Robot Navigation Project

Introduction to Design Team B:
William Frankland, Jason Michnovicz,
John Schilling, Beryl Wootton

CONTENTS

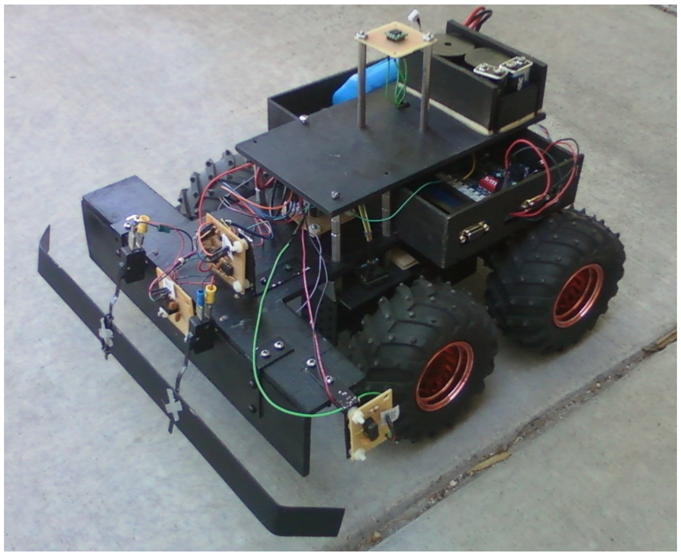
Fig. 1. Team B, Completed Robot

LIST OF FIGURES

***Abstract*—The design, testing, and implementation of a wheeled autonomous robot is described. The robot incorporates optical wheel encoders, a digital compass, infrared proximity sensors, collision-triggered bump sensors, and an ultrasonic receiver into a sensor suite which provides information about the environment to an on-board microcontroller. The microcontroller then controls the navigation of the robot through an outdoor obstacle course, utilizing the sensors to find pre-specified objects that contrast highly with the environment.**

Dead reckoning and ultrasonic sensing were the two primary methods of navigation utilized. The robot successfully found four out of five waypoints in the obstacle course and three out of four objects within the waypoints.

I. Introduction

ROBOTS able to autonomously navigate an area can be utilized to improve the productivity and safety of humans in a multitude of complex situations. In order to navigate its environment effectively, a robot needs to know three things: where it is, what its goals are, and how to achieve them [1]. It gains information about its environment using sensors, utilizes specific instructional paradigms to interpret that information, and employs navigational techniques in order to obtain its goals.

A robot's equipment will include different sensors depending on its intended application. Generally, a robot is interested in sensing specific characteristics of an object, such as its temperature, color, reflectivity, or illumination. It might also communicate with external devices via radio, infrared, or ultrasound [3]. In addition, cameras are often used to image the environment [5].

After gaining information with sensors, a robot must decide where to move based on its programming. There are three types of location information that aid movement:

- Absolute location with respect to the entire environment (ie. GPS)
- Location with respect to objects in the environment
- Location of the robots body parts with respect to itself and objects it may be handling.

A robot can use this information in different ways. It can simply execute a sequence of commands, or perhaps implement a state machine where its actions depend on an interconnected flow of data from its sensors and memory.

A robot's programming should allow for obstacles, backtracking and error handling. A robot might come across an obstacle it needs to avoid or accumulate sensor error so that the previously collected information is no longer useful. In this case, adjustments must be made to its instructions in order to deal with the situation. Preparing a robot for the unknown is the main problem faced by designers of autonomous mobile robots.

This paper will describe Team B's approach to the design and implementation of these sensors and navigational subsystems and their use to complete the requirements of the course. Course information will be provided, followed by a plan to cope with the difficulties of finding objects in the course. Hardware and sensors will be described, and this paper will conclude with a budget and the results of this team's final course evaluation.

A. *Course Requirements*

For the final course evaluation, we were required to design an autonomous vehicle capable of navigating outdoors through specific waypoints. The course included concrete sidewalk and grassy areas. There were identifying objects in each waypoint (e.g., something emitting heat, RF signal, etc.). The exact objects were not determined until late in the semester.

B. *Provided Hardware*

In order to complete the requirements of this course, we were provided a robot chassis with high-torque motors shown in Fig. 2, two H bridge motor drivers, and a 6.6Volt long-life hobby battery.

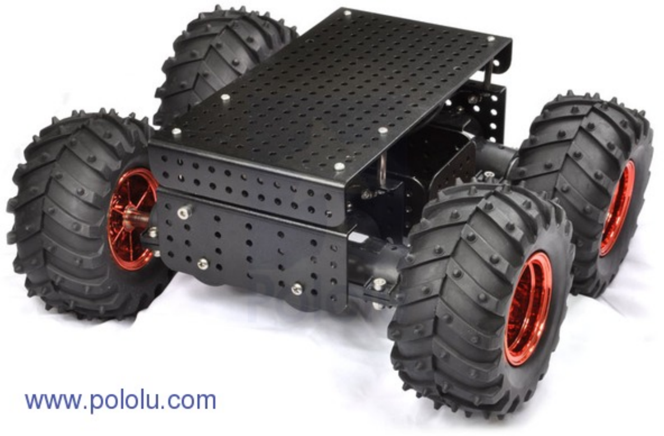

Fig. 2. Robot Chassis

We were also required to keep the budget of the robot under $750 including $125 of department funds that were allocated to our team.

