Cobra Automated Tabletop Sentry
(C.A.T.S.)

Dana Preble
Intelligent Machines Design Laboratory
Spring 2009
Final Report

Professors:
Dr. Eric M. Schwartz
Dr. A. Antonio Arroyo

Teaching Assistants:
Mike Pridgen
Thomas Vermeer

Table of Contents

Abstract	2
Introduction	2
Integrated System	2
Mobile Platform	3
Actuation	5
Sensors	5
Behaviors	7
Experimental Layouts and Results	9
Conclusion	9

Abstract

C.A.T.S. stands for the Cobra Automated Tabletop Sentry. After being placed on a table and activated, it will scan the tabletop for any targets, and push them off of the table. Hostile targets are laser-equipped G.I. Joe figures. In reality, these will be large red LEDs. It will disregard any object that doesn't have these lights. After pushing the targets off of the table, it will turn around and resume the search.

Introduction

While I was a child, I loved playing with G.I. Joes. Many other children, and even some adults, have over the years. What if a G.I. Joe craft could play back? What if an evil Cobra vehicle was out to get your heroic Joes? What if the forces of Cobra were out to wreck all of your precious China on your kitchen table? Cobra Commander would love it. My own childhood, and the childhoods of many others, inspired this project.

Integrated System

The system controlling this robot will be the PVR board, designed by the Teaching Assistants. It has an Atmega 128 microcontroller on board. This board has four built-in analog to digital converter channels, six PWM ports for controlling servos, and six general I/O ports. In C.A.T.S., four sensors dictate the robot's behavior. Two infrared sensors and an ultrasonic rangefinder use the analog to digital converters for accuracy and efficiency. A CMU Camera, from Carnegie Mellon University and Seattle Robotics, uses serial communication to track objects by color. All of these sensors cooperate to locate a target, move toward a target, push it off of a table, locate the edge of the table and stop, and then it turns around and repeats the process.

Mobile Platform

I was considering a simple, yet effective and efficient (in terms of space) body for my robot. I was searching through some childhood memories, and I found a very old G.I. Joe vehicle, a Cobra snowmobile, that was the perfect size for my idea. Along with being the perfect size and lightweight, this inspired me to make the theme G.I. Joe based. The original model was called the “Ice Snake”, shown here.

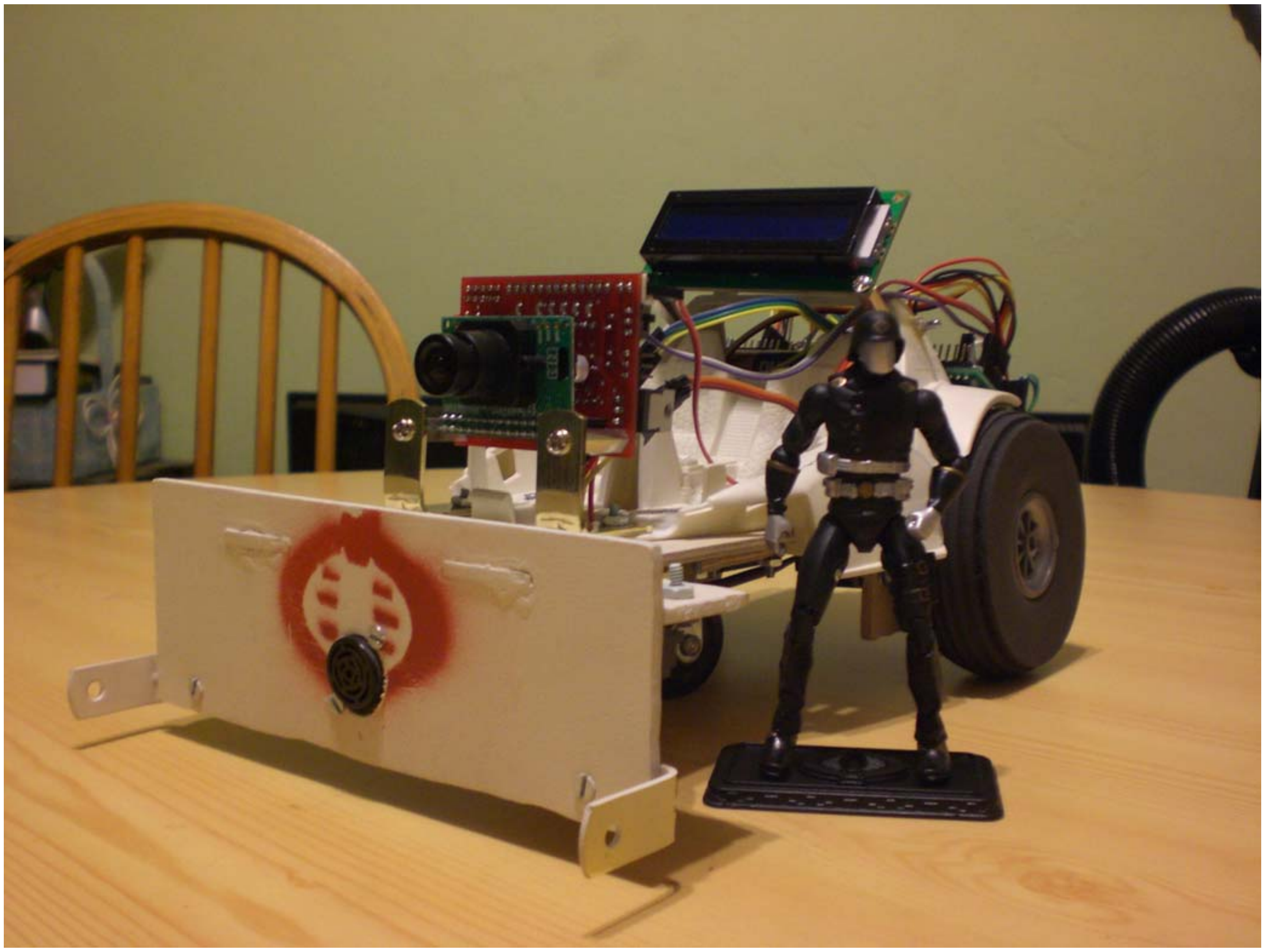
However, this picture is from an online source, showing the toy in perfect condition. My vehicle was purchased in 1992, and through many years of rough play, the skis are bent out of shape, the wheels and cannon are missing, and the once white plastic is was a yellowish hue.

