Jason Reeder

Final Report

EEL5666C IMDL

Arroyo, Schwartz

Table of Contents

1. Abstract

The robot will mimic a dog that always wants to find and be next to its owner. The owner will then be able to repel the dog by pressing a push button on the platform. The dog will then turn and run away for a short period of time before searching for its owner again.

2. Executive Summary

This project was to make an autonomous robot that had some sort of special feature. I decided to create a robot dog that would be able to find its owner and then also run away from its owner once it was signaled. The 'dog' is made up of a mobile platform of two circular tiers that house all of the electronics and parts used for movement. The main board used as the brain of the robot is the Arduino Mega. This board interfaces with two motors drivers, four sonar sensors, a CMUcam, and a pushbutton. The two motor drivers are used to drive two twelve volt motors in both directions allowing the robot to move forward, backward, or spinning in either direction.

The sonars on the robot were used for object detection and also avoidance. The robot is able to move around a room and avoid obstacles that appear in its path. The CMUcam was used in order to detect the color red and return an X value for its left and right position and also a number of pixels that lets the robot know how close it is to its owner. The push button on the top of the robot was used to signal the robot to turn and run away from its owner.

The robot used a fairly simple state machine in order to create its behaviors. The first state was the searching state in which the robot would spin in a circle and try to find the owner. Once the robot saw the owner it switched to the approaching state in which it would approach the owner. Once in the approaching state if the robot lost sight of the

owner it would switch back to the searching state. If in the approaching state and the robot got close enough to the owner it would switch into the waiting state where it would either wait for the owner to move or it would wait for the owner to press the push button to signal it to run away. If the owner moved away the robot would switch back into the searching state, but if the owner pressed the button the robot would go into the running state. In the running state the robot moves away from the owner, turns around, and begin to move in the opposite direction while avoiding obstacles. After a period of time the robot changes back to the searching state and starts its behaviors all over.

3. Introduction

The project is to create a fully autonomous robot. The robot is expected to be able to move around freely while being able to avoid obstacles using attached sonar sensors. The robot will also use a CMU camera to track and follow a primary color, which will be a bank of red LEDs. The robot should also be able change its behavior once a pushbutton is pressed signifying that the owner wants the robot to move away from them. The rest of this paper will cover the platforms, systems, and methods that will make all of this possible.

4. Integrated System

The brains behind my system will be the Arduino Mega board based on the ATmega1280 chip. For movement my robot will interface with two nine amp motor drivers connected to two twelve volt DC gear motors. To accomplish object detection and avoidance the robot will use four sonar sensors that will return the distance to nearby objects in each direction. The main board will take these analog values from the sonar sensors and make determinations on which direction to move if an object becomes too close. A CMU camera will be used to track the color red in order to identify the dog's

owner. A push button will be used in order to signal the robot to change its behavior and run away from the owner.

5. Mobile Platform

The platform for this robot will be based around the need for it to be symmetrical about its center axis in all possible ways. The motors will need to be mounted carefully in the center in such a way as to not make the robot lean to either side. The boards and especially the battery packs will need to be centered and lifted up as far as feasible in order to give the platform a higher center of gravity. Two circular platters will sit above the motor brackets and the wheels to hold the electronics and power source. In order to allow movement there will be a third wheel added that will be unpowered and will only be used for balance. Even though time did not permit me to make the robot balance itself, much of the platform designed was geared towards allowing the robot to self balance on its two wheels in the future.

6. Actuation

The only actuation that will be occurring on this robot will be used for the robot's motion and it will occur through two DC gear motors. The motors that I have chosen are 12 volt 131:1 gear ratio metal gear motors. At twelve volts they will free run at approximately eighty-two revolutions per minute while pulling five hundred milliamps each. The stall current is five amps for each motor when ran at twelve volts. Since the stall current on these motors is fairly high I have to use two separate motor drivers. Pololu manufactures the motor drivers that I selected and they can handle nine amps of current continuously and a peak of thirty amps. The motor drivers have two digital inputs that are used as a truth table in order to make the motors move either clockwise, counterclockwise,

brake to VCC, or brake to ground. Once the direction is selected the speed of the motor is sent to the driver using pulse width modulation which is passed on to the motor as an analog voltage between zero and twelve volts. In my design I ended up only supplying the motors with around eight volts and only running them at fifty percent speed. This means that my motors were only running at around one third of their maximum speed, which was still fast enough for my purposes.