II. Course Information

When identifying a waypoints identifying object, the robot must perform an unambiguous action signaling to judges and observers that the intelligence of the platform believes that it has found the item. All waypoints excepting four consisted of a circle of three meters radius outlined with high-contrast paint. We were permitted to place one item anywhere within the evaluation area to assist the platform in its operation. The specifications for this item were vague; however, some suggestions have been that groups may place a second ultrasonic beacon, a barrier, or a visually high-contrast landmark near areas that the platform is having difficulty navigating.

A. *Waypoints*

The contents of waypoints one and five were not known until the week before evaluation day, but we were given some general information about them. They would be objects chosen from the following:

- 900MHz RF Beacon
- 40KHz Ultrasonic Beacon
- Extreme Temperature Item (either higher or lower than the environment)
- Reflective Item
- Brightly-Colored Item

Waypoint two contained an object of relatively low temperature in contrast with its surroundings. It was hinted that this object would be a bag of ice. Waypoint three contained an ultrasonic beacon broadcasting in the 40KHz range. Waypoint four was a position relative to waypoint three to which the robot was required to navigate without the aid of a high-contrast object marking the waypoint's location.

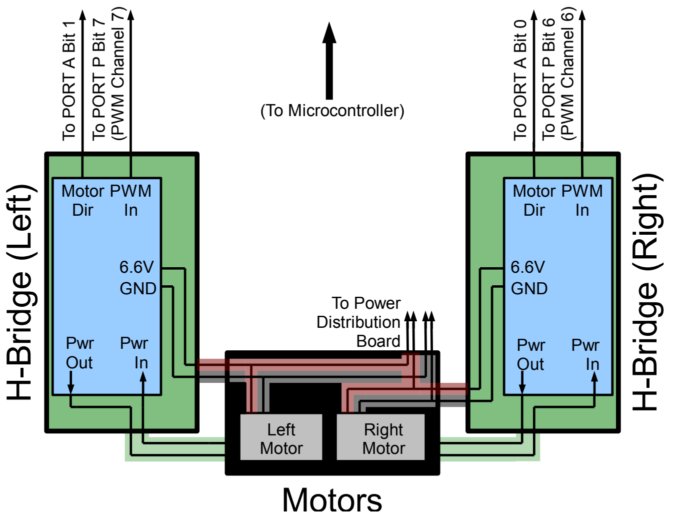

Fig. 3. Motor Control Subsystem

B. Obstacles

The course contained several trouble areas for a robot. Between the first and second waypoints, there were metal posts along the left side of the sidewalk, and we were concerned about the robot thinking they were the waypoint object. There were also trees and mulch along the sidewalk continuously until after the turnoff towards waypoint three. Here we faced additional challenges. There was a three inch drop off the sidewalk for ten feet along the path. Also, the grass was uneven in many places, which caused compasses to read the wrong direction, and the grass itself impeded any kind of motion.

III. Design Plan

To overcome these difficulties and be able to find the identifying objects in the waypoints, we implemented the following design.

A. Subsystems

To be able to complete the tasks necessary for this project, we chose to provide the following functionality:

- Motor Control
- Dead Reckoning
- Waypoint Area Searching
- Wall-Following
- Collision Detection and Obstacle Avoidance
- Ultrasonic Beacon Convergence
- Music Playing

1) Motor Control: For motor control our robot used the pulse width modulation subsystem on our microcontroller. This signal was sent to dual h-bridges which powered the motors off of our 6.6Volt hobby battery. The duty cycle we sent to the h-bridges was adjusted based on the type of terrain we expected in a certain section of the course (ie. the duty cycle in grass was higher than that on sidewalk).

2) Dead Reckoning: The process of dead reckoning became a priority system later in the semester. Although we did not originally plan to so heavily rely on dead reckoning, this system turned out to be a majority of our robot's ability and far more accurate than we would have anticipated. This system can be further broken down into subcategories: turning to a heading, going straight, and going for a set distance.

Fine tuning our robot's ability to turn to a specific heading began with magnetometer communication and continued throughout the semester. We averaged sixteen magnetometer readings on our microcontroller each and every time we read from the magnetometer, and we had to experiment with the minimal timing between readings in order to improve the smoothness of turns. Originally, while debugging, we had determined that motor interference was going to be a problem despite our mounting the magnetometer well above the motors. We thus had code written to stop regularly as we were turning to read the magnetometer without the motor interference. Later, it became apparent that this was largely eliminated with the fine tuning done to the averaging subroutine. Still unsatisfied with the smoothness of turns, we implemented proportional control based off the difference between the current and desired headings to control the power on the motors with great success. It was important to limit the minimum of the duty cycle with this proportional control, and some experimentation was required with this. If we had had more time, simple rpm control could also have been implemented to boost power to turns when the robot was stuck, and our separate handling of grass and concrete control might have been unnecessary. Ultimately, we were able to produce a smooth turn routine and controlled our robot with as low as 2.5 degree tolerance.

