Pink Pants Stalker

EEL4665C

Steven Settle

Instructors:

A. Antonio Arroyo

Eric M. Schwartz

Teaching Assistants:

Tim Martin

Ryan Stevens

Table of Contents

Abstract

The purpose of the pink pants stalker is to follow a person wearing pink pants and avoid obstacles that are in the robots way, as well as squirt the person wearing pink pants with water.

Executive Summary

This robot was designed to follow someone wearing pink pants. The color pink was chosen because of the rarity of people that where pink pants and could possible mess up the demo of the robot. Later on it was decided to add on a water pump and nozzle that will squirt the person wearing pink pants.

This robot has a single layer platform that was bought at Lowes. The platform is 15 inches in diameter and one inch thick. This board was chosen because of the shape, weight, and durability of the wooden board.

Obstacle avoidance used three different types of sensors: IR range finders, sonar range finders, and bump sensors. The IR sensor is used to detect how close a person is in front of a robot. The IR sensor has a narrow field of view which makes it ideal for "seeing" if someone is in front of the robot. The sonar sensors are used to detect if there are obstacles in front/sides of the robot. These have a wider viewing angle so they have a greater chance of detecting an object in front of the robot. The bump sensors were useful for when the wheels kept getting cause on legs of chairs. This was also fixed with better placement of the sonar sensors.

The special sensor is a camera on a HTC Thunderbolt smartphone. This phone has an app that sends a video feed to the computer. On the computer there is a program that sets up this video feed as a webcam so it can be used with OpenCV.

Introduction

This project is to accomplish the goal of being able to follow someone with a robot, more specifically someone wearing pink pants. This will be accomplished by using image processing on a video feed to obtain information about the colors in the video. There will a IR range finder on the front of the robot which will detect how close the robot is the the person wearing pink pants. If the person is too close the the robot will squirt them with water. There will also be sonar sensors on the side to find obstacles. Lastly, there will be bump sensors on the robot to allow the robot to know when it has run into an obstacle if the sonar fails.

