Worcester Polytechnic Institute
Robotics Engineering Program

RBE 2002 Final Project

SUBMITTED BY
Michael Abadjiev
Qingyuan Chen
Nicholas Johnson

Date Submitted:	**5/1/2018**
Date Completed:	**4/25/2018**
Course Instructor:	**Prof. Putnam**
Lab Section:	**D02**

Abstract

This project was the capstone project for RBE 2002: Unified Robotics II. The goal of the project was to apply the material we were covering in class, particularly related to sensing. We had to build a robot that navigates a maze and extinguishes a candle. We successfully created a robot that was robust and consistent, and was able to navigate the maze and record the candle's location, before attempting to return home.

Contents

Table of Figures

Introduction

The goal of this RBE 2002 final project was to apply the knowledge we learned in class about sensors to a real-world application. For the challenge, our robot had to autonomously navigate through a maze in order to find and extinguish a candle. Once the candle had been extinguished, the robot had to record its location in the X, Y, and Z dimensions, relative to the starting location, and then return home. All of this had to be accomplished within a 5 minute time limit. In order to test our understanding of sensor usage, we were also instructed to use at least an Inertial Measurement Unit (IMU), encoders, range-finding sensors, and an infrared flame sensor to determine the candle's location.

An IMU is a sensor that measures the change in a robot's position and orientation. It measures the rate of change of the robot's position in three axes, and rotation around the axes. Some IMUs simply report these rates back to the robot controller, requiring it to do any computations necessary to determine absolute position. Other IMUs are slightly more expensive, however they do the integration calculations onboard, and report absolute positions back to the robot. Encoders are sensors used to measure the rotations of a spinning shaft, in this case the drive motor outputs. Encoders with higher resolutions record more than one "tick" per revolution, allowing for greater positional accuracy. Rangefinders are used to determine a robot's distance from another object, and they come in two typical varieties: Ultrasonic, and Infrared. Both types rely on beams (either a sound wave or IR beam respectively) shot from the front of the sensor that bounce off of an object, and return to a recording unit. The two methods used to calculate the distance of an object are to either measure the time of flight (TOF) of the beam, or to shoot the beam at an angle and record its return angle. Finally, the flame sensors that were used for this project detect infrared light emitted from fire sources, like a candle.

Materials & Methodology

Strategy

In order to complete this challenge, we decided to use a classic approach to solving a maze: right wall following. Our robot followed the wall to its right, making left turns to navigate the maze until it saw a flame. Once the flame was detected, the robot then turned towards it and identified its location in the X, Y, Z. Because of how inaccurate the flame sensor was compared to the accuracy of the rangefinders (see graph in appendix) we chose to use it to find the angle of elevation of the flame, and use the robot's distance to the base of the candle for trigonometric calculations, rather than distance of the flame from the sensor. Finally, the flame was extinguished and the robot returned to its home location. The various sensors were used to keep the robot oriented straight while following a wall, track its position relative to the start, and to find and extinguish the flame.

Mechanical

For our robot, we decided to focus on keeping the mechanical systems simple and function-focused. In order to accomplish this, we used a 3-tiered design that allowed us to separate the major system of the robot. As can be seen in figure 1, the systems were separated by putting the drive components, as well as the Arduino, on the first floor, the primary sensing apparatus, including the rangefinders, the IMU, and the flame detectors on the second floor, and the primary actuator, the fan, on the third floor.

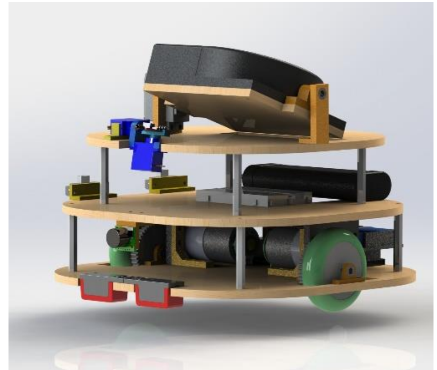


Figure 2: full robot, note 3-floor design

In order to drive the robot, we decided to use a classic, two-wheel drive robot with a roller caster wheel in the back. Each wheel was driven by one of the RBE 2002 bookstore pololu motors, with a 1:3 gearing ratio after the initial planetary gearbox. This resulted in a 1:150 final drive ratio and can be seen in figure 2, allowing for slow, easily controllable robot movement. The first floor contained these drive motors, along with their motor controllers, the Arduino, and two VEX line-followers for detecting when the robot was on a cliff.