I painted the car, and attached a lightweight front-end dozer to the frame. With a front-end dozer, it can ram and push any object in front of it. The servos provide more power than expected, and it can push heavier objects than originally predicted. A CMU camera is placed on the front of the vehicle, allowing for an unobstructed view of the front of the vehicle. Servos have replaced the wheels, and a caster has been placed near the center of the vehicle. The battery pack for the sound module and the microcontroller makes the vehicle's center of gravity directly over the servos. This keeps the platform stable when traveling over any bumps or colliding with any heavy objects.

All four sensors are placed on or above the front-end dozer of the vehicle. The sonar and two infrared sensors are under the dozer, for optimum use and neatness. The CMU camera is towards the front of the vehicle.

This platform is sturdy and lightweight, which is ideal for the small amount of torque that the servos provide. The vehicle gets around quite well and provides more than enough pushing power for its application.

Here is a picture of the completed platform:



Actuation

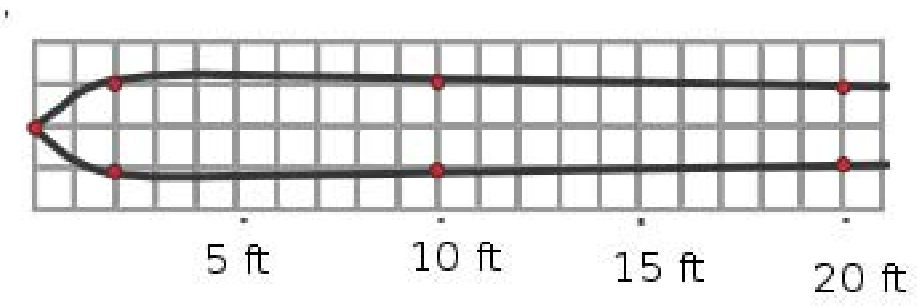
Since this frame is very small and lightweight, not much torque is needed to get it moving. I have decided to use servos to drive my vehicle. I chose them mainly for their simple implementation and their low cost. My movement algorithm relies on the ability to drive straight ahead to knock an object off of the table top, and I know servos are imperfect devices that rarely go in a straight line. I plan on implementing a self-straightening algorithm, along with a convex dozer to “catch” the object it is pushing to compensate for any imperfect driving.

I am using HiTec HS-325HB servos. They have a stall torque of 42oz/in, according to their statistics sheet. It is low power, but the platform is lightweight plastic. I intend to simulate G.I. Joe battles in presentation, so the targets will also be very light. However, the servos can push objects much heavier than the designated targets.