Integrated System

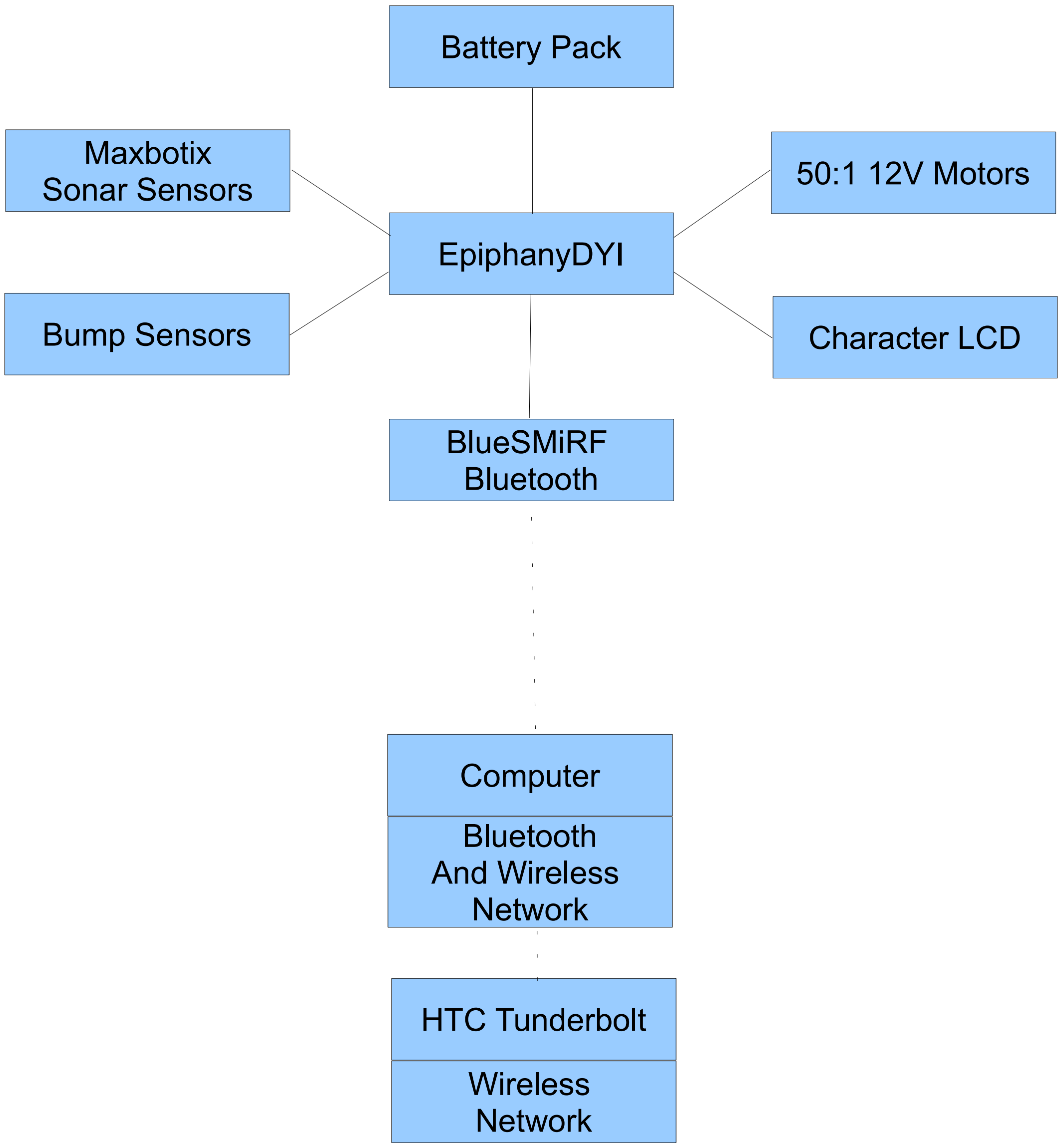

Figure 1: System Design

The main controller of this system will be the EpiphanyDYI board. This boards main processing chip is an Atmel ATXMEGA64A1. This processor will communicate with the computer through the bluetooth modem. The Thunderbolt will send a video feed to the computer wirelessly as well. The processor will also receive information from the sensors and send information to the motor drivers and the LCD.

Mobile Platform

My design will be based off of a round platform with two motor on opposite sides and the microprocessor board in the center. The design that was built is shown below as Figure 2.