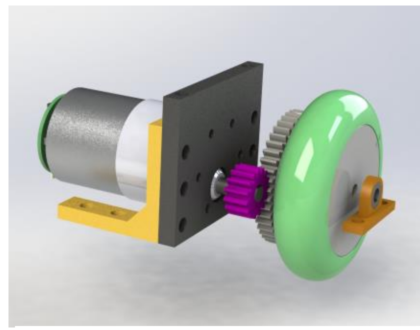


Figure 1: drive system

The rest of the sensory equipment, including the infrared rangefinders, the IMU, and the solid-state flame sensing array were housed on the second floor. The third floor housed the fan seen in figure 3, mounted on a servo for easy positioning. The third floor was also designed to allow it fold on hinges along a center seam to allow for easy access to the second floor and all of its sensors as can be seen in figure 4.

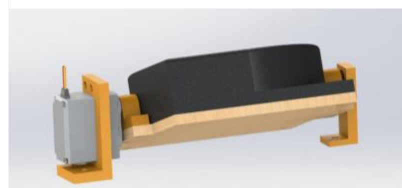


Figure 3: fan mounting

The most important aspect of the robot was the flame sensor gimbal mounted on the third floor. This gimbal was modeled similarly to the gimbals found in many stabilization systems for high-end camera equipment. It utilizes two servos, one for each axis, to rotate and point the lab kit flame sensor at the candle. It was mounted on the third floor, just below the fan, in order to easily translate the angle of the gimbal to an angle for the fan to point at. Figure 5 shows this gimbal, along with the flame sensor on it, and figure 6 shows the mounting of the gimbal.

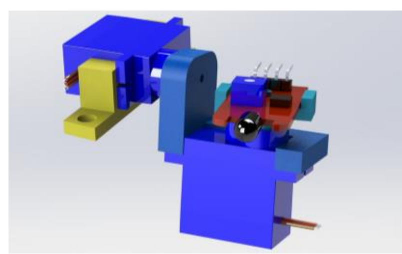


Figure 4: fire gimbal

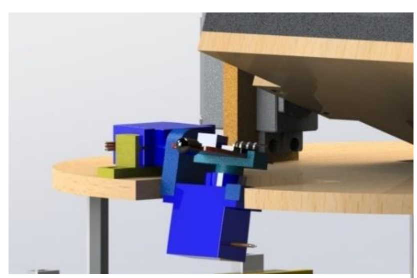


Figure 5: gimbal location

Electrical

To drive the robot's two pololu 37D mm metal gear motors, two MC33926 full bridge motor controllers were used. Each motor controller requires two PWM signals to control forward and reverse speed. On the back of each motor is a magnetic quadrature encoder. The encoder is capable of 64 counts per revolution (CPR), with the 1:150 gearing, this results in in 9600 counts per revolution of the drive wheels.

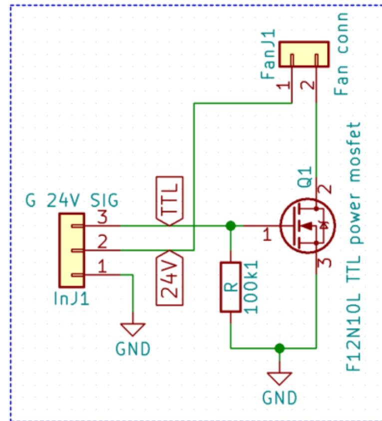


Figure 6: fan control circuit

While everything else on the robot is powered by the 12V battery, the fan required 24V. An additional battery was connected in series to create 24V. A RFP12N10L logic level power MOSFET(metal-oxide-semiconductor field-effect transistor) was then connected in series on low side as seen in figure 6, to allow the 5V signal from the Arduino to control the fan.

The key sensors in this challenge are the flame detectors. They are made up of an infrared sensitive phototransistor and a transresistance amplifier. The phototransistor is simply a Bipolar Junction Transistor (BJT) with the junction exposed. A normal BJT is modeled as a dependent current source controlled by the base current. The phototransistor is the same but controlled by light intensity. With a coating that allows only Infrared light to pass, the output current will be proportional to the amount of light received.