Sensors

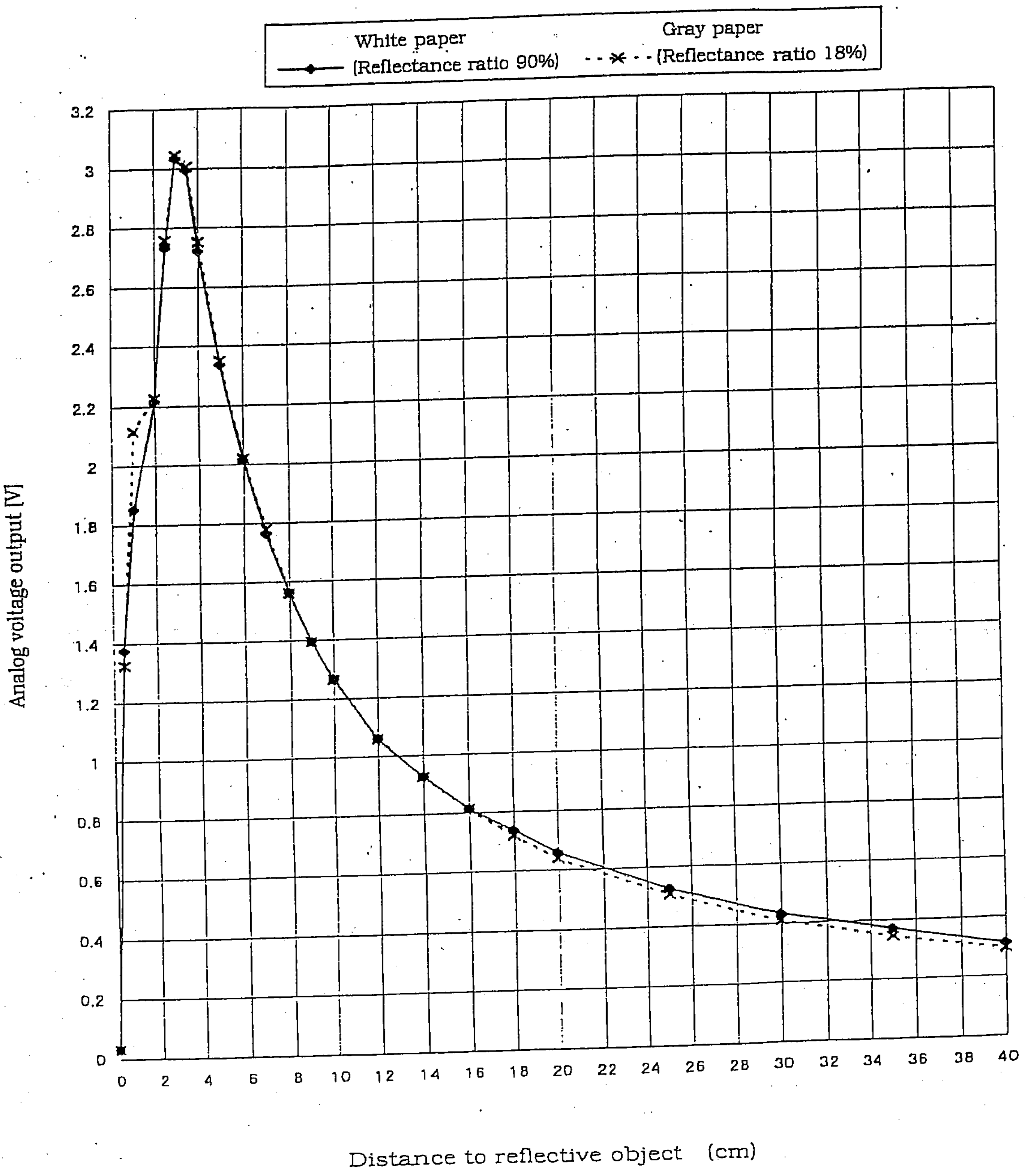
The tasks for the C.A.T.S.S. are pretty simple, but require an array of sensors to function. Most importantly, it must not travel off of the table. It must be able to detect the edge of the table, stop in time, back up, and resume searching for objects. In addition to this, it must find an object and tell if that object is a designated color.

A sonar range-finder will be fixed on the hood to detect objects. I chose the model with the more narrow range, so that it will only detect objects directly in front of the car. That, in conjunction with the camera, helps the vehicle pinpoint targets. Its cone of vision is shown here:
(Vendor: Sparkfun Electronics, item number SEN-00242.)



