computers and Communications*. In ISCC, 2005. Pages 51-56

[2] Michael A. Goodrich, Alan Schultz. Human-Robot Interaction: A Survey. *Foundations and Trends in Human-Interaction,* Vol. 1, No. 3, (2007), pages 203-275.

[3] Christian Lochert, Bjorn Scheuermann, Martin Mauve. A Survey on Congestion Control for Mobile Ad-Hoc Networks. *Wireless Communications and Mobile Computing*, volume 7, issue 5, pages 655 – 676. John Wiley & Sons Ltd, April 2007

[4] Hoa G. Nguyen, Narek Pezeshkian, Anoop Gupta, and Nathan Farrington. Maintaining Communication Link for a Robot Operating in a Hazardous Environment. In *ANS 10th Int. Conf. on Robotics and Remote Systems for Hazardous Environments,* Gainesville, FL, March 28-31, 2004.

[5] Mauve, M. Widmer, A. Hartenstein, H. A Survey on Position-Based Routing in Mobile Ad Hoc Networks. In *Network, IEEE,* volume 15, issue 6, pages 30-39.

[6] Xiaoyan Hong, Kaixin Xu, Gerla M. Scalable Routing Protocols for Mobile Ad Hoc Networks. In *Network, IEEE,* volume 16, issue 4, pages 11-21.

[7] Paulo Rogerio Pereira, Antonio Grilo, Franscisco Rocha, Mario Serafim Nunes, Augusto Casaca, Claude Chaudet, Peter Almstrom, and Mikael Johansson. End-to-end Reliability in Wireless Sensor Networks: Survey and Research Challenges. *EuroFGI Workshop on IP QoS and Traffic Control,* Lisbon, Portugal, December 6-7, 2007.

[8] Chonggang Wang, Kazem Sohraby, Yueming Hu, and Weiwen Tang. Issues of Transport Control Protocols for Wireless Sensor Networks. In *Proceedings of Communications, Circuits and Systems, 2005.* Volume 1, pages 422 – 426.

[9] Ahmad Al Hanbali, Eitan Altman, Philippe Nain. A Survey of TCP over Mobile Ad Hoc Networks. *Technical Report No 5128.* May 2004.

[10] Joseph Polastre, Jason Hil, David Culler. Versatile Low Power Media Access for Wireless Sensor Networks. In *the Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems.* 2004. Pages 95 – 107.

[11] TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *the Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems.* 2004. Pages 162 - 175.

[12] A Survey of Applications of Wireless Sensors and Wireless Sensor Networks. In *the Proceedings of the 2005 IEEE International Symposium on Mediterrean Conference on Control and Automation.* June 2005. Pages 719 – 724.

[13] A. Perrig, J. Stankovic, D. Wagner, Security In Wireless Sensor Networks, *Communications of the ACM*, June 2004, Vol. 47. No. 6, pages 53-57.

[14] G. A. Jacoby, N. J. Davis, Mobile Host-Based Intrusion Detection And Attack Identification, *IEEE Wireless Communications*, August 2007.

[15] H. Karl, A. Willig, Protocols and Architectures for Wireless Sensor Networks, 2005, John Wiley & Sons Ltd, WestSussex, England.

[16] A. S. Tanenbaum, Computer Networks, 2004, 4th Edition, Pearson Education.

[17] M. Bishop, Introduction to Computer Security, 2005, Pearson Education Inc.

[18] R. Anderson, Security Engineering – A Guide to Building Dependable Distributed Systems, 1 Edition, 2001, Wiley.

[19] http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf, "Crossbows, Micaz mote Hardware/Software specification".

[20] B. C. Neuman, T. T'so, Kerberos: An Authentication Service for Computer Networks, *IEEE Communications Magazine*, Volume 32, Number 9, pages 33-38, September 1994.

[21] T. Amtoft, et. al., Specification and Checking of Software Contracts for Conditional Information Flow, Technical Report SAnToS-TR2007-5, Manhattan, Kansas.

[22] J. Barnes, "High Integrity Software, The Spark Approach to Safety and Security", 2003, 2006, Praxis High Integrity Systems, Inc.

[23] "P3-AT Robot Specification" http://www.activrobots.com/ROBOTS/specs.html

[24] CC2420 Radio Stack, http://www.tinyos.net/tinyos-2.x/doc/html/tep126.html

[25] 802.15 standard, http://www.ieee802.org/15/pub/TG4.html

[26] RFC 1662 – PPP in HDLC-like Framing, http://www.faqs.org/rfcs/rfc1662.html

[27] Soo Young Shin, Hong Seong Park, Sunghyum Choi, Wook Hyun Kwon. Packet Error Rate Analysis of IEEE 802.15.4 under IEEE 802.11b Interference. In *the 63rd Proceedings of the IEEE Vehicular Technology Conference*, 2006.