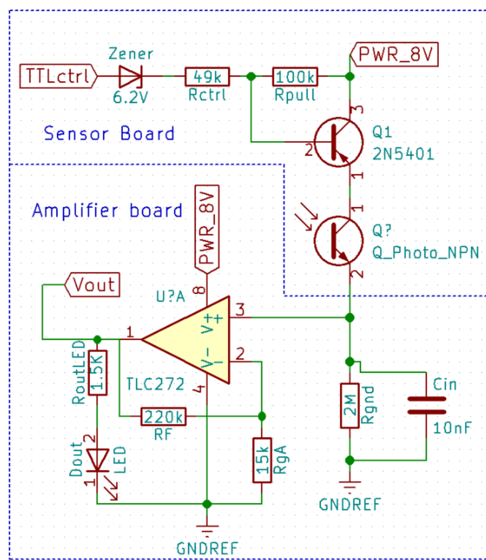


Figure 7: custom fire sensors

With the provided 140C001 flame sensor on a 2D gimbal, a sweeping motion was required to cover a wide angle which introduces innacuracy. Since the flame sensor circuit is simple and cheap to build, a custom flame sensor array was built to monitor the field instead of large sweeps, allowing better accuracy and reserving the gimbal for smaller angle pinpointing. The basic construction of the custom sensor is shown in figure 7. The IR phototransistor is connected to a resistor to ground. The current through the resistor will create a voltage (at pin3 of the OpAmp) at about 0-200 mV. An amplifier of 15 V/V is required to make this voltage readable by an Arduino. Since the impedance is about 50 Mohm, the current is at nA range. For this reason, a CMOS (Complementary metal-oxide semiconductors) OpAmp is used to avoid the OpAmp's input offset current.

To cover a 90 degree angle for detection, we used total of 16 phototransistors. In order to the number of pins and OpAmps required, as well as allow for easy mounting arrangements, the phototransistors were arranged in groups of 4 on their own sensor board, as seen in figure 8. Power for each sensor board (total of four) is controlled by the Arduino through a PNP transistor and a zener diode. A multiplex configuration is formed when the output of each sensor board uses the same set of OpAmps. A detailed schematic of the board can be seen on the next page in appendix section. In this way, only 4 control pins and 4 output pins are needed to use all 16 phototransistors.

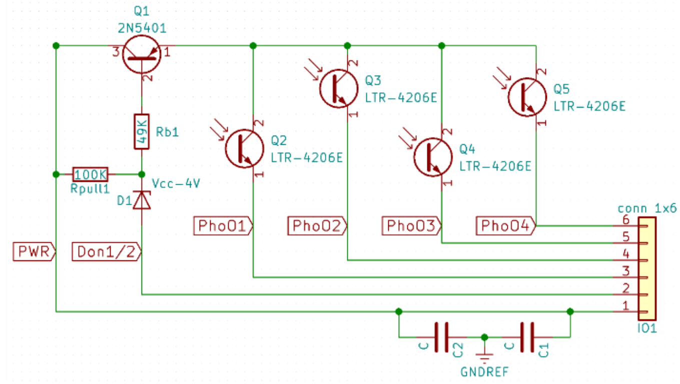


Figure 8: individual sensor board configuration

Since motors directly draw current from the battery as well, there will be quite a bit of PWM noise to the sensor array. A LM7809 9V linear power regulator is used to provide clean DC voltage for the sensors.

There are many other off the shelf sensors equipped. Two GP2Y0A41SK0F Sharp IR sensors on the side, and three time of flight (TOF) sensors, two VL53l0x and one VL6180x, in the front. Although they all operate in infrared range of about 940nm, they did not have interference with each other, or the flame, as the Sharp sensors use angle of arrival and the others use time of flight, neither of which depend on intensity of IR signals. To free up more computing power for the Arduino to focus on other tasks, the BNO005 absolute position IMU is used.

To wire all the sensors together required more space than was available on the Arduino shield. In order to avoid a common issue of wires coming off a breadboard and to make wiring easier, multiple breakout boards were made with prototyping boards and connectors soldered onto them. Figure 9 shows the break out board for motors, motor drivers and encoders, and figure 10 shows the breakout for all I2C sensors.

