2 Dimensional Infrared Robotic Mapping System

Kosuke Sato
School of Engineering Science
Osaka University
sato@sys.es.osaka-u.ac.jp

Bradley Nobuo Watanabe
Jack Baskin School of Enineering
University of California, Santa Cruz
(949) 632-0744
watanabebrad@gmail.com

ABSTRACT
This project intents to take the initial steps towards increasing the mobility and effectiveness of robotic movements. It does this by creating a 2 dimensional map of the given environment. This gives the robot the ability to avoid obstacles more efficiently and take a more direct route to its destination. In this prototype, the system will only create a map of a small well defined area. This concept will later be expanded to a larger, more rugged environment.

General terms: Algorithms, Design, Experimentation, Measurement

Keywords: Wii IR Camera, Arduino, mapping, robotics

INTRODUCTION

Whenever we travel to a new location, we always look for a map. It is our ability to use maps that helps us successfully travel from one location to another in an efficient manor. The SENS Laboratory Object Recognition System (SENSORS) applies this very concept towards robotics.

On the commercial market there are many examples where this system could be used. For instance, the Romba robot doesn't have a mapping system and therefore it has a hard time cleaning an entire room. This is due to the robots inability to determine if it has cleaned the area more than once, because of its random cleaning pattern. If a mapping system was implemented, the robot would be able to determine if it has been to that location before. Thereby making sure it doesn't clean the same area over and over again. With a mapping system the Romba would also be able to determine which areas of a room get dirty more frequency. With this information the robot could adapt its cleaning schedule to clean certain areas of the room more often than others.

The mapping system that we propose will be capable of mapping out and generating a 2 dimensional map of the given room. The map that is created will be displayed in a 2 dimensional matrix. This system utilizes the mobility of a microcontroller to collect all the mapping data and a PC to perform many of the complex mathematical calculations.

SENSORS BREAKDOWN

PC interface

As mentioned earlier there are two different components of the SENSOR System, the PC and the microcontroller (mobile unit). The main operation of the PC is to handle the complex mathematical operations needed to generate the map. The PC based SENSORS program retrieves both location and distance data via a text file. Figure 1a depicts the flow of data from the Mobile SENSORS Unit to the PC. It then goes on to show to flow of data within the PC SENSORS program.

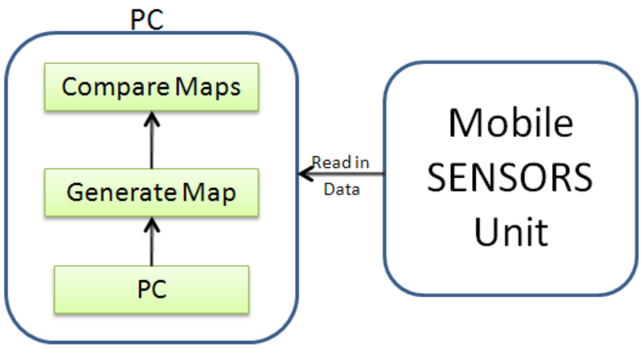
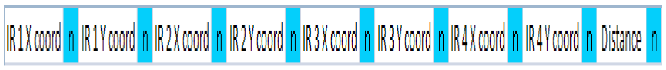


Figure 1a: This depicts the flow of data within the PC.

The data in the text file is then read into the program as a packet of nine data points. The data must be sent in a specific order. The x and y coordinates for the first Wii IR tracker followed by the x and y coordinates of the other three Wii IR trackers. The Wii IR tracker data is obtained by the Wii IR camera and depicts the relative location of the IR beacon. The final data point is the distance data. This is obtained by the IR range finder and shows the distance from the SENSORS mobile unit (figure 3b) and an obstacle. Each of the data points is followed by the character 'n'. The reason for this is because the program needs to know where one data point ends and the next data point begins. The layout of the packet can be found in figure 1b. It should also be noted to the character 'e' must be placed at the end of the text file. All of this is taken care of on the microcontroller.



IR1 X coord	n	IR1 Y coord	n	IR2 X coord	n	IR2 Y coord	n	IR3 X coord	n	IR3 Y coord	n	IR4 X coord	n	IR4 Y coord	n	Distance	n

Figure 1b: This show the order of each of the data packets are read in by the SENSORS program.

This data is then put into a stack in the PC program. Once in the stack the data can be accessed and manipulated to carry out the mapping operations.

