Department of Electrical Engineering

EEL 5666

Intelligent Machine Design Lab

Mad Max: An Autonomous Tank

James Curlee

Instructor: Dr. Arroyo

December 8, 1999

Table of Contents

Abstract 3

Executive Summary 4

Table of Contents 5

Introduction 6

Integrated System 7

Mobile Platform 8

Actuation 9

Motor Driver Data Sheet 10

Sensors – Part One. 11

Sensors – Part Two 12

Behaviors – Part One 13

Behaviors – Part Two 14

Experimental Layout and Results 15

Conclusion 16

Appendix A 17

Abstract

The following report discusses the design of an autonomous twin-track tank-like vehicle. This twin-track vehicle began with a simple RC platform and was developed into a autonomous (ideally) all terrain vehicle that could achieve obstacle avoidance, bump sensing, targeting, and the rapid triggering of a simple gun. The robot uses a 68HC 11 microprocessor that relies on feedback from sensors to and controlling algorithms to achieve the attributes mentioned above. The main idea of this robot was to be able to control its motion through some type of motor control circuitry and develop software to implement a targeting system which can track down a signal (beacon) and activate a gun once this target is acquired.

Executive Summary

Mad Max is an autonomous twin-track vehicle that was designed to act out like an automated tank which could track down targets and blow them away with some type of automated weaponry. The challenge is in having a smooth motor controlling device along with a reliable targeting system and accurate firing device. In addition this vehicle must be able to avoid objects quickly and sustain collisions with stationary or moving objects.

Mad Max's primary motion control is through the use of motor driver boards which can control a fast DC motors like the ones present in RC cars. The robot uses three different sets of sensors to interact with its environment and calculate where a target is along with avoiding obstacles and recovering from collisions. The targeting system relies on 32 kHz IR sensory that can achieve a target range between 1 and 10 feet.

The entire robot is controlled with a 68HC11 TJ PRO Board that takes in analog sensor readings to determine its next course of action. (Avoid, hunt, fire weapon) The main controlling algorithm switches between 40 kHz and 32 kHz sensory to achieve obstacle avoidance and target finding respectively. There are two main routines that are switched between depending on the values of the 32 kHz sensors.

Introduction

My goal coming into IMDL was to design an autonomous tank-like vehicle to be equipped with a reliable targeting system and triggering mechanism. I wanted a non-traditional platform so I elected to build upon a twin-track RC car that could maneuver like a tank. Another quality I wanted in my robot was for it to be run by motors instead of servos. The only requirement for a servo I desired dealt with my triggering device. I needed an automated semi-rapid fire gun, so I choose to use a servo to accomplish this. I also wanted to include an interesting bump sensor design into my robot. I felt the MIL golf cart bumper system was a neat idea so I choose to model my system after it. I wanted the overall complexity of my robot to lie in the integration of all my behaviors.

The objective of my robot was to have a fast moving vehicle that could track down targets and fire multiple rounds at each. I wanted a robust platform that could withstand sudden crashes yet achieves quick obstacle avoidance. Eventually, it would b fun to have multiple twin-track robots that could play war games.

The following sections discuss in detail each system of the robot. The report starts with the integrated system describing the role each component plays in the overall design of the robot. It moves on to discuss the mobile platform and what each sub-platform contains. Actuation discusses how the robot achieves its motion. The sensor section describes how each sensor is used to achieve the desired behaviors. The behaviors section goes into detail what each behavior involves and how they are achieved. The Experimental Layout section discusses the approach I took in putting together and testing each component of my robot. Finally, my conclusion talks about my accomplishments and problems I had with getting my robot to work, along with some discussion of the future work I’d like to do on the robot.