7. Sensors

For object avoidance there is an array of four sonar sensors used. When an object gets within a preset threshold of one of the sonar sensors the robot recognizes this and maneuvers in such a way as to avoid a collision. The sonar sensors are daisy chained together and pass along a signal between them to ensure that only one sonar is collecting data at a time. To initialize the chain effect there is a sequence of delays and signals that must be used. First you must wait at least two hundred fifty milliseconds after applying power to allow the sensors to boot up correctly. Then you must use a digital IO pin on the microcontroller to send a high signal to the RX pin on the first sensor in the chain. This high pulse must last for at least twenty microseconds. Once that is finished you must return the IO pin to a high impedance state so that it will not interfere with the chaining signal. This is accomplished by changing the IO pin to an input. Once the sonar chain is running you can check the distance from any of the sensors by reading their analog voltage value. This will always return the most recent data that the sonar has collected. Once you have read the voltage from the sonar the conversion is $V_s/512$ per inch. Since I am supplying the sensors with five volts this value comes out to around 9.8 millivolts per inch. Through

experimentation I have been able to accurately measure distances ranging from 5.68 inches all the way up to 10 feet under optimum conditions.

In order to find the dog's owner I am using a CMUcam1 to detect the color red. The camera communicates with the microcontroller using serial so there are only five wires connecting the camera to the main board, including one power and two ground wires. The default baud rate of the camera's serial communication is 115,200 but through the use of jumpers it can be reduced all the way down to 9600 baud if necessary. The lower baud rates may cause the camera to operate at a slightly slower frame rate however. The default frame rate of the camera is 27 frames per second but that can also be varied all the way down to 4 frames per second using the camera's internal registers if desired. A specified protocol is used to communicate with the camera and it is fully specified in the CMUcam1 datasheet that is widely available on the internet. In general basic commands are sent to the camera over serial and the camera responds first with an 'ACK:' response to acknowledge the command and then the camera may optionally send a response pertaining to the command. In my use I will only need to use two commands. The first command that I will use is 'PM 1' and this will change the camera out of polling mode and cause it to only respond to each tracking command with one frame. This will allow me to query the camera for only the information I want and when I want it so there won't be overflowing data on the serial port when I don't need it. The second command that I will use is the track color command. This command takes a range of each red, green, and blue and then looks for a blob in the range of colors and then returns location information about the blob. An example track color command is 'TC 200 255 0 30 0 30'. This command will cause the

camera to send back location information for the color red since that is the range that was given to it.

A digital bump sensor has also been used on the robot in order to detect when the owner would like the robot to run away. This bump sensor has three wires that are connected to power, ground, and a digital input on the board. The board can call a digital read on the input pin at any time and the value return will be a zero if the button is not pressed and a one if the button is currently being pressed. The switch used is already debounced which means that it does not bounce between zero and one before settling after only one press of the button. This allows me to save processor cycles because I do not have to debounce the switch in software.

