EEL5666
Kari Bowen
Dr. Arroyo
4/12/2005
Nerfherder Final Report

TABLE OF CONTENTS

ABSTRACT..... 3
EXECUTIVE SUMMARY..... 4
INTRODUCTION..... 5
INTEGRATED SYSTEM..... 6
MOBILE PLATFORM..... 7
Wheels & Movement..... 7
Shooting Mechanism..... 7
SENSORS..... 7
CMUCam Vision Sensor..... 7
Devantech SRF-04 Ultrasonic Ranging Module..... 11

CdS Cell ……..... 14
BEHAVIORS..... 15
Avoiding/Roaming..... 15
Target Detection……15
Shooting……...15
CONCLUSION..... 15
ACKNOWLEDGEMENTS..... 16

Abstract

This autonomous robot searches a room for a red target. While avoiding obstacles, the robot attempts to find the correct concentration of the color red. Once it finds the correct red object, it moves toward the target, continuing to get data from the camera in order to figure out its position. Once it is at the proper angle and distance to the target, it then launches a ball at the target. Once the ball has been fired, the robot stops and waits for reset.

Executive Summary

Nerfherder is a color-detecting robot equipped with a vision sensing CMUCam. Nerfherder performs a search behavior while avoiding obstacles in order to find the red target. Once Nerfherder finds the color red that meets the threshold defined in the program, it attempts to correct its angle and distance from the target so that it can get a straight shot at the target. Once it has positioned itself to the target, it starts the shooting mechanism, which fires the projectile at about a five second delay. Once the shooting mechanism has completed its job, Nerfherder waits patiently for a reset to do target practice all over again.

Introduction

Most people have been to a carnival or some like event where there was a target game. My favorite was for the Chemistry Honor Society when we threw bean bags at a large Mole cutout and tried to earn points for the targets hit. This common game was the inspiration for my robot. It is already fun enough to lose at these carnival games, why not compete against a robot and lose even more!

Nerfherder certainly has ties to warfare applications as well. In fact, many fellow IMDLers have told me that Nerfherder is the most violent robot of the group. The association is obvious: find the enemy target, and shoot it.

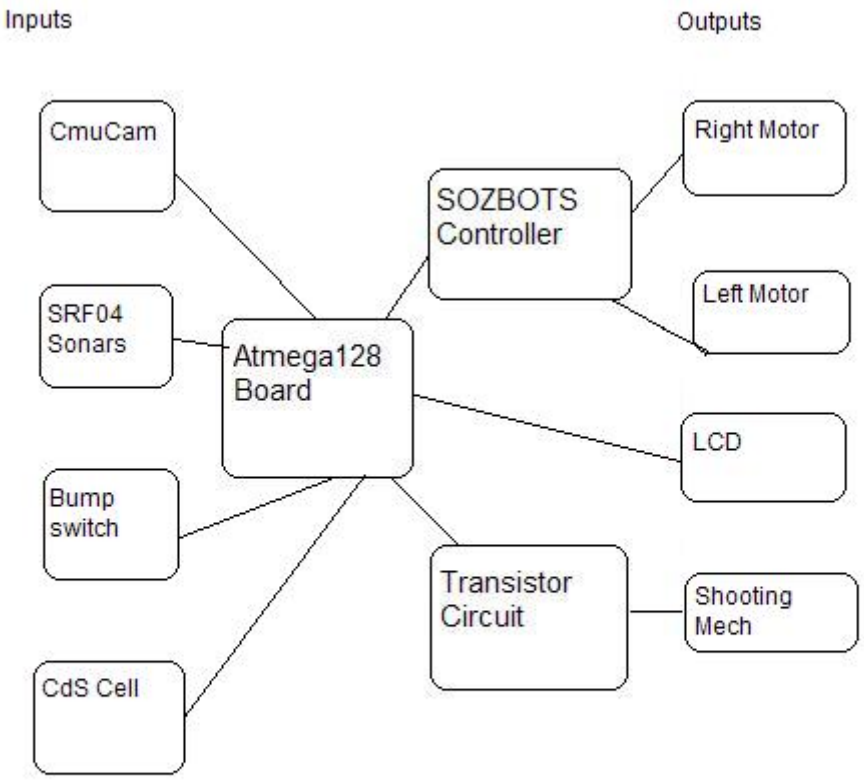
In the coming pages, Nerfherder's design and implementation will be described, including but not limited to its platform, sensors, and behaviors.