Going straight was the first segment of control code written for our robot. When this code was first implemented by giving both motors the same signal, it was found that drift factored in rather quickly due to uneven motor output and wheel wobble. We then attempted to compensate simply by experimenting with a constant difference between the duty cycles, but the drift depended on so many factors that it benefitted us greatly when we were finally able to base this off the magnetometer rather than guesswork. By reading from the magnetometer constantly, arcing to compensate for minor drift and going back to our turning routine for major offsets, we eventually implemented very accurate straight-line driving.

For distance measurement, we used our optical encoders and input capture timer to count wheel rotations. By discounting distances during turns and averaging the left and right encoder values, we were able to arrive at a very reliable distance measurement. The most challenging aspect of this subsystem was determining the best way to mount the encoders, as we were unable to alter the wheels or axles permanently. Through trial and error, we eventually settled on hot glue, since it was cheap and it held better to metal than superglue or epoxy. Combining both of these subsystems, the digital compass and wheel encoders, into a dead reckoning system was nearly flawless. Problems in our dead reckoning resulted from basic bugs in the waypoint instructions and measured values. We are confident that, given another few days, our robot would have navigated the course without any problems at all.

3) Waypoint Area Searching: Searching a waypoint itself was originally going to be based on color sensing. Since this option was abandoned fairly late into the semester, the search algorithm was a last-minute addition and based off dead reckoning. It was important to leave the circle at a known point and to cover as much of the circle with a minimal amount of dead reckoning error accumulated. We thus settled on a simple pattern of three parallel passes through the circle, with the robot exiting before the second pass if the object was found during the first. Thus, our routine was not demonstrated during our final run on the first waypoint, but was shown to some degree on the fifth.

4) Collision Detection: Collision detection was based off of our proximity sensors and bumper switches. If we were between circles, the left proximity sensor was ignored since we were only interested in going around obstacles directly in our path. Inside circles, all these sensors were monitored and triggered the victory routine, but the left proximity sensor did not trigger obstacle avoidance. Noticing that the robot would often avoid non-existent obstacles, we added a routine to stop briefly and check again on suspected collisions, which greatly improved the performance of this routine.

Obstacle avoidance was also based off dead reckoning, and it simply involved going around the obstacle to the right. Distance traveled was restored on exit from this routine, plus the offset from traveling around the obstacle. Given more time, a more adaptive obstacle avoidance routine that was sensitive to left versus right collisions and which allowed for multiple iterations might have been implemented.

5) Ultrasonic Convergence: We implemented this subsystem to be able to travel towards the highest power received ultrasonic signal. Ultrasonic convergence simply involved spinning in place and storing any values of measured signal strength and the corresponding heading that were greater than the previous greatest measured signal strength, then going that distance for one meter and repeating the process until collision occurred. This was an extremely durable and reliable routine which helped convince us to make our wildcard object an ultrasonic beacon.

6) Music Playing: Our music player was a sequencer previously designed by Jason Michnovicz, which was improved upon for greater compression and versatility. While it required us to program our microcontroller into flash memory, since there was not enough room in EEPROM, the sequencer gave us our victory signal and greatly added to our robots individual appeal. Three melodies were implemented, the overworld theme from the Legend of Zelda video game, the underworld theme from Super Mario Bros., and the fanfare from Final Fantasy. The first was played while the robot believed itself to be between waypoint circles (based off of dead reckoning), the second while the robot believed itself to be within a waypoint circle, and the third when the robot believed itself to have found a waypoint object (accompanied by a pause so our evaluators could more easily confirm). This sequencer ran on interrupts, and did not seem to affect the processing ability of our microcontroller.

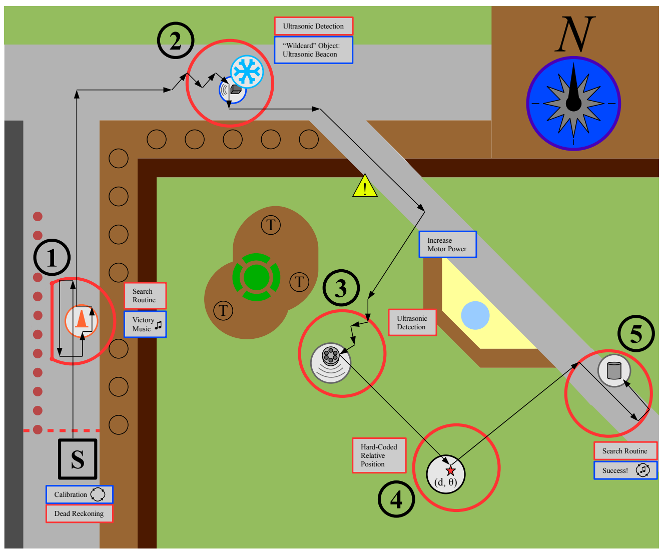

Fig. 4. Course Map

B. Waypoints

1) Unknown 1: Waypoint one was initially unknown. This waypoint was a bright-orange traffic cone. Our original plan included a color sensor which we would use to detect any changes from the main environment colors. However, when we scrapped the color sensor, we were forced to find another way to detect this object. We therefore implemented a three-pass search routine for waypoint one, as you can see in the first leg of the robot's journey in Fig. 4. The robot would dead-reckon to the waypoint area, make three passes up and down the sidewalk, and if nothing was found, it would continue on its way through the course. We chose to make three passes because the width of the sidewalk was roughly three times the range of our left-side proximity sensor plus the width of the robot. Three passes would therefore let us search the entire waypoint area.

