



# Bluetooth Bots

## An Investigation of Bluetooth's Applicability to a Light Finding Robotics Problem

Submitted By  
Laurel M. Hesch

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR AN  
UNDERGRADUATE THESIS WITH A

**BACHELOR OF SCIENCE IN ELECTRICAL ENGINEERING**

School of Engineering  
Tufts University  
Medford, Massachusetts

May 2004

---

Signature of Author:  
Laurel Hesch

---

Certified By:  
Dr. Ron Lasser  
Department of Electrical Engineering  
Tufts University

---

Committee:  
Professor Chris Rogers  
Department of Mechanical Engineering  
Tufts University

---

Committee:  
Professor Joseph P. Noonan  
Dept. of Electrical Engineering  
Tufts University

## Table of Contents

1. Abstract.....	4
2. Problem Statement.....	5
3. Theory.....	9
4. Experimental Setup.....	17
5. Implementation.....	22
6. Results.....	28
7. Statement of Limitations, Obstacles and Deficiencies.....	30
8. Most Critical Obstacles.....	33
9. Suggestions for this Project.....	37
10. Lessons Learned.....	39
11. Acknowledgements.....	40
Bibliography.....	41
13. Appendix One: Technical Information.....	42
13.1 Mathematical Basis for the Algorithm.....	42
13.2 Technical Information for Infrared Communications.....	43
13.3 Technical Information for Bluetooth Communications...	44
14. Appendix Two: Pictures.....	48
15. Appendix Three: Figures.....	51
16. Appendix Four: Code	
16.1 Mobile Robot Code.....	52

16.2 Stationary Robot Code #1 (DTE Device).....	57
16.3 Stationary Robot Code #2 (S/M Device) .....	61

## 1. Abstract

A team of light finding robots was constructed to investigate whether Bluetooth is a good fit for wireless robotics. Bluetooth has the potential to greatly increase the accuracy and reliability of robotics communications.

The basic approach to find a spot of light was for two stationary robots positioned at the corners of a rectangular area to each sweep through ninety degrees and search for the angle of brightest light. Once this angle was found, the stationary robots transmitted the angle to a mobile robot that calculated the point of intersection between the two received angles. After moving to this destination, the mobile robot followed the spot of light as it moved, by comparing the values of three photoresistors. A wireless Bluetooth system was developed to send and receive angles of brightest light.

The Bluetooth communications structure developed was highly successful in terms of reliability and accuracy of transmission. A limitation of this architecture was the lengthy amounts of time to connect as well as a one-to-one master/slave relationship. Bluetooth has the potential for one master to connect with many slaves, but the technology available for purchase only enables one-to-one connections.

Light sensing, motor control, and communications were all implemented successfully. Future work includes establishing a multi-slave Bluetooth architecture, and the global awareness of the mobile robot after its initial movement.

## 2. Problem Statement

On Monday, March 20, 1995 members of the Japanese terrorist organization Aum Shinrikyo released deadly Sarin gas into subway cars in five coordinated attacks. These attacks killed 12 people and injured 6000 more, marking this incident as the most serious terrorist attack in Japan's modern history. In the aftermath, the Subway Authority was severely criticized for its handling of the event, where in some cases trains were not halted despite reports of passenger injury. On the Marunouchi line, the contaminated train was only put out of service an hour and 40 minutes after the gas had been released<sup>1</sup>. The Subway Authority's slow response time to this tragedy was the cause of needless death and injury<sup>2</sup>.

The ineffectiveness of the decontamination efforts on the part of the Subway Authority highlight the human error involved in finding and disposing of deadly chemicals. One simple but effective preventative measure would be to have passive chemical sensors mounted on the walls of terrorist-sensitive areas (such as subway cars or federal buildings). These sensors, much like smoke detectors, would serve as an early warning system against such attacks.

An effective method of decontamination would be to deploy teams of robots with nerve gas sensors to sweep an area and attempt to find a source of nerve gas. The advantage of robots is that they do not need protective gear and could potentially cover an area more quickly and effectively than their human counterparts. A team of fifteen or more mobile robots could be released which would communicate with each other to find the most likely location of the gas. These robots would have to have chemical sensors, movement capabilities, and a method of wireless communication. Teams of such small

robots would be effective in quickly finding locations. Finally, the robots would either have to report their coordinates back to a central station or have the capabilities to decontaminate the area autonomously.

To this end, the Team Spot problem was created. The goal was to build team of robots with the functionality to communicate with one another and find a location. The problem was redefined to light detection from chemical sensing to facilitate the building of a viable model. While chemical sensing is the eventual goal, it was neither safe or feseable to obtain enough sarin gas to be able to test the functionality of nerve gas-finding robots. The problem was adapted to light sensing because light is similar to chemicals in that both dissipate away from a spot of highest intensity. We thought that with inexpensive photoresistors a good approximation for chemical leak conditions would be achieved.

Another simplification was to build only one mobile robot instead of many. We postulated that a single moving robot was a necessary first step before a team of robots could be constructed. The development of algorithms to establish communication with stationary robots, calculate coordinates, control motors, and detect light were a good first step to constructing a team of robots with these functionalities.

The team defined several other design constraints in an attempt to simplify and clarify the problem to solve. In addition to the physical constraints addressed earlier (communication, motor control, and light detection), it was decided to develop a team of autonomous robots. Autonomous robots would be able to communicate with one another and find a spot of brightest light without a central processing unit (CPU) controlling all robots. A CPU was not used in order to construct a system which would be resilient to

attacks on itself. A central control unit that would render all the robots useless if destroyed would be an inherent weakness in a terrorist-resistant system. The team decided to design interconnected robots with the ability to function if one or more was disabled. To this end, each robot has its own embedded microcontroller and ability to understand and interpret various conditions autonomously.

Environmental constraints were the second major factor in designing this system. Outdoors, chemical attacks are not especially potent, so it was decided to try to simulate an enclosed space such as a room in an airport or a subway car. Therefore stationary robots, which could be mounted on walls, would work.

In this project, it was decided to design a team of robots which would operate within the Robotable environment. The Robotable is a project being developed at Tufts to connect two or more sites via an interactive physical table. Two or more remotely connected sites can communicate using multitude of tools including 3-D modeling and virtual whiteboards. It was decided to use the Robotable for this project based on the versatility of this platform. Within the Robotable environment, it would be easy to create moving spots of light, multiple spots of light, or light which had a gradient towards a brightest spot. Furthermore, the fixed boundaries of a table are similar to the walls of a room, lending the project for easy transition to the real world. By choosing a small space for the robots to search, an additional physical constraint was added to the problem. The robots had to be sufficiently small that they did not take up a large amount of the table.

The secondary goal of this project was to design a problem which could be easily imported into the classroom. Meeting this goal was another reason the Robotable was used. The team of light-finding robots could be demonstrated in other sites with a

Robotable. Students in remote locations could create light spots and watch robots find these spots from a home location. Students could watch as the robots found the spot of light at their Robotable. Lessons which could be taught through the light-finding robots include trigonometry, principles of wireless communication, light sensing, motor control, gearing, and geometry.

Finally, a design process was defined which contained several stages. First, it was decided to develop robots that could communicate with one another to find a single spot of light on a black background. Then, the problem was complicated with moving spots, spots of different intensity, and gradient spots.

A vital link to solving this problem was the issue of enabling the robots to communicate with each other. The communications functionality was limited to wireless solutions, as connecting the robots directly with cables would be cumbersome and limit the functionality of the robots. Furthermore, a system of robots which did not have a wireless communications system could be disabled by cutting wires, which would be a security risk. Wireless solutions allowed the robots to have a free range of motion. Main considerations in designing a wireless communications system were the need for security, accuracy, and reliability. Security was important because systems of communications that could easily be disabled or tampered with are inherently weak. One security consideration was the need for accurate authentication between devices, such that the robots could identify where the communications were coming from. Accuracy (sending correct serial data with few errors) and reliability (correctly establishing a connection between devices) were also important considerations. Other considerations were data transfer speed, power, range, and connection speed. Ideally, a wireless system would

initially establish a connection, and keep it open throughout, thereby allowing constant sending and receiving of information. Finally, the communications method had to be in accordance with the idea of autonomous robots, meaning that communication methods relying on a central PC to send and receive information were rejected.

### **3. Theory**

Many robotics applications depend on reliable wireless communications. From simple remote controls to interdependent groups of robots, devices that move independently and communicate require some form of wireless communications. The requirements for robotics technologies are in some cases similar to the short-range wireless industry in general, and in other ways quite different.

The main development efforts of short-range wireless technologies are in the direction of faster speeds and lower costs. Cable replacement technology, such as infrared and Bluetooth, are effective only if they operate at comparable speeds to physical cables. Furthermore, these technologies must be inexpensive if they are to be integrated into cell phones, automobiles, and personal computers.

While speed and cost are important in robotics applications, a host of other factors are equally critical. Because many robots have the ability to move, wireless technologies must ideally have an extensive range and an ability to “see” through obstacles. Similarly, as robots are typically powered through batteries rather than power cords, low power requirements are necessary. In general, data transfer speeds are less important for most robotics applications, as communication usually consists of a series of commands. Where other wireless applications may stream video or audio data, robots typically do not

require this level of complexity. Finally, reliable and accurate connection and data transfer are important factors in choosing a wireless technology.

As subsequent sections of this thesis will cover wireless technologies in depth, some of the basic concepts of communications theory will now be discussed, as this will make understanding of higher-level concepts easier.

Serial data transmission is a method of sending and receiving data one bit at a time in a sequential fashion. In order to make equipment from different manufacturers compatible, the RS-232 standard was developed. This standard specifies connections between terminals and modems. A basic scheme for connecting two RS-232 ports involves connecting three wires between two devices. Two of these wires send and receive data in opposite directions, where a third is a common ground. (See figure 3.1 for a diagram of this operation.) To prevent two devices from trying to send and receive data along the same lines, devices which use pin two for output are known as DTE (Data Terminal Equipment), and devices which use pin two for input are called DCE (Data Communications Equipment). DTE devices are referred to as masters and DCE devices are called slaves.

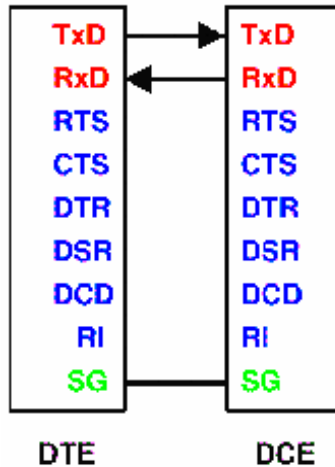


Figure 3.1: Basic RS-232 Functionality

Devices that talk to each other using RS-232 signals operate in full duplex mode, meaning that both devices can transmit and receive data simultaneously. Beyond the simple send and receive wires, optional flow control lines can be connected between two devices. These lines go high when the each device is ready to begin transmissions. The nine standard RS-232 connections are outlined in figure 3.2. In this figure, note the six flow control lines, two data lines, and common ground. See figure 3.3 for the configuration of these pins.

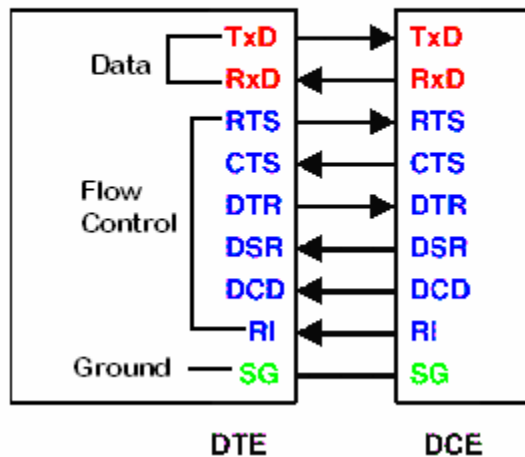


Figure 3.2: Full RS-232 Pinout

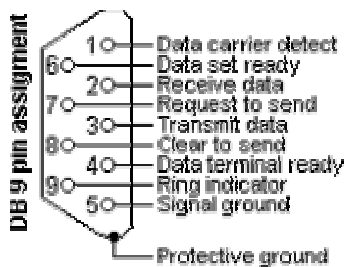


Figure 3.3 Physical layout of DB9 Pins for a DTE Device

In order to send and receive data over distances of up to 50 feet, RS-232 signals operate between +5 and +15 volts for high (logical one) signals and -5 to -15 volts for low (logical zero) signals<sup>3</sup>. Typical voltage levels for RS-232 signals are  $\pm 12$  Volts. For lower-power applications, Transistor-to-Transistor Logic (TTL) signals are used. These signals operate between zero and five volts, with any voltage above 3.5 volts operating as a logical one. TTL signals are typically used by microchips powered by a five-volt supply. These signals are used for short-range data transmission between logic devices.