Integrated System

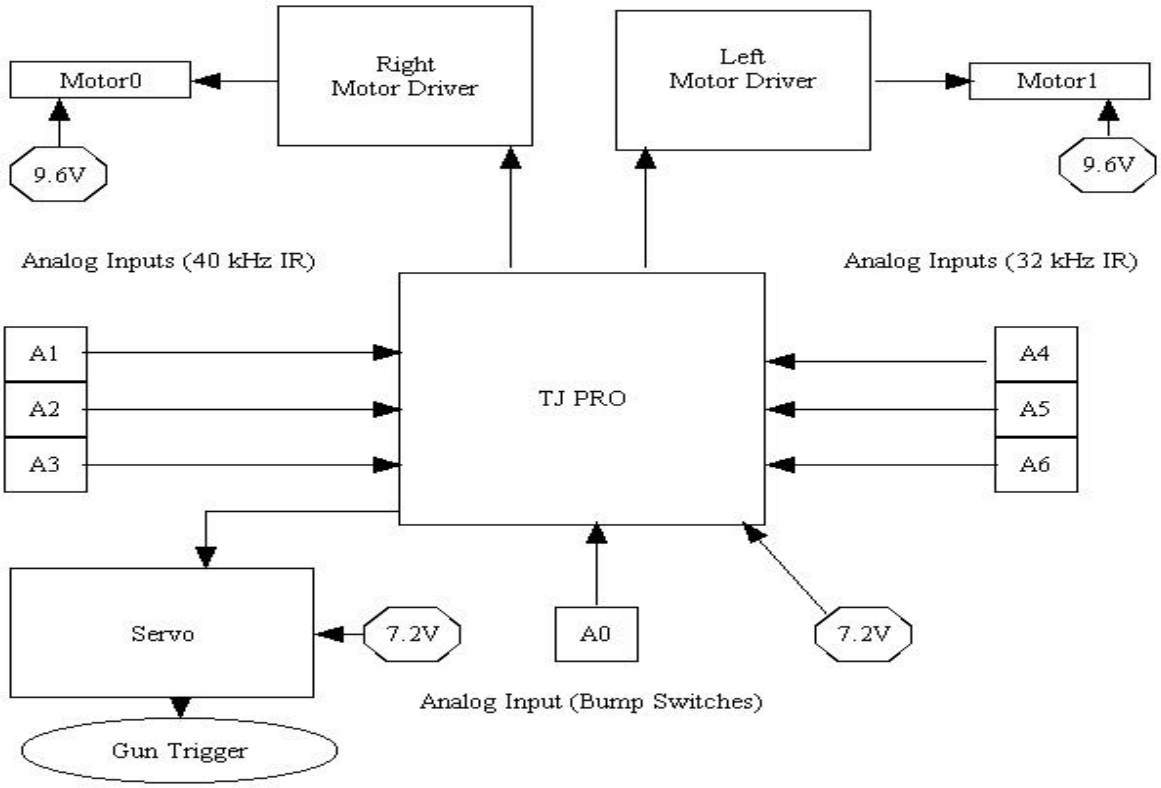


Figure 1: Integrated System for Mad Max

Mad Max's integrated system consists of two DC motors, two motor drivers, a TJ PRO board developed by Mektronix, six Sharp IR sensors, two roller switches, two 9.6V NiCad batteries (1200mAh), two 7.2V Nicad batteries, a 300 oz.-in. servo, and a foam disc shooting gun.

Eric Anderson developed the motor driver boards this summer in MIL. They are rated to handle 5A and contain a slow-blow fuse for safety. These control the two DC motors on the RC car. The TJ PRO board accepts 7 analog inputs, 6 analog readings (A6-A1) from both 40 and 32 kHz Sharp IR detectors, and one analog input (A0) to take the readings of the roller switches. The motor direction is determined by a direction bit signal sent from the TJ PRO board to the motor drivers. Likewise, the servo is sent a signal from the TJ PRO board to control its range of motion. The servo in turn triggers the foam disc gun. Each DC motor is powered by a 9.6 V (1200mA/h) Ni-Cd battery, while both the TJ PRO board and servo are powered by a 7.2 V battery pack.