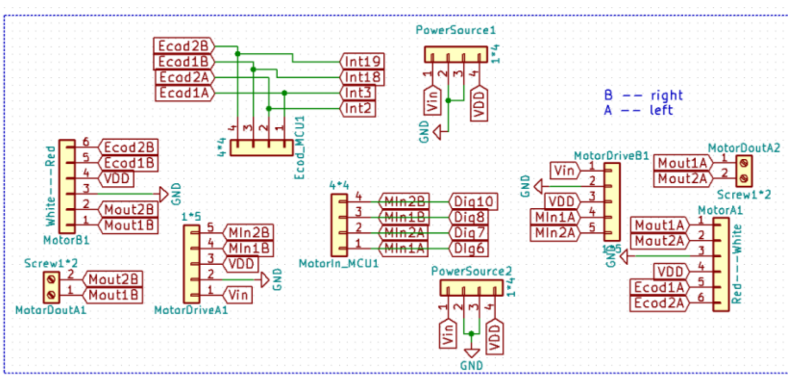


Figure 9: motor breakout board

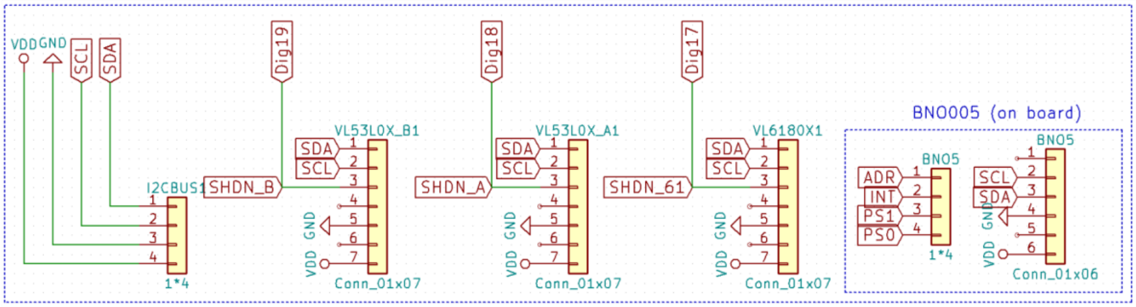


Figure 10: I2C breakout board

Software

As this project was sensor heavy, those sensors needed to be frequently updated by the Arduino. To accomplish this the program was built as a state machine from the ground up. Sets of moves were divided up into action groups, each of which was a group of states that would execute in a row, for example; going around an edge would require driving past the edge, checking for a wall, then turning right if there was no wall, or following the wall if there was one. The state machine held its current “mode” to execute while checking the important sensor readings every time through the program.

In order follow the right wall of the maze, the gyro (IMU) kept the robot mostly straight while the two side IR sensors checked the rate of change of the robot to the wall and correcting the gyro heading accordingly. The three forward facing TOF sensors informed the robot when an object, or a wall was in front of it. If the object was not a candle, the robot would take a 90 degree left turn and continue following the wall. This was the basis for the robot’s drive system.

To find the candle in the maze the solid-state array of photo-transistors was used to search for any indication of the flame. Once the long range array spotted the flame, the robot would determine which orientation the flame was in relation to the robot. If the flame was 90 degrees to the left, it would enter the flame extinguish action group. It would position itself perpendicular to the flame, in order to keep all turns 90 degrees, then drive to a distance of 250 mm from the base of the candle

Once the robot was in position the gimbal was used to determine an accurate angle to the flame by sweeping horizontally and determining the highest IR value. Once the horizontal angle was determined, the gimbal would sweep vertically to determine the height of the candle by identifying the angle with the highest IR reading. Once the X, Y, and Z coordinates were calculated the fan was swept up and down for multiple passes to put out the candle. After each attempt to put out the candle, the flame sensor would wait and check again to determine if the flame was extinguished and if not try again.

For both finding the position of the candle and for backtracking to the origin, encoders were used. The robot kept track of all of the movements in the X and Y directions to know where it was located. This was used to output the location of the candle and to return back home. To backtrack the robot would simply continue following the wall around until it reached (0,0) again.