Short-range wireless communication is a class of technologies meant primarily for indoor use over short distances<sup>4</sup>. In contrast to cellular systems, which are limited to two megabits per second because of constraints imposed by physical laws governing bandwidth, short-range wireless technology has no such limitation. The four main technologies currently available are WiFi (802.11), HomeRF, Infrared, and Bluetooth. The first two technologies, WiFi and Home RF, are designed for Local Area Networking between Personal Computers. These technologies are much more expensive than Bluetooth and Infrared technologies, and have an additional level of complexity to implement. See table 3.4 for a comparison of these technologies. The \$70-\$100 cost of HomeRF and the \$100-\$300 cost of WiFi make these technologies prohibitively expensive for most robotics applications. Furthermore, the high data transfer rates of these two technologies are typically unnecessary.

Bluetooth and Infrared, at \$5-\$15 and \$2 per unit respectively, emerge as the obvious choices for wireless robotics applications. It is important to note that these prices reflect costs for only the physical technology. Integrated circuit solutions for both Infrared and Bluetooth are more expensive. Both were designed as cable replacement technology, or simple point-to-point data transfer. Robotics communications needs are typically closer to cable replacement than local area network requirements. Moreover, Infrared and Bluetooth both have ad hoc networking capabilities. An ad hoc network is one in which all devices in the network are treated as peers. In other words, no single device communicates in a fundamentally different fashion than any other. Unlike a more structured network, where certain devices have enhanced capabilities over others, ad hoc networks are formed on the fly and require no base station to control network

communications. The capability of forming ad hoc networks is important for applications where teams of robots would have to form communications networks on the fly.

	<b>BLUETOOTH</b>	<b>HOMERF</b>	<b>IEEE802.11B/ Wi-Fi</b>	<b>IrDA</b>
Technology	RF	RF	RF	Infrared
Primary Use	Cable replacement and ad hoc device-to-device connections	Home or small office LANs	Corporate or campus LANs	Cable replacement and ad hoc device-to-device connections (narrow angle)
Maximum speed	1Mbps	10Mbps	11Mbps	4Mbps
Range	30 feet	150 feet	300 feet	3 feet
Connects through walls	Yes	Yes	Yes	No
Connection angle	360 degrees	360 degrees	360 degrees	30 degrees
Data support?	Yes	Yes	Yes	Yes
Native voice/ telephony support?	Yes	Yes	No	Yes
Frequency sharing	FHSS	FHSS	DSSS	
Requires separate access points (base stations)?	No	No	Yes	No
Susceptibility to RF interference	Medium	Medium	High	None
Power requirements	Low	High	High	Low
Manufacturing cost per device	\$15 now; dropping to \$5	\$70-\$120	\$100-\$300	\$2

Table 3.4: A comparison of wireless technologies. <sup>5</sup>

Infrared communications (IR) is currently the standard for most wireless robotics applications. Infrared is low-cost and widely used. IR can be found on over 150 million computing devices, and the technology is growing at a 40% annual rate<sup>6</sup>. Infrared's low power requirements and simple structure make it the industry standard for most robotics applications.

However, infrared technology has several serious drawbacks limiting its usefulness for robotics. The most significant of these is the issue of line of sight.

Infrared wavelengths are slightly longer than visible light, but share many properties with visible light. To transmit data between an IR transmitter and receiver, the devices must be visible to one another and within a “cone” of  $30^{\circ}$ . In other words, IR transmission will only take place if the two devices have direct line of sight and are both positioned within the correct connection angle. Furthermore, even in ideal conditions, IR has a maximum range of only three feet, which means that devices must come to within this range before they have the ability to communicate.

Bluetooth technology, developed by Ericsson Mobile Communications, a worldwide telecommunications company based in Sweden, is fast becoming the worldwide standard for short-range wireless communication. The technology was originally developed to easily connect cell phones with peripheral devices such as personal computers. However, other potential applications for home and office quickly emerged. Today, over 2000 companies have joined the Bluetooth Special Interest Group, which coordinates the royalty-free licensing of Bluetooth technology. Bluetooth is growing so rapidly that Merrill Lynch predicts that by 2005 there will be more than 2.1 billion Bluetooth compatible devices on the market<sup>7</sup>.

Bluetooth uses Radio Frequencies (RF) to transmit and receive signals in the ISM Band. This is an unlicensed frequency range operating between 2.4 and 2.48 GHz. The ISM band is used by a host of wireless applications including some cordless phones, WiFi and HomeRF technologies, baby monitors, garage door openers and microwaves. In order to prevent interference with other devices, Bluetooth employs a spread-spectrum frequency-hopping technique which takes a narrowband signal and spreads it over a broader portion of the available radio frequency band. This spreading is accomplished by

frequency-hopping, a technique developed for encryption that relies on the sender and receiver agreeing on a pre-defined “hop” sequence for frequency variation. Figure 3.5 shows a diagram of this functionality. Since only the intended receiver is aware of the transmitter’s hopping pattern, only that receiver can make sense of the data being transmitted. Spread spectrum frequency hopping ensures Bluetooth’s security and limits interference.

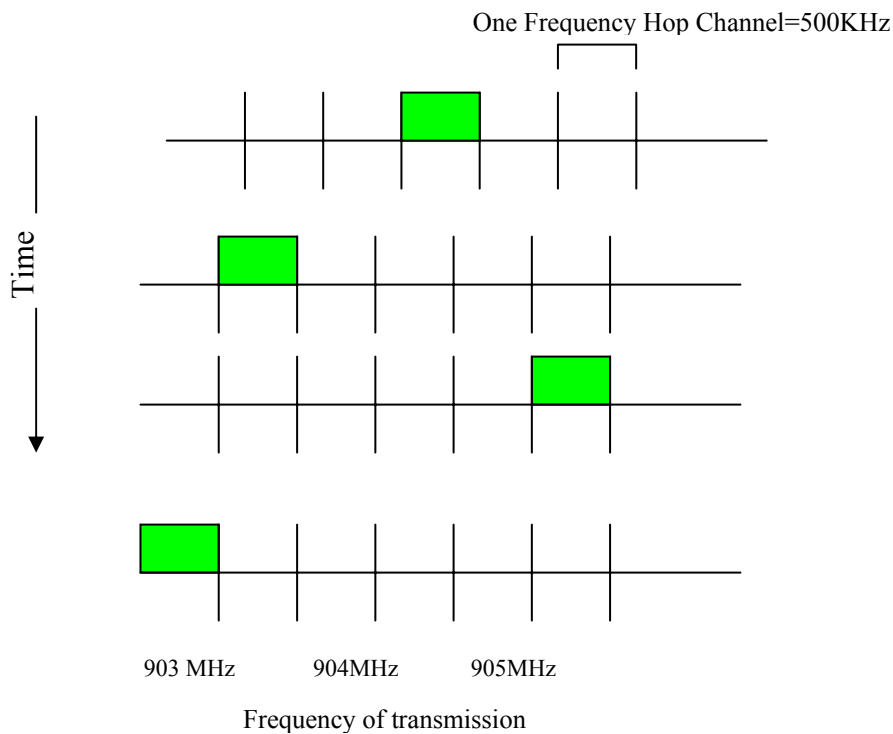


Figure 3.5: Spread Spectrum frequency hopping.

The Bluetooth 1.0 specification allows for a gross data transfer rate of 1Mbps. However, this is a theoretical maximum for half-duplex data transmission. Full duplex transmission is accomplished at 432.6 Kbps. The Bluetooth specification can support three synchronous voice channels at 64 Kbps each. Devices typically require 1mW of power to operate.

When two Bluetooth devices establish a connection, they create a network called a piconet. A piconet can contain up to eight devices, with one device serving as a master (DTE) and up to seven serving as slaves (DCE). All devices in a piconet share the same frequency-hopping scheme, which is established when a slave initially connects to a master. Bluetooth devices can operate in either a single slave piconet, in which a master connects to only one slave, multi-slave piconet, where a master connects to multiple slaves, or in a scatternet structure. A scatternet is enabled by masters and slaves with capability to connect to more than one device at a time (ie. two masters can connect to one another and a slave can connect to more than one device at a time). Scatternets are accomplished by masters having the ability to act as a slave if necessary when connected to another master. See figure 3.6 for a diagram of this functionality.



Figure 3.6 Piconet Structure

#### 4. Experimental Setup

Once the design criteria were established, several configurations were considered before a final design was chosen. These options will be briefly outlined here, and the final design covered in detail.

The first significant design decision was whether robots with different functionalities would exist within the problem. A solution where several mobile robots would move to the edge of predefined quadrants looking for light, then slowly converge

on the light source was considered (see figure 4.1). This design was rejected because of the complication and inaccuracy involved in calculating and moving to various quadrants.

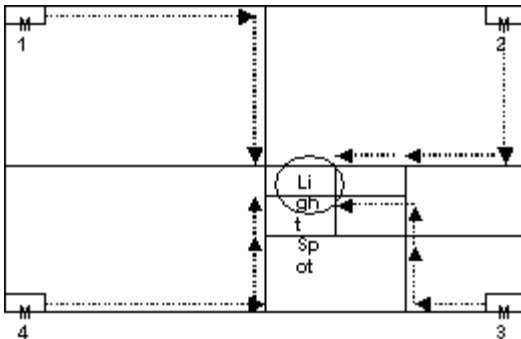


Figure 4.1: Quadrant Design

Another design was to incorporate robotic arms which could move in either the x or y direction and report their coordinates to a mobile robot. This robot, which had capabilities to move in both x and y direction, would extrapolate the “brightest spot” from the position of the robotic arms (see figure 4.2). While this solution is similar to the final design, the idea of moving robotic arms was rejected because of the inaccuracy of knowing the exact coordinates of the robotic arm at any given moment.

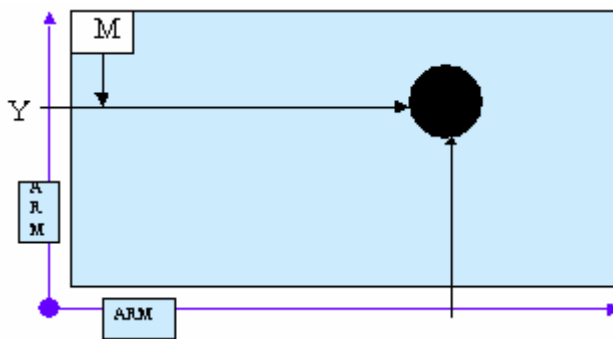


Figure 4.2: Robotic arm design

A third design involved mobile robots moving “elbow to elbow” along either the x or y direction and keeping track of who found the brightest light (see figure 4.3) was

considered. This design was rejected because it was not a feasible real-world solution to this problem. Huge teams of mobile robots searching areas “elbow to elbow” are not a cost-effective solution.

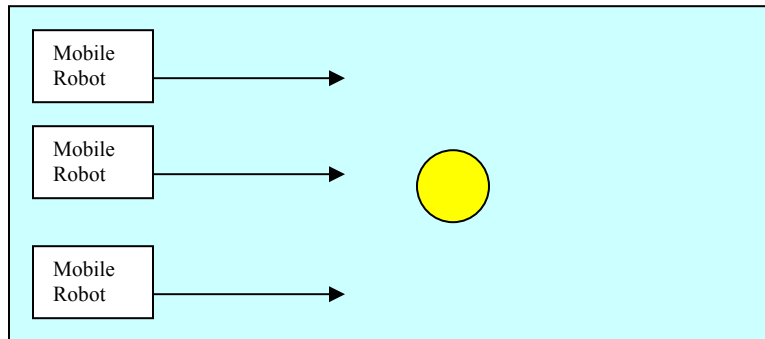


Figure 4.3: Elbow to Elbow Robotic Movement

The design which was finally settled upon involved both stationary and mobile robots. The robots that performed the initial light finding sweep were placed at the corners of the area to be searched. Two stationary robots performed sweeps through ninety degree angles and sent these coordinates to the mobile robots. The mobile robots subsequently extrapolated a rough location for the light. See figure 4.4 for a diagram of the stationary robot sweeping pattern and figure 4.5 for a diagram of the full design.