Mobile Platform

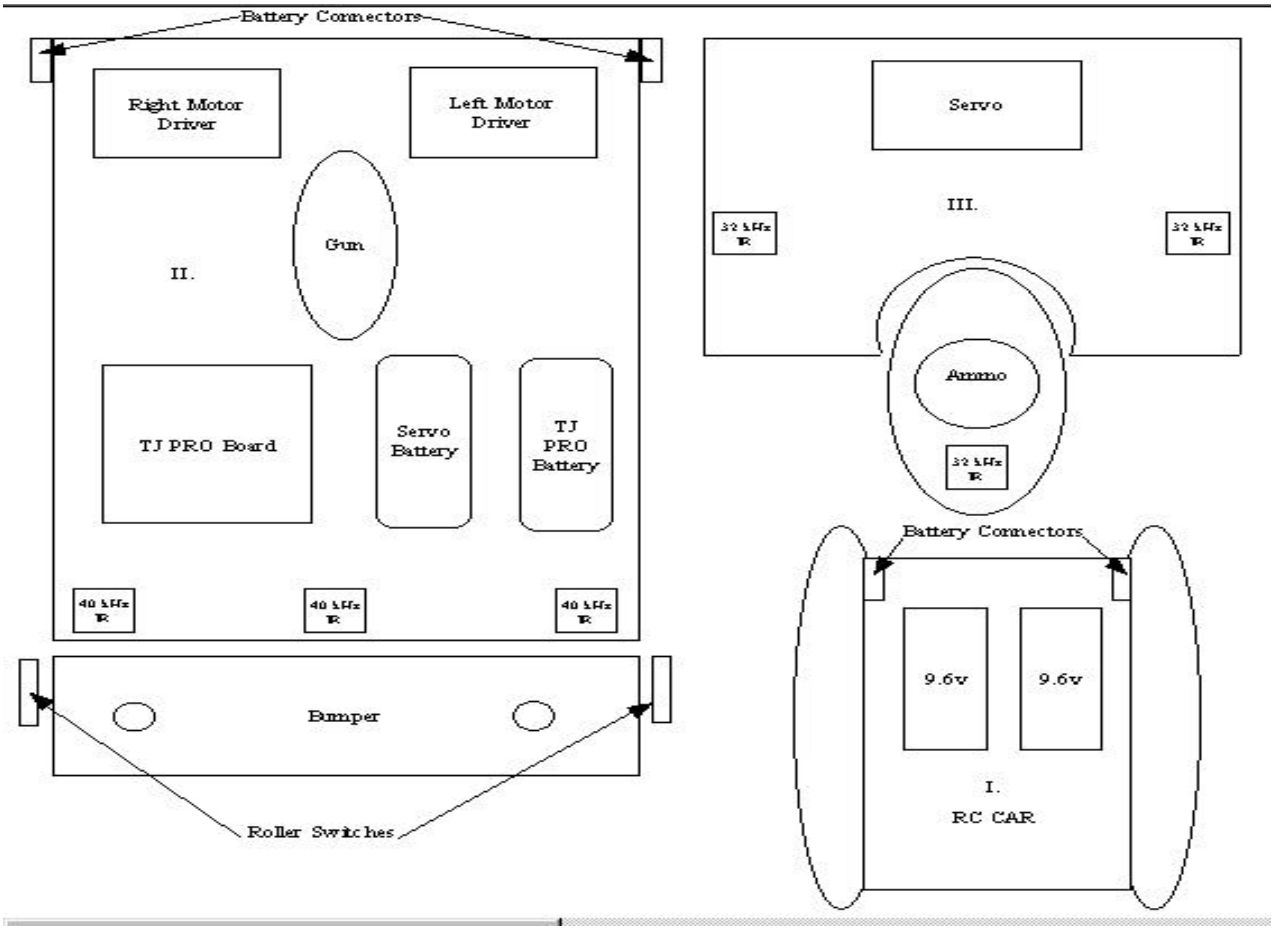


Figure 2: Mad Max Platform

Mad Max's platform is a combination of three smaller platforms, labeled I., II. III. in Figure 2 above. The first sub-platform is the RC car body itself. It originated from a TYCO R/C Nitro Dozer Twin-Track Vehicle purchased at TOYS-R-US. I stripped out the original control board and top portion of the car body to build upon. The second and third platforms were designed in AutoCad and cut out on the T-Tek machine in lab. The second platform is the guts of the robot so to speak containing the two motor driver boards, TJ PRO board, two 7.2V Ni-Cd batteries, IR detectors and is attached to a bumper system. The bumper system is very similar to the one used on the MIL golf cart, which is where I came up with the idea. Attached to the bottom of platform II is a 5¾" by ½" by 2" box. Through the box run two 3" bolts (rods) which are connected to an 8" by 1¾" by ¼" piece of plywood which acts as the "bumper" for Mad Max. Whenever Mad Max runs into an object the rods push backward against one or both of the roller switches indicating a bump has occurred. Mad Max has three 40 kHz IR detectors located at the front of platform II as its "eyes" to avoid obstacles.