Now that the information has been retrieved we must convert it from its raw form to something more useable. To do this we convert the raw distance values into actual distances. The output of the IR range finder is a digital value between 0 and 1023. After some experimentation and calibration we found the IR range finder follows equation 1.

$$D = 69601n^{-1.373}$$

Equation 1:
D = distance in centimeters
n = digital output of the IR range finder

For further information on how this equation was found please refer to the IR range finder section.

Once the distance data has been converted, the Wii IR Camera data needs to be evaluated. To do this the program needs to find the location of the mobile SENSORS unit. It does this by converting the relative coordinates of the Wii IR camera into matrix coordinates. With the location of the mobile unit know we can then find the angle at which it is looking. These processes are further explained in the Wii IR Camera Section.

With the distance, direction, and location information we now know enough information to create a complete 2 dimensional map of the explored area. To display the map we simply print out the 2D matrix, printing the character '.' in spaces where nothing was found. In locations where an object was found we have placed the character 'X'. The character 'O' is placed at the location of the mobile unit when an object was detected. Figure 2 shows an example of the 2D matrix map.

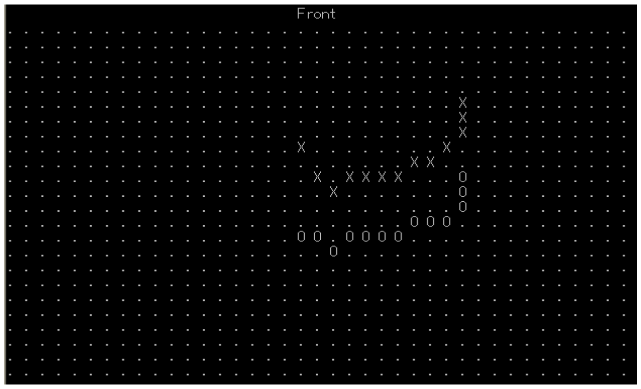


Figure 2: There we see a map of a small wall depicted by the character 'X'. The character 'O' depicts the location of the mobile unit.

MOBILE SENSORS UNIT

The second unit consists of 5 parts; the microcontroller, range finder, SD card reader, beacon, and Wii camera. Each of these components play an important role in the mapping process, without which the system would be rendered inoperable. The flow of data between components in the mobile unit is depicted in Figure 3a below. Figure 3b show the Mobile SENSORS Unit.

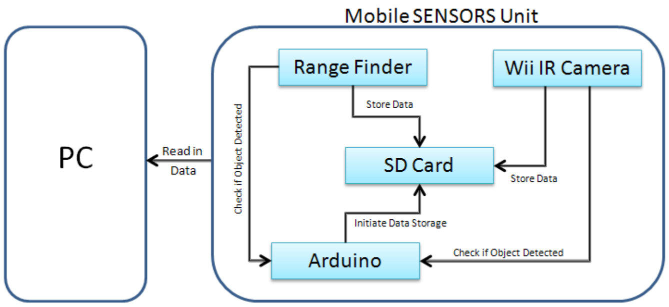


Figure 3a: This show the order of each of the data packets are read in by the SENSORS program.

Microcontroller

The microcontroller used on the Mobile SENSORS Unit is a 3.3 volt Arduino Mini Pro. As the name implies this microcontroller runs off of 3.3 volts, which makes communication between other components very simple. This is because all the other components of the mobile unit require 3.3 volts for operation.

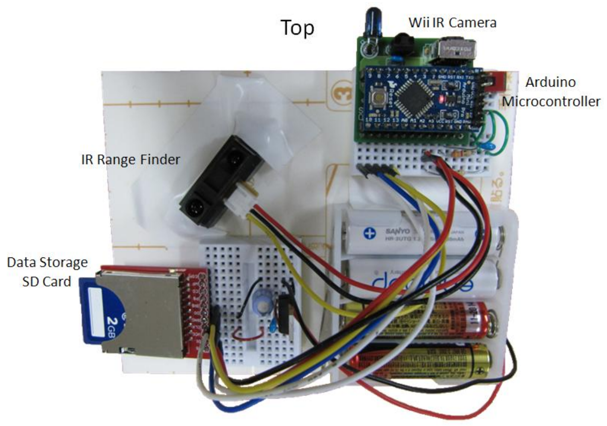


Figure 3b: This shows the mapping system. The Wii IR Camera must always be facing up to make sure the Wii IR Camera can see the IR beacon. The IR range finder will always point forwards