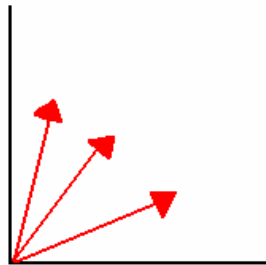


Figure 4.4: Stationary robot sweeping pattern

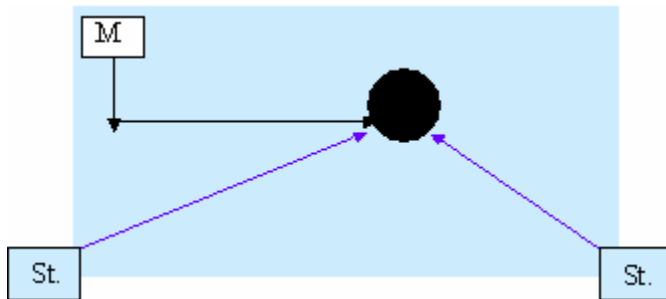


Figure 4.5: Final Design- Triangulation

Once the angle of brightest light was established, both stationary robots had to transmit this value. The information that had to be sent wirelessly was an angle measurement and a “left” or “right” identification to the mobile robot. The stationary robots initiated communications and transmitted both their location and the angle at which they detected the brightest light. See appendix one for the mathematical basis for this algorithm.

The mobile robot’s basic operation involved first waiting to receive two angle measurements from the stationary robots, calculating the coordinates at which these angles intersected, and then moving to those coordinates. When the mobile robot moved to this initial “rough” estimate of the location of brightest light, it then moved around in a circle, locating the direction of the brightest light, and returned to that angle. Finally, the robot used the three photoresistors on the front to establish whether the light spot was located left, right, or straight ahead, and moved towards its brightest spot of light accordingly. In this matter, the mobile robot could follow a moving spot of light. This “inching” behavior continued indefinitely. See figure 4.7 for a flowchart of the algorithm’s full functionality.

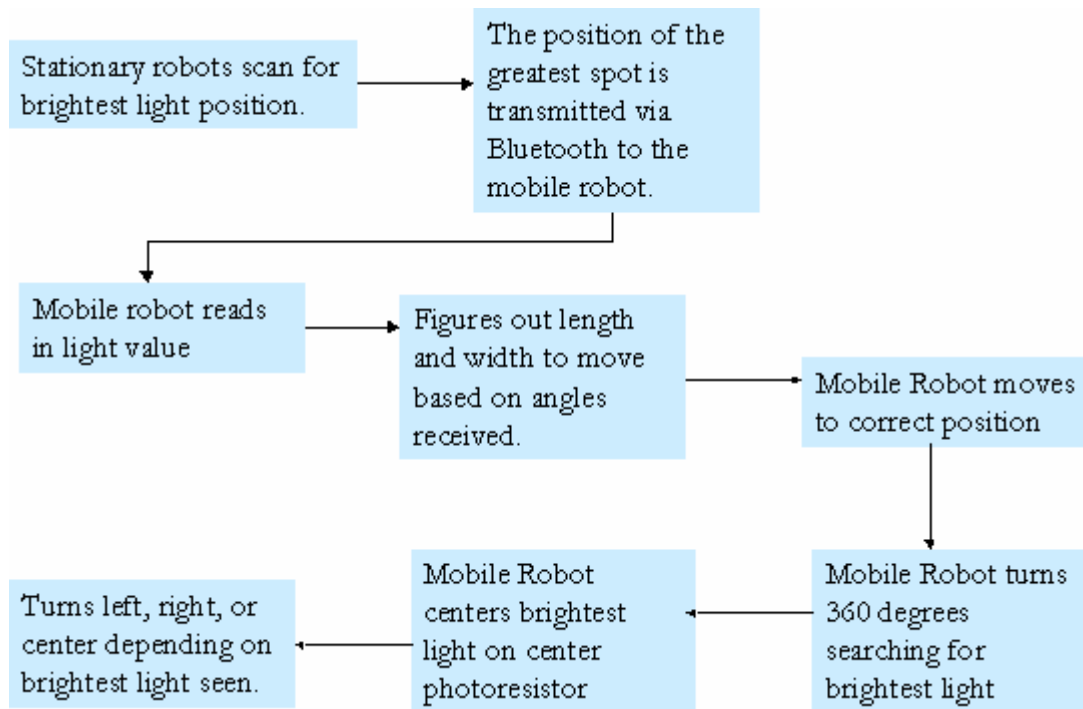


Figure 4.7: Flowchart of Functionality

A two-phase process for the development of communications architecture was conceived. The first phase involved implementing infrared wireless communications. Infrared was considered as a “base” technology because of its widespread use in the robotics industry. The design for infrared communications involved the mobile robot pointing its infrared receiver towards each of the stationary robots, and waiting to receive coordinates. Because this design included the mobile robot physically turning towards each of the receivers, only one piece of information had to be sent. In other words, the left robot did not need to identify itself as such because the mobile robot already was aware of its identity by turning towards that robot to receive coordinates.

The second phase of this project involved implementing Bluetooth communications. Bluetooth, as a more sophisticated technology, was developed second because it was postulated that our skills set would be greater after implementing infrared. Another reason was that the Bluetooth integrated circuits we used were not available for

purchase until December 2003. After researching the capabilities of Bluetooth networks, a multi-slave structure was designed. In this structure, the master device would be placed on the mobile robot, and it would connect to the two stationary robots upon startup. Because of the two-way nature of Bluetooth communications, once a connection was established, the stationary robots could send angle coordinates once they were available. It was postulated that the Bluetooth architecture would be upwardly compatible, with this communications structure enabling more sophisticated information exchanges between mobile and stationary robots.

## **5. Implementation**

Figures 5.1 and 5.2 contain systems diagrams of the interconnection between modules for this project. The stationary and mobile robots each contained a microprocessor, status LED's, a Voltage Regulator, Motors, A/D Converters, and a communications module. The stationary robot (figure 5.1) sent out a communications signal and read in light from one A/D converter. The mobile robot (figure 5.2) received a communications signal and read in light values from three A/D converters.

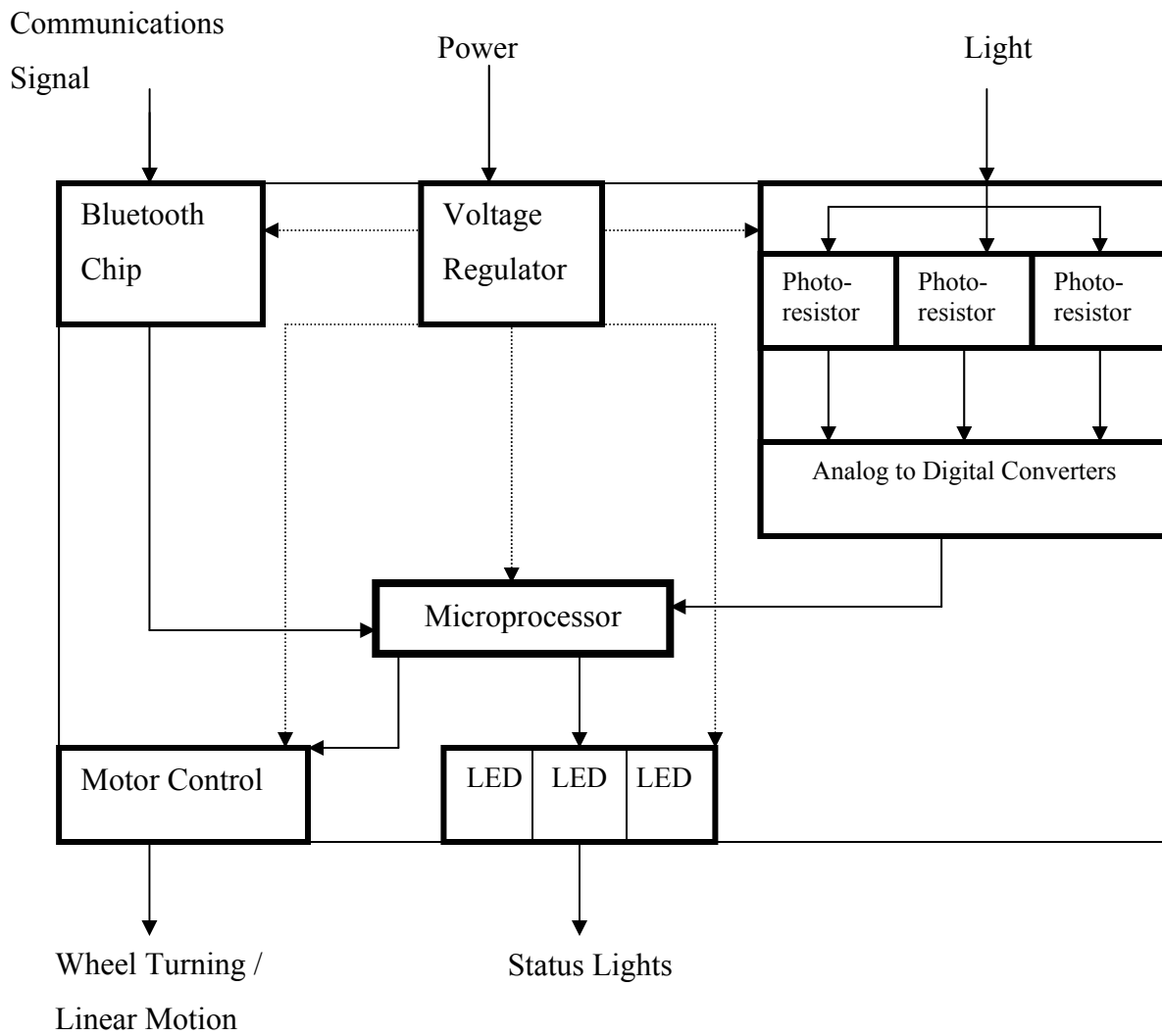


Figure 5.1 Mobile Robot Functionality

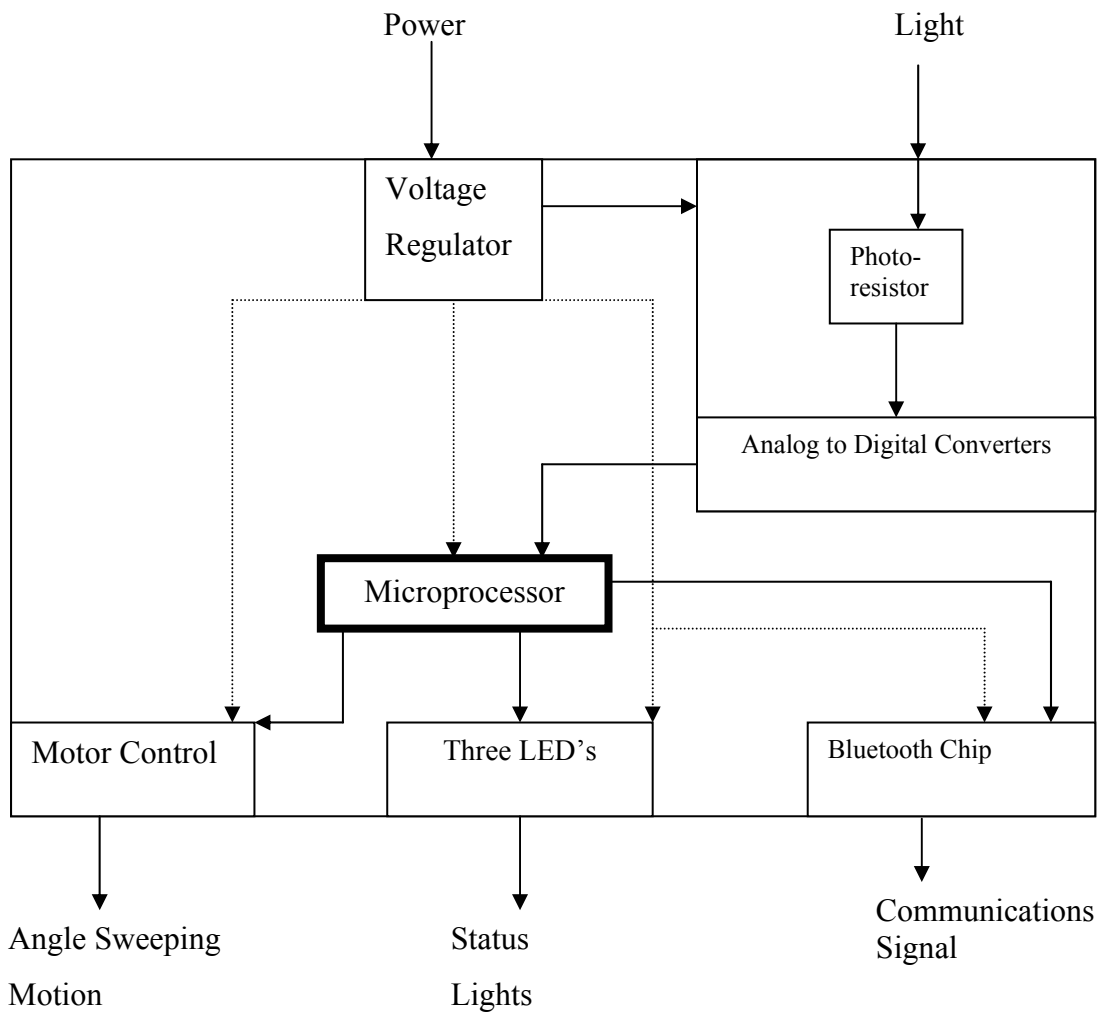


Figure 5.2: Stationary Robot Functionality

In implementing the design, it was decided to “over-engineer” the solution, in other words to add more sophisticated microprocessors, communications, and sensors than strictly necessary wherever possible. The reasoning behind this decision was to build robots which would be upwardly compatible as the problem became more involved. The purpose of this project was to develop algorithms which would be functional in a more complex robotics environment; this was enabled by the decision to over-engineer some of the components.