Actuation

Battery Power

Mad Max uses a total of four batteries for his normal operation. It uses two 9.6V, 1200mA/h Ni-Cd batteries to power the motor driver boards. In addition, Mad Max
uses two 7.2V battery packs, one to power the TJ PRO board and the other to power the servo.

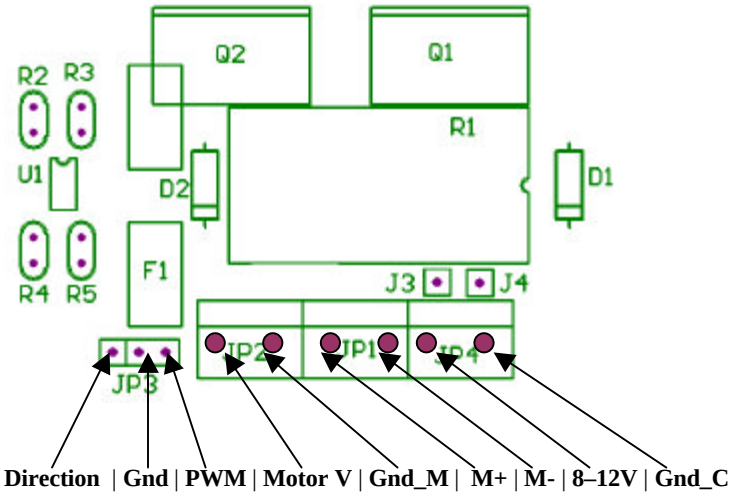
Motor Drivers

Mad Max uses two of the motor driver circuits shown in the Motor Driver 5A data sheet on the page below. Eric Anderson in MIL originally designed this motor driver circuit last summer. He happened to use these same motor drivers to control his submarine. (They work very well if I may add) This circuit uses a Power MOSFET MTP75N to switch on and off each DC motor at a high rate. Pulse widths generated from the OC2 and OC3 pins on PORTA of the TJ PRO board determine the duty cycle (speed) at which the DC motors run. The direction bit determines whether the motors run in forward or reverse. A 0V input means the motors run in the forward direction (which conserves battery life) while a 5V input causes the motors to run in reverse.

Motors

Two DC motors control the speed at which the twin-tracks run at. Since they were designed for an R/C Car the motors can obtain a great deal of speed, so I currently only have them running at a 10-15% duty cycle. This is more than enough speed for me to demonstrate the desired behaviors of Mad Max. Included below is the schematic and components list of the motor driver boards along with a table of signal definitions.