Integrated System

Nerfherder's "brain" is the ATmega128 which is housed on a Mavric-II development board. Features of the board are 128K Program FLASH, 4K Static RAM, 4K EEPROM • dual level shifted UARTs • RS485 on-board • I2C ready w/pull-up resistors installed • up to 53 digital I/O pins • Clock frequency of 14.7456 MHz• Advanced, low drop-out voltage regulator on-board accepts 5.5-15V input with reverse polarity hookup protection • Small size at 2.2 x 3.6 inches

The brain does all of the main processing for the robot, including reading from the sensors and running the behaviors.

Nerfherder operates off



Mobile Platform

The mobile platform was created with one constraint in mind: being able to balance the weight of the shooting mechanism properly without the robot tipping over. Thus the robot was created on a circular platform with a ball caster mounted in the rear to prevent tipping. Below the circular platform, the motor mounts keep the motors in place. Wheel cutouts in the circle prevent the chafing of wheels with the platform as well as giving Nerfherder the lowest ride possible.

On top of the circular platform lies a modified cube (6 inches wide and long, 4 inches high). This cube houses the "guts" of the robot. Mounted on the surface of the cube are the CMUCam and SRF-04 sonar devices in the front, and the LCD panel in the back.

Atop the cube lies the roof. The shooting mechanism is mounted to the roof. The shooting mechanism consists of two wheels and motors mounted an inch apart with a track between them for the ball to travel through.

Wheels and Movement

Nerfherder is equipped with two Green Dot Sumo, 2.5 inch tires with 4 mm hubs. The tires are mounted to the sides, but slightly angled towards the front for balancing purposes. The hubs connect the tires to the motors, which are spur gearhead motors from Lynxmotion which run at 290 rpm and 12 vdc. These motors are mounted on the underside of the platform, and are actually quite fast for the application at hand.

Shooting Mechanism

The shooting mechanism consists of two wheels with motors in plastic casings. These were originally purchased from the surplus store for use as Nerfherder's wheels, but due to their lack of torque, they could not support much weight. Instead, they turned out to be perfect for the shooting mechanism, which only needs to push a light ball through.

The shooting mechanism is mounted on top of the robot. The two wheels are separated by about 1.5". A track lies beneath the wheels to support them as well as to hold the ball. An additional piece connected to the tops of both wheels also acts as support to keep the mechanism from moving too much. Originally, a solenoid was to be used to push the ball into the chamber, but it was discovered that the air flow from the motors would pull the ball through.

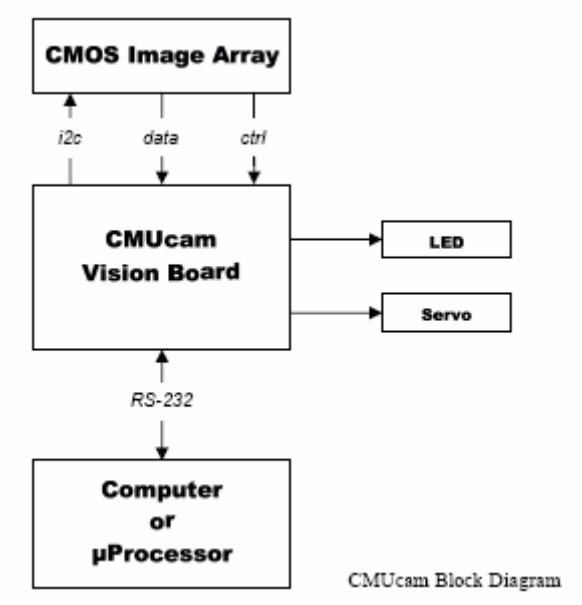
The shooting mechanism is controlled by two MOSFETs, which connect to Port F and turn the motors on when Port F goes high.

The shooting mechanism was one of the most difficult parts for me to figure out!

CMUCam

Theory

The CMUCam is a simple camera with microprocessor which was created at Carnegie Mellon University. It has built in functions such as tracking an object of a certain color, finding the average color of a frame. It utilizes the SX28 microcontroller and an OV6620 Omnivision CMOS camera. It communicates in RS232 or TTL serial ports.