2) Temperature Difference: Waypoint two was a metal bucket filled with ice. Initially, we wanted to detect the temperature of this object, but we were unable to establish communications between our microcontroller and the temperature sensor we had chosen. Once we realized this, we chose to use our wildcard object at this waypoint. This also helped the robot determine its global position, so it could find the turnoff towards waypoint three. In the end, the robot dead-reckoned from waypoint one to the corner, turned towards waypoint two, and then moved towards waypoint two approximately three meters. From there it began converging on our wildcard object, which we describe in a later section.

3) Ultrasonic Beacon: Waypoint three was an ultrasonic beacon. We were worried about reflections and multipath with this waypoint, but we never saw any results that would suggest problems like these. We designed, built and tested an ultrasonic receiver early in the semester. This helped us greatly because we were able to converge on the beacon without fail. This system was so reliable that we modeled our wildcard object, an ultrasonic transmitter, on the beacon from this waypoint. The robot was able to converge on the beacons, both waypoint

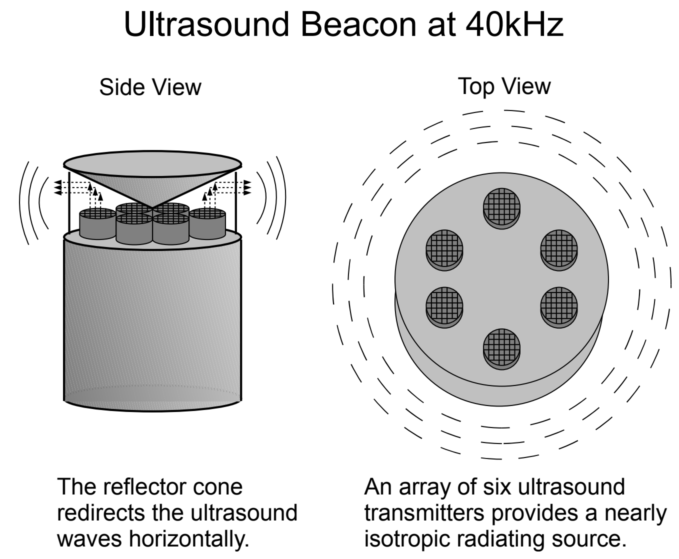
Ultrasound Beacon at 40kHz
Side View
Top View
The reflector cone redirects the ultrasound waves horizontally.
An array of six ultrasound transmitters provides a nearly isotropic radiating source.

Fig. 5. Waypoint 3: Ultrasonic Beacon

three and our wildcard, from a distance greater than ten meters.

4) Relative Vector: Waypoint four was a relative distance and heading from waypoint three. This waypoint gave us trouble throughout the semester. Our dead reckoning system worked very well on the sidewalk, but in the grass, it was error-ridden. The grassy areas were uneven, hilly, and resistive to motion. We found that our digital compass was very sensitive to tilting, and as the robot traversed the course, it could get off by as much as twenty degrees from its intended course. We tried to work around it by trial and error, choosing slightly different paths, but the robot could never get close enough to this waypoint for it to count.

5) Unknown 2: Waypoint five was similar to waypoint 1 in that both were large objects that differed in color from the environment. Additionally waypoint five was very reflective. It was a large metal can. We had two ideas for finding this waypoint: the robot could dead-reckon to the sidewalk and begin its search routine, or it could move towards the fountain area and follow the railroad ties around to the sidewalk and then search the area. Given our time restraints, we chose to go with the first option. Unfortunately, we encountered the same kinds of problems we had with waypoint four. We were forced to input headings and distances that were obtained by trial and error, but in the end the robot went about two meters further than it should have before starting the search pattern. Given more time, wall-following could have been a viable solution to our problems at this waypoint.

IV. Hardware

Aside from the hardware given to us at the beginning of the semester, we added the following systems to make our robot functional.

A. Microcontroller

We chose to reuse our Dragon-12 development board from EE 308 Microcontrollers Lab. We chose this board because we already had experience programming and interfacing it. The board also included all the various systems we required, e.g. IIC bus, input capture timer, ADC, PWM, etc. We were also able to program the microcontroller in the C programming language because we had access to a free MCU-specific compiler, which would not necessarily be true for other boards.

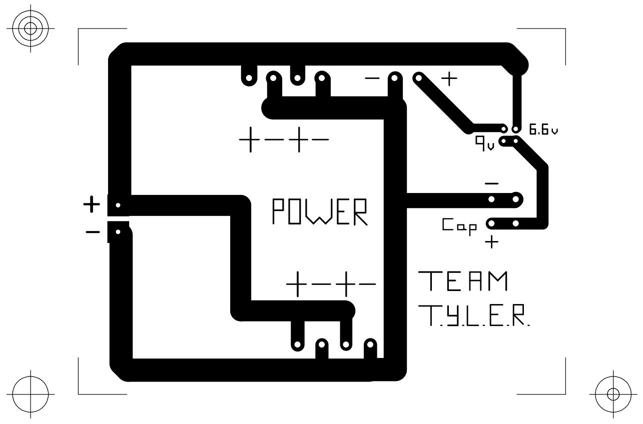

Fig. 6. Power Distribution Board PCB

B. Power Distribution Board

We etched a power distribution board ourselves, making large traces to handle the current we expected, so that we could easily provide power to several components. We included screw terminals for easy attachment of power cables and a dongle to plug in the battery. At one point, wires between the power board and our sensors shorted together, and the amount of current caused the traces to partially seperate from the dielectric. The traces themselves remained intact, and we were able to salvage the board.