8. Behaviors

The robot will begin in a search state. In this state the robot will turn in a circle looking for its owner denoted by a bank of red LEDs. If a specified number of loops occurs without the robot seeing its owner it will stop spinning and move forward to find another spot to spin and look for the owner. Once the robot sees its owner it will immediately switch from the search state into the approaching state. In the approaching state the robot will check the X value and the number of pixels returned from the CMU cam to determine whether or not to turn left or to stop. If the X value gets outside of the threshold then the robot will turn right or left in order to center the red LEDs in front of it. If the CMUcam loses sight of the LEDs the robot will revert to the search mode and will begin to circle again in the last direction that it saw the lights. If the CMUcam can still see the LEDs and the number of pixels is above the preset threshold then the robot knows that it is close enough and it will stop and switch into the waiting state. In the waiting state the robot is waiting for the LEDs to move further away so that it can switch into the searching or approaching state again, or the robot is waiting for the push button to be pressed which will switch it into the running state. Once the push button is pressed the robot changes to the running state and it will back up turn around and begin to move away from where the LEDs were while using the sonars to avoid collisions. After a period of time the robot will change from the running state back into the searching state and will begin the cycle all over again.

9. Conclusion

At the end of the semester I was able to accomplish nearly all of the goals that I set out to achieve. The robot can move around and use its sonars to avoid obstacles. The robot can find and track a color using the CMUcam. I was able to get the behaviors programmed correctly and make the robot find its owner, wait for a signal, and then run away.

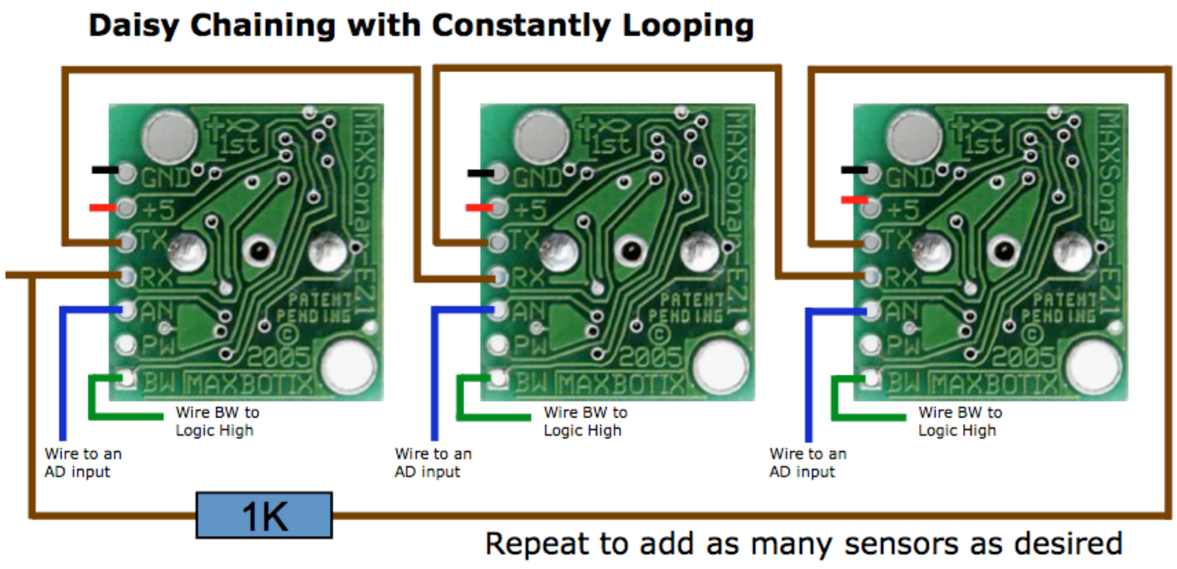
If I would have had more time and more money I would have like to have purchased an accelerometer and a gyroscope in order to get the robot to balance itself on two wheels. I am not sure if the processor would have been able to run enough loop to keep the robot upright though since the CMUcam slowed the overall loop down to only around 8.5 Hz. One area that can definitely be improved is the speed of the loop. I need to find a way to read the values from the CMUcam through the serial port much faster. The loop should have been able to run at twice the speed that it is currently running at.

If I could start the project over I would have definitely tried to get the robot moving and tracking the color much sooner in the semester so that I could have focused on the balancing aspect at the end of the semester. I also would have spent more time making cables and measuring wires in order to make my robot look cleaner instead of having wires

going all over the place. If I were to do it again I would have purchased much smaller motors since the ones that I am using are way too fast for my project. I also would have used more sonars sensors and possibly some fuzzy logic to ensure that my object avoidance would work at all angles and more intelligently.