Results and Discussion

Overall, we were pleased with the success of our robot, its ability to follow a wall consistently meant that we were usually able to easily navigate to the candle. Once it was operating at the full 24V, it had no trouble consistently extinguishing it. The drive system worked perfectly, and allowed the fine movement control we required for accurate positioning. The gimbal also worked perfectly for pinpointing the location of the flame. The major mechanical complaint with our robot was the lack of easy access to plug sensors and other pins into the Arduino.

The major electrical issues were revealed when the solid state phototransistor array proved to be problematic in a few ways. The first of these issues was that the initial design of the sensor board let to overly sensitive sensors; they would detect a flame from across the field, when we only wanted it within a few feet of the robot. This was rectified by changing some resistors to lower the gain values on the OpAmps. Another problem was that the initial design called for phototransistors with a very narrow cone of operation, however we accidentally purchased the wrong ones and were unable to find replacements in time for the final demo. This led to us making shields for some of the sensors out of cardboard.

The custom flame sensors was designed to pick up the flame from far away with only a very rough indication of flame direction. Thus, the phototransistors are designed to have a small angle offset to cover the full 110 degree angle and the output gain is quite high. After some testing, it is found that doing non 90 degree turn and move are hard to implement. On the other hand, if one group of the custom flame sensors could be dedicated to line up the robot with the flame, everything would be easier. Thus, the sensor board on the left is mount vertically with a cardboard shroud to narrow the detection range to a vertical line and the output gain is decreased.

Several improvements could be made to the robot’s software. The first and most important is an improvement in accuracy of the Z calculation of the flame. Sometimes the gimbal when sweeping would catch an incorrect noisy value and provide the calculation with a wrong angle. This could be rectified with additional filtering of the flame sensor.

For backtracking, the robot continued to follow the right side wall and check when it was at the origin. An improvement in this would have been to record the moves and then execute them in reverse. This was attempted and partially worked but was not as reliable as our wall following and because it was optional, resources were directed elsewhere. The source of the reliability issues were certain small moves that would cause the robot to stutter and reduce the accuracy of the trip home.

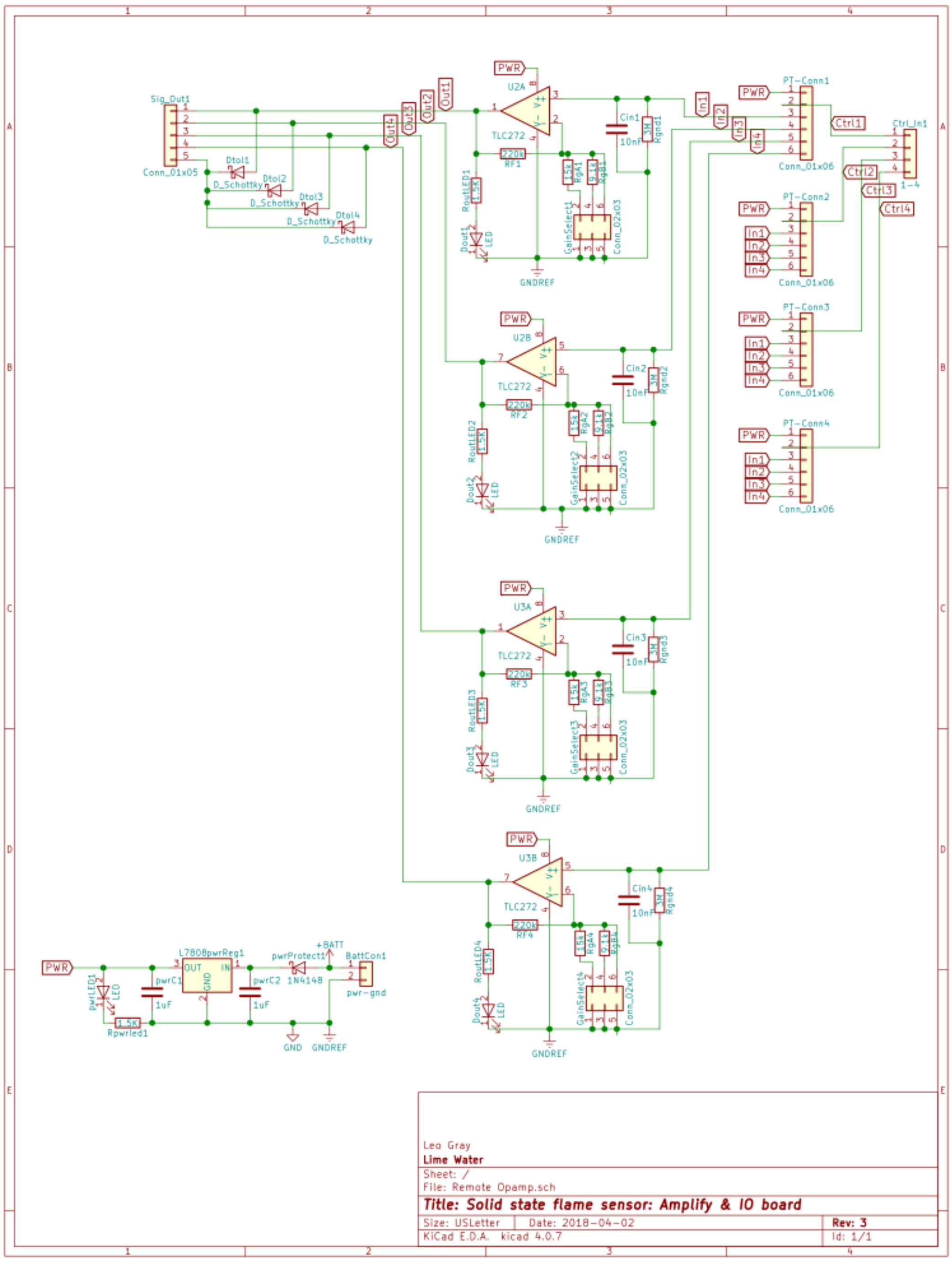
The overall cost of the robot came out to $335.27, of which $121.86 came from materials that were supplied to us by the robotics lab, and $213.41 came from our own spending, on materials like sensors and building materials. A full cost breakdown can be seen in the appendix section. We believe that this was a fair price for the robot, especially considering that the vast majority of the items we bought can be reused in future projects.