C. Power Regulator Board

Our sensor suite required several different DC voltage levels to operate, so we built a regulator board to satisfy this need. Our board took power from the 6.6Volt battery through the power distribution board and output $\pm$ 5Volts and ground. This can be seen in the schematic in Fig. 7.

D. Chassis Additions

We realized early in the semester that the stock chassis was not large enough to mount everything that we needed, so we built additions to it using standoffs and thin plywood as illustrated in Fig. 8. We added a fender and bumper, two horizontal platforms, and two trays. To the fender we mounted both proximity sensors, and the bumper-actuated proximity threshold violation detectors. In the bowels of the robot we placed the power distribution board and the H bridge motor drivers. On the first platform, we attached the ultrasonic sensor, the power regulators, and the microcontroller. We included a tray on the second platform which contained the batteries for the microcontroller and two metal weights to even out the frame. We attached the magnetometer on top of the robot, as far from the motors as possible.

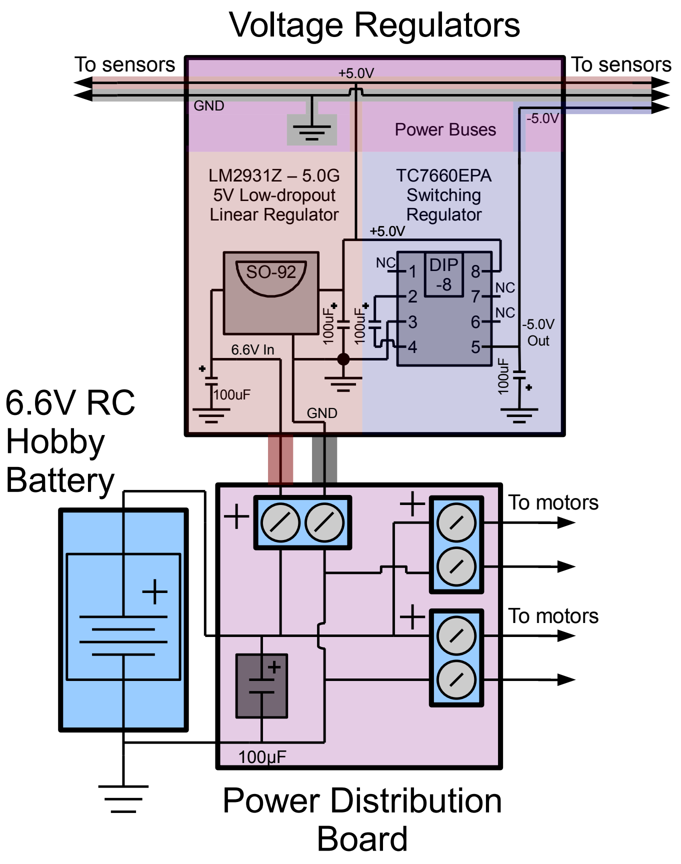

Fig. 7. Power Distribution and Regulation

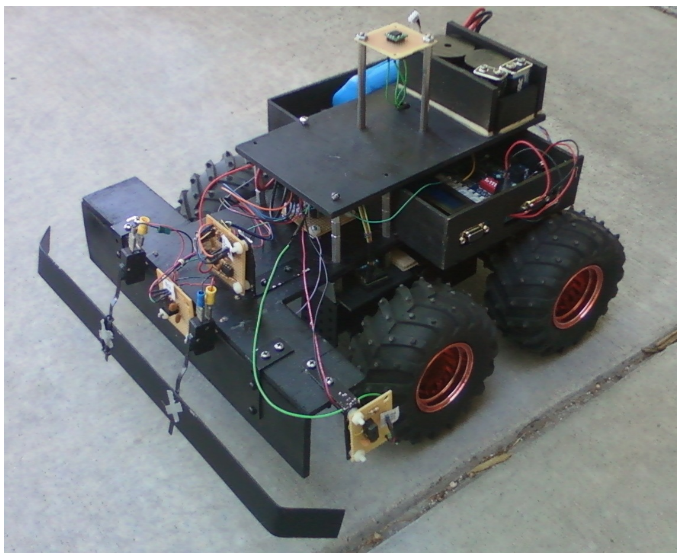

Fig. 8. Completed Robot

V. Sensors

Down selection of sensors occurred about one third of the way into the semester. At this time, our design required sensors for detecting heading, acceleration, collisions, proximity, ultrasonic, color, temperature, and radio frequency. Later in the course, the radio frequency, temperature, and color sensors were dropped and we added wheel encoders. Below is a description of each sensor: our initial intentions for it, the difficulties we encountered with it, and our final evaluation of its functionality.

