Computer Engineering Senior Design Projects II
EECC 657
Final Report

EX-Bot
Map it once and use it forever.

Submitted By: Timothy C. LaForest
TCLaforest@aol.com

Eric Shields
ers3268@rit.edu

Merve Evran
merveevran@gmail.com

Submitted To: Dr. Roy Czernikowski
Date Submitted: February 14, 2006

Table of Contents

Index of Figures

Index of Tables

Introduction

In this system, a robot will roam an enclosed area with random objects. If allowed to, the robot will do this autonomously. If the robot gets stuck in a loop or the user simply wishes to speed the robot's progress to a certain area, the user can click on a virtual map to designate a new place to map from. This map is displayed to them on the computer running the User Interface. The virtual map is updated as the robot's mapping progresses and is saved until the system is reset or exited. This interface also allows the user to convert the robot to a controlled mode, in which the user can navigate the robot through the mapped area.

Overall System

The objective of this project is to create a robot that will be capable, with the assistance of a PC, of mapping an area, 7.5ft by 7.5ft (2.286m by 2.286m), then navigating the mapped area. The system will have a java-based user interface with a display of the mapped area and some controls which will communicate via 900MHz RF modules with the robot. The robot will make use of ultrasonic range finders, a magnetic compass, and an infrared phototransistor for mapping and navigation. The robot will use the I2C standard for communication with the sensors (except for the phototransistor).

