ABSTRACT

Caddy, the golf ball collecting robot, is made up of a TALRIK platform, EVBU with 32k of memory added and a servo mechanism that will open a door to capture the golf balls. The sensors used on this robot are a bump sensor, break-beam sensor and 5 IR sensors. The behaviors are object avoidance, ball detection, color differentiation and ball capture. The IR sensors are involved in all four behaviors allowing for a simple set of sensors, but a more complicated program. The robot is propelled by two gear head motors and wheels. The robot was controlled by a C program complied by IC.

EXECUTIVE SUMMARY

The robot Caddy is a golf ball collecting robot. It is a simple robot that satisfies the requirements of the Intelligent Machine Design Lab. These requirements include four sensor including a bump sensor, and obstacle avoidance sensor as well as at least four integrated behaviors. The complete system designed and built allows the robot to wander around in its environment and interact with the objects it encounters.

For this course, the four sensors were an obstacle avoidance sensor, color differentiation sensor, break-beam sensor and a bump sensor. The four behaviors are obstacle avoidance, ball detection, color differentiation and ball collection. There are 2 IR sensors mounted on the platform and micro-switches mounted on the board for the bump sensor. The three IR sensors mounted on the door of the robot are protected from the objects in the room by putting them behind the door and they are protected from the golf balls in the collection bin by a wood beam.

INTRODUCTION

The objective of this project was to design a robot that would be able to collect golf balls as well as differentiate between colors. This report starts with a general overview of the robots functions. It then proceeds to discuss the physical systems of the robot including the servo that opens the robots “mouth”. From there, the sensors used on the robot are discussed and explained. Finally, the behaviors that the robot exhibits and the experiments performed are discussed.

INTEGRATED SYSTEM

The robot consists of two basic subsystems. These are the base on which the robot is mounted and the door on the front of the robot which traps the balls underneath the robot. The platform consists of the circuit board, wheels and two IR cans. The door mechanism consists of a wood door, servo, 3 IR sensor pairs and cage underneath the robot.

The door part of the robot is the source of many of the robots basic behaviors. The three IR cans mounted on the door allow for ball detection, obstacle avoidance and color determination. Capturing the ball is another behavior that the door mechanism provides. The cage under the robot hold the balls there and the door prevents them from escaping. The platform subsystem holds the brains of the robot as well as the propulsion of the robot.

MOBLIE PLATFORM

The platform chosen for this robot was the TALRIK board that was provided through Dr. Doty. The TALRIK platform had a castor mounted to the back of it to provide balance as the robot traversed the room. The battery pack was attached to the underside of the board using a double sided Velcro strip. This would provide weight on the back of the TALRIK board to offset the weight of the door on the front. Finally the EVBU board was mounted lengthwise on the EVBU board between the two wheel bays using 4 4/40 1.5 inch screws. Two 3" wheels were mounted to the robot giving it about 2" of clearance off the ground. This board was then modified by adding a wood cage underneath it here the golf balls would be trapped when the robot encountered one. The cage was made out ¼" plywood and fits between the two wheel mounts located under the TALRIK board. Finally, the a MS410 servo was mounted to the front of the robot. This servo controls the door of the robot. Figure 1 shows the top and bottom of the board.

ACTUATION

Wheel Motors

Two motors were chosen to drive the robot for two reasons. The first was that the board that was chosen, the TALRIK board, had two holes already cut for the wheels. Even if the board was not used, two wheels are easier to manipulate and precision steering was not need for this robot. The robot simply had to maneuver around objects and when a golf ball was encountered, backup to determine the color. To balance the robot, a castor was mounted to the back of the TALRIK board. The motors