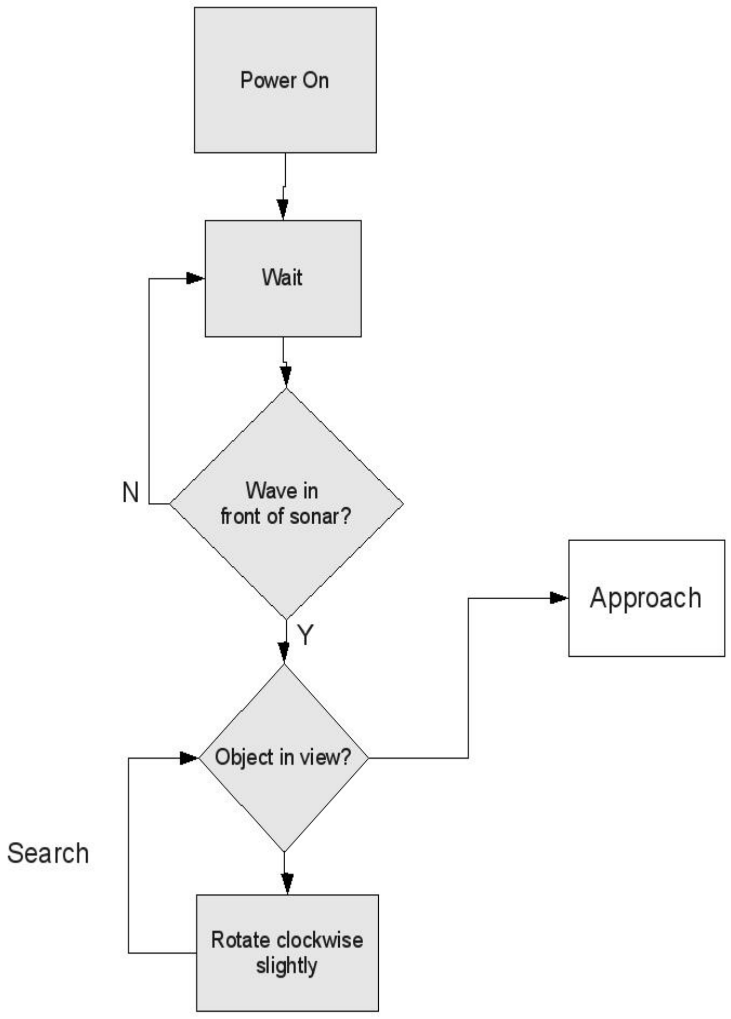
The CMU Camera is used for color tracking. It communicates with the PVR board via UART serial communication. In my code, there are a few functions to take input from and transfer commands to the CMU camera. I only use the x coordinate of the middle mass tracking mode of the camera, so it simplifies my code. However, I could use it for tracking in the y-direction if I wanted to improve upon my design.

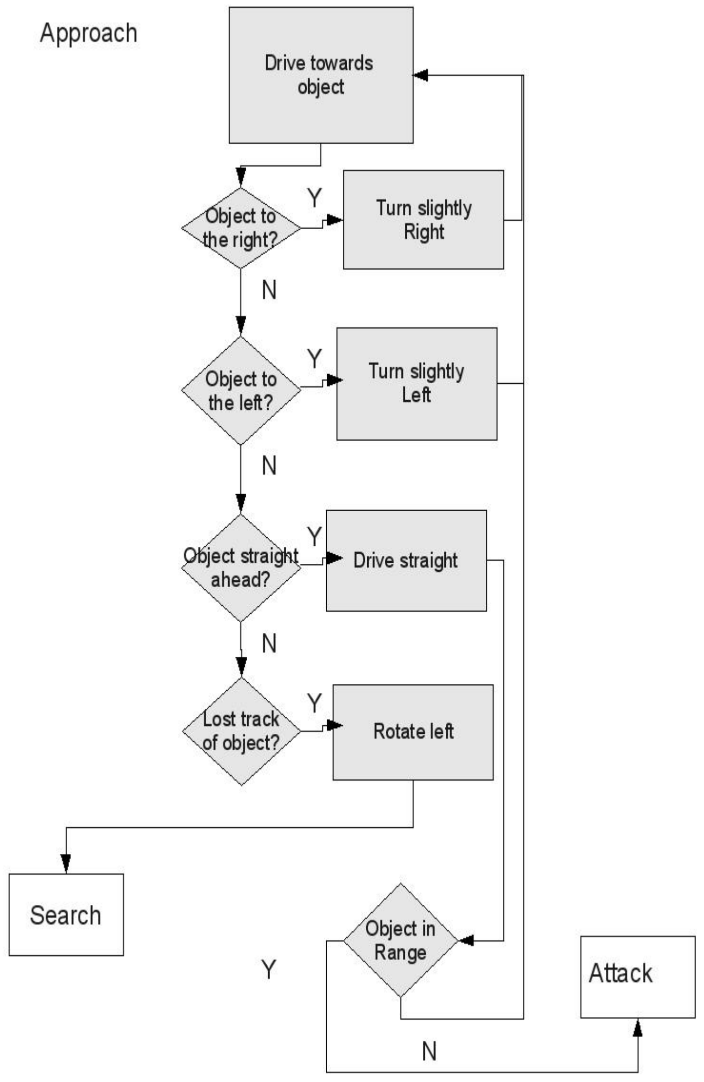
I have chosen a pair of infrared sensors for sensing the edge of the tabletop. These are very accurate and, more importantly, react faster than other methods. This is the diagram that plots the voltage output versus the distance to a given object. Notice the sharp decline when the object goes further away from the sensor. This makes determining when the vehicle approaches the edge of the table quick and easy. (Vendor: Sparkfun Electronics, item number SEN-08959)