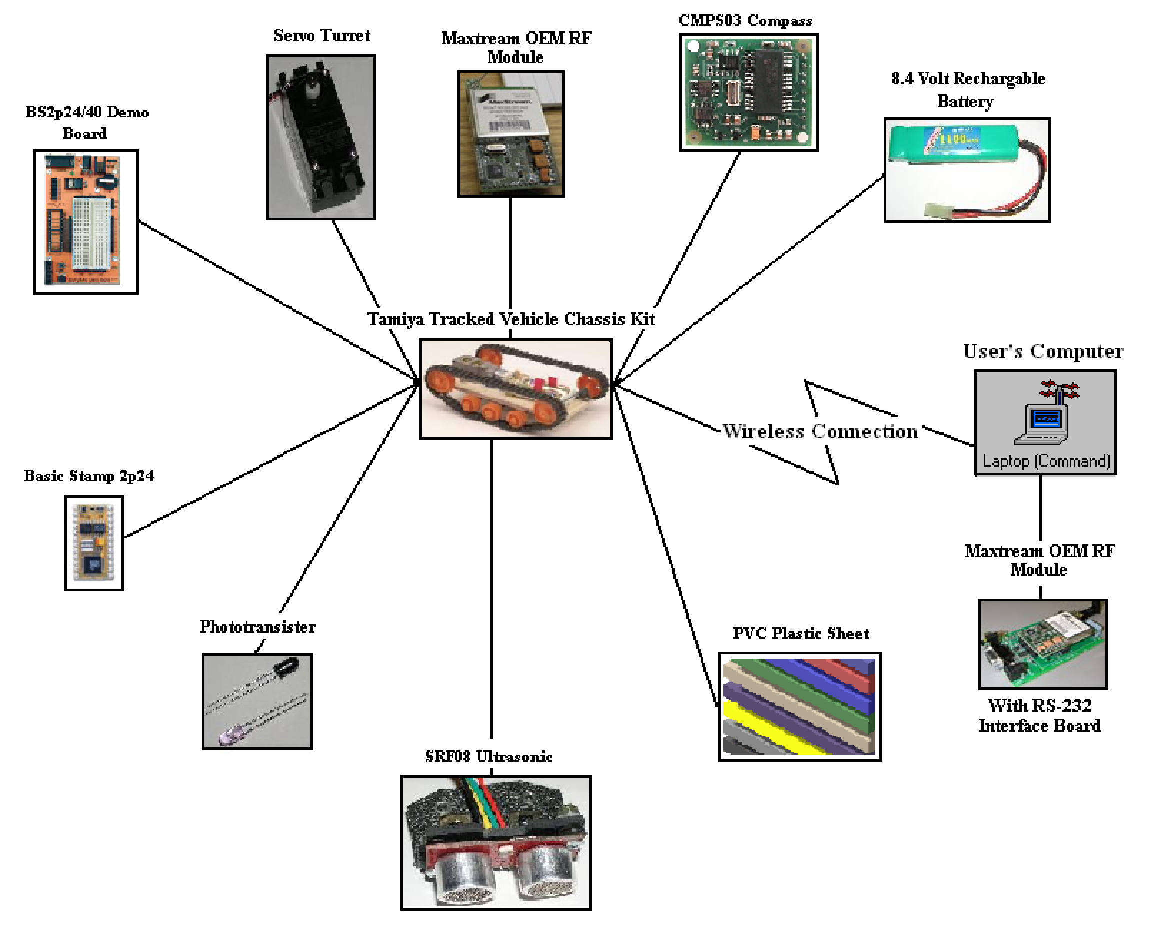

Figure 1: System Overview Component Diagram

The complete system consists of three components: The robot, a laptop serving as the user interface (also known as Control), and the communications between them. A component diagram of the system is shown above in figure 1.

The overall system design is as follows: Both components have an RF wireless module. Using standard RS-232 and UART protocols, a series of signals will be transmitted, in the form of specifically assigned segments of the payload of each packet, to and from the robot.

Some of the operating environment will be created specifically for the demo. This will include 4 outer walls, to eliminate the issues related to outer boundaries, and objects of various shapes and sizes to act as obstacles. The only restriction on the placement of objects is that they cannot be moved without restarting the user interface.

The next three sections detail each of the major pieces of the system, which are stated above.

Robots

The robot will use the following as a base: The Tamiya Tracked Vehicle Chassis with the Low Cost H-Bridge provides the motor and drive. There will be three tiers, constructed of colored PVC plastic sheets, one for the motor housing, one for the RF and development board, and one for the turrets. On the second tier will sit the BS2p24/40 Demo Board from Parallax, Inc. This board was chosen for the fact that it has an embedded I^2C controller and Parallax, Inc provides PBasic, an easy to use programming language.

Connected to the development section of the board will be the CPMS03 Magnetic Compass, MaxStream 900MHz, 9600 baud, RF transceiver, and QRB1134 Photo-transister. The compass and ultrasonic range finders will be connected to the I^2C bus. The compass will determine the direction that the robot is facing. The phototransistor will allow the user's computer to determine the distance travelled by sending a signal to the BS2 everytime a tread hole passes. The RF module will allow communication with the user's computer.

Finally, the robot will have 2 SFR08 High Performance Ultrasonic Range Finders on the top tier. Each one will sit atop a "turret," a servo motor, which will rotate so as to take measurements at different angles. These "turrets" will be placed such that they are placed exactly 5 inches apart at opposite ends of the robot and are in the center of the robot's width. The two sensors will always be facing in opposite directions when taking measurements.

Figure 2: Robot Schematic

The details of robot system are contained in the schematic for the robot, Figure 2 above, and the component list, Figure 3 below.

Figure 3: Robot Component List

Wireless Communications

The “glue” that holds this project together is the wireless communications that occur constantly between the robot and the user interface. These communications will use a specially-designed packet structure, shown in Figure 2 below.

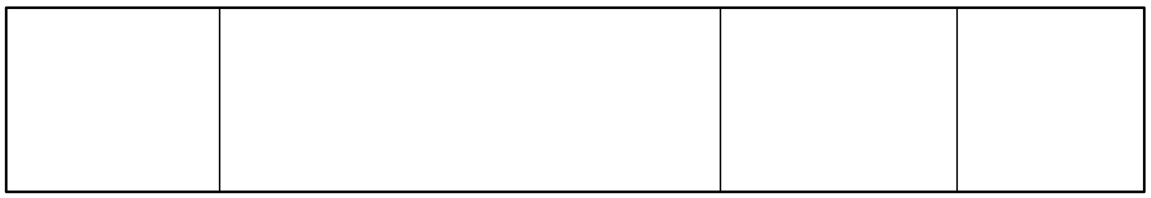

Figure 4: Communication Packet Setup

In most cases, this packet will be sent only once, with the Ultrasonic Measurements zeroed out. However, in the case of mapping mode, this packet will be sent 5 times in order to send all of the sensor readings. This is due to memory constraints of the BS2. A 6th packet will then be sent as part of the normal communications.

The state bits represent what the robot is currently doing or where the user interface wants the robot to be next, for Robot-to-Control and Control-to-Robot communications, respectively. The state diagram is shown on the next page in figure 5.