The microprocessors which controlled both the stationary and mobile robots had to be able to implement serial input and output, analog to digital conversion, motor control, perform mathematics algorithms, and monitor inputs and outputs. Object-oriented PIC16F877 chips (oopic) were used because of their high level of flexibility and the amount of technical support available from other hobbyists. The Oopic-R evaluation boards features 31 IOlines, built in analog to digital conversion, and serial communications capability. A voltage regulator, speaker, and status LED's were included on the evaluation board. Finally, object-oriented firmware specifically configured for robotics applications was embedded on the PIC microcontrollers. The oopic-R's advantages over the Basic Stamp include a larger number of IO Lines, networking capability, and a more flexible programming environment. This object-oriented platform resulted in simple coding algorithms. See appendix one for pinouts and technical diagrams of these microcontrollers, and appendix three for code.

The light sensors used in this project had to accurately detect light with enough accuracy to pinpoint a spot of brightest light. Using the analog to digital conversion pins of the oopic board, voltage over a photoresistor was converted to a meaningful value. Simple light circuits were constructed in which a current limiting resistor was put in series with a photoresistor, and the voltage over the photoresistor was measured. In the case of the stationary robots, one photoresistor was mounted to a servo motor in the front of the robot, enabling the  $90^{\circ}$  sweeps. In the case of the mobile robot, the values of three photoresistors were read in to determine whether the spot of brightest light was to the left, right, or straight ahead of the robot.

The motors on the stationary robots had to be able to sweep through  $90^{\circ}$  and accurately determine the angle at which the brightest light was detected. Servo motors were used for this purpose, allowing specific angle measurements to be sent to the robots, and several sweeps to be performed. In the case of the mobile robot,  $90^{\circ}$  stops were removed from servo motors to allow for constant rotation. The potentiometers were also removed from the servos, thereby disabling their ability to move to precise angles. Simple pulse-controlled motor control was implemented. In the case of the mobile robot, the motors controlled high friction foam wheels to prevent slipping. The motors were powered from a regulated five-volt power supply from the microcontroller.

The objective of the wireless communications portion of this project was to implement accurate, reliable, and secure serial communications. The first iteration was to implement wireless infrared communications. Infrared communications were implemented using a high speed infrared communications chip from Reynolds Electronics. This chip ramps up the voltage from TTL serial output to provide sufficient voltage such that an infrared LED will transmit signals to be picked up by an infrared receiver. See appendix one for a technical description of this setup.

Infrared receiver and transmitter circuits were built, but found to be highly unreliable in testing. The range was much shorter than the technical specifications suggested, and serial communications was only intermittently accurate. The “work around” solution in this first iteration was to send pulses instead of true serial signals. In other words, instead of sending several bytes of information serially, each stationary robot sent pulses corresponding to the position where it found the brightest light. The mobile robot would then interpret these pulses as angles and calculate the location of

brightest light. The most significant drawback to this method of communication was that the range was so small the mobile robot had to be manually moved in range of each of the stationary robots before communications was successful. If the robots missed one or more pulses, they would calculate incorrect coordinates, and move to the wrong spot. Furthermore, this method was not secure, as other infrared sources could interfere with the functionality of the robots. Because infrared communications was not secure, accurate, or reliable, a better method of wireless communications was necessary.

The second iteration of this project involved the implementation of Bluetooth wireless communications. Bluetooth master/slave devices were purchased from Wireless Futures BlueWAVE line of integrated circuits, which are designed to be “out of the box” Bluetooth solutions. In other words, this line of products is designed to be a simple one chip method of providing wireless cable replacement. These devices implement a point-to-point Bluetooth connection (i.e. no Multi-Slave Piconet structure). Three types of Bluetooth devices were purchased, a dedicated master (DTE) device, a dedicated slave (DCE) device, and a device which could act as either a master or slave device (DTE/DCE).

Several significant difficulties were overcome in the development of the Bluetooth communications which merit discussion. Of the three kinds of Bluetooth device, the dedicated master could not connect with the combined master/slave device. As a result, the original design, which placed the master device on the mobile robot, had to be modified such that two masters were placed on the stationary robots and each, in turn, initiated a connection with the slave device on the mobile robot. One obstacle was the difference in processing speeds between the Bluetooth device and the microprocessor.

The microprocessor sent commands significantly faster than the Bluetooth device could receive them, such that the microprocessor had to be programmed with a series of built-in delays to insure an accurate connection.

Another challenge between interfacing the microprocessor and Bluetooth was that certain programming constructs caused the PIC to stop running and wait until a serial value was received. In other words, reading in the serial receive variable caused the microprocessor to wait until serial data was received. This unusual functionality was overcome by reading the serial value into a dummy variable, and then looking at the dummy's value.

The communications algorithm for the two stationary robots is essentially the same, except that a five second delay was built into the second stationary robot's functionality to prevent interference. While this delay is not strictly necessary, the algorithms performed more efficiently with a staggered start. The reason for this is that Bluetooth devices, as a security measure, exponentially increase the time delay between pair fails. Therefore, if one device repeatedly tries to connect to a device which is already connected, each successive pair fail will result in a greater and greater time delay. See appendix one for a more detailed discussion of the technical aspects of the Bluetooth architecture.

## **6. Results**

The first iteration of this solution worked in a very limited way. The robot itself was big and bulky, communication was highly inaccurate, and motor control was a problem. The most successful component of this design was the light-sensing portion.

The photoresistors were a good way to detect light. However, it was identified that the robots' ability to detect light was hampered by interference by ambient lighting. The microprocessors were highly effective, and proved to be upwardly compatible.

Motor control, or the means to achieve robotic movement in a straight line, was functional but inaccurate. Servo motors with the potentiometers and 90<sup>o</sup> stops removed were used in both the mobile and stationary robots. Because the potentiometers were removed, the stationary robots relied on motor pulses to calculate angles. Furthermore, in the mobile robot, the motors were inaccurate and responded differently to the same power application, resulting in the mobile robots not going straight. This problem was simply due to motors which responded differently to the same power. One motor was much stronger than the other.

Infrared communications was successful in only a very limited fashion. Serial communications between infrared sender and receiver was so inaccurate that pulses were sent rather than true serial output. A coding scheme where numbers one through six represented six different angular positions was used. The stationary robots sent out a number of pulses corresponding to the position at which the brightest light was found. The mobile robot then input these pulses and interpreted them as angles. Sending pulses rather than true serial output led to accuracy problems. Furthermore, range and line of sight were obstacles.

In the second iteration, problems with motor control and communications were fixed. The physical robot was redesigned to be much smaller, resulting in a tighter turning radius. By enabling the potentiometers in the stationary robots, the angle sensing became much more accurate. The robots could sweep back to the original angle and zero

in on it rather than performing only one sweep. The servo motors were kept in the stationary robots, and the potentiometers were put back in the motor. They were disabled to allow constant rotation in the first iteration.

Bluetooth communications was implemented successfully. The communications portion of the project was successful 91% of the time. Once a connection was established, it was very accurate. However, establishing a connection was sometimes problematic. Bluetooth was secure, accurate, and low power. The approximately 100 ft range was much larger than necessary for this project, and the devices could “see through walls”. As a whole, all components were implemented to a high level, with the entire algorithm performed successfully 83 % of the time.

## **7. Statement of Limitations, Obstacles, and Deficiencies**

In previous sections, the technical aspects the construction phase of this project were outlined. The remainder of this investigation will focus on the successes and failures of various components, and the implications of this work for future robotics research. The four main components that will be discussed are light sensing, motor control, communications, and the effectiveness of the project design in general.

Light sensing, as discussed in the problem statement, was chosen for this project because of its similar behavior to chemical sensing. Light sources emanate out from a brightest spot, much like dissipation of chemicals from a source. Photoresistors are also cheap and easy to implement. However, unlike dangerous chemicals, light is abundant in the everyday world. The problem was made worse by our design. At close range, photoresistors are marvelous at detecting light sources. However, because we designed

the stationary robots to detect light from a long distance, it became harder and harder for the robot to distinguish the target from even dim sources of ambient light. Light sensing was very effective in the dark sound room we used for testing, but when the robots were demonstrated outside of this space, the stationary robots found the light with much less regularity. Because of these errors, it seems that our assumption that light sensing would substitute for chemical sensing was faulty. Better substitutes would have been sonar sensors, temperature sensors, or ultrasonic sensors to detect a target. Another possibility would have been to use wavelength-specific photoresistors, such that the robots searched for a certain color of light only.

Irrespective of what the robots were trying to find, another important consideration is that of motor control. Servo motors were a good choice for stationary robots because servos are designed for precise angular positioning. The choice to use servos in the mobile robots, however, led to a high level of inaccuracy in the motor control. When the 90° stops and the potentiometer were removed to allow for constant rotation, the motors became much more difficult to precisely control. One of the results of this difficulty was that the motors were not very well matched. In the case of the stationary robots, the speed in which they moved from angle to angle was unimportant. However, the fact that similar pulses led to different motor outputs meant that the mobile robot did not travel in a perfectly straight line over long distances. The result of this error was that the mobile robot tended to arrive at a slightly different destination than the coordinates to which it was directed. Stepper motors, which are designed for precise positioning within a constantly rotating drive-shaft would have been a better choice.

Furthermore, after accumulating errors, the mobile robot did not have any way of correcting its positioning. This limitation will be discussed later.

In terms of communications, the robots were limited by the technology available for purchase at this time. After researching Bluetooth technology, we determined that a multi-slave piconet structure would be appropriate for this application. This structure involves a master being connected to two slaves at once (see figure 3.4). In this case, the master robot would be the mobile robot, and the two slaves would be the stationary robots. The choice of master and slave was fairly arbitrary, because once a connection is established information can be sent both directions. The model of single mobile master initiating connection intuitively makes sense.

However, the only Bluetooth devices available for small-volume purchase implement a one-to-one master/slave connection. In other words, instead of one master connecting simultaneously to two slaves, each stationary robot had to take turns connecting with the mobile robot. Because each connection took up to two minutes to establish, the model of stationary and mobile robots constantly sending and receiving data was not feasible.

A discussion of the components of this project would not be complete without touching on the functionality of the algorithm in general. The inaccuracy of sensing light at a distance was discussed earlier, but another fundamental limitation of the algorithm was the fact that the mobile robot did not track its own coordinates. Once the mobile robot calculated and moved to the coordinates specified by the two stationary robots, it began “inching” behavior to track a moving spot of light. This behavior involved the mobile robot comparing the values of three photoresistors and keeping the brightest spot

of light straight ahead by turning and incrementally moving towards this light source. This algorithm was extremely effective in tracking a moving spot of light. However, once the robot began this behavior, it no longer had any sense of its location. In effect the mobile robot “got lost” after its initial movement.