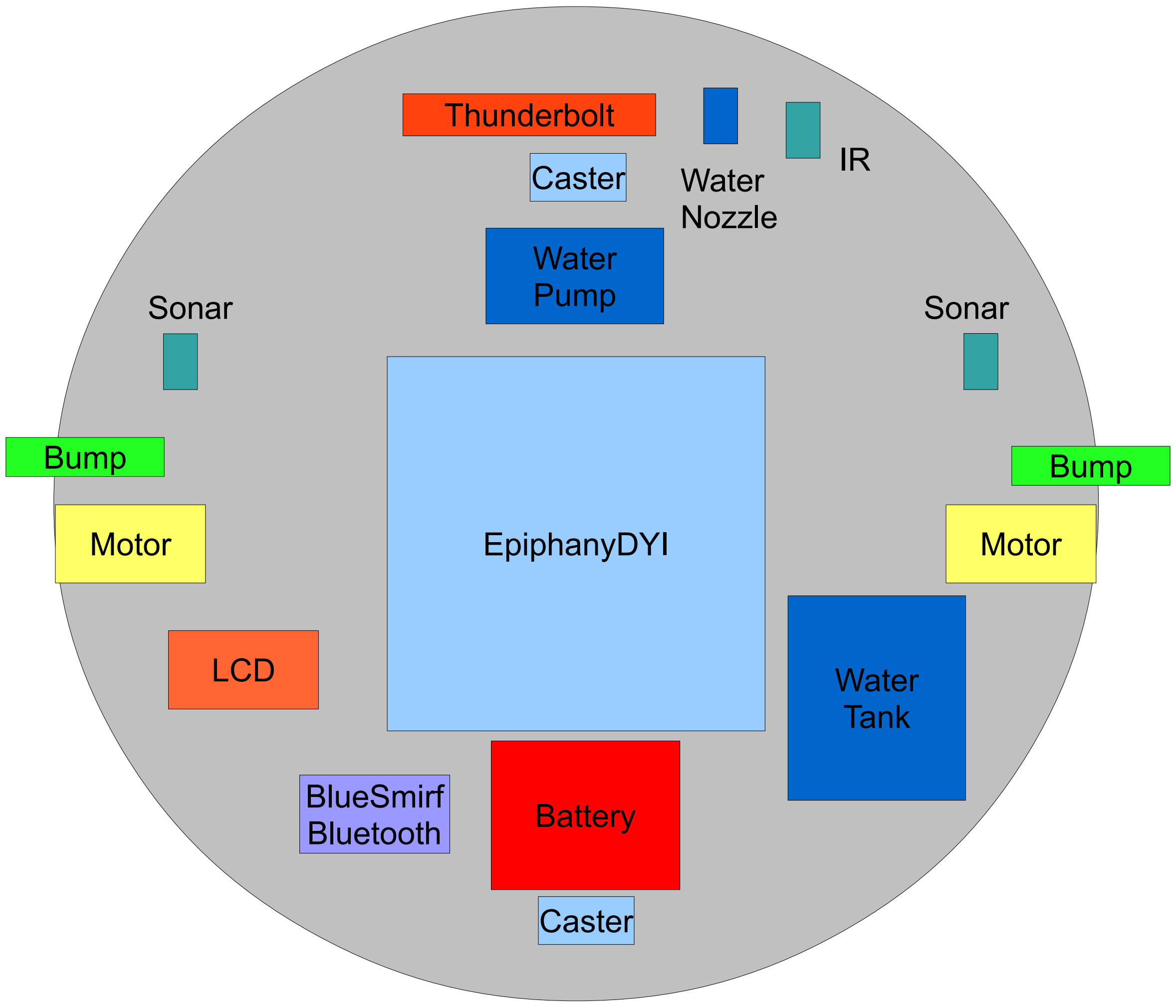

Figure 2: Mechanical Layout

This design was chosen because I didn't want to have to many motors, but I still wanted it to remain mobile. Having two motors on opposite sides will allow the robot to turn while remaining in one spot.

Actuation

The robot will be driven by two motors attached outer rim of the robot. At 12V the motors have a torque of 170 oz·in, a max rotation of 200 RPM, 300 mA free-run , and a stall current of 5 A.

Sensors

There are three types of sensors that will be used in this project: a camera, two sonar sensors, and bump sensors. The camera is included in the smartphone. The smartphone will send a video feed to the computer. Then a program called DroidCam will process this video and allow the computer to see it as a webcam. Now the image processing can be done on the video by grabbing single frames and processing them.

Figure 3: HTC Thunderbolt

Bump sensors will be used to report collisions with objects not detected by the sonar sensors. These sensors are placed in front of the wheels and are used as a last resort if the range finder missed objects.

The sonar and IR range finding sensors will give information to microprocessor so the processor can figure out where obstacles are and how far away they are. The IR has a narrower viewing angle than the sonar so it is used in the front of the robot to help avoid large objects. The sonar sensor are used as backup for the objects that the IR range finder missed

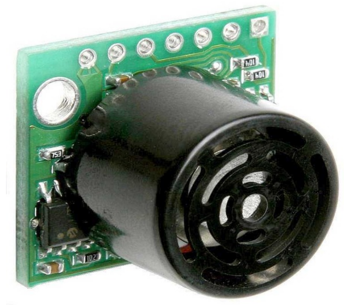
Figure 4: Sonar Range Finder

Behaviors

The behavior of my robot can be see in the flow chart no the following page. It is a simple behavior where the robot will obtain information about the presents of pink colors in front of it through bluetooth with the computer. Then the robot will decide to move forward depending if there is pinks colors in front of it. With the sonar range finders the robot will also “see” if there are objects in front of it and make the decision to move forward still or rotate to avoid the object. If the person wearing pink pants gets too close to the IR sensor the robot will then quirt them with water.