The Arduino is the brain of the mobile unit it tells each device what to do and when to do it. It determines if data from the Wii IR camera and IR rangefinder are valid and should be stored on the SD memory card. In order to determine this, the microcontroller will run the following algorithm. It checks to see if anything has been detected by the IR rangefinder. If nothing has been found by the IR range finder, the microcontroller will wait 500 milliseconds and then check the IR range finder again. This process will continue until the IR range finder detects something. Once the IR range finder detects an object the microcontroller will store the distance data from the range finder as well as the location data from the Wii IR camera.

IR range finder

As described in the microcontroller section the IR range finder initiates the mapping process. It is used to calculate the distance between the mobile mapping unit and detected object. The IR range finder works by projecting infrared light from an infrared light emitter. That light is then reflected off the surface of an object. The light then returns to the range finder where it can be absorbed by the infrared light detector. The infrared detector than converts the captured light into a propositional voltage though digital signal processing.

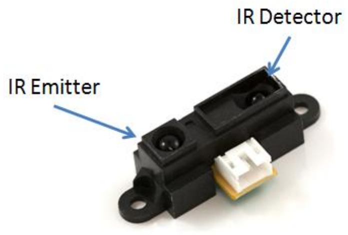


Figure 4: This is the GP2Y0A21YK Sharp IR range finder.

The IR range finder gives an output in the form of an analog voltage between 0 and 3.3 volts, where a lower voltage indicates a further distance. This analog voltage is then sent into an analog port on the Arduino microcontroller. This converts the analog voltage into a proportional binary number between 0 and 1023. In this case 0 volts is represented by 0 volts and 3.3 volts is represented by 1023. This process needs to be done so that the microcontroller can appropriately evaluate the data.

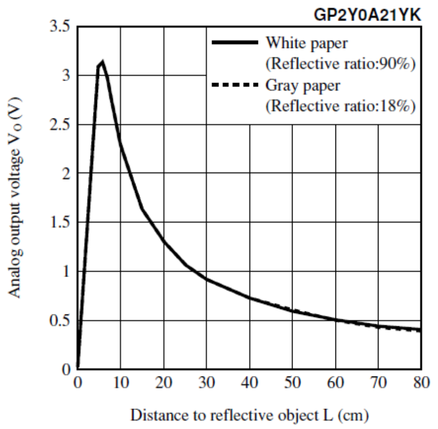
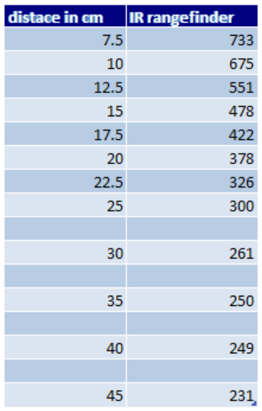


Figure 5: This is a graph of the output of the IR range finder given in the Sharp data sheet.

In figure 5 we see the theoretical output of the IR rangefinder, as given by the Sharp data sheet. From this figure we see that there is an exponential correlation between the voltage output and the distance in centimeters. To find the exact relationship we tested the digital output for several different distances. From this we acquired the data in table 1 and figure 6.



distace in cm	IR rangefinder
7.5	733
10	675
12.5	551
15	478
17.5	422
20	378
22.5	326
25	300
30	261
35	250
40	249
45	231

Table 1: IR Range Finder Calibration. This is the raw data that was collected from testing the sensor.

To obtain the data point in table 1 we measured out the distances in column 1, then a flat white surface and the IR range finder were separated by that distance. The distance was then measured by the range finder. A special program was written in order to obtain this information. The program printed out the digital equivalent of the distance measured by the IR range finder. This process gave the data found in table 1.

This data was then plotted in Microsoft Excel where the exponential approximation was then calculated. This is given in equation 1. Using this equation we were able to determine the distance between the mobile unit and detected object.