Programming the mobile robot to keep track of its coordinates while inching would be fairly simple. However the robot tracking its own location would be of limited use because of accumulated errors in positioning due to inaccurate motor control. Because the robot already accumulated error after its first movement, each successive movement would increase the difference between the robot’s true and perceived position. Even with extremely accurate motors, some positional error exists. This error is a major obstacle to solving the “where am I” problem.

## **8. Most Critical Obstacles**

The implementation of this project revealed two critical problems that must be overcome to have robust design in terms of reliability and performance. The most pertinent is the overall communication architecture. While Bluetooth is an ideal architecture for a robotics environment because of its range, networking capabilities, and high level of accuracy, the learning from this project revealed a collection of hardware limitations, telemetry and physical obstacles, and deficiencies in algorithm logic. The second most critical obstacle is the lack of global awareness on the part of the mobile robot. Global awareness, or the capability of the robot to accurately track its own location, is integrally linked with the motor control on the mobile robot. The communications architecture is the more critical problem to solve because a fully

functional system would enable alternate methods of global awareness. The interaction between global awareness and communications will be discussed as well.

The issue of communication between devices was not anticipated as a problem at the outset of this project. Theoretically, a Bluetooth multi-slave piconet structure has the ideal functionality for this application. However, the devices available for purchase limited the communications architecture. Bluetooth was designed to be a cable replacement technology, and much of the technology available is designed for this application. While the Bluetooth specification does allow for one master to connect with multiple slaves, these capabilities are only realized in large-volume industrial applications. Many Bluetooth vendors will only sell volumes in excess of a thousand units. For this reason, we were limited to purchasing evaluation kits from one such vendor. These evaluation kits, at \$200 a unit, were much more expensive than the cost of Bluetooth chips (around \$5.00) and Bluetooth IC's (\$40-\$60)<sup>8</sup>. Furthermore, because many Bluetooth applications involve point-to-point connections, the devices purchased only had the capability of connecting one master with one slave. Devices that implement multi-slave piconet or even scatternet structures may be available in the near future, this functionality is currently not available for small volume purchases.

To accommodate the technology currently available, a single point-to-point master/slave connection was implemented. In this configuration, each master took turns connecting with the slave device. Because of slow connection algorithms which took anywhere from five seconds to two minutes, it was deemed impossible for the mobile robots to receive input from the stationary robots for every pulse of movement. The reasoning behind this decision was that if the stationary and mobile robots could take up

to four minutes for each new set of coordinates to be received, the movement of the mobile robot could never keep up with any spot moving at a reasonable rate.

The importance of establishing a multi-slave piconet structure was not obvious from the outset of the project because the slow initial connection speeds were not well documented. Furthermore, when Bluetooth technology was researched, it was unclear that devices with the capability of implementing a multi-slave structure would be unavailable. Few Bluetooth robotics applications have been attempted, and the lack of documentation for this technology was a serious limitation.

The secondary issue of global awareness on the part of the mobile robot can be initially regarded as a motor control issue. Accumulated error on the part of inaccurately matched motors is a limitation which could be solved either by motors which have a high level of accuracy, or enabling the mobile robot to reposition itself. While highly accurate motor control is clearly a desirable attribute to a mobile robot, the assumption that it is possible to find motors accurate enough to indefinitely operate without positional error is naïve. A robust design must include periodic error adjustment or the ability to reconfigure coordinates after an error is sensed. Several different error adjustment schemes are possible. The simplest scheme would be to mount touch sensors on the sides of the mobile robot and provide some stationary walls or obstacles which would provide fixed coordinates if they were bumped. If four stationary poles were provided on the table, the robot could move towards the nearest pole and move around until one of its sensors was pushed in, signaling a fixed location to the robot. A better scheme would be to place Hall effect sensors on one or both wheels. These sensors respond to the presence of a magnetic field by producing either a digital or analog output. If a magnet is placed at

one location on the wheel, the number of rotations of this wheel can be easily counted, and the distance the robot has traveled calculated in this way. The problem with this scheme is that a Hall Effect sensor only measures the number of rotations a wheel has made. This type of sensor would not be able to detect if the robot was tending to the left or to the right due to imperfectly matched motors. Furthermore, a Hall sensor could not account for slipping.

Schemes of enabling global awareness by precision motor control and sensing on the part of the mobile robot do not definitively solve the problem of accumulated error. Both of the schemes mentioned in the previous paragraph are not sufficient for the mobile robot to have a robust global awareness. A much more effective scheme involves the stationary robots tracking the mobile robot's position. In the current design, the stationary robots sweep through the area only once looking for the spot of brightest light. After communicating this value to the mobile robot, the stationary robots no longer contribute. If sonar sensors were placed on the front of the stationary robots, the stationary robots could search not only for the brightest spot, but also the nearest object detected (i.e. the mobile robot). Inaccurately matched motors would not be nearly as important if the mobile robot was constantly receiving information about its position from the stationary robots. The possible error from this scheme comes from errors in motor control on the stationary robots, and inaccuracies in the sonar sensors. Because highly accurate positioning has already been implemented on the stationary robot servo motors, and sonar sensing fundamentally has little error, this configuration has a likelihood of being highly effective.

Moving the responsibility of global awareness from the mobile to stationary robots comes back to the issue of Bluetooth multi-slave piconet behavior. The mobile robot's ability to initiate sonar sweeps (in effect to ask "where am I?"), and the response of the stationary robots (answering "I see you at this location.") is contingent on constant two-way communications being available to both receivers. Assuming multi-slave piconet devices will not be available until the demand for Bluetooth IC's for robotics applications increases, an alternative implementation may have to be substituted. A fourth Bluetooth module, placed on the mobile robot would allow the mobile robot to simultaneously maintain two open connections with each stationary robot. Because PIC chips only have the ability to send and receive one serial signal at a time, it is possible that a UART would have to be used to enable dual serial communications. A UART (universal asynchronous receiver-transmitter) is a device that translates parallel to serial data and serial to parallel data over several channels. While placing two transmitters on the mobile robot is costly and overly complex, this is a feasible work-around until the appropriate technology becomes available.

## **9. Suggestions for this Project**

In the previous section, I discussed several possible methods of improving communications and global awareness. Fixing or improving these two factors is the logical next step for this project. Because of the limitations of the mobile robot correcting its own global positioning, I believe that tracking the robot's position through sonar sensors on the stationary robots is the most robust design. Four major factors would have to be developed for this improvement to be made. The first is implementing

the algorithms and hardware associated with sonar sensing. Sonar sensing is fairly common within robotics and a large amount of information is available for interfacing with PICs. The next factor is interfacing the PIC with a UART to enable serial communications to two separate Bluetooth devices on the mobile robot. The PIC would have to be connected to a dual UART, which in turn would have to be connected to the two Bluetooth devices. The third factor is to develop algorithms to establish both Bluetooth connections. Because the work has already been done to connect two devices, this step should be fairly straightforward. Finally, algorithms for robotic behavior involving this additional level of complication would have to be developed. One possible behavior is outlined in figure 9.1.

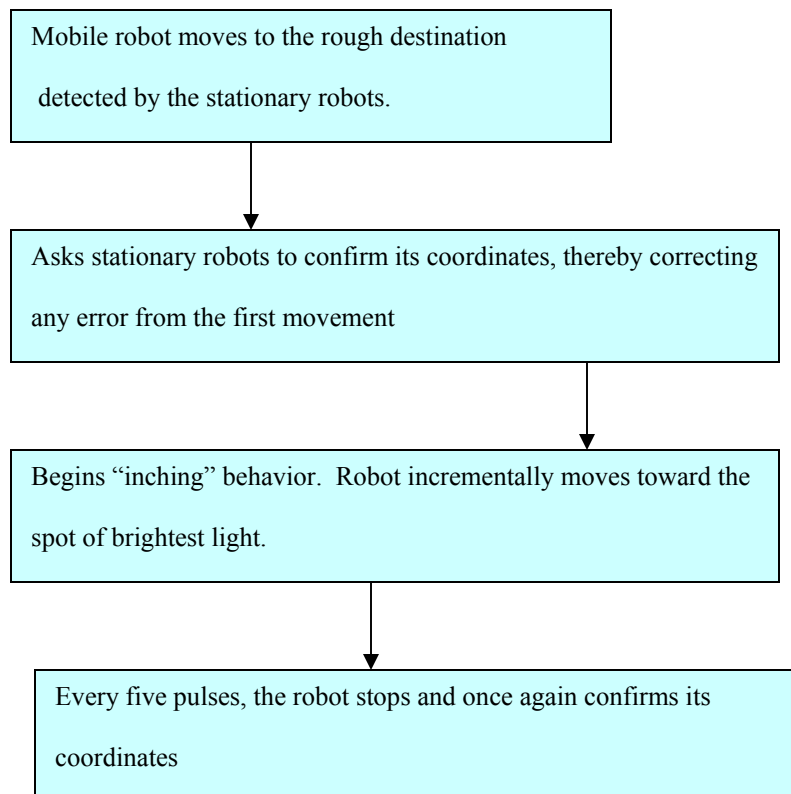


Figure 9.1: Potential Functionality of Robots

## 10. Lessons Learned

Beyond the technical aspects of this project, I think a few words about the lessons learned working as a team is in order. This project was an interdisciplinary effort between two electrical engineers, a mechanical engineer, and a child development major. My main work was on the communications architecture, but I got the opportunity to learn about a host of topics including servo motor gearing, the design and prototyping process for the construction of robots, the fundamentals of Lego robotics building, and the diverse ways in which robotics is applicable to the classroom. Additionally, I learned an incredible amount about microprocessors, power supplies, circuit building, analog-to-digital conversion, motor control, and project troubleshooting (to name a few). I am deeply indebted to my teammates for their enthusiasm and their willingness to keep at it until we got it right. A few lessons we learned which might be helpful for future participants of the Robotics Academy are as follows.

1. Model early, model often.
2. After prototyping, revisit your design to make sure is a good idea before beginning the main building process.
3. When dividing tasks, define each task to a high level of specificity. Match tasks to personal skills, capabilities, motivation, and interests.
4. Hold project status reviews to discuss progress and obstacles. Problems resolved by the group can move the project along more quickly or uncover a serious situation that must be addressed.
5. Communications is a vitally important part of any robotics problem. Implement a robust protocol and technology which is upwardly compatible.

6. It is often much more productive to work in a team than to work alone.
7. When everything stops working, go home and try it again tomorrow.
8. The best solution is often the simplest.

I hope that future participants of the Robotics Academy enjoy this amazing opportunity to learn their hearts out. It's been wonderful.

## **11. Acknowledgements**

I would like to extend a special thank you to my team-mates Louise Flannery, Emily Mower and Addie Sutphen. This project would not have been so great were it not for your commitment and enthusiasm. A special shout out goes to Emily, without whom I would still be trying to detect infrared light. Thanks to Matt Dombach, who has kept us all on track and been our fearless leader through thick and thin. I would like to thank Dr. Ron Lasser, who helped me go above and beyond what I thought was possible. Thanks to Dr. Chris Rogers, whose leadership makes the Robotics Academy possible. I would like to thank Dr. Joe Noonan for his flexibility and willingness. Thanks to James Caceese, for being the knight in shining armor of Bluetooth communications and Warren Gagosian, for giving us all the parts we needed with only a minimum of grumbling. Thanks to everyone in the TUFTL lab for being incredibly gracious about letting me use their computers and borrow their flashlights. Finally, I would like to thank my family and friends, who have had to listen to me talk about robots far beyond the bounds of normal human decency.

## 12. Bibliography

1. Murakami, Haruki. Underground. Trans. Alfred Birnbaum and Philip Gabriel. New York: Vintage International; 2001. p. 105.
2. Murakami, Haruki. Underground. Trans. Alfred Birnbaum and Philip Gabriel. New York: Vintage International; 2001. p. 225.
3. Gofton, P. Mastering Serial Communications. San Fransisco: Sybex; 1986. p. 14.
4. Leeper, David G. "A Long Term View of Short Range Wireless". CISE Portal. June 2001 p 39.  
[http://www.cs.huji.ac.il/course/2003/postPC/docs/Wireless\\_and\\_Bluetooth/Leeper\\_UWB\\_r6039.pdf](http://www.cs.huji.ac.il/course/2003/postPC/docs/Wireless_and_Bluetooth/Leeper_UWB_r6039.pdf)
5. Miller, M. Discovering Bluetooth. San Fransisco: Sybex, 2001. p. 208.
6. Miller, M. Discovering Bluetooth. San Fransisco: Sybex, 2001. p 196.
7. Miller, M. Discovering Bluetooth. San Fransisco: Sybex, 2001. p 15.
8. Miller, B. and C. Bisdikian. Bluetooth Revealed: The insiders guide to an open specification for global wireless communications. Upper Saddle River: Prentice Hall; 2001. p. 6.
9. oopic.com?