Conclusion

In this project, we successfully applied the knowledge we learned from class about sensors, filters, and creating a robot that could properly sense and react to its environment. The robot was able to consistently navigate the maze using right wall following, detect the flame, and extinguish it. It used a wide variety of sensors, from the IMU and IR rangefinders for wall following, to the sensor array for flame detection, to sense its environment and interpret when different actions should be performed. The robot’s major shortcoming was the accuracy of the provided flame sensor, which led to inaccurate readings for the height of the candle. Future iterations of this project could utilize more complex, accurate algorithms for determining candle height, as well as simply more accurate sensors to solve this issue.

Appendix

Custom Sensor Amplification board



Cost Breakdown				
	Item	count	unit price ($)	price ($)
Lab Materials	arduino Mega	1	38.50	38.50
	arduino Mega rbe lab shield	1	0.00	0.00
	140C001 flame snesor	1	1.69	1.69
	Vex line sensor	2	13.33	26.67
	GP2Y0A41SK0F Sharp IR	2	9.55	19.10
	MC33926 Motor Driver Carrier	2	17.95	35.90
Our materials	30:1 Metal Gearmotor 37Dx68L mm with 64 CPR Encoder	2	40.00	80.00
	RFP12N10L N-channel MOSFET	1	0.92	0.92
	SG92R mini servo	2	3.95	7.90
	big servo	1	5.10	5.10
	Computer Fan	1	20.00	20.00
	VL53l0x TOF distance sensor	2	15.00	30.00
	VL6180x TOD distance sensor	1	14.00	14.00
	BNO005 absolute position IMU	1	35.00	35.00
	Prototying board	4	0.30	1.20
	schottky diode	5	0.50	2.50
	Bule LED	4	0.05	0.20
	TLC272 dual OpAmp	2	1.30	2.60
	L7808 LDO	1	0.62	0.62
	2n5401 PNP	4	0.51	2.04
	zener diode 6.2V	4	0.13	0.52
	LTR-4206E IR transistor	16	0.13	2.11
	Surface mount resistor 1%	30	0.07	2.10
	multilayer ceramic capacitor X7R	16	0.10	1.60
	Connectors/headers	10	0.50	5.00
	Lab Cost			121.86
	Our Cost			213.41
	Total Cost			335.27