State
(8 bits)

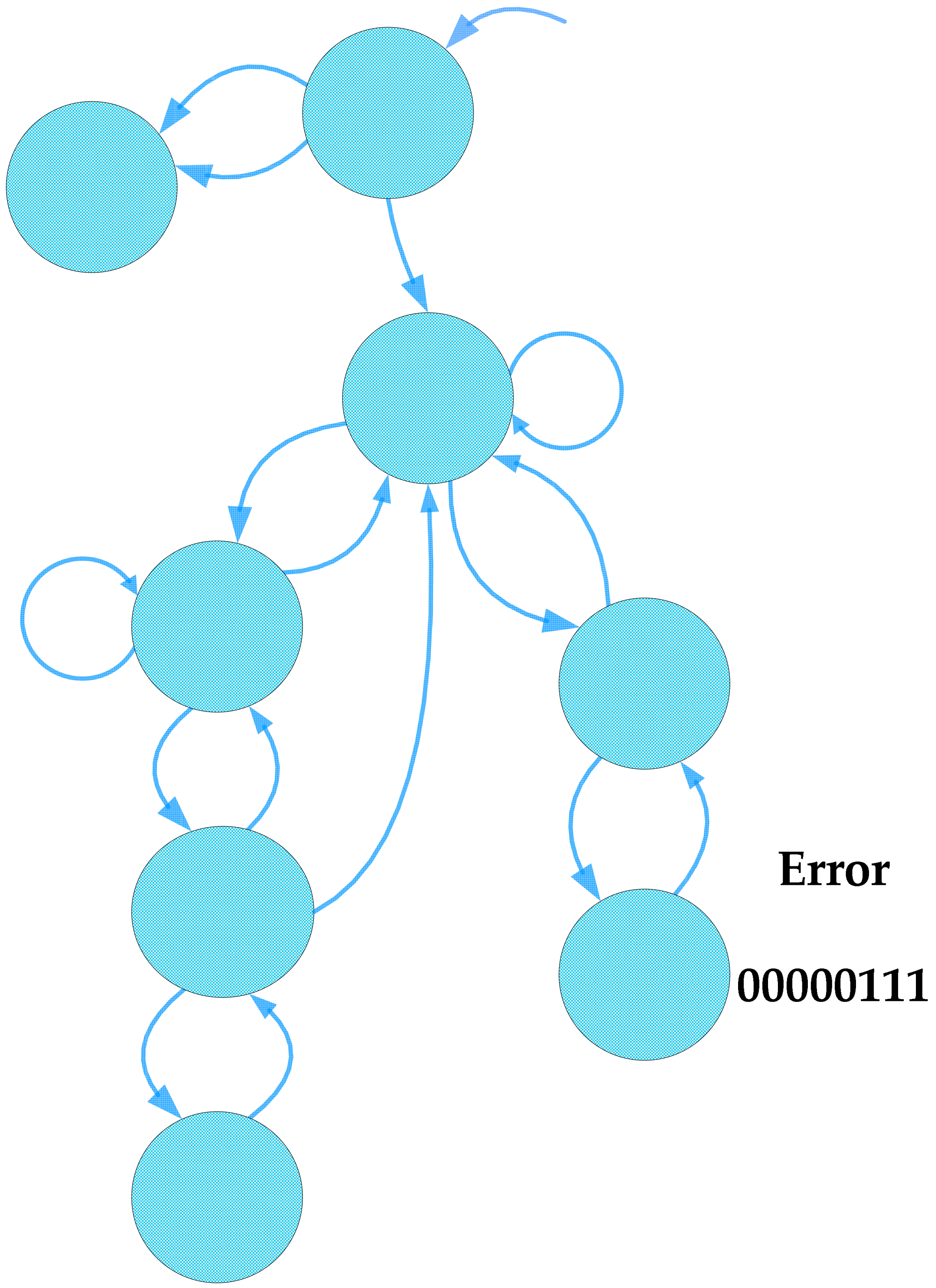

Figure 5: Robot State Diagram

The compass value will be an integer from 0 to 3599, representing 0 to 359.9 degrees. This value will contain a non-zero value when giving the robot directions in controlled or user-mapping mode or when the robot is communicating it's heading to the control.