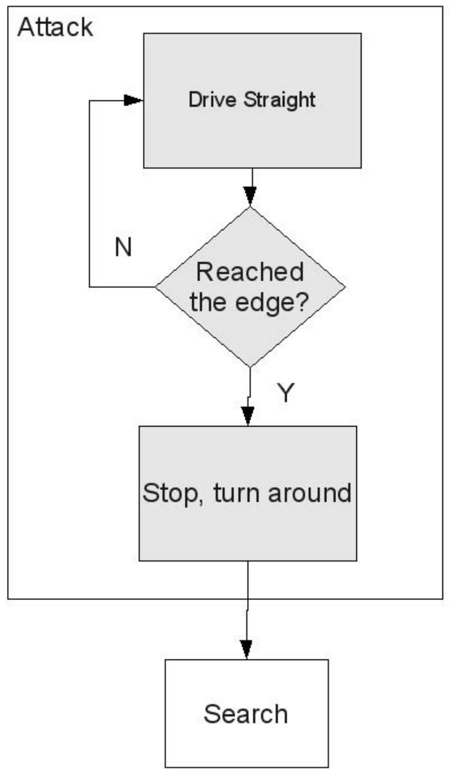


Behavior

There is only one similarity between my robot, C.A.T.S.'s behavior and that of my cat's: They both like to knock things off of my table. The behavior of my robot is best described in a flowchart:




Approach
Drive towards object
Object to the right?
Y
Turn slightly Right
N
Object to the left?
Y
Turn slightly Left
N
Object straight ahead?
Y
Drive straight
N
Lost track of object?
Y
Rotate left
Search
Object in Range
Y
N
Attack


Attack
Drive Straight
N
Reached the edge?
Y
Stop, turn around
Search

Experimental Layouts and Results

To test the speed of the Sharp IR sensors, I ran a routine that would drive the car randomly about the table, and it would stop if either one would detect the edge of the table. This was a success. The infrared sensors are very responsive and I could let my robot drive about and it would not fall off of the table.

One major issue came up with these infrared sensors, however. If the robot was not perpendicular to the edge of the table, the target would not always be pushed off of the table. I wrote a routine that would have the robot rotate and adjust itself until the robot was perpendicular to the table's edge.

The ultrasonic rangefinder has a narrow field of vision. This is a form of double-edged sword. The sensor can accurately detect the distance from the car to an object. I set the threshold to ignore values longer than about three feet, and it will go into its attack mode when an object is six inches or less in front of the vehicle.

The CMU camera's speed and accuracy controls my entire target finding algorithm. I originally had the car spin continuously until it found an object, but the camera, and gathering data from the camera, was not fast enough. The robot would rotate past its object and then attempt to track/attack it. To compensate, I have the robot rotate in small increments and it stops when it attempts to track a color with the camera. The camera also has a narrow field of vision – and this field gets more narrow as the car gets closer to the target. To compensate for this, whenever the camera loses track of an object, it will rotate to the left and then re-scan the area for the target. These adjustments make for a rather fluent target tracking algorithm.

Conclusion

As a software engineer, most of my problems were corrected with software. As a result, my software is rather robust for a simple platform. However, my code is efficient and easier to understand.

An added bonus is that my robot plays sound. I tapped into a toy that plays the G.I. Joe theme, wired a signal to an output pin. My bot plays the song whenever it enters its attack mode.

After tweaking several aspects of my robot, it has a rather clean tracking algorithm and it can seek out red targets thanks to my bright red LEDs. It is sure to impress any G.I. Joe fan, or even any robotics enthusiast.