### 13. Appendix One: Technical Information

#### 13.1 Mathematical Basis for the Algorithm.

The math behind this design involves fairly simple trigonometry. See figure 13.1.1 for angle references. When a light at two angles is established, and the distance is known between two stationary robots, the law of sine's can be used to find two sides of the angle (A and B in this representation):

$$B/\sin(b) = A/\sin(a) = C/\sin(c) \quad [\text{equation 13.1.1}]$$

A and B can be found using the relationships in equation 13.1.2 and 13.1.3.

$$A = \sin(a) * C/\sin (c) \quad [\text{equation 13.1.2}]$$

$$B = \sin(b) * C/\sin (c) \quad [\text{equation 13.1.3}]$$

Once A and B are established, the horizontal and vertical distance to the point of intersection can be established using equations 13.1.4 and 13.1.5.

$$\text{Horizontal Distance} = A * \cos (a) \quad [\text{equation 13.1.4}]$$

$$\text{Vertical Distance} = A * \sin (a) \quad [\text{equation 13.1.5}]$$

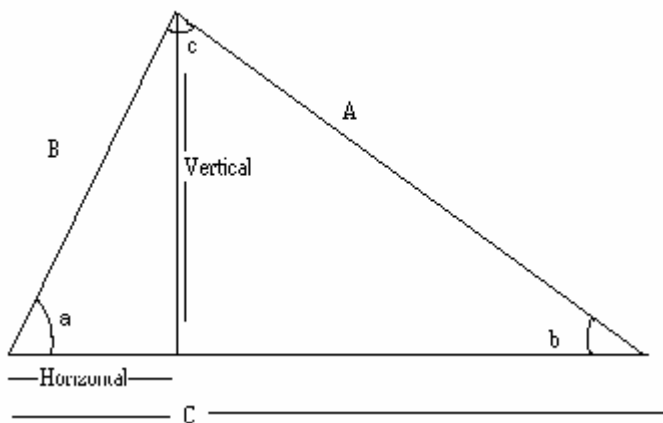


Figure 13.1.6: Geometry behind the light sensing algorithms.

## 13.2 Technical Information for Infrared Communications

The infrared transmitter circuit is found in figure 13.2.1. The oscillation was provided by a 20 MHz ceramic resonator, which was connected to pins 2-4, where pins two and three provided the oscillation, and the voltage on pin four determined the duty cycle. Data in was provided to pin six, and data out to pin seven. Two current limiting resistors and an NPN transistor inverted the output from the TX-IRHS chip.

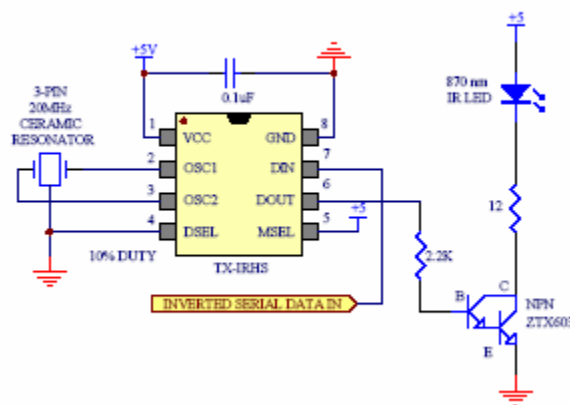


Figure 13.2.1 Infrared transmitter circuit.

Figure 13.2.2 indicates the receiving end of serial communications. Once again, a PNP transistor inverts the serial output, such that the output of this circuit is identical to the input. The data is inverted because inverted data, by transmitting a majority of ones, has less error.

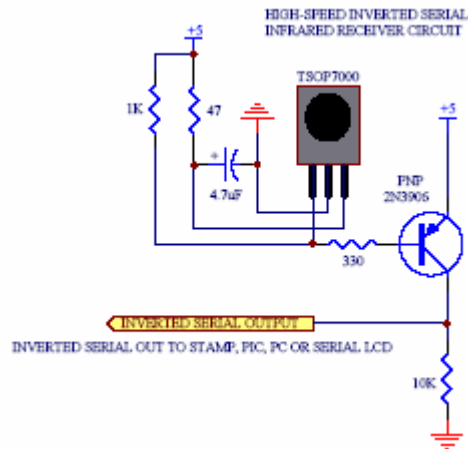


Figure 13.2.2 Infrared receiver circuit

### 13.3 Technical Information about Bluetooth Communications

BlueWave devices can be configured with AT strings, and can support RS232 or UART signals. See figure 13.3.1 for the pinout to the BlueWave devices. Pins 3, 4, 5, and 6 provided the flow control, transmit, and receive for the RS-232 pins, while pins 7, 8, 9, and 10 provided the flow control, transmit, and receive for the UART pins. The first step before connecting the Bluetooth devices to the microprocessor was to configure the device via a PC. Pins 5, 6, and GND were connected to a DB9 connector, which was connected to the COM port of a PC. The devices were configured via HyperTerminal. HyperTerminal is a Windows application used to send and receive communications directly through a PC's COM ports. In HyperTerminal, commands were tested and device "discoveries" initialized. In order to connect two Bluetooth devices, the unique authentication code for every slave needed to be "discovered". By connecting the master devices to the PC, and powering up the slave, it was possible to figure out which commands were necessary to connect two devices. Furthermore, the default speed for the Bluetooth devices was 19200 bps, which is faster than the maximum serial output of the PIC. The speed of all three Bluetooth devices was configured to 9600 bps via a PC.

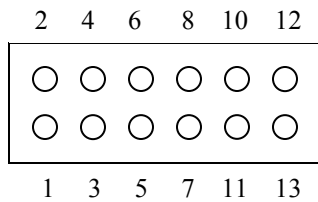


Figure 13.3.1 BlueWAVE pinout

Because a PIC sends and receives signals at TTL (between +5 volts and zero) levels, the UART pins were used to connect to the microprocessor and the RS-232 level signals were disabled (RS-232 operates between  $\pm 12$  Volts). Flow control was disabled for both RS-232 and UART communications by tying the CTS/RTS pins together. It was discovered that the flow control pins for the UART signals were not enabled, and that the RS-232 flow control had to be connected in order to send and receive TTL level signals. Very little documentation or support was available for the UART communications on the BlueWave devices.

Once the UART pins were connected to the microprocessor, serial commands could be sent or received to the Bluetooth devices. The set of commands outlined in figure 13.3.2 were used to establish a connection to the Bluetooth device.

### Bluetooth Connection Algorithm

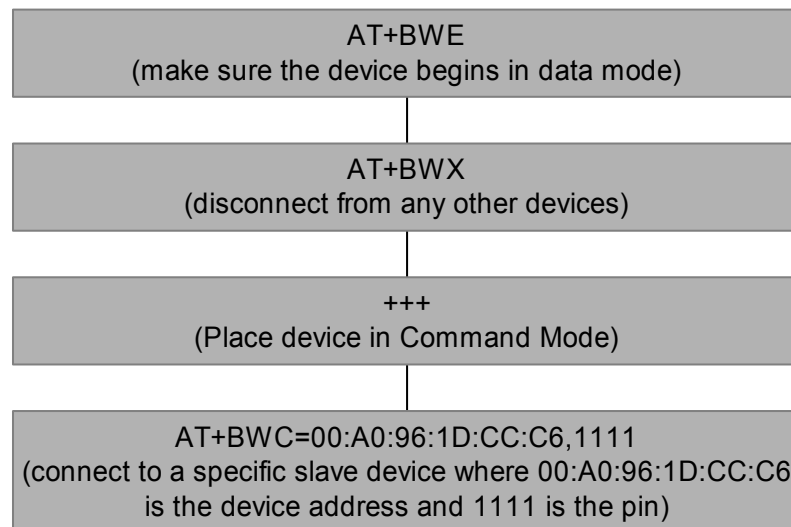


Figure 13.3.2: Bluetooth connection algorithm.

After the commands in figure 13.3.2 were sent, the program then waited for a “PAIR FAILED” or a “CONNECT”, which signifies whether the connection was successful. If the device was already connected to another device, a pair fail was returned. In the event of a pair fail, the connection was attempted again. In this way, the two stationary robots simultaneously tried to make a connection, and the unsuccessful one tried to connect over and over until it was successful. Once a successful connection was established, the master went into data mode (command “AT+BWE”), and transmitted the angle and its numerical identifier (i.e. each stationary robot was assigned a numerical identifier signifying the left or right robot). Finally, “AT+BWX” was sent to disconnect the devices after data was sent. These AT strings are specific to the BlueWAVE devices, although many IC’s have a similar command structure.

On the receiving end, the mobile robot initially looped until both values were received. This functionality was much simpler. If the left/right identifier was received,

the left/right angle value was set. Once both angle values were nonzero, the robot calculated the angles and moved to the corresponding coordinates. Because of the First-In-First-Out structure of the Bluetooth stack, the receiving device got the data bytes in the reverse order they were sent. In other words, the transmitter (stationary robot) sent the angle followed by the numerical identifier, and the receiver (mobile robot) received the numerical identifier followed by the angle.