This distance value will contain one of three things: Most commonly it will contain a number of tread holes that the robot needs to travel. It may also contain all 0's if the data isn't relevant. Finally it may contain an error code. If the value is all ones, then the robot received an invalid state and if the value contains alternating 1's and 0's, then the robot is stuck between two objects and cannot move without the help of the user.

The user's computer will be running a Java program and using an RS-232 interface, so no restrictions are anticipated on what computer it could be set up on. Each robot will have a single RF module operating in the 900MHz frequency range. This range is satisfactory for use within the US, but is subject to restrictions in other countries.

User Interface/Control

Figure 3 below shows the general look that the user interface will have.

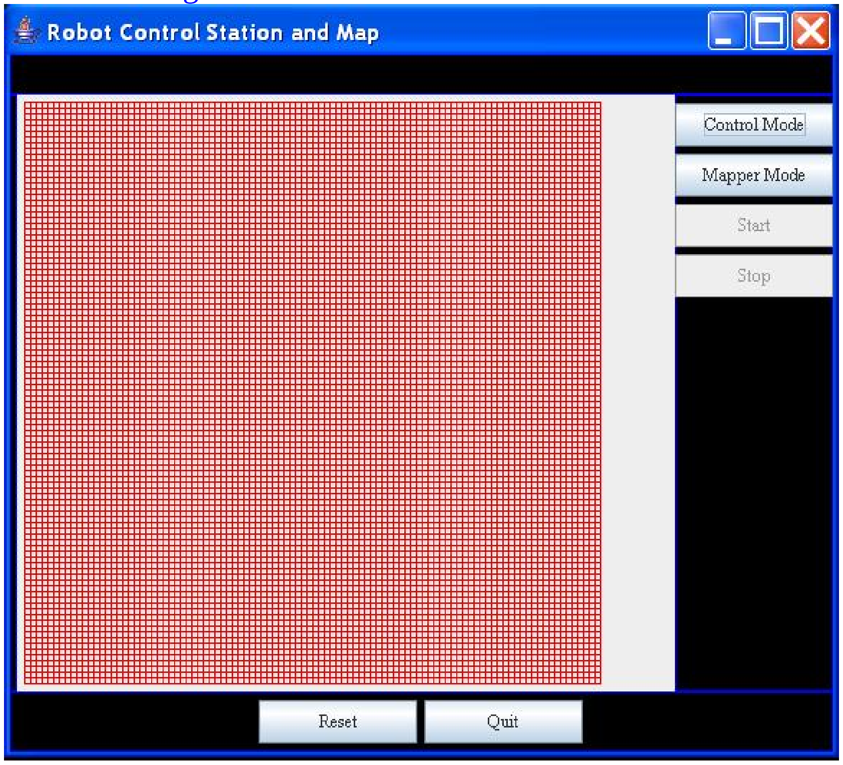

Figure 5: User Interface

In this view, the user can control the general actions of the mapping robot. There are four basic functions, plus the ability to change from Mapper mode to Controlled mode, or vise versa. Of the basic functions, two deal with the robot directly, one deals with the interface alone, and one deals with both. In the grid used, each box represents a 5 inch by 5 inch square.

Pressing the start button will cause the robot to either start mapping from the current position or continue idling until a destination is provided by the user.

Pressing the stop button will cause the robot to cease travel after the current instruction completes or to cease scanning after the current scan completes, depending on its current state.

Pressing the Reset button will clear all of the mapped area from the coordinate grid and replace it with black boxes. The coordinates of the robot, however, will NOT be reset. This way the mapping can begin anew without replacing the robot at the start and the new map will directly correspond with the old one. This will also act as if the stop button was pressed.

Pressing the Quit button exits the program. All coordinate data is lost. The robot will act as if the stop button were pressed.

The Mapper Mode and Controlled Mode buttons are used to change the current mode of the robot, to Mapping and Controlled, respectively. In Mapper mode, the robot will follow the mapping algorithm, listed below in the System Functionality and Algorithms section. In Controlled Mode, the robot will wait until it is given a destination. This is done by clicking a location of the map.

The destination must be in a mapped area. The destination cannot be in an object. (objects and unmapped areas will be represented by black areas.) The destination cannot be outside of any predefined boundaries. (The boundary feature may or may not be implemented in the final product.)