CMUcam Block Diagram

Objective

For Nerfherder, the CMUCam will be used to find the red target. Once it identifies the red target, it will calculate the distance from the target. As the robot moves, the CMUCam will continue to tell the distance from the target, until the robot determines it is close enough. The robot will then launch its projectile.

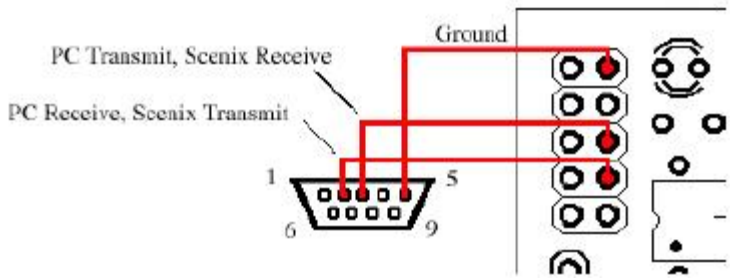
References

Vendor Name: Acroname
Part number: R140-CMUCAM-KIT
Price: $109
Contact Phone: 720.564.0373
Website: www.acroname.com
Email: info@acroname.com
Product manual: http://www-2.cs.cmu.edu/~cmucam/

Integration

The CMUCam will communicate with my MAVRIC-II board via RS232 communication. For initial testing purposes, it can also interface with a PC using the serial connection and a terminal program such as hyperterminal. Upon connection to the microprocessor, the commands and responses can be sent and received fairly easily.

The CMUCam uses only 3 pins on the serial connector: a ground pin, and two transmit/receive lines. See below. The serial connection can operate at 4 different baud rates: 115,200; 38,400; 19,200; and 9600 baud. It uses 8 bits with 1 stop bit and no parity or flow control.