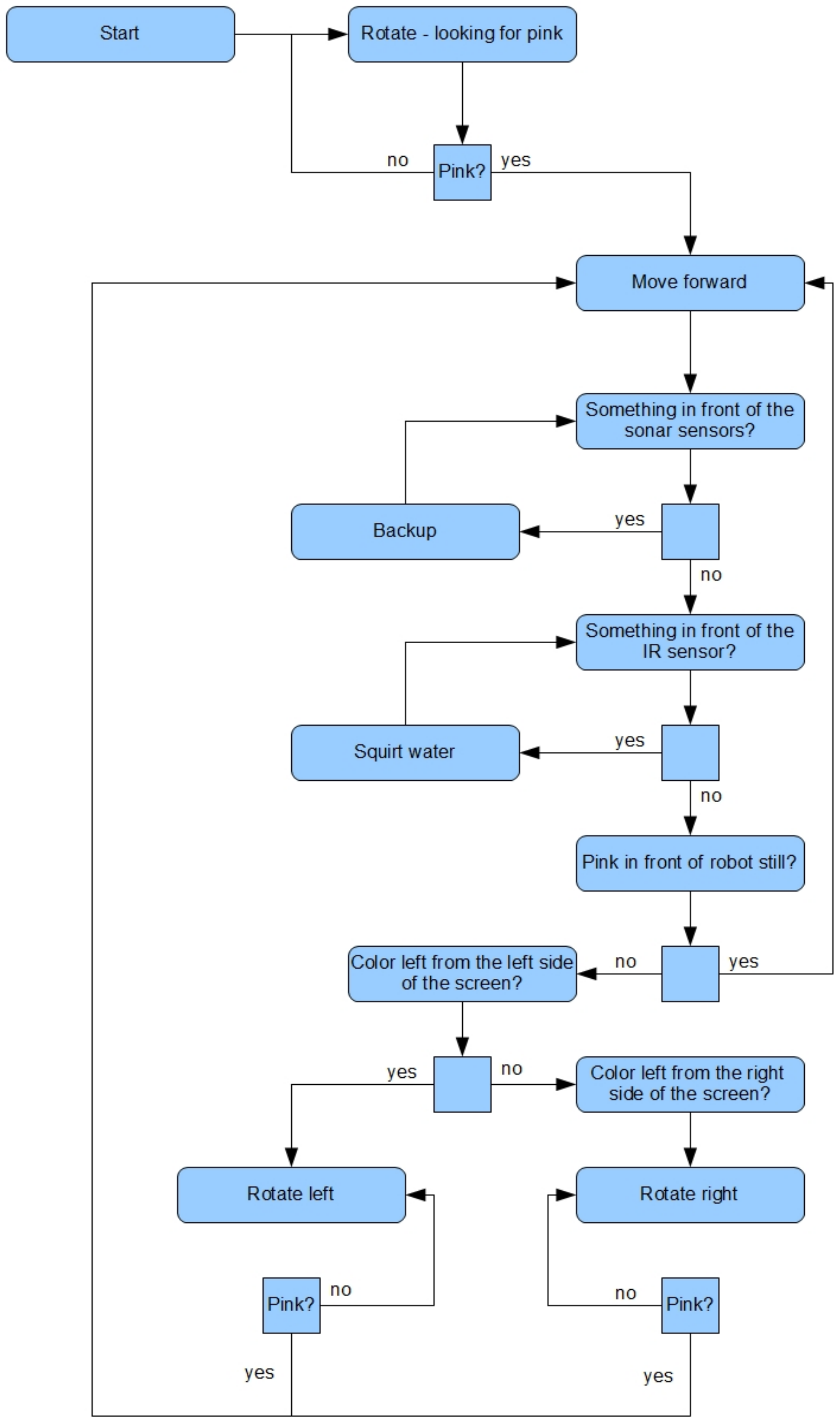
Start
Rotate - looking for pink
Pink?
no
yes
Move forward
Something in front of the sonar sensors?
yes
Backup
no
Something in front of the IR sensor?
yes
Squirt water
no
Pink in front of robot still?
no
yes
Color left from the left side of the screen?
yes
no
Color left from the right side of the screen?
Rotate left
Rotate right
Pink?
no
yes
Pink?
no
yes

Figure 5: Basic Flowchart for the Pink Pants Stalker

Experimental Layout and Results

Several different experiments were performed when building this robot. Experiments were run with the IR sensors and the sonar sensors. To do this experiment the A/D register that was connected to the sensors was output on the LCD. A piece of paper was used to change the distances. The results were the ADCs vlaues increased when the piece of paper got to closer to the sensor. It was decided to use a register value of 335 for the sonar sensors and 600 for the IR sensors.

To test the robots behavior it was let loose in my apartment or the lab and I observed it for problems. The main issues I had involved the dead zones due to the positions of my sensors. I have tried several different positions but they all have this problem and the only way to fix it would be to add more sensors.