## 14. Appendix Two: Pictures

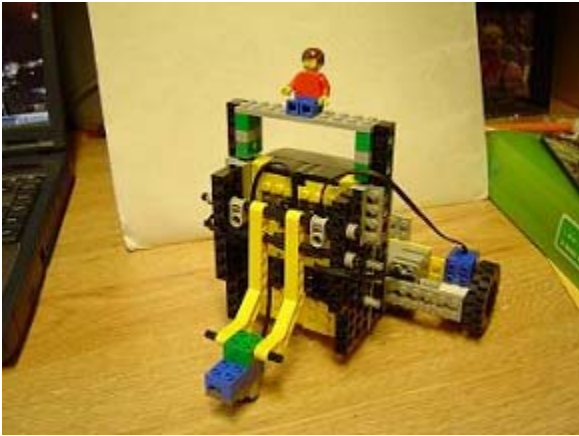


Figure 14.1: Lego prototype of Mobile Robot

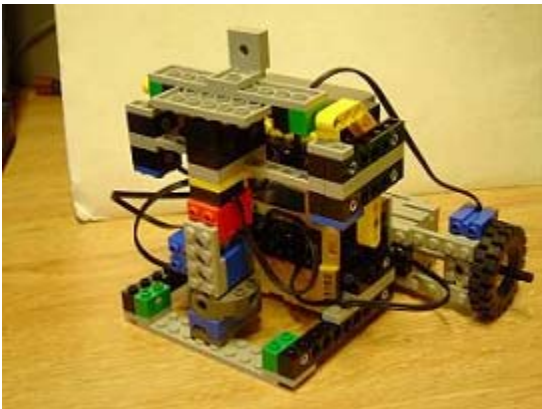


Figure 14.2: Lego prototype of Stationary Robot

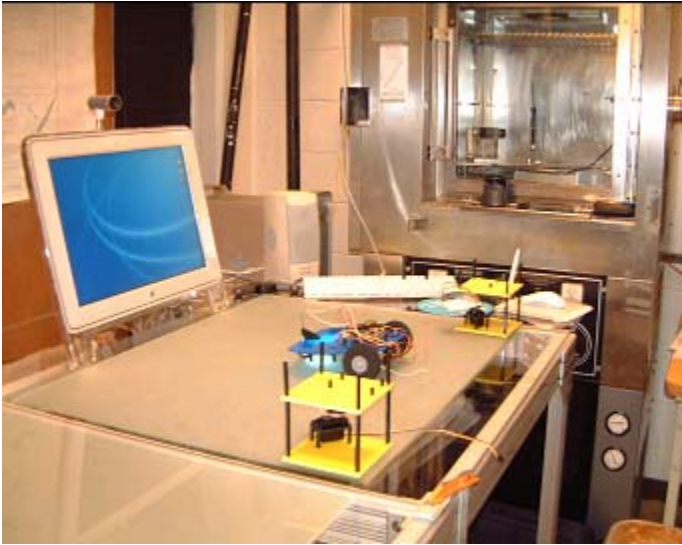


Figure 14.3 Stationary and Mobile Robots on the RoboTable

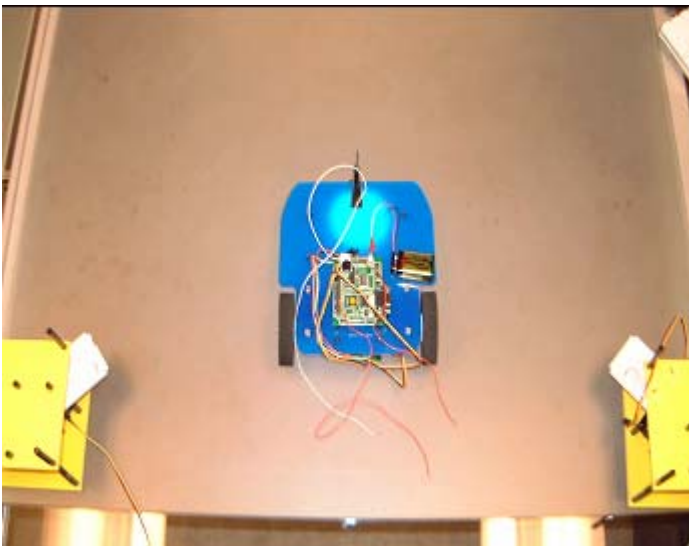


Figure 14.4 Initial Stationary and Mobile Robot designs.s

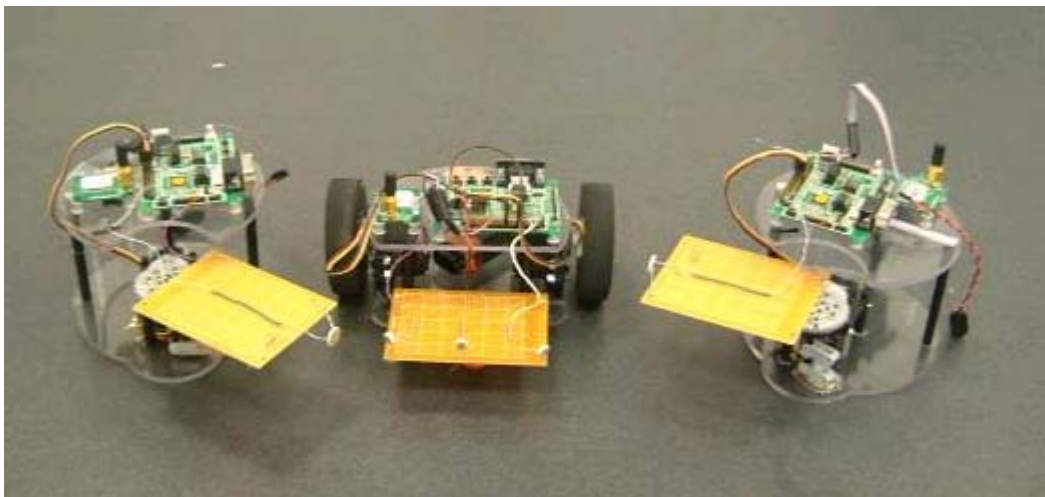


Figure 14.5: Final Stationary and Mobile Robots.

### 15. Appendix Three: Figures

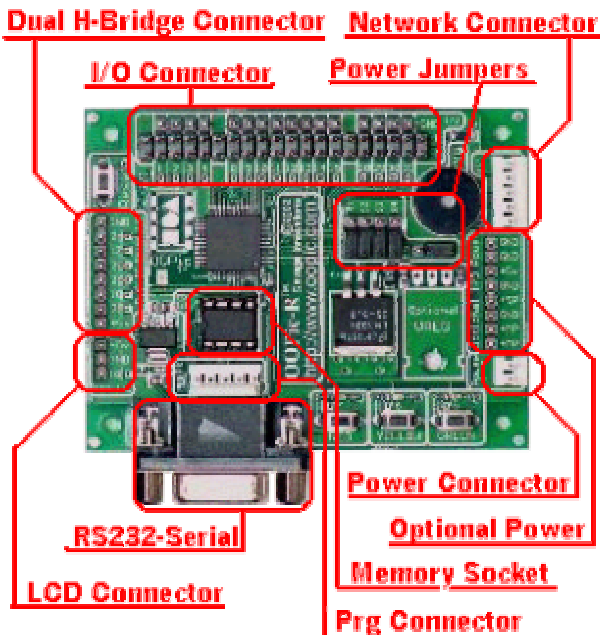


Figure 16.1 OOPic-R board

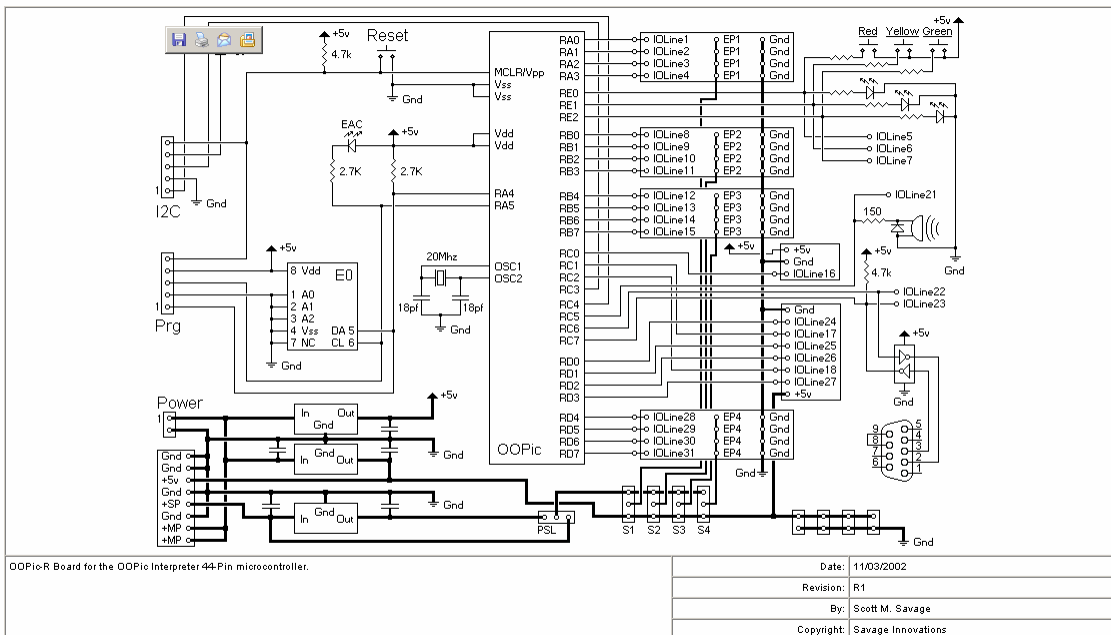


Figure 16.2 pinout to oopic-R board

## 16. Appendix Four: Code

### 16.1 Mobile Robot Code

```
'Mobile Robot Code
'By Emily Mower and Laurel Hesch

'-----bluetooth variables-----

Dim output As New oSerialX
Dim input As New oSerialX

'-----movement variables-----

Dim RightMotor as New oServo
Dim LeftMotor as New oServo
Dim Counter as New oByte
Dim Counter2 as New oByte
Dim RH as New oBit
Dim FH as New oBit
Dim LH as New oBit
Dim Right as New oA2D
Dim Front as New oA2D
Dim Left as New oA2D
Dim HighestLight as New oByte
Dim Position as New oByte
Dim Dummy as New oByte
Dim Angle1 as new oByte
Dim Angle2 as new oByte
Dim TableD as new oByte
Dim TableH as new oByte
Dim Vertical as new oWord
Dim Horizontal as new oWord
Dim H1 as new oWord
Dim J as new oWord
Dim K as new oWord
Dim SinAngle1 as new oWord
Dim SinAngle2 as new oWord
Dim CosAngle1 as new oWord
Dim SinRemainder as new oWord
Dim SumAngles as new oWord
Dim Extra as new oByte

'-----LED Variables-----
Dim Red as New oDio1
Dim Yellow as New oDio1
Dim Green as New oDio1

Sub Main()

'delay for bluetooth to stabilize before program starts

  oopic.delay=500

'LED initialization

  Red.IOLine=7
  Red.Direction=cvoutput
```

```

yellow.IOLine=6
yellow.Direction=cvoutput

green.IOLine=5
green.Direction=cvOutput

'Serial initialization for Bluetooth

input.IOLineS = 14 'Data from Bluetooth IC
input.IOLinef = 0 'Flow Control Disabled
input.Baud = cv9600

output.IOLineS = 15 'Data to Bluetooth IC
output.IOLinef = 0 'Flow Control Disabled
output.Baud = cv9600

input.operate=1
output.operate=1

'-----recieve angle values from stationary robots-----

angle1=0
angle2=0

dim stop as new obit

stop=0

do until stop=1

if input.value=65 then 'DTE validation
    angle1=input.value
    red=1
elseif input.value=75 then 'SM validation
    angle2=input.value
    green=1
end if

'if both angles have been recieved, exit loop

if angle1>127 then
    if angle2>127 then
        stop=1
    end if
end if

loop

'-----math-----

'Initialize motors

Right.IOLine = 1
Front.IOLine = 2
Left.IOLine = 3
Right.Operate = 1
Front.Operate = 1
Left.Operate = 1

RightMotor.IOLine=8
LeftMotor.IOLine=11

```

```

'All three LED's blink when value recieved

Red.Value=1
Green.Value=1
Yellow.Value=1

OOPic.Delay=100

Red.Value=0
Green.Value=0
Yellow.Value=0

TableD=30
TableH=24
'Angle1=145
'Angle2=160
if Angle1<(255+-64) then
    CosAngle1=(sin(63+Angle1)+-124)*8/10
End If
Extra=64
Dummy=0
If Angle1>(255+-64) then
    For J=0 to 63 step 1
        if Angle1<255 then
            Angle1=Angle1+1
        End If
        if Angle1=255 then
            Angle1=0
            Extra=(63-J)
            J=65
        End If
    Next J
    For K=0 to Extra+-1 step 1
        Angle1=Angle1+1
    Next K
    CosAngle1=(sin(Angle1)+-124)*8/10
End If
Green.Value=1
SumAngles=Angle1+Angle2+-2*128
'I=sin(SumAngles)
SinAngle1=(sin(Angle1)+-127)*8/10
SinAngle2=(sin(Angle2)+-127)*8/10
SinRemainder=((sin(255-SumAngles))+-127)*8/10
If SinRemainder=0 then
    SinRemainder=1
End If
H1=TableD*SinAngle2/SinRemainder
Vertical=H1*SinAngle1/100
Horizontal=H1*CosAngle1/100

Horizontal=TableD-Horizontal
Vertical=TableH-Vertical

RightMotor.Center=28
RightMotor.Value = -127
LeftMotor.Center=28
LeftMotor.InvertOut = 0

Green.Value=0

```

```

For Dummy=0 to Horizontal step 1
    RightMotor.Operate = 1
    LeftMotor.Operate = 1
    OOPic.Delay=8
    RightMotor.Operate=0
    LeftMotor.Operate=0
Next Dummy

For Dummy =0 to 10 step 1
Next Dummy

For Dummy =0 to 30 step 1
    RightMotor.Center=28
    RightMotor.Value=127
    RightMotor.Operate = 1
    LeftMotor.Center=28
    LeftMotor.InvertOut = 1
    LeftMotor.Operate = 1
Next Dummy

RightMotor.Operate=0
LeftMotor.Operate=0

RightMotor.Center=28
RightMotor.Value = -127
LeftMotor.Center=28
LeftMotor.InvertOut = 0

For Dummy=0 to Horizontal step 1
    RightMotor.Operate = 1
    LeftMotor.Operate = 1
    OOPic.Delay=8
    RightMotor.Operate=0
    LeftMotor.Operate=0
Next Dummy

'SETS A BASE VALUE FOR HIGHESTLIGHT
HighestLight.Value=100
Position.Value=0

'THIS SECTION ROTATES THE ROBOT 360 DEG.
'CHECKING FOR THE AREA OF BRIGHTEST LIGHT
For Counter = 0 to 31
    'THE MOVING PART
    For Dummy =0 to 2 step 1
        RightMotor.Center=28
        'RightMotor.InvertOut = 0
        RightMotor.Value=127
        RightMotor.Operate = 1
        LeftMotor.Center=28
        LeftMotor.InvertOut = 1
        LeftMotor.Operate = 1
    Next Dummy
    RightMotor.Operate=0
    LeftMotor.Operate=0

    'THE LIGHT PART
    If Front.Value>HighestLight.Value Then
        HighestLight.Value=Front.Value
        Position.Value=Counter.Value
    End If
Next Counter