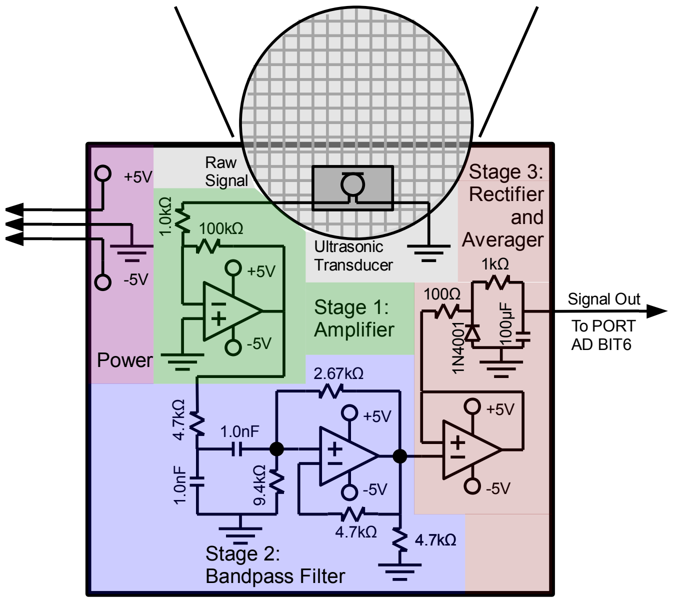

Fig. 9. Ultrasonic Receiver

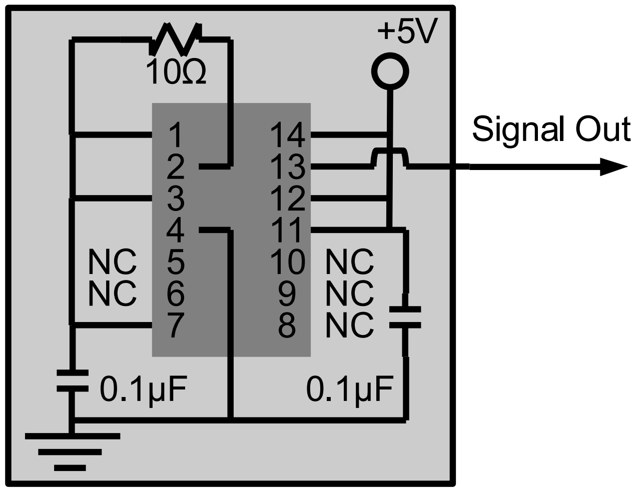

Fig. 10. Proximity Sensor

A. Implemented

1) *Ultrasonic Receiver:* For detecting the ultrasonic beacon, we purchased a 400SR16 ultrasonic receiver and included it with an amplifier, band-pass filter, and rectifier/averager built by Jason. We then fed this signal into the A2D converter on board the microcontroller. We observed register values from the A2D ranging from reliably zero in the absence of a source up to 800 or more right next to the beacon. Using this and our ultrasonic search routine, we were able to easily converge on an ultrasonic source from more than ten meters away.

2) *Proximity Sensor:* In addition to collision detection, we wished to expand our field of view to avoid objects preemptively and for edge following. To do this, we selected two