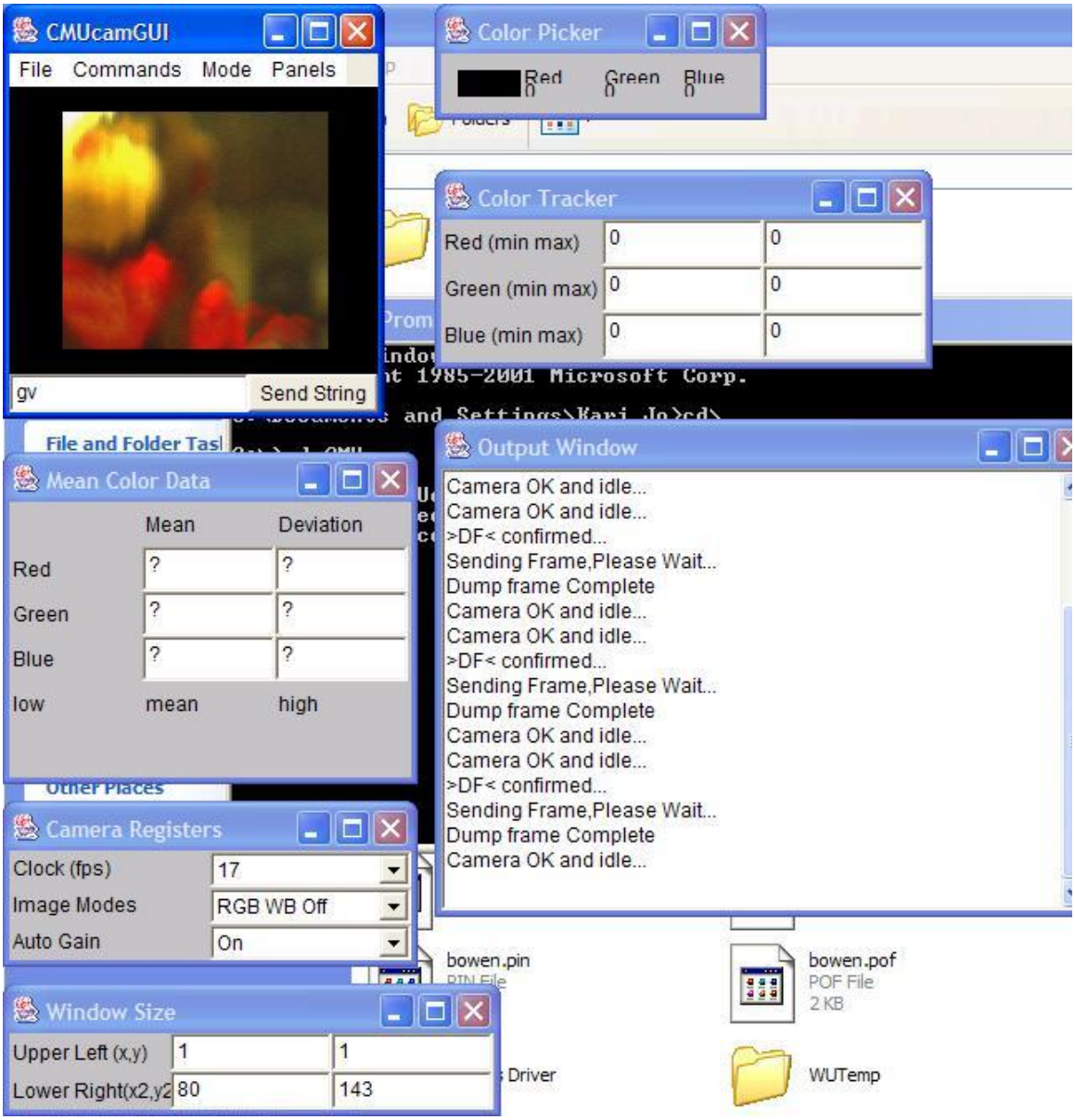


CMUCam Commands

Command	Action
/r	Set board into idle state
CR[reg1 value 1 [reg2 value2 … reg16 value16]]\r	Set internal register values
DF\r	dump frame onto serial port
DM value\r	sets delay before transmission
GM\r	get mean color value
GV\r	get current version of firmware
HM active\r	puts into half-horizontal resolution mode
II \r	uses servo port as digital input
L1 value\r	control the tracking light
LM active\r	turns on line mode
MM mode\r	sets middle mass mode
NF active\r	noise filter setting
PM mode\r	poll mode
RM bit_flags\r	raw serial transfer mode
RS \r	Reset
S1 position \r	set position of servo 1
SM value \r	enable switching mode of color tracking
SW [x y x2 y2] \r	set window size of camera
TC [Rmin Rmax Gmin Gmax Bmin Bmax]\r	track a color
TW \r	track the color in the central region of the currect window

Experimental Data

I have tested the CMUCam using the CMUCamGUI provided by the Carnegie Mellon website. The camera can capture images and dump frames. Commands can be used to find the colors at certain positions. The cam correctly identifies red, blue and green. See below screen shot for example of a dumped frame.



The camera is more sensitive to the color red than any other color. I found that usually a value of 120 or greater signaled a significant amount of red, whereas 100 or more would be a significant amount of blue or green. For my code, I used a simple get mean function with values red>130, blue<80 and green<80.

Lessons Learned

One major lesson learned is to not expect a sensor to work right out of the box. I dragged my feet to finish soldering this kit together, and I have suffered for it! I had trouble making the serial cable as it was a snap-on device. I did not put the cable far enough into the end piece, so I was missing a connection on one of the wires. Upon trying to fix this, I broke part of the snap-on piece.

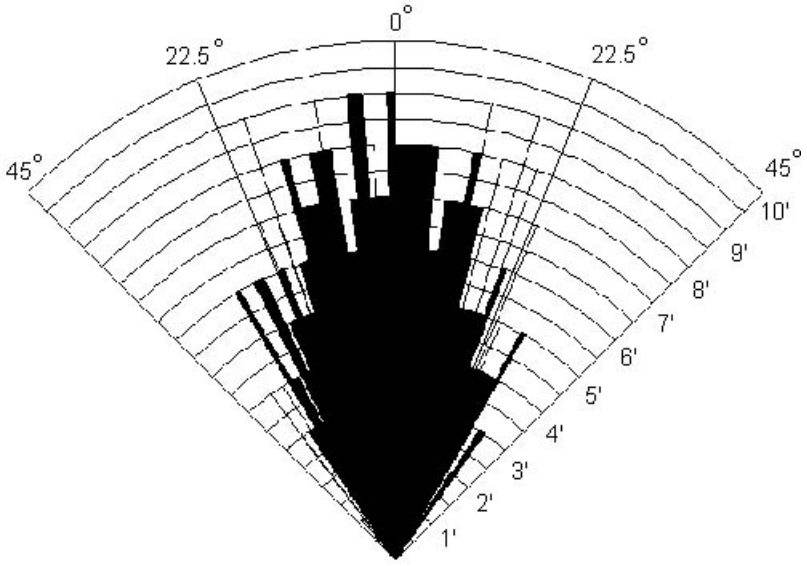
Once I fixed the cable, I plugged the serial connector into my laptop. I then turned on the camera and expected to see "CmuCam v. 1.12." Instead, I got garbage. After a few hours of attempting to debug this, I realized that the supply voltage for the camera should be 6-9v, not 5v. After fixing that, I realized my settings were incorrect in hyperterminal.