```

```

RightMotor.Operate=0
LeftMotor.Operate=0
Oopic.Delay=25

```

```
'THIS BRINGS THE ROBOT BACK TO THE SPOT OF BRIGHTEST LIGHT
```

```

For Counter = 0 to Position.Value
  'THE MOVING PART
  For Dummy =0 to 2 step 1
    RightMotor.Center=28
    'RightMotor.InvertOut = 0
    RightMotor.Value=127
    RightMotor.Operate = 1
    LeftMotor.Center=28
    LeftMotor.InvertOut = 1
    LeftMotor.Operate = 1
  Next Dummy
  RightMotor.Operate=0
  LeftMotor.Operate=0
Next Counter

```

```

For Counter2 = 0 to 10 step 1
  'ROBOT GOES FORWARD FOR 10 BEATS
  RightMotor.Center=28
  RightMotor.InvertOut = 0
  RightMotor.Operate = 1
  LeftMotor.Center=28
  LeftMotor.InvertOut = 0
  LeftMotor.Operate = 1

```

```
OOPic.Delay=20
```

```

RightMotor.Operate=0
LeftMotor.Operate=0

```

```
'THIS IS WHERE I FIND OUT WHICH SENSOR SEES THE BRIGHTEST LIGHT
```

```

HighestLight.Value=Right.Value
RH.Value=1
LH.Value=0
FH.Value=0

```

```

If Front.Value > HighestLight.Value Then
  HighestLight.Value = Front.Value
  RH.Value=0
  FH.Value=1
End If

```

```

If Left.Value > HighestLight.Value Then
  RH.Value=0
  FH.Value=0
  LH.Value=1
End If

```

```

If RH.Value = 1 Then
  For Dummy =0 to 15 step 1
    'RightMotor.Center=28
    'RightMotor.InvertOut=0
    'RightMotor.Value=127
    RightMotor.Operate = 0
    LeftMotor.Center=28
    LeftMotor.InvertOut = 0
  Next Dummy

```

```

        LeftMotor.Operate = 1
        Green.Value=1
        Red.Value=1
        Yellow.Value=1
    Next Dummy
    RightMotor.Operate=0
    LeftMotor.Operate=0
End If

If LH.Value =1 Then
    Green.Value=0
    Red.Value=0
    Yellow.Value=0
    For Dummy =0 to 10 step 1
        RightMotor.Center=28
        RightMotor.Value=-127
        'RightMotor.InvertOut=0
        RightMotor.Operate = 1
        LeftMotor.Center=28
        LeftMotor.InvertOut = 1
        LeftMotor.Operate = 1
    Next Dummy
    RightMotor.Operate=0
    LeftMotor.Operate=0
End If

If FH.Value=1 Then
    Green.Value=0
    Red.Value=0
    Yellow.Value=0
End If

OOPic.Delay=20

Next Counter2

Red.Value=0
Green.Value=0
Yellow.Value=0

do
loop

End Sub

```

## 16.2 Stationary Robot Code #1 (DTE Device)

```

'DTE
'By Laurel Hesch and Emily Mower
'This is the code for the right stationary robot

'initialization for LED's

Dim Green as New oDio1
Dim Red as New oDio1
Dim Yellow as New oDio1

'initialization for light finding

```

```

Dim Motor as New oServo
Dim Dummy as New oByte
Dim Photo as New oA2D
Dim MaxPhoto as New oByte
Dim Place as New oByte
Dim Value1 as New oWord
Dim ValueF as New oWord

```

```
'each servo has slightly different min/max vals
```

```

Const Mn=48
Const Mx=95
Const Dly=1

```

```
'initialization for serial communication
```

```

Dim output As New oSerialX
Dim input As New oSerialX

```

```
Sub Main()
```

```
'init. LED's
```

```

Red.IOLine=7
Red.Direction=cvoutput

```

```

yellow.IOLine=6
yellow.Direction=cvoutput

```

```

green.ioline=5
green.direction=cvoutput

```

```
'-----initialize bluetooth-----
```

```

input.IOLineS = 14
input.IOLineF = 0
input.Baud = cv9600

```

```

output.IOLineS = 15
output.IOLineF = 0
output.Baud = cv9600

```

```

input.operate=1
output.operate=1

```

```

output.String="AT+RESET" 'reset the device
output.value=13

```

```
oopic.delay=20
```

```
'-----stationary robot light finding-----
```

```
Motor.IOLine=13
```

```

motor.center=0
motor.value=mx 'move robot to max position

```

```

motor.operate=1

Photo.IOLine=4
Photo.Operate=1

MaxPhoto.Value=Photo.Value
Place.Value=0
OOPic.Delay=200

For Dummy =mx to mn step -1

    motor.value=dummy

    if Photo.Value < MaxPhoto.Value then
        MaxPhoto.Value=Photo.Value
        Place.Value=Dummy
        Red.Value=1
    elseif Photo.Value > MaxPhoto.Value then
        Red.Value=0
    End If
    oopic.delay=dly
Next Dummy
motor.value= mx
oopic.delay =50

motor.value=place

'TOTAL SPREAD IS 64 (128-192)
Value1=1.3617*place+62.6383

'This value is sent
ValueF=(12800+Value1)/100

OOPic.Delay=50
Red.Value=0
Motor.Operate=0

'-----bluetooth send-----

oopic.delay=500 'delay for bluetooth to stabilize before program starts

'go into data mode before delay

output.String="AT+BWE"
output.value=13

oopic.delay=500

output.String="+"
oopic.delay=15
output.String="+"
oopic.delay=15
output.String="+"
oopic.delay=15
output.value=13          ' 13 is code for carriage return

```

'DTE should be in cmd mode

```
output.String="AT+BWX" 'disconnect
output.value=13
```

oopic.delay=100 'delay for DCE to disconnect and reset itself before we try to connect (this delay is important!)

'connect to mobile robot

```
output.String="AT+BWC=00:A0:96:1D:CC:C6,1111"
output.value=13
```

oopic.delay=10 ' delay to prevent echoed C in AT+BWC... from being mistaken for C in CONNECT

'loop until devices connect

```
dim stop as new oByte
dim i as new oByte
stop = 0
do until stop = 1
  i = input.value
  If i=67 then '67 is "C" the first letter in "CONNECT"
    green.value=1
    stop = 1
  elseif i=80 then ' 80 is "P" for "PAIR FAILED"
    ' did not get connect, red light on halt
```

```
red.Direction=cvoutput
red.invert
'exit sub
```

```
  'try to connect again
  output.String="AT+BWC=00:A0:96:1D:CC:C6,1111"
  output.value=13
```

```
end if
loop
'we are connected
```

```
green.invert
yellow.invert
```

```
output.String="AT+BWE"
output.value=13
'we are in data mode
```

```
red.invert
yellow.invert
```

'To be safe, send values 20 times

for i=1 to 20 step 1

```
red.invert
yellow.invert
```

```
'Send A and then valueF
output=valueF
```

```

        output=65

        green.invert
        oopic.delay=10
        green.invert
next i

green=0
yellow=0
red=0

oopic.delay=200

'place in data mode

    output.String="+
    oopic.delay=15
    output.String="+
    oopic.delay=15
    output.String="+
    oopic.delay=15
    output.value=13          ' 13 is code for carriage return

oopic.delay=200

output.String="AT+BWX" 'disconnect
output.value=13

oopic.delay=20

output.String="AT+BWE" 'back into data mode
output.value=13

End Sub

```

### 16.3 Stationary Robot Code #2 (S/M Device)

```

'S/M Code
'By Laurel Hesch and Emily Mower
'This is the code for the left stationary robot

'initialize LED's

Dim Green as New oDio1
Dim Red as New oDio1
Dim Yellow as New oDio1

'initialization for light finding

Dim Motor as New oServo
Dim Dummy as New oByte
Dim Photo as New oA2D
Dim MaxPhoto as New oByte
Dim Place as New oByte
Dim Value1 as New oWord

```

```

Dim ValueF as New oWord

'Minimum and Maximum angle values for this
' servo motor.

Const Mn=48
Const Mx=93
Const Dly=1

'initialization for serial communication

Dim output As New oSerialX
Dim input As New oSerialX

Sub Main()

'init. LED's

  Red.IOLine=7
  Red.Direction=cvoutput

  yellow.IOLine=6
  yellow.Direction=cvoutput

  green.ioline=5
  green.direction=cvoutput

'initialize serial communications

  input.IOLineS = 15
  input.IOLinef = 0
  input.Baud = cv9600

  output.IOLineS = 14
  output.IOLinef = 0
  output.Baud = cv9600

  input.operate=1
  output.operate=1

  output.String="AT+RESET" 'reset the device
  output.value=13

  oopic.delay=20

'-----stationary robot light finding-----

  Motor.IOLine=13

  motor.center=0
  motor.value=mx 'move motor to maximum value
  motor.operate=1

  Photo.IOLine=4 'initialize A/D conversion
  Photo.Operate=1

  MaxPhoto.Value=Photo.Value
  Place.Value=0

```

```

OOPic.Delay=200

'sweep through 90 degrees
For Dummy =mx to mn step -1
    motor.value=dummy
    'if a max light value is found, store location
    if Photo.Value < MaxPhoto.Value then
        MaxPhoto.Value=Photo.Value
        Place.Value=Dummy
        Red.Value=1
    elseif Photo.Value > MaxPhoto.Value then
        Red.Value=0
    End If
    oopic.delay=dly
Next Dummy

motor.value= mx
oopic.delay =50

motor.value=place

'TOTAL SPREAD IS 64 (128-192)

'Spread values over 128-192 range
value1=1.4222*place+59.7333

'Scale value to be sent
ValueF=(12800+Value1)/100

OOPic.Delay=50
Red.Value=0
Motor.Operate=0

'-----bluetooth send-----

oopic.delay=500 'delay so that DTE has a head start

' Place device in command mode

output.String="+
oopic.delay=15
output.String="+
oopic.delay=15
output.String="+
oopic.delay=15
output.value=13          ' 13 is code for carriage return

output.String="AT+BWX" 'disconnect
output.value=13

oopic.delay=100 'delay for DCE to disconnect and reset itself before we try to connect (this delay is important!)

```

'Look for Mobile Robot

```
output.String="AT+BWC=00:A0:96:1D:CC:C6,1111"
output.value=13
```

oopic.delay=10 ' delay to prevent echoed C in AT+BWC... from being mistaken for C in CONNECT

'Keep trying to connect after each pair failed recieved

```
dim stop as new obyte
dim i as new oByte
stop = 0
do until stop = 1
  i = input.value
  If i=67 then '67 is "C" the first letter in "CONNECT"
    green.value=1
    stop = 1
  elseif i=80 then ' 80 is "P" for "PAIR FAILED"
    ' did not get connect, red light on halt
    red.Direction=cvoutput
    red.value=1
  'exit sub

  'try to connect again
  output.String="AT+BWC=00:A0:96:1D:CC:C6,1111"
  output.value=13
```

```
end if
loop
'we are connected
```

```
green=1
```

```
output.String="AT+BWE"
output.value=13
'we are in data mode
```

```
red.invert
yellow.invert
```

```
for i=1 to 20 step 1
```

```
red.invert
yellow.invert
```

'Send A and then valueF (FIFO structure)

```
output=valueF
'output=140
output=75
```

```
green.invert
oopic.delay=10
green.invert
next i
```

```
green=0
yellow=0
red=0
```

```
oopic.delay=200
```

```
'go into command mode after data is sent to disconnect
```

```
output.String="+"
```

```
oopic.delay=15
```

```
output.String="+"
```

```
oopic.delay=15
```

```
output.String="+"
```

```
oopic.delay=15
```

```
output.value=13          ' 13 is code for carriage return
```

```
oopic.delay=200
```

```
output.String="AT+BWX" 'disconnect
```

```
output.value=13
```

```
oopic.delay=20
```

```
output.String="AT+BWE" 'Return to data mode
```

```
output.value=13
```

```
End Sub
```