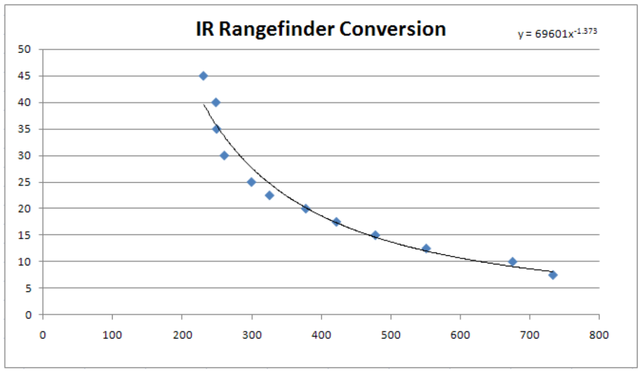


Figure 6: IR Range Finder Calibration. This is the plotted data, used to find an equation to convert the raw data.

While testing the IR range finder we found that it behaved very unpredictably when measuring distance over 12.5 cm. We believe that this is due to infrared interference from the lighting. Because of this limitation we only trigger the IR range finder when the measured distance is between 7.5 cm and 12.5 cm; this corresponds to the digital values of 725 and 520 respectively.

SD card reader

The SD card is being used to increase the storage capacity of the mobile unit. A significant amount of data is being generated in order to create a complete map. On the Arduino Mini there are 256 bytes of EEPROM available for use. This is non-volatile memory, meaning if the microcontroller loses power the data stored in the EEPROM will still be saved. This is not the case for RAM which doesn't retain any information if power is lost. Because the Arduino Mini is being powered by a battery and it holds valuable mapping information we have deemed it necessary to protect the data that is being collected, hence using the SD card.

To use the SD card a FAT16 protocol has been used. This allowed the data to be stored in a universally used format. The FAT16 protocol was utilized thank to the efforts of Bill Greiman for providing an Open Source FAT16 library, which can be found at http://code.google.com/p/fat16lib/. By using the library created by Mr. Greiman we were able to successfully store and read all of our mapping data.

IR Beacon

The IR beacon is an essential hardware platform for the mapping system. Without which it would be impossible to determine the mobile unit's position within the environment. The IR beacon uses 3 IR LED's so that the Wii IR camera can determine the location and direction. If only 1 LED was used, the relative location could be determined but there is no way to determine which direction the mobile unit is faceing. On the other hand if 2 LEDs were used then it is possible to determine directions parallel to the LEDs but not in the adjacent directions.

By using 3 LEDs we are able to determine any direction without extra calibration. The reason for this is because we have made the distances between each of the LEDs different. This makes it easier for the Wii camera to differential between the IR LEDs. Further explanation can be found in the Wii IR camera section.

Wii IR camera

The Wii IR camera is possible the most important part of the mapping system because it gives location data as well as directional data. The Wii IR camera works by tracking up to 4 infrared light sources at once. The location of these light sources is given as a number between 0 and 1023. The center position is indicated by the coordinates 511 for both the x and y coordinates.

To find this data we received a considerable contribution from Ms. Keiko Ishikawa. She has provided a substantial amount of the base code for the Wii IR camera. We were able to use Ms. Ishikawa's code to store the x and y coordinates of all 4 of the Wii IR trackers. To keep from getting an overflow of unusable raw data we have decided to only store the Wii IR camera data when we detect an object with the IR range finder.

After all the data has been collected it will be uploaded to the PC, where the usable data can be extracted. The raw

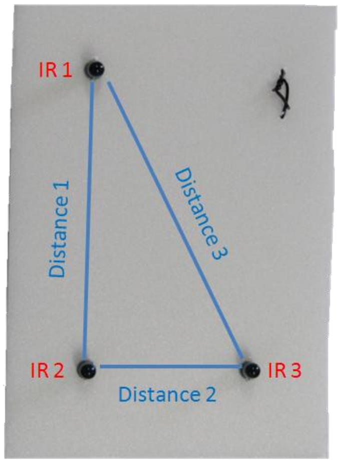


Figure 7: There is the IR beacon with the labeled marker indicators.

Wii IR camera data is then converted into directional data and location data though complex mathematical conversions. Before these calculations could be preformed the orientation of the IR beacon must be analyzed. This extra analysis is necessary because the Wii IR camera randomly chooses which IR marker to follow. This means that everytime the Wii IR camera looks at the IR beacon it will chooses a different way to represent the IR beacon. In order to make calculations easier we have assigned names to the three markers on the IR beacon. Which are indicated in the figure 7.