Additionally, both for speed and to prevent deadlocks, the user will have the option to click on the map during the mapping process. Doing this will act the same as pressing stop, setting the mode to Controlled, pressing start, clicking on a destination, clicking stop once it arrives, setting the mode to Mapper, and finally clicking start again. Once the robot reaches the last segment of its journey to the destination, a pop-up window will appear asking for the orientation of the robot. The user should input a value from 0 to 359, with 0 meaning to face west, 90 meaning north, 180 meaning east, and 270 meaning south. No change in which buttons are grayed out will occur during this process.

Once the user selects a location for the robot to travel to, in either mode, the software will calculate the best path for that robot to travel to reach the destination, avoiding all objects. For a description of the algorithm used to determine the shortest path, see the System Functionality and Algorithms section.

On startup, the interface will default to Mapper mode and stopped. When Mapper mode is active, the Mapper Mode button will be grayed out and the Controlled Mode button active. They will switch when Controlled mode is active. The same holds for stop and start. When the robot is doing something, the start button will be grayed out and the stop button active, and vise versa when the robot is stopped. Reset and Quit will always be avtive.

Testing Strategy

The testing for this project will be long and difficult. First and foremost, each of the robot's individual components and the wireless interface software must be individually tested to ensure functionality.

The user interface software will be constructed piece by piece and tested for functionality (e.g. buttons send correct signals to network, information is displayed accurately, etc.). It will also be tested by creating a series of fake packets and ensuring that the correct map details are created.

The sensor network will be tested individually for the robot to ensure functional sensors and accurate reading of data from the sensors by the robots. The following is a Built-In Self-Test (BIST) for the robot that will be run upon startup.

Figure 6: Robot BIST

The ability of the robots to make appropriate turns on the carpet must be tested. If it is not plausible, an alternative surface must be created.

Finally the mapping and path finding algorithms must be tested for cases such as open area, impossible to reach destinations, destination out of range, and various types of obstacles.

Equipment/Software

The required components for this project to be replicated are listed in the following table.

Manufacturing Costs			
Component	*Cost Per*	*Quantity*	*Total*
AC Adapter for Development Board	$ 10.00	1	$ 10.00
Battery	$ 8.99	1	$ 8.99
Battery Recharger	$ 23.99	1	$ 23.99
Compass	$ 49.00	1	$ 49.00
H-Bridge	$ 19.99	1	$ 19.99
IR Detector	$ 0.99	1	$ 0.99
IR Emitter	$ 0.99	1	$ 0.99
Microcontroller and Development Board	$ 199.95	1	$ 199.95
Miscellaneous Electical	$ 80.37	-	$ 80.37
Miscellaneous Mechanical	$ 20.63	-	$ 20.63
Red LED	$ 1.00	1	$ 1.00
RF Devolopment Kit	$ 149.00	1	$ 149.00
Robot Base	$ 39.95	1	$ 39.95
Servo Turrets	$ 18.95	2	$ 37.90
Ultrsonic Sensors	$ 56.99	2	$ 113.98
		Total Manufacturing Cost	$ 756.73

Table 1: Component Listing to Replicate Project

The next table contains the items that were actually purchased in the process of completing this project.

Actual Project Cost				
Component	*Cost Per*	*Quantity*	*Shipping*	*Total*
5V to 3.3V Regulator	$ -	1	$ 9.00	$ 9.00
AC Adapter for Development Board	$ 6.00	1	$ -	$ 6.00
Antenna	$ -	1	$ -	$ -
Battery	$ 8.99	2	$ 7.95	$ 25.93
Battery Recharger	$ 23.99	1	$ 11.90	$ 35.89
Bluetooth Dongle	$ 12.49	1	$ -	$ 12.49
Bluetooth Module	$ 1.98	8	$ -	$ 15.84
Compass	$ 49.00	1	$ -	$ 49.00
Dual, Bidirectional, Voltage Level Translat	$ -	1	$ -	$ -
H-Bridge	$ 19.99	1	$ 30.14	$ 50.13
IR Detector	$ 0.99	1	$ -	$ 0.99
IR Emitter	$ 0.99	1	$ -	$ 0.99
Microcontroller and Development Board	$ 60.00	3	$ 24.00	$ 204.00
Miscellaneous Electical	$ 80.37	-	$ 68.28	$ 148.65
Miscellaneous Mechanical	$ 25.63	-	$ 14.05	$ 39.68
Module Adapter	$ 68.40	1	$ -	$ 68.40
Red LED	$ 1.00	1	$ -	$ 1.00
RF Devolopment Kit	$ 149.00	1	$ 49.00	$ 198.00
Robot Base	$ 39.95	1	$ -	$ 39.95
Servo Turrets	$ 18.95	2	$ -	$ 37.90
SOT-23-8 to DIP Adapter	$ 3.19	1	$ 26.00	$ 29.19
Ultrsonic Sensors	$ 56.99	2	$ -	$ 113.98
Demo Materials	$ 45.00	-	-	$ 45.00
			Total Project Cost	$ 1,132.01