Conclusion

This robot has the ability to follow someone wearing pink pants. The designs that were chosen worked fairly well and there was little that could be changed. The IR sensors field of view was very narrow and if this was increased the squirting action would work better. Also, having two smaller platforms placed on top of each other would have made this design more compact, which I believe would be a better alternative.

Documentation

EpiphanyDYI:

https://sites.google.com/site/epiphanydiy/home

Maxbotix LV-MaxSonar-EZ1 Sonar Range Finder:

http://www.pololu.com/catalog/product/726

50:1 12V Motor:

http://www.pololu.com/catalog/product/1104

Basic 16x2 Character LCD - White on Black :

http://www.sparkfun.com/products/9052

BlueSMiRF Silver Bluetooth Modem:

http://www.sparkfun.com/products/10269

OpenCV documentation about blob and color tracking:

http://www.technical-recipes.com/2011/tracking-coloured-objects-in-video-using-opencv/

Appendix

Robots Code

```
#include <asf.h>

#include <avr/io.h>
#include <ctype.h>
#include <stdint.h>
#include <stdio.h>
#include <util/delay.h>
#include "motor.h"
#include "lcd.h"
#include "uart.h"
#include "RTC.h"
#include "ATtinyServo.h"
#include "ADC.h"
#include "sonar.h"

#define DbLedOn()          (PORTR.OUTCLR = 0x02)            //Turns the debug led on.  The led
is connected with inverted logic
#define DbLedOff()         (PORTR.OUTSET = 0x02)            //Turns the debug led off.  The
led is connected with inverted logic
#define DbLedToggle()      (PORTR.OUTTGL = 0x02)            //Toggles the debug led off.  The
led is connected with inverted logic

void turn90(void);
void backup(void);
void turn90other(void);

int main (void)
{
       board_init(); /*This function originates in the file init.c, and is used to initialize
the Epiphany DIY
                                    motorInit() is declared within because by default you the
user should define what your
                                    motor setup is to prevent hurting the Epiphany.  You can do
this by
                                  */
       RTC_DelayInit();//initializes the Real time clock this seems to actually take an
appreciable amount of time
       DbLedOn();    //I like to do this by default to show the board is no longer suspended in
the bootloader.
       ATtinyServoInit();
       uartInit(&USARTE1,115200);//as can be seen in the schematic.  This uart is connected to
the Xbee port.  This function initializes this uart
       uartInit(&USARTD1,9600);
       ADCsInits();//this function initializes the ADCs inside the Xmega
       sei();
       LCDInit();
       PORTB.PIN0CTRL = PORT_OPC_PULLUP_gc;
       PORTB.PIN1CTRL = PORT_OPC_PULLUP_gc;
       unsigned int i,j,distance,turn,k,hold=0;


       _delay_ms(1000);
```

```
        while (1)
        {



//demo ADC code

        //      fprintf(&lcd_str,"ADC %d = %d                   ",0,analogRead_ADCA(7));
        //      fprintf(&lcd_str,"\r");
        //      _delay_ms(100);
//      }
//}


                //fprintf(&lcd_str,"Reg: %x,                          ",USARTE1.DATA);
                        //fprintf(&lcd_str,"\r");
                if(USARTE1.DATA == 0x32)
                {
                        hold=0x32;
                }
                if(USARTE1.DATA == 0x33)
                {
                        hold=0x33;
                }
                        //      fprintf(&lcd_str,"%x                      ",hold);
                if(USARTE1.DATA == 0x31)
                {


                        _delay_ms(10);//delay // milliseconds

                        if (analogRead_ADCA(7) <325 )
                        {
                                fprintf(&lcd_str,"Back up1            ");
                                fprintf(&lcd_str,"\r");

                                for(i=2;i<5;i=i+2){
                                for(j=0;j<640;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                                _delay_ms(10);//delay 10 milliseconds
                                }

                                for(j=0;j<640;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                                _delay_ms(10);//delay 10 milliseconds

                                for (k=0; k<15;k++)
                                {
                                        for(i=2;i<5;i=i+2){
                                for(j=0;j<640;j++) setMotorDuty(i,j,MOTOR_DIR_BACKWARD_gc);
                                _delay_ms(10);//delay 10 milliseconds
                                }

                                }
                                for (k=0; k<8;k++)
                                {
                                turn90other();
                                _delay_ms(2);//delay 10 milliseconds
                                }
                        }
        else if (analogRead_ADCA(6) <325 )
        {
                fprintf(&lcd_str,"Back up2              ");