Sensor Suite

Devantech SRF04 Sonar

Theory

The SRF04 sensor works on the principle of sonar. The sensor sends out a ping and waits for a response. The time it takes for the ping to return is an indication of how far away an object is. The SRF04 in particular has a range of 3cm to 3 m. The angles at which the SRF04 pings are shown below.



Objective

The SRF04 will be used on Nerfherder for obstacle avoidance. The sensors will ping continuously to determine how far away from any walls or objects Nerfherder is. If Nerfherder gets too close to an object (within some threshold that I will define), an

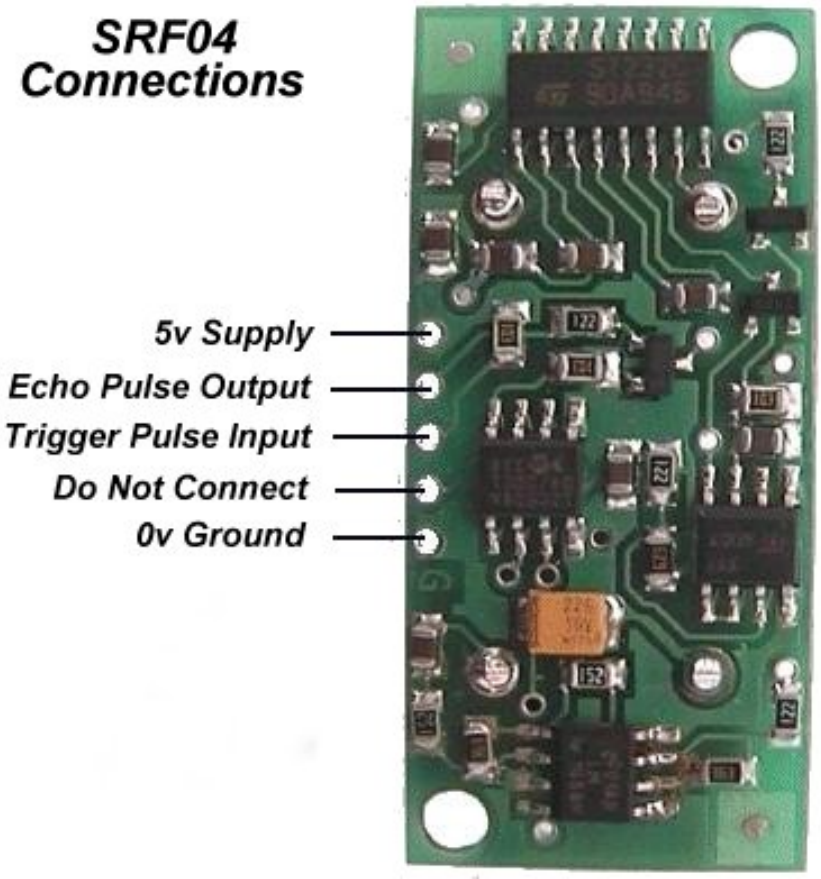
obstacle avoidance routine will run in which Nerfherder will turn away from the obstacle(s) and then continue what is was doing.