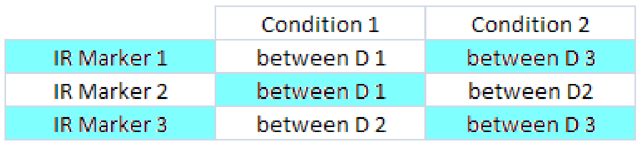
As mentioned earlier, when the Wii IR camera tracks these markers, there is no way of knowing which of the four IR trackers is following IR marker 1. This meant that the SENSORS software needed a way to recognize which marker is which. To do this we made the distances between each of the markers different. By doing this we were able to calculate the distances and based off of those values, determine which marker is where.

We first determined that the angle formed at IR 2 is approximately a right angle and because of this we know that the distance, D3 (D = Distance) is the greatest and that D2 is the smallest. Using this information we can apply the magnitude equation to find the distance between the IR markers (See equation 2).

$$d = \sqrt{(x_a - x_b)^2 - (y_a - y_b)^2}$$

Equation 2:
d = between the IR markers
x_a = IR marker a's x coordinate
x_b = IR marker a's y coordinate
y_a = IR marker b's x coordinate
y_b = IR marker b's y coordinate

By using equation 2 we are able to find the distances between all of the IR markers. With this knowledge we can determine which IR tracker is related to which IR marker.



	Condition 1	Condition 2
IR Marker 1	between D 1	between D 3
IR Marker 2	between D 1	between D2
IR Marker 3	between D 2	between D 3

Table 2: The two conditions dedicate the two criteria that make each marker differentiable.

We ultimately want to make sure that the IR trackers will match that in figure 7. To do this we must determine the criteria of each of the IR markers. This is listed in table 2.

Using these criteria and the length relations for D1 through D3 we can determine which IR marker correlates with which each IR tracker. After this knowledge is know we can then reorder the IR trackers to correspond with the formation shown in figure 7.

With the IR markers in the correct arrangement we can now find the location of the mobile unit. To do this we look at the x and y coordinates of IR marker 2. We have chosen to use only one IR marker to limit the calculations needed by

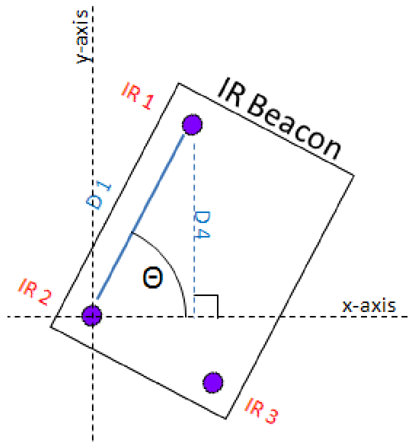


Figure 8: This shows the angle Θ being measured to determine the direction at which the mobile unit is facing. This figure depicts the view of the Wii IR camera looking at the IR Beacon.

the PC and microcontroller. If we were to use all 3 IR markers to determine the location of the mobile unit we would average the three locations. Thereby increasing the work load of the microcontroller. Marker 2 was chosen as the location marker because it is closest to the center of the IR beacon as compared to markers 1 and 3.

To find the direction or angle at which the mobile unit is facing a simple algorithm is used. The algorithm finds the angle between an imaginary x-axis and the line created by

$$sin(\Theta) = \frac{D4}{D1}$$

Equation 3: In this equation Θ is the angle of interest while D1 and D4 are still unknown.

$$D1 = \sqrt{|x_1 - x_2|^2 - |y_1 - y_2|^2}$$

Equation 4: x_1 and x_2 correspond to IR_1 and IR_2 respectively. The same goes for y_1 and y_2.

$$D4 \; = |y_1 - y_2|$$

Equation 5: D4 shows the equation for the absolute change in height y.

D1. This is depicted in figure 8.

In order to find the angle Θ, the sine relationship is used. This relationship can be found in equation 3.

To find D1 we can use the distance equation found in equation 2. We found that, D1 is described by equation 4.

Finally to find the distance D4 we observed that it is the change in height Δy from makers IR_1 and IR_2. We found that D4 is given by equation 5. After putting equations 3, 4, and 5 together we were able to solve for the direction or angle Θ. This can be found in equation 6.

$$\Theta = sin^{-1}\left(\frac{|y_1 - y_2|}{\sqrt{|x_1 - x_2|^2 - |y_1 - y_2|^2}}\right)$$

Equation 6: This shows the final equation to determine the angle at which the mobile unit is facing.