10. Appendices

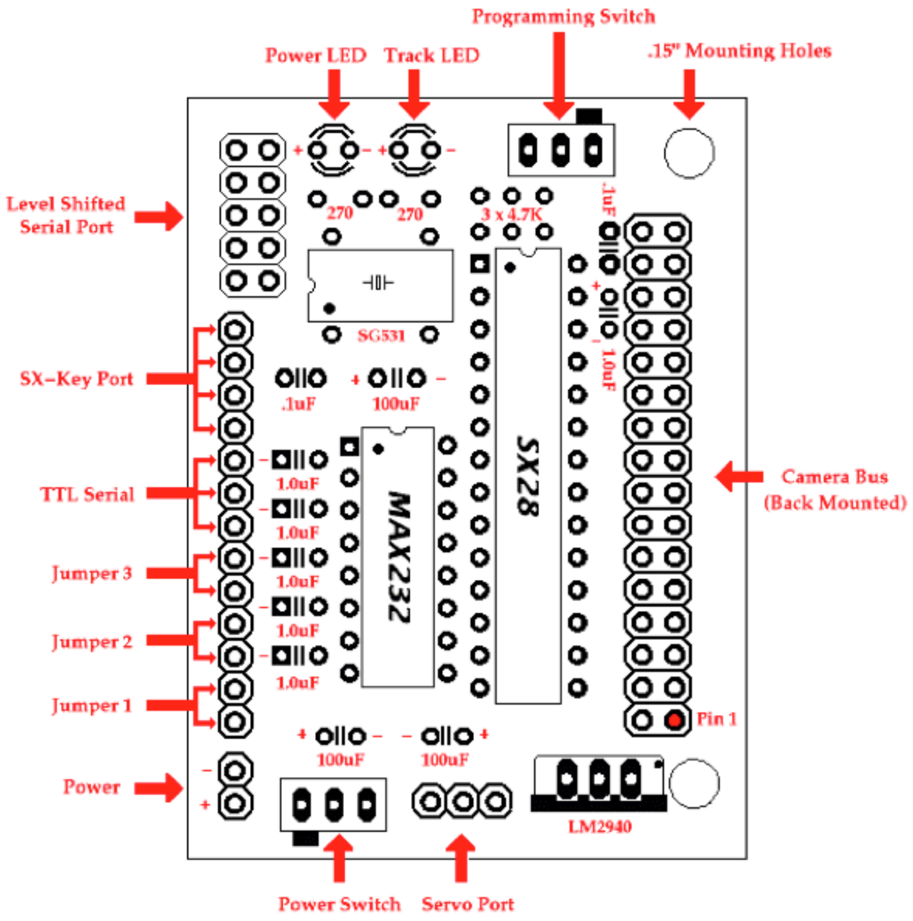
Sonar Circuit


Daisy Chaining with Constantly Looping
GND
+5
TX
RX
AN
PW
BW
MAXBOTIX
MAXSonar-EZ1
Wire BW to Logic High
Wire to an AD input
1K
Repeat to add as many sensors as desired

Sonar Code

```
void sonarSetup(){
  delay(250);
  pinMode(signalPin, OUTPUT);
  digitalWrite(signalPin, HIGH);
  delay(1);
  pinMode(signalPin, INPUT);
}
float sonarReadingToInches(float analogValue){
  return analogValue / 1023 * 5000 / 9.55078125;
}
```

Camera Diagram



Camera Code

```
void camSetup(){

  cmuSet("RS");

  cmuSet("PM 1");

}

//sends a command, ignoring the result

int cmuSet(char * command)

{

  int byteCount;
  //send command and \r
  Serial1.print(command);
  Serial1.print("\r");
  delay(10);


  //rcv "ack\r:"
  while(Serial1.available() == 0);
  while(Serial1.read() != ':');
  Serial1.flush();
}
```