References

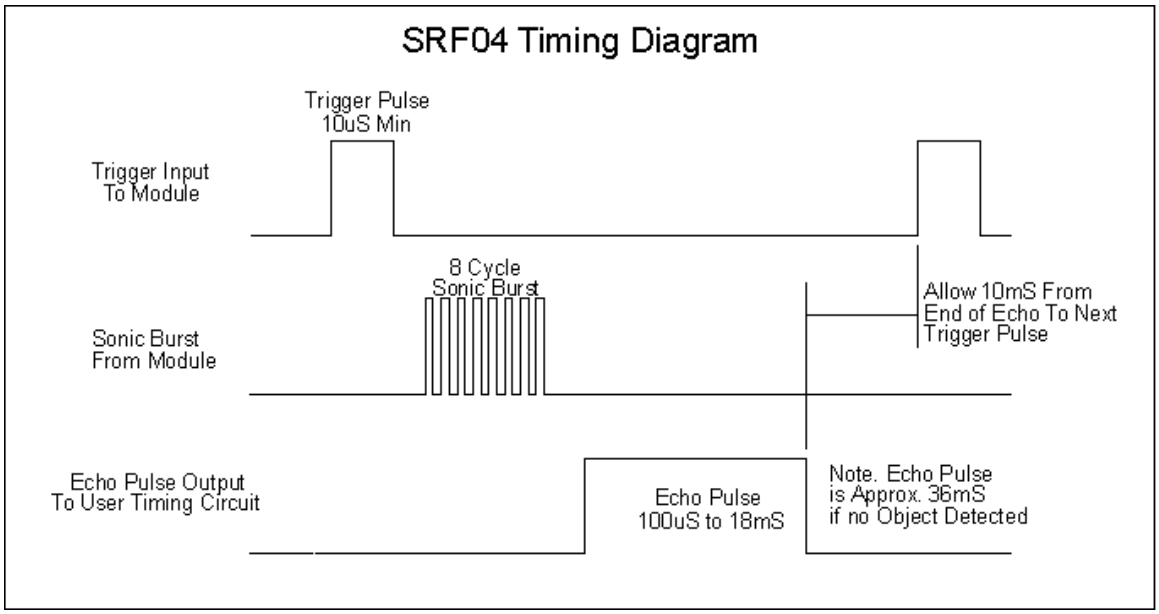
Vendor Name: Acroname
Part number: R93-SRF04
Price: $36 (however, I got them for free from a previous IMDL student).
Contact Phone: 720.564.0373
Website: www.acroname.com
Email: info@acroname.com

Integration

The sonar will be interfaced to the microprocessor via a digital port (for me, Port E and on the MAVRIC-II). The connections for the sensor are shown below.



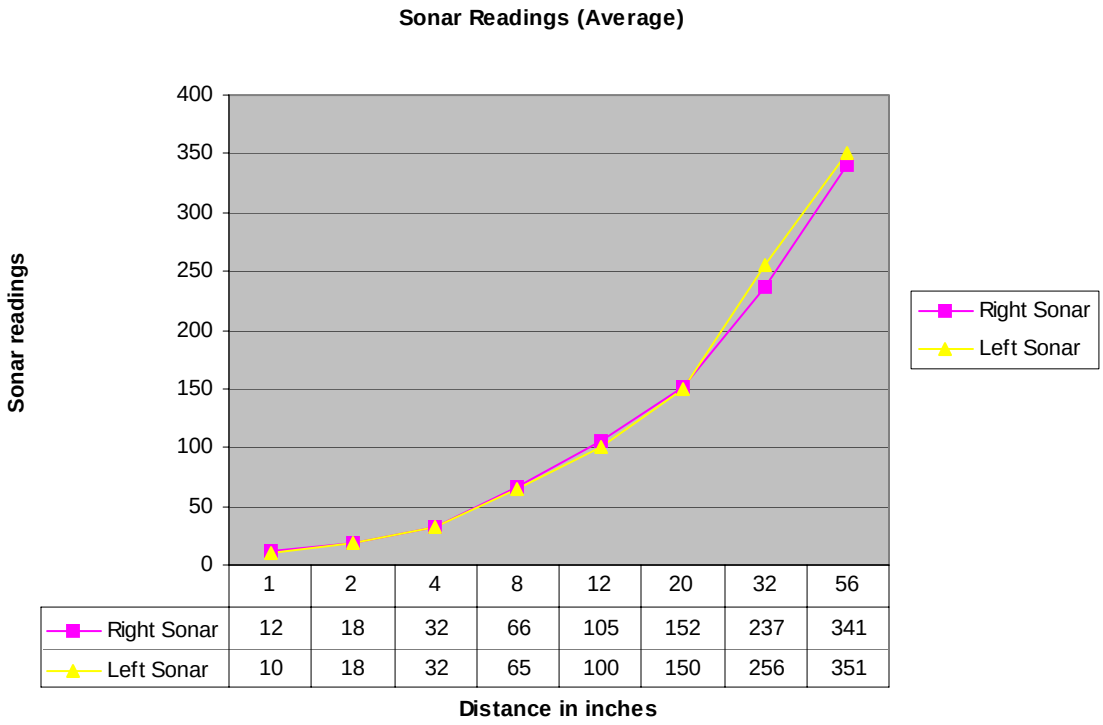
The microprocessor will send out a pulse on the "trigger pulse input" line. This will enable the SRF04 to begin pinging. The microprocessor will then monitor the echo pulse output line to find when it goes low. A timer running will then determine what the distance is to the nearest object. The timing diagram is also shown below.