                fprintf(&lcd_str,"\r");

                for(i=2;i<5;i=i+2){
                for(j=0;j<640;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                _delay_ms(2);//delay 10 milliseconds
                }

                for(j=0;j<640;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                _delay_ms(10);//delay 10 milliseconds

                for (k=0; k<15;k++)
                {
                        for(i=2;i<5;i=i+2){
                for(j=0;j<640;j++) setMotorDuty(i,j,MOTOR_DIR_BACKWARD_gc);
                _delay_ms(2);//delay 10 milliseconds
                }

                }
                for (k=0; k<8;k++)
                {
                turn90();
                _delay_ms(2);//delay 10 milliseconds
                }
        }

        else if (analogRead_ADCA(0) > 600)
        {
                fprintf(&lcd_str,"Turn                          ");
                fprintf(&lcd_str,"\r");

                for(j=0;j<680;j++) setMotorDuty(1,j,MOTOR_DIR_FORWARD_gc);
                _delay_ms(10);//delay 10 milliseconds


                for(i=2;i<5;i=i+2){
                        for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                        _delay_ms(10);//delay 10 milliseconds
                }
                for(i=40;i<120;i++){
            setServoAngle(5,i);
            _delay_ms(10);//delay 50 milliseconds
    }
    for(i=120;i>40;i--){
            setServoAngle(5,i);
            _delay_ms(10);//delay 50 milliseconds
    }


                /*while (turn <= 10){

                        for(i=2;i<5;i=i+2){
                        for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                        turn++;
                }
                }
                while (turn > 10 && turn <=50){
                //backup();
                turn++;
                for(j=0;j<680;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                }
                 if( turn >=50)
                 {
                        turn=0;
                        for(i=2;i<5;i=i+2){
                        for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                }
                 }
                _delay_ms(2);//delay 10 milliseconds
        */
        }
        else if (analogRead_ADCA(0) <= 600)
        {
                fprintf(&lcd_str,"Go                           ");
                fprintf(&lcd_str,"\r");

                for(j=0;j<680;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                _delay_ms(10);//delay 10 milliseconds

                for(j=0;j<640;j++) setMotorDuty(2,j,MOTOR_DIR_FORWARD_gc);
                _delay_ms(10);//delay 10 milliseconds

                for(j=0;j<640;j++) setMotorDuty(4,j,MOTOR_DIR_FORWARD_gc);
                _delay_ms(2);//delay 10 milliseconds
                turn=0;

        }

        else if ((PORTB.IN & 0x01) ==0x00)
        {
                fprintf(&lcd_str,"Back up3            ");
                fprintf(&lcd_str,"\r");

                for(i=2;i<5;i=i+2){
                for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                _delay_ms(10);//delay 10 milliseconds
                }

                for(j=0;j<680;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                _delay_ms(10);//delay 10 milliseconds

                for (k=0; k<15;k++)
                {
                        for(i=2;i<5;i=i+2){
                for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_BACKWARD_gc);
                _delay_ms(10);//delay 10 milliseconds
                }
                }

                for (k=0; k<8;k++)
                {
                turn90other();
                _delay_ms(2);//delay 10 milliseconds
                }
```

```
                    }
                    else if ((PORTB.IN & 0x02) == 0x00)
                    {
                           fprintf(&lcd_str,"Back up4              ");
                           fprintf(&lcd_str,"\r");
                           for(i=2;i<5;i=i+2){
                           for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_NEUTRAL_gc);
                           _delay_ms(2);//delay 10 milliseconds
                           }
                           for(j=0;j<680;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                           _delay_ms(10);//delay 10 milliseconds
                           for (k=0; k<15;k++)
                           {
                                  for(i=2;i<5;i=i+2){
                           for(j=0;j<680;j++) setMotorDuty(i,j,MOTOR_DIR_BACKWARD_gc);
                           _delay_ms(10);//delay 10 milliseconds
                           }
                           }
                           for (k=0; k<8;k++)
                           {
                           turn90();
                           _delay_ms(2);//delay 10 milliseconds
                           }