Table 2: Actual Components Bought Listing

All cost values are subject to change at the will of the companies. All components are subject to upgrading, being made obsolete, etc. at the will of the companies. The listed project cost is not necessarily accurate due to fluctuating currency conversion rates, shipping costs, sales, and bulk purchasing.

The robot will be powered by an 8.4V, 1100mAh rechargeable battery.

The software needed for this project includes a Java Runtime Environment (JRE) from Sun Microsystems, Parallax free Pbasic compiler, and X-CTU software provided by MaxStream.

System Functionality and Algorithms

Upon starting the demo, the robot will perform a Built-In Self-Test to ensure that the components are functioning as expected. See figure 5 in the Testing Strategy section for details of the BIST. The robot will then wait for instruction from the User Interface. The User Interface will send the command to be a Mapper along with the first command to start. This is because there's no possible location for the user to command the robot to go until at least one mapping scan has completed.

Once the robot receives a start command, it will first take a compass reading, perform a scan. A scan consists of taking a measurement from each ultrasonic sensor, moving the sensors by 10 degrees, in opposite directions from one another, taking another measurement, moving 10 degrees, etc., until each sensor has completed a 180 degree arc. The sensors should be facing opposite directions at all times. The compass and ultrasonic sensor data is sent to the user interface as one packet. If a stop command was issued, this is where it would stop.

Once a scan completes, the robot will move forward by two grid boxes. If a stop command was issued, this is where it would stop. The robot will then ask if mapping has been preformed from this location already. If not, a new scan will commence. If so, the robot will move on two more grid boxes.

If a wall is encountered such that the robot cannot move forward two boxes, it will turn 90 degrees and try again; continuing this process until an opening is discovered or a stop command is received.

If at any time the user clicks on a location on the map, the robot will proceed until reaching a stop point, then follow the same instructions it would if acting as a controlled robot (described later in this section) until that location is reached, then it will rotate to the orientation designated by the user and continue the mapping procedure.

The algorithm just described is given in block diagram format in the following figure:

Figure 7: Mapping Algorithm

If at any point the robot is switched to Controlled mode, the robot will stop at the first stopping point and wait for instructions. When the instructions have been received, the robot will turn to the designated direction, then move the distance specified. It will then send an acknowledgement to the user interface and wait for another set of instructions. If a stop command is issued, it will stop either after the turn is completed or after the move is completed, dependant on the timing of the command. This algorithm is shown in block format in the following figure:

Figure 8: Controlled Algorithm

The final and most important algorithm is the path finding algorithm. The purpose of this is to find the best path, or at the least a path, from a starting point to a destination point. This algorithm is used both when the robot is in Controlled mode and when it is told to proceed to another location to continue mapping.

The algorithm is as follows:

Analysis of Difficulties

This project was fraught will difficulty from the start.

The project was behind schedule right away due to a combination of changing professors and a lack of communication from a hopeful sponsor. This resulted in the late discovery that the proposal written for Senior Design I was not accomplishable in one quarter. This resulted in a complete redevelopment of the project for the design review at the start of Senior Design II. The in turn caused numerous additional delays due to the lack of explicit detail in the new proposal, which had to be written in very short order.

Once the project was truly underway, many more problems cropped up.

The Java API (Application Programming Interface) for communication with the serial port of a computer was difficult to integrate, especially when trying to use a virtual serial port created by a Bluetooth USB device. This led to many problems with establishing packet sizes and communication timing. One of the results was that the packet size had to be made divisible by bytes (in other words a multiple of 8 bits) for the information to be successfully sent over the wireless and that the individual sections of the packet also had to be divisible by bytes for parsing purposes.