Motor Driver 5A

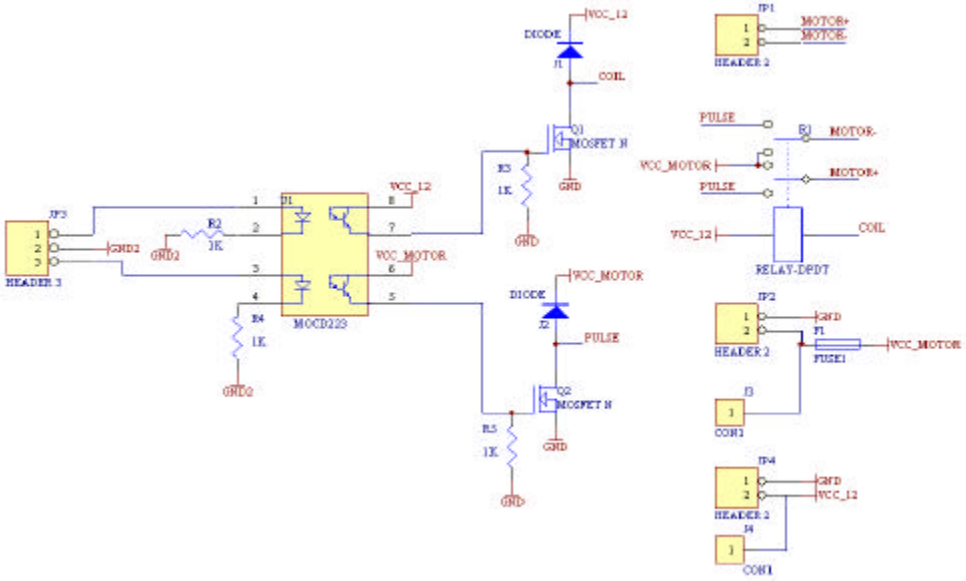


List of terms

Direction - digital input to change polarity of motor
Gnd - same ground used on microprocessor board
PWM – pulse width signal from microp using the output compares
Motor V – positive battery power for motor
Gnd_M – negative battery power for motor
M+ - motor terminal, this can be either the positive or negative. Wire motor such that default direction has coil of relay off.
M- - motor terminal, see M+.
8-12V – indicates voltage input range for coil (battery positive)
Gnd_C – negative battery terminal for coil. Tied to Gnd_M on board

Parts List				
Part Type	**Designator**	**Description**	**Radio Shack#**	**Qty**
1K	R2,3,4,5			
CON1	J3,4	Jumper		
DIODE	J1,2	4N4004		
FUSE1	F1	5A Slow Blow	270-1056	1
	F1	Fuse Holder	270-744	1
HEADER 2	JP1,2,4	2 Position	276-1388	1
HEADER 3	JP3	3 pin Header		
MOCD223	U1	MOCD223		
MOSFET N	Q1	MTP7N20E		
MOSFET N	Q2	MTP50N		
RELAY-DPDT	R1	5A DPDT	275-249	1

Special Note: If motor voltage is between 8 and 12 volts, JP4 can be omitted by placing a jumper between J3 and J4



Schematic diagram of Motor Driver board

Sensors

Mad Max is composed of the following sensory:

- Three 40 kHz Sharp IR detectors
- Three 32 kHz Sharp IR detectors
- Two roller(bump) switches

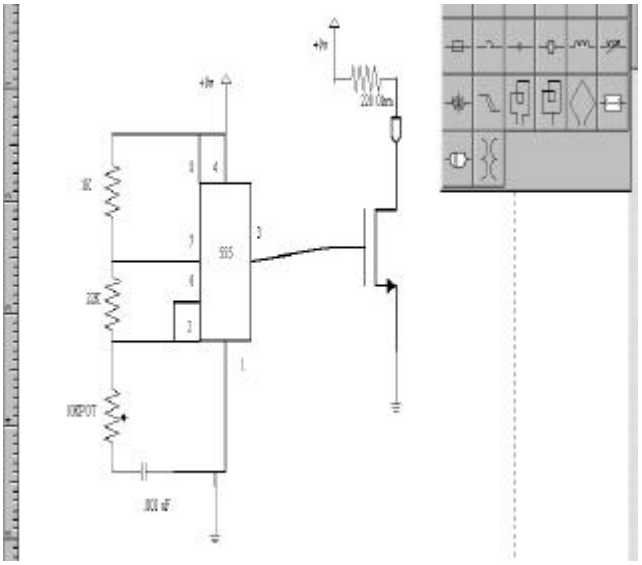
40 kHz Sharp IR detectors

Mad Max includes three hacked 40 kHz Sharp IR detectors to read the intensity of IR reflections in front of the robot. The 40 kHz signal is generated when a 74HC390-decade counter divides the 68HC11 E-clock, activating the LEDs. The placement of the IR detectors on the front of platform II achieves approximately a 120-degree arc in front of the robot. With just these three sensors, Mad Max is able to achieve collision avoidance from most objects.

32 kHz Sharp IR detectors

Mad Max uses three 32 kHz Sharp IR detectors for long range detection. To generate a frequency between 29 and 32 kHz I built the 555 timing circuit shown below. I choose C1 = .001 mf, R1 = 1K, R2 = 22K which is in series with a 10K potentiometer. With these parameter values I am able to achieve a variable frequency between 28 and 32 kHz. To obtain the desired range for my targeting system I used a high current (1.2A) LED with the 555 Timing Circuit. Initially there wasn't enough current being driven through the LED (33 ma!) so I added a TIP120 MOSFET to increase the current through it. With the output of the 555 tied to the gate of the FET, I was able to run 500mA of current the LED when it was across a 33-ohm resistor. This proved to be too much current (it blinded my robot for all practical purposes) I used a 220 ohm resistor instead, This gave me range or detection up to 10 feet.