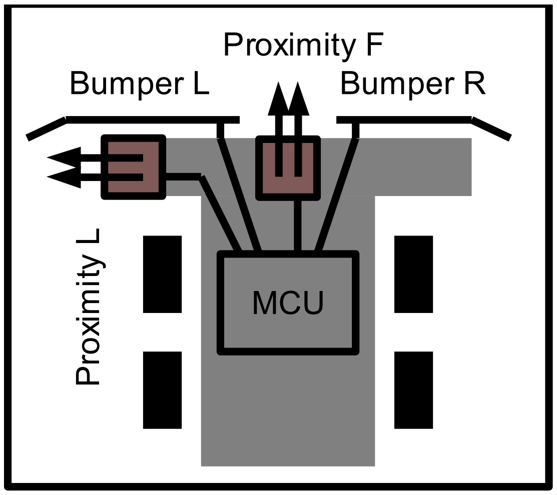

Fig. 11. Proximity Sensors and Bumper Switches

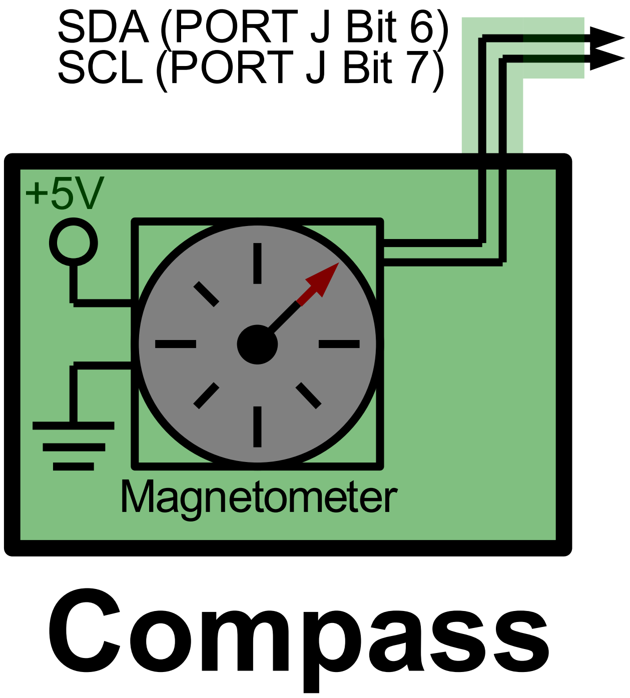

Fig. 12. Digital Compass

gp2y0d810z0f proximity sensors from sharp microelectronics . These were wired up so as to give a basic logic high or low depending on the presence of an object within 10cm and fed into PORTA. One proximity sensor was placed on the left of the front chassis and one on the front. The front one could give more precise information on object location for collision avoidance, though this was not implemented, and the left one allowed us to prevent turning collisions while searching and to search a wider area. These worked well though their limited range precluded more advanced application.

3) Bumper Switch: Two Defond switches were attached to the front of the chassis and a bumper connected between them so that the robot would have a directional knowledge of collision. These switches were simply powered from the regulators and connected directly to the microcontroller's general purpose input/output pins. In retrospect it may have been wiser to power the switches directly from the microcontroller or else place a resistor in series with them, since our microcontroller's LCD screen lit up upon switch actuation, which is worrisome. Given more time, we may also have wished to connect these and the proximity sensors to input capture ports so that we wouldn't have to check PORTA in while loops. Difficulties included the attachment of the bumper, since aluminum does not weld well, and the switches not opening on their own. The first problem was addressed with epoxy and wire lashing and the latter by bending the actuators such that a desired sensitivity was established. Ultimately, these worked well and were vital in object detection.

4) Wheel Encoder: Later into the semester, we deemed it necessary to develop encoders for distance measurement. The encoders consist of two parts and were placed on each of the back wheels. These two parts are an optical switch (Fairchild Semiconductor H21LTB) which reads the presence of an object between two elements, and metal disks which have evenly-spaced holes around their circumference. These were designed by John Schilling and worked well with the exception of our method of attachment. This difficulty occurred because of our limitation in not being allowed to modify our wheel attachments in any sort of permanent manner. If this were not a prototype, the encoders would be more permanently attached and this would not be a problem. Our encoders worked exceptionally well and were heavily relied on.

5) Magnetometer: For heading detection, we chose and implemented the Parallax HMC6352 Compass Module. We planned to use this to be able to turn and travel along a desired heading. We selected this magnetometer above its peers for its convenient package and low price. It communicates over an IIC bus and has tenth-of-a-degree resolution. Integrating this sensor was one of our group's first priorities; however, we encountered a few problems along the way. Our first difficulty with this sensor was the ordering of commands used in start up in order to program EEPROM. Some basic guess and check was required to get the module running. After that, we had a basic coding error caused by shifting bytes while they were considered signed. This caused quite a few intermittent errors including erroneous readouts between the headings of 12 and 27 degrees. We also noted that calibration was required directly following any program crashes during magnetometer operation. We found this sensor fairly reliable once communication was established, and relied upon it heavily.

B. Scrapped

1) Color Sensor: For color detection, we selectedTCS3210D-TR from TAOS. This sensor was small and had to be surface mounted, but provided color information on red, blue, green, and total channels. Our plan was to use this to verify our position when encountering the circles around way points. We achieved communication with this sensor, but never managed to get truly useful information out of it. Even with software averaging, the signals from this device had a large range and varied from environment to

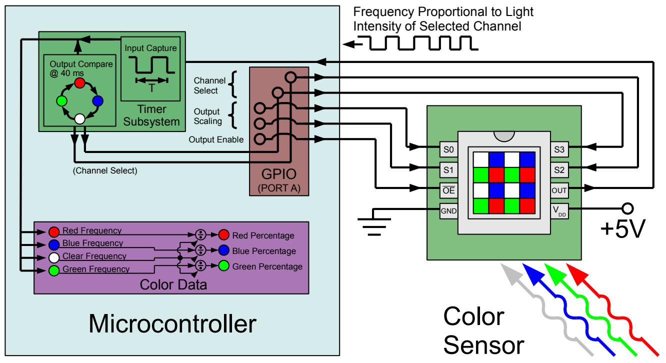

Fig. 13. Color Sensor

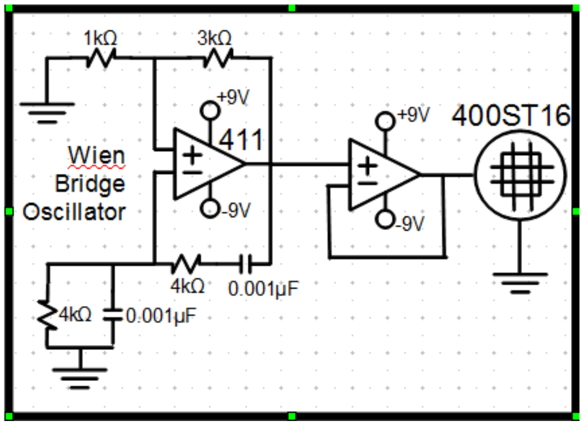

Fig. 14. Wildcard Ultrasonic Beacon

environment based off of light level and background surface. Given extensive overtime, we might be able to threshold program if we got a working light shield around the sensor. It was eventually determined that our dead-reckoning was sufficient to navigate the course, and this sensor was left off the final design.