Experimental Results

The sonar performed at a roughly linear level at mid-range (1-4 feet). Above and below that threshold, there were slight variations. Data shown below.

Measured Distance (in inches)	Right Sonar	Left Sonar
1	12	10
2	18	18
4	32	32
8	66	65
12	105	100
20	152	150
32	237	256
56	341	351



	1	2	4	8	12	20	32	56
Right Sonar	12	18	32	66	105	152	237	341
Left Sonar	10	18	32	65	100	150	256	351

CdS Cell

Theory

The CdS cell contains cadmium (Cd). When it is exposed to light, the resistance inside the cell changes.

Objective

The CdS cell is to be used on Nerfherder as my last sensor, to detect if Nerfherder is loaded. Depending on if the ball is in the chamber, the CdS cell will be exposed to less light.

Integration

The CdS cell must be hooked up to an A to D pin on the microcontroller. The A to D then calculates the value of the voltage coming from the CdS cell so that one can determine the light value.

Experimental Results

I had no luck with the CdS because of the material of the balls being used. They are fairly sheer and let a lot of light through. This is a problem because depending on the lighting of a room (sunlight vs. fluorescent, etc.), the values change. So, even if the ball is blocking some light in the sunlight, it could be comparable to the amount of light

allowed with no ball in the chamber in fluorescent lighting. To use these, there needs to be a calibration due.

Lessons Learned

Since this sensor was not really vital to Nerfherder, I left it until the end to do, and then I realized the calibration issues, etc. If I were to redo this I would start with even the small sensors at the beginning.

Behaviors

Avoiding/Roaming

The obstacle avoidance/roaming behavior uses the sonar to figure out how far away from any objects Nerfherder may be by polling the sonar values in a while loop. If Nerfherder gets within 6 inches of an object, it will turn. Sometimes, however, it manages to get within 2 inches of an object. In this case, it will just stop and wait for reset to avoid damage to the robot. Nerfherder also has a variety of slight turns it performs if the sonar readings are biased to the left or right.

Target Detection

The target detection is a team effort by the sonars and the CMUCam. The CMUCam is constantly sending the average values of the frame. If the frame has the target values of red >130, green<80 and blue <80, then it will stop Nerfherder and start the shooting mechanism.

Shooting

The shooting is controlled by a normal I/O Port, Port F. The lines are simply set high, which causes the transistors to turn on the motors. The motors stay on for approximately 10-12 seconds to shoot the ball.

Conclusion

Nerfherder is a robot that can find a red target and shoot a ball at it. The original scope of the project was to have Nerfherder shoot a ball through the target, but due to difficulties with the CmuCam this did not occur. As I am sure many other IMDLers have said, more time spent on the robot at the beginning of the semester would have produced a much better result. I still plan to get Nerfherder fully working this summer, however.

Acknowledgements

I would like to thank the following people for their contributions to Nerfherder:

Steven Pickles for being a great TA and especially for help with my platform design.

William Dubel for knowing everything!

Mike Bonestroo for buying and shipping me parts and for assistance with the shooting mechanism.

Sara Keen for code tidbits and moral support.

Drs. Arroyo and Schwartz for all their assistance.