MAP EVALUATION

Once all of the raw data has been input into the PC and converted into a 2 dimensional map, we must evaluate that data. This process entails comparing past maps with the recently generated map. In Figure 9 we see an old map (left) and a new map (right). All characters other than '.' represents the object that were mapped. The characters 'B', 'R', 'A', and 'D' represent the corners of the mapped object. It is our ability to map the corners of the object that allow the program to determine how far the object has moved.

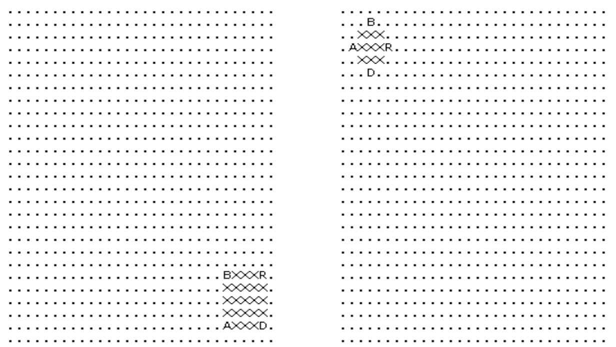


Figure 9: Image to the left is the new map and the image to the right is the old map.

The process of determining the corners of the object requires 4 different searching patterns. Pattern A will search horizontally from left to right going from top to bottom. Pattern B will search from top to bottom moving from columns on the right to columns on the left. Pattern C searches from bottom to top, moving from columns on the right to columns on the left. Pattern D scans from right to left moving up row by row. All of these search patterns continue until a none '.' character is found. The 4 patterns are depicted in Figure 10.

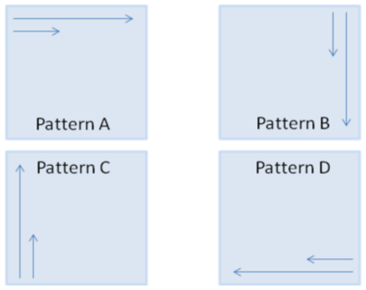


Figure 10: Image to the left is the new map and the image to the right is the old map.

This search pattern is essential because it makes sure that four different corners will be detected in a square object. This can be seen by closely examining Figure 9.

RESULTS

After several test we were able to successfully generate a map of a single object. In the first test we mapped out one side of a cubic object that had dimension of 9 cm x 9 cm x 9 cm. The results of this experiment can be seen in figure 11. Superimposed on the map is a picture of the object.

The object was placed at a slight angle so that the accuracy of the mapping system could be evaluated. Based off of the data that we have collected the mapping system will map each point on the matrix to a 1 cm^2 area. Therefore from this map we can see that the box is actually 9 cm wide.

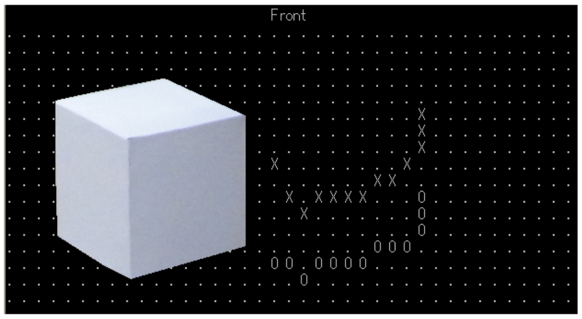


Figure 12: Here we have superimposed the object being mapped onto the 2D map created by the SENSOR System.

CONCLUSION

The SENSOR System has successfully shown that it is able to correctly map out a small area with distinguishable detail. This has become the first step in creating a system that can map a building in its entirety. As a next step for this project we hope to increase the range of the IR beacon. This is currently one of the major limitations. The IR beacon can only be 1.5 meters away from the Wii IR camera. We believe that over coming this problem will increase the range of the mapping system, inevitably bringing us closer to our goal.

ACKNOWLEDGMENTS

Professor Daisuke Iwai

Ms. Keiko Ishikawa

Mr. William Greiman

SENS Laboratory

REFERENCES

1. Evans, Allan. *Fat16 Interface for MSP430*. Michigan State University, 2004. Web. May 2010.
2. Greiman, William. *Arduino Fat16 Library*. Tech. 2008. Print.
3. Sharp. *GP2Y0A21YK*. Data Sheet