                    }
                    }

                           if(USARTE1.DATA == 0x30)
                           {
                                  for(j=0;j<680;j++) setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                               _delay_ms(10);//delay 10 milliseconds
                                         if (hold==0x33)
                                         {
                                                //fprintf(&lcd_str,"Turn left
");
                                                for(j=0;j<680;j++)
setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                                                _delay_ms(10);
                                                for(i=2;i<5;i=i+2){

                                                for(j=0;j<640;j++)
setMotorDuty(4,j,MOTOR_DIR_FORWARD_gc);
                                                _delay_ms(10);
                                                for(j=0;j<640;j++)
setMotorDuty(2,j,MOTOR_DIR_BACKWARD_gc);
                                                _delay_ms(10);
                                         //turn90();
                                         }
                                         }
                                         else if (hold==0x32)
                                         {
                                                for(j=0;j<680;j++)
setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                                                _delay_ms(10);
                                         for(i=2;i<5;i=i+2){
                                         //for(j=0;j<640;j++) {
                                                //fprintf(&lcd_str,"Turn right
");
                                                        for(j=0;j<640;j++)
setMotorDuty(2,j,MOTOR_DIR_FORWARD_gc);
                                                        _delay_ms(10);
                                                        for(j=0;j<640;j++)
setMotorDuty(4,j,MOTOR_DIR_BACKWARD_gc);
                                                        _delay_ms(10);
                                                //}//turn90();
                                                }
                                                }
                                                else
                                                {
                                                        fprintf(&lcd_str,"NO PINK
");
                                                        for(j=0;j<680;j++)
setMotorDuty(1,j,MOTOR_DIR_NEUTRAL_gc);
                                                        _delay_ms(10);//delay 10 milliseconds
                                                        for(i=2;i<5;i=i+2){
                                                        //for(j=0;j<640;j++) {
                                                                for(j=0;j<640;j++)
setMotorDuty(4,j,MOTOR_DIR_FORWARD_gc);
                                                                _delay_ms(10);
                                                                for(j=0;j<640;j++)
setMotorDuty(2,j,MOTOR_DIR_BACKWARD_gc);
                                                                _delay_ms(10);
                                                        //}//turn90();
                                                        }
                                                }
                                        }

                        }
                }



void turn90 (void) // no longer turns the robot, this just backs up the robot
{
        int i,j=0;
        for(i=2;i<5;i=i+2){
                        for(j=0;j<680;j++)
                        {
                                //setMotorDuty(2,j,MOTOR_DIR_FORWARD_gc);
                                setMotorDuty(2,j,MOTOR_DIR_BACKWARD_gc);

                                setMotorDuty(4,j,MOTOR_DIR_BACKWARD_gc);
                        }
                        }
                        _delay_ms(10);//delay 10 milliseconds
}

void turn90other (void)// no longer turns the robot, this just backs up the robot
{
        int i,j=0;
        for(i=2;i<5;i=i+2){
                        for(j=0;j<680;j++)
                        {
                                setMotorDuty(2,j,MOTOR_DIR_BACKWARD_gc);
                                setMotorDuty(4,j,MOTOR_DIR_BACKWARD_gc);
                                //setMotorDuty(4,j,MOTOR_DIR_FORWARD_gc);
                        }
                        }
                        _delay_ms(10);//delay 10 milliseconds
}

void backup (void)
{
        int i,j=0;
        for(i=2;i<5;i=i+2){
                        for(j=0;j<640;j++)
                        {
                                setMotorDuty(2,j,MOTOR_DIR_BACKWARD_gc);
                                setMotorDuty(4,j,MOTOR_DIR_BACKWARD_gc);
                        }
                        }
                        _delay_ms(10);//delay 10 milliseconds
}
```