555 Timing Circuit to generate 30 kHz frequency



Bump Sensing

Mad Max includes a bumper attached to the second platform to achieve bump sensing. The bumper system is similar to the one on the MIL golf cart, which is where I originally came up with the idea. Glued to the bottom of platform II is a 5¾" by ½" by 2" box. Through the box run two 3" rods which are connected to an 8" by 1¾" by ¼" piece of plywood. The rods have springs that rest between the bumper and the box so when Mad Max runs into something the majority of the impact is absorbed by the springs, keeping the robot intact. At the same time the springs contract, the rods are pushed backwards allowing the roller switches that rest against them to open momentarily. When this occurs a signal is sent to analog (0) (PA0) on the TJ PRO board. This signal reads an analog value between 40 and 140 depending on which side was bumped. Analog values of 40, 130, and 140 are read for a left, right, and center bump respectively.

Sensor Integration

My sensor integration was an interesting challenge. Since I am using two different frequencies of IR detectors, I had to account for the case of interference between the two. I used the fact that my target(s) can be easily turned on or off by simply disconnecting the 9V supply to get around this. In my code, I have two main while loops, one for targeting and the other for collision avoidance. If the target is on, my 32 kHz IR sensors will pick up analog readings above 100 very quickly. Due to this, I give the 32 kHz IR sensors priority over the 40 kHz IR sensors. So while the target is on the robot should always be trying to target unless it is significantly far away (> 10 feet). On the other hand, if the target is turned off, then the 40 kHz sensors are given priority and my collision avoidance routine is run. Since I have the bumper switches active at all times, I need not worry about running into objects while looking for a target.

***

One lesson I did learn is to check all your init files you use in your project. Earlier on I experienced a problem with my servos not being able to run while init_motors was being called. With help from Ivan, I was able to locate and fix the bug in my code. In this case, the OC4 bits were never being cleared in init_motors and as a results the interrupt was never occurring.

Behaviors

Mad Max was initially designed to have four distinct behaviors:

- Bump Sensing
- Obstacle Avoidance
- Target Finding
- Triggering

Bump Sensing

This is perhaps the easiest of behaviors to implement, but not particularly with a RC car. To achieve bump sensing with Mad Max I had to design around the fact that my platform was going to be fast. The most I dare run my motors at during behaviors is about 15% duty cycle, and even this proves too fast at times (like when I have freshly charged batteries ☺). I wanted my bump sensing to grant my robot a certain degree of robustness, so I decided on a bumper system similar to the Mil golf cart. Since I've already explained the platform setup of this bumper I won't go into any more detail here. In my code I have a set of conditions my robot checks for to avoid obstacles. Three of these conditions involve bump sensing. By having two roller switches (one for the left and right side of bumper) I have three cases: my robot was bumped from the left, the right, or in the center. Each case yields a different analog reading (ranging from 40 to 140) on analog (0) (PE0 on the TJ PRO board). If you examine the code for bump sensing included in the appendix, you can see the actions Mad Max will perform for each case.

Obstacle Avoidance

Obstacle Avoidance was interesting to implement with a RC car. Since my robot is often traveling at fast speeds relative to say a TJ robot; I had to adjust my avoidance code to make Mad Max react quick enough to avoid objects in time. This mainly involved adjusting the analog threshold values to which Mad Max responded. One example being say if Mad Max was cruising along and suddenly came upon a wall in front of him. I had to write the code such that it gave priority to the center IR so that Mad Max would know to stop and reverse rather than turn a quick left or right and possibly smack into a wall. A better understanding of how I achieved this behavior is illustrated in my avoidance code included in the appendix.

Targeting

To achieve targeting on my robot I decided to use IR sensory versus sonar for a couple of reasons. The first was the relative success IR sensory has had versus sonar in the past. The second reason was that I felt I could achieve better target accuracy with IR sensory. IR sensory gives me a better line of sight targeting scheme, which is fairly straightforward to implement. For Mad Max, I choose to use three 32 kHz Sharp IR detectors to home in on a target. The way I designed my code was to have Max sweep left and right for a short while recording the maximum reading the Center IR found during that sweep. I then would have Mad Max turn right or left again depending on which IR had a stronger reading at the time. If during the turn the center IR decreased in value I had Max turn back again until it found this value again. Once this was determined I sent Mad Max forward for 1.5s. The robot would then repeat this process until the Center IR became greater than the threshold value I set for it initially. Overall this scheme seems to work quite well. The one problem I did discover is that since my robot is significantly heavier than when it first began this semester, it has an annoying amount of gear slippage due to the poor traction it has with the floors in lab.