2) *IR Thermometer:* For temperature detection, we selected Parallaxs MLX90614 IR Thermometer Module. This sensor was selected due to its easy-to-connect pin out, example code, and simple communications protocol. The module is simply an extension of the MLX90614 which communicates over serial lines rather than IIC or PWM and has an alarm for a temperature threshold built in. The necessary wiring of this sensor involves three lines: SCIO, +5V, and GND. In retrospect, the price increase from sensor to module was probably not worth the benefit. After spending three plus weeks attempting to communicate with the module, this sensor was abandoned. Our plan was to use this similarly to the ultrasonic sensor and home-in on the object or else store a threshold voltage that we would look for during a search routine. Given more time, we feel confidant that we could get the sensor communicating, but the results from other teams show that it might not have been useful anyway.

3) *RF Receiver:* For detecting a RF beacon, we selected the Mica-Z antenna provided to us by Dr. Aly El-Osery. This package had an estimated cost of $120. As the semester progressed, the emphasis on being able to detect an RF beacon diminished until it was dropped from the course requirements entirely. Thus, it became apparent that this sensor suite was unnecessary, a very significant part of our budget, and the time required to implement it was better spent elsewhere. It was not included in our final design.

4) *Accelerometer:* We had originally planned to implement an accelerometer for use in dead reckoning, due to the negative air placed on optical encoders and our expectation of wheel slippage. The MMA8450Q was cheap and had enough resolution that we thought it would be useful. When it arrived, we came to realize that the package was miniscule enough that the resources to implement it were better directed elsewhere, especially since, with the addition of weight to our chassis, wheel slippage was looking to be less and less of a problem. Thus, we scrapped this sensor and ended up using encoders instead.

VI. Wildcard Object

Due to the success of our ultrasonic reciever, we decided to build our own ultrasonic beacon to use as our wildcard object. We designed a Wien bridge oscillator to output a frequency of approximately 40 kHz, then put that signal through a buffer to an ultrasonic transmitter.[cite transmitter datasheet] The beacon was powered by two 9-Volt batteries oppositely polarized to provide positive and negative inputs to the operational amplifiers.

The beacon was placed inside of a weighted box to prevent damage to the circuitry should the robot collide with it. Placed at Waypoint 2 near the ice bucket, the ultrasonic beacon was successfully found by our robot.

VII. Budget

At the beginning of the semester, we were provided the chassis, battery, H bridges, and battery, given $125 from the Electrical Engineering department, and told that our final design could not cost more than $750. Our projected cost for the project was $747, but by discarding the radio frequency sensor, our total design costs dropped to $643 and without the temperature sensor, color sensor, and accelerometer, our robot prototype costs only $555. Of the department funds, we only spent $85. We are thus surprisingly under budget and our design is quite economic. If given the chance, we might have invested some of these extra funds into better motors and wheels. (See attached!)

Early on, we projected based off some rough estimates that our robot would be able to run for 28 minutes, with the motors on the 6.6V being the limiting factor. With all sensors, the microcontroller, and the motors running, we measured the drain on the 6.6V after 19 minutes and still estimate a run time of 28 minutes. In addition, our robot has completed the course and was still functioning fully at the end. Our estimates were surprisingly accurate and our robot performs the specified task with some leeway.

VIII. Conclusion

Autonomous robot navigation affords many challenges. The foremost difficulties we encountered in this project pertained to sensor integration and reliability. Even though we were forced to deal with less information than our original design had assumed, due to failure to interface with our color sensor and lack of consistent compass data, we were able to implement a robust design which utilized dead reckoning, collision detection, and software-based error compensation to achieve complete navigation of the course and successful location of over half of the waypoint objects. Solving design problems of this type corresponds heavily to a number of real-world autonomous navigation tasks, and in completing this project, we have learned a lot about robotic design and have been able to develop a greater appreciation for the elegance and robustness of the design solutions accomplished every day in the field of robotic navigation.

References

[1] Jonathan Dixon and Oliver Henlich, *Mobile Robot Navigation*, 1997 June

[2] Andreas Bartel and Frank Meyer, *Real-Time Outdoor Trail Detection On A Mobile Robot*

[3] Cliff Randall and Ian MacColl and Henk Muller and Yvonne Rogers, *Exploring the Potential of Ultrasonic Position Measurement as a Research Tool*

[4] Martin Adams and Wijerupage Sardha Wijesoma and Andrew Shacklock, *Autonomous Navigation: Achievements in Complex Environments*,*IEEE Instrumentation & Measurement Magazine*, 2007 June, 1094-6969/07

[5] Francisco Bonin-Font, Alberto Ortiz and Gabriel Oliver,*Visual Navigation for Mobile Robots: a Survey*

[6] James Andrew Bagnell and David Bradley and David Silver and Boris Sofman and Anthony Stentz, *Learning for Autonomous Navigation*, *IEEE Robotics & Automation Magazine*, 2010 June, 1070-9932/10

[7] Stefan Baten, *Techniques for Autonomous Offroad Navigation*, *IEEE Intelligent Systems*, 1998 December, 1094-7167/9