Bluetooth and Open CV code

```
#include "stdafx.h"

#include "cv.h"
#include "highgui.h"
#include "stdafx.h"
#include <fstream>
#include <iostream>
#include <sstream>
#include <string>
#include <vector>
#include <stdio.h>
#include <stdlib.h>
using namespace System;
using namespace std;
using namespace System::IO::Ports;

#include "opencv/highgui.h"
#include "opencv/cv.h"
#include "BlobResult.h"
#include <iostream>
#include <stdlib.h>
#include <stdio.h>

#include "stdafx.h"

#include <opencv2\opencv.hpp>

IplImage* GetThresholdedImage(IplImage* img)
{
       IplImage* imgHSV = cvCreateImage(cvGetSize(img), 8, 3);
       cvCvtColor(img, imgHSV, CV_BGR2HSV);

       IplImage* imgThreshed = cvCreateImage(cvGetSize(img), 8, 1);

       cvInRangeS(imgHSV, cvScalar(150, 72, 5), cvScalar(175, 255, 255), imgThreshed);

       cvReleaseImage(&imgHSV);

       return imgThreshed;
}

int main()
{
       CBlobResult blobs;
    CBlob *currentBlob;
       CBlob biggestBlob;
    CvPoint pt1, pt2;
    CvRect cvRect;
       double center;

       string porttemp;
        Console::Write("Enter your port (COMx): ");
        getline(cin,porttemp);
        cout << "You entered: " << porttemp << endl << endl;
        String^ port=gcnew String(porttemp.c_str());
        Console::Write("\n\n");
        SerialPort^ serialPort = gcnew SerialPort(port, 115200, Parity::None, 8,
StopBits::One);
     try
        {
               serialPort->Open();
        }
        catch(...)
        {
               Console::WriteLine("An error has occurred. Did you use the right com port?");
               system("pause");
               return 0;
        }
        Console::Write("Connected!\n\n");

CvCapture* capture = 0;
    capture = cvCaptureFromCAM(1);

if(!capture)
    {
        printf("Could not initialize capturing...\n");
        return -1;
    }

    cvNamedWindow("video");
cvNamedWindow("thresh");

IplImage* imgScribble = NULL;

while(1)
    {

       IplImage* frame = 0;
       frame = cvQueryFrame(capture);

                     if(!frame)
                            break;

       if(imgScribble == NULL)
       {
       imgScribble = cvCreateImage(cvGetSize(frame), 8, 3);
       }

       IplImage* imgThresh = GetThresholdedImage(frame);

       blobs = CBlobResult( imgThresh, NULL, 0 );
       blobs.Filter( blobs,B_EXCLUDE,CBlobGetArea(),B_LESS,300 );


       int num_blobs =blobs.GetNumBlobs();

                     for ( int i = 0; i < num_blobs; i++ )           {
                            currentBlob = blobs.GetBlob( i );
                            cvRect = currentBlob->GetBoundingBox();


                            pt1.x = cvRect.x;
                            pt1.y = cvRect.y;
```

```
                        pt2.x = cvRect.x + cvRect.width;
                        pt2.y = cvRect.y + cvRect.height;

                        cvRectangle( frame,pt1, pt2, cvScalar(0, 255, 255, 0),1,8,0 );
                }
                if (num_blobs==1)
                {
                        printf("%u %u  ", pt1.y,pt2.y  );
                        if ( pt1.y<20 && pt2.y<100)
                        {
                                serialPort->Write("3"); //turn right
                                printf("Sending out 3 \n");
                        }
                        else if ( pt1.y>80 && pt2.y>220)
                        {
                                serialPort->Write("2"); //turn left
                                printf("Sending out 2 \n");
                        }
                        else
                        {
                                serialPort->Write("1");
                                printf("Sending out 1 \n");
                        }

                }
                else if (num_blobs>1)
                {
                        printf("Sending out 1 \n");
                        serialPort->Write("1");
                }
                else
                        {
                                printf("Sending out 0 \n");
                                serialPort->Write("0");
                }

CvMoments *moments = (CvMoments*)malloc(sizeof(CvMoments));
cvMoments(imgThresh, moments, 1);

double moment10 = cvGetSpatialMoment(moments, 1, 0);
double moment01 = cvGetSpatialMoment(moments, 0, 1);
double area = cvGetCentralMoment(moments, 0, 0);

static int posX = 0;
static int posY = 0;

int lastX = posX;
int lastY = posY;

posX = moment10/area;
posY = moment01/area;

printf("# of blobs %u\n", num_blobs);

cvShowImage("thresh", imgThresh);
cvShowImage("video", frame);

int c = cvWaitKey(10);
```

```
        if(c!=-1)
        {
                                break;
        }

        cvReleaseImage(&imgThresh);

        delete moments;
                }

        serialPort->Close();
        cvReleaseCapture(&capture);
                return 0;
}
```