Triggering

This behavior was by far the most fun to implement on my robot. To achieve an automated firing mechanism, I purchased a simple firing mechanism. The gun I decided on is a foam disc shooter. It is ideal for my application in that it has a reloading chamber that holds up to twelve discs and has a simple triggering method. The one downside to a gun like this is that it often jams or misfires due to the lightweight of the discs. To implement the automatic triggering I placed a 300oz-in servo about three inches behind the gun. I connected the servo to the trigger via a soldered loop of electrical wire. Tony (ME) helped me in designing this and it has thus far proven very reliable. The discs may not always come out, but the gun can fire at a variable rate (depending on the pulse I send to the servo) and a variable number of times. The only problem I ran into was getting the servo and motors to work together, which turned out to be an initialization error in my init_motors routine. Overall, once the target has been acquired, the robot then releases three to four discs at the target with good precision. (Within 10% of the target most of the time)

Experimental Layout and Results

To test my behaviors I first made sure each behavior worked separately. The first experiment was to test my TJ PRO board and make sure each sensor worked before anything. Once I had determined each of my IR and bump sensors worked, I tested each of my actuators. After completing tests to assure my motor drivers and motors worked properly, I began coding for my obstacle avoidance routine. I started with the simple obstacle avoidance code that was written for the TJ robots and adjusted it to meet my robots needs. (Since my platform is fast I needed lower threshold values and a few more cases) Upon getting my obstacle avoidance to run, I put together my timing circuits to test the range of my 32 kHz sensors. At first it didn't appear I'd get the range I needed from them, but after adding the MOSFET at the output of the 555 chip I was able to drive a lot more current through my 1.2A LEDs. (500 ma at one point) Once I decided upon a targeting range suitable for my robot (10 feet due to the limitation of my gun range) I began coding my targeting routine. The first few routines were quite unsuccessful; basically my robot would run around the target but never stop when it got near. I finally came up with a sweeping algorithm that involved keeping track of the maximum Center IR value as my robot swept an area. With this routine I am able to center on the target with little error. The next task was to design my triggering mechanism. With the help of some of the ME students in class I was able to come up with the design mentioned in the behavior section of this report. To date this has worked great for my robot; he hasn't missed a triggering yet. (Can't say the same for the ammo I am using though) The last and final test was to integrate all of these behaviors together. This was a problem at first for I wasn't able to have my motors working alongside with my servo. The problem as I've mentioned was with the initializations of the output compare bits in init_motors. All I had to do was clear the OC4 bits so they would be able to interrupt. (Previously this wasn't happening)

Conclusion

Overall, I accomplished the behaviors I set out to in the beginning of class.
I achieved bump sensing, obstacle avoidance, targeting, and triggering all working together.

The limitations posed upon my robot were:

- The range constraint of my gun (foam discs are accurate up to about 8 feet)
- Occasional gear slippage due to the weight of my robot (causes my motors to get stuck more than I like)

The areas I was the most impressed with were the triggering and obstacle avoidance system of my robot. The triggering mechanism was straightforward and easy to implement yet is very robust and reliable. The bumper system on Mad Max increased the overall robustness of my design. I felt my design was fairly compact for all the components I needed.

One area my robot could be improved upon is the gear slippage problem. I need to look into a way to make the overall movement of my robot smoother. Often times the tracks will get stuck or not turn as well as they should. Also, I could come up with more reliable ammunition. (foam discs are not the wave of Mad Max's future)

Future Work

If I had it to do all over again I would have tried to find a RC platform with better traction under heavier weight. Also, I would have liked to have a more enclosed system. I could have made a top piece to cover everything except the gun and IR detectors.

Future work for my robot would include expanding my targeting routine to tracking down moving targets as well at stationary ones. I would also like to add an indicator light that lets me know when I'm out of ammunition. Another neat little feature would be to have a way to tell if the gun has misfired. (I.e. ammo never was released)

Appendix A: Final Demo Code