The ultrasonic data algorithms were built on a turret arc of 180 degrees. Unfortunately it was discovered that they only seemed to allow a 90 degree arc. This caused many changes in the communications and algorithm processing, as well as lowering the overall accuracy obtained from each mapping position.

The sensor offsets on the robot, in conjunction with the GUI using the exact center of the robot as the “origin” of the robot, rather than one of the sensors, caused extensive problems with the algorithm that interprets the ultrasonic data.

The original motors for the robot needed to be replaced with higher torque ones at the last minute, which required disassembling the robot and reassembling it again.

Several times, a group member would be incapacitated for a period of time due to sickness or disability problems. This caused problems with planned due dates and meetings, as well as class attendance. Also several times the project had to be delay do to the ridiculous amount of work that had to be done for other classes being taken at the time, which was at least 5 fold the work load of any quarter previous.

It was discovered that the Bluetooth modules purchased for the project didn’t have the capability to connect with the robot in the way that was needed, so the whole Bluetooth concept had to be dropped near the end of the project and replaced with easier to use RF modules.

Conflicts in the way the coordinate systems were set up in the User Interface and the Robot caused problems with integration until the cause was determined.

Lack of willingness to modify one’s own part to help interact with the rest of the project was another difficulty. Many painful compromises needed to be made in communication size and other aspects due to this problem.

Power draw on the robot caused the microcontroller to shutdown when powering the motors; resulting in another, separate battery being added. In addition, several parts of the robot operate at different input voltages, so voltage level translators had to be bought and installed in several places, and then some of them removed when the Bluetooth was replaced with the RF modules.

Finding and implementing the path finding algorithm was an extensive task. First the algorithm used would not actually find all paths or by any means a good path if it did find one. Then the algorithm would find a path, but it would use far more turns than the robot could feasibly accomplish without a much better guidance system.

Trying to find a way to place an updatable map in the GUI was another major problem. Java applets do not function inside of swing GUI elements and there was no easy way to create a map without the use of the applet. This resulted in using a rougher map system without a lot of the features the Applet offered.

Deliverables

The project demo will have the following format:

1. Demonstrate mapping of an area
2. Demonstrate stopping the mapping
3. Demonstrate continuing the mapping
4. Demonstrate sending the robot to a location while mapping
5. Demonstrate switching the robot to Controlled mode
6. Demonstrate sending the Controlled robot on an easy path
7. Demonstrate sending the Controlled robot on an difficult path

8. Demonstrate sending the Controlled robot on an impossible path
9. Demonstrate resetting the map
10. Demonstrate exitting the program

Time Line

Item Description	Primarily Responsible Member	Completion Date
Setup User Interface	ME	12/6/2005
Finalize Project Algorithms	ALL	1/12/2006
Conclude Bluetooth Research	ERS	12/16/2006
Create Website Template	ERS	12/13/2005
Aquire All Hardware	TCL	12/16/2005
Implement Buttons/Map Function	ME	12/16/2005
Test Component Functionality	TCL	1/12/2006
Implement Mapping Software	ME	1/31/2006
Construct Robots	TCL	1/31/2006
Write Avoidance Alrorithm	TCL	1/13/2006
Rewrite Project Proposal	ERS	1/24/2006
Implement Packet interface	ERS	1/31/2006
Test Avoidance Algorithm	TCL	1/31/2006
Write Path Finding Algorithm	ME	1/31/2006
Test User Interface Functionality	ME	1/31/2006
Test Path Finding Algorithm	ME	1/31/2006
Find New Java Interface Tool	ME	1/31/2006
Update website	ERS	1/31/2006
Test Terrain Mapping	ALL	1/31/2006
Bluetooth Module Functionality	ERS	N/A
Order RF Modules	ERS	2/7/2006
RF Module setup	ALL	2/10/2006
System Functionality Testing	ERS	2/14/2006
Write Final Report	ERS	2/14/2006
Demo Poster	TCL	2/14/2006
Team Assesments	ALL	2/22/2006
Finish Website	ERS	2/14/2006

Table 3: Project Time Line

Conclusion

This project started as a great idea with a lot of enthusiasm. Unfortunately, it fell on some very hard times and had to be continuously altered to be simpler and/or cheaper. The final product was nothing like what we wanted at the start and is, by itself, not overly useful, simply a small stepping stone to larger things. This project was a great learning experience in feasibility and the usefulness of predicting constraints. This knowledge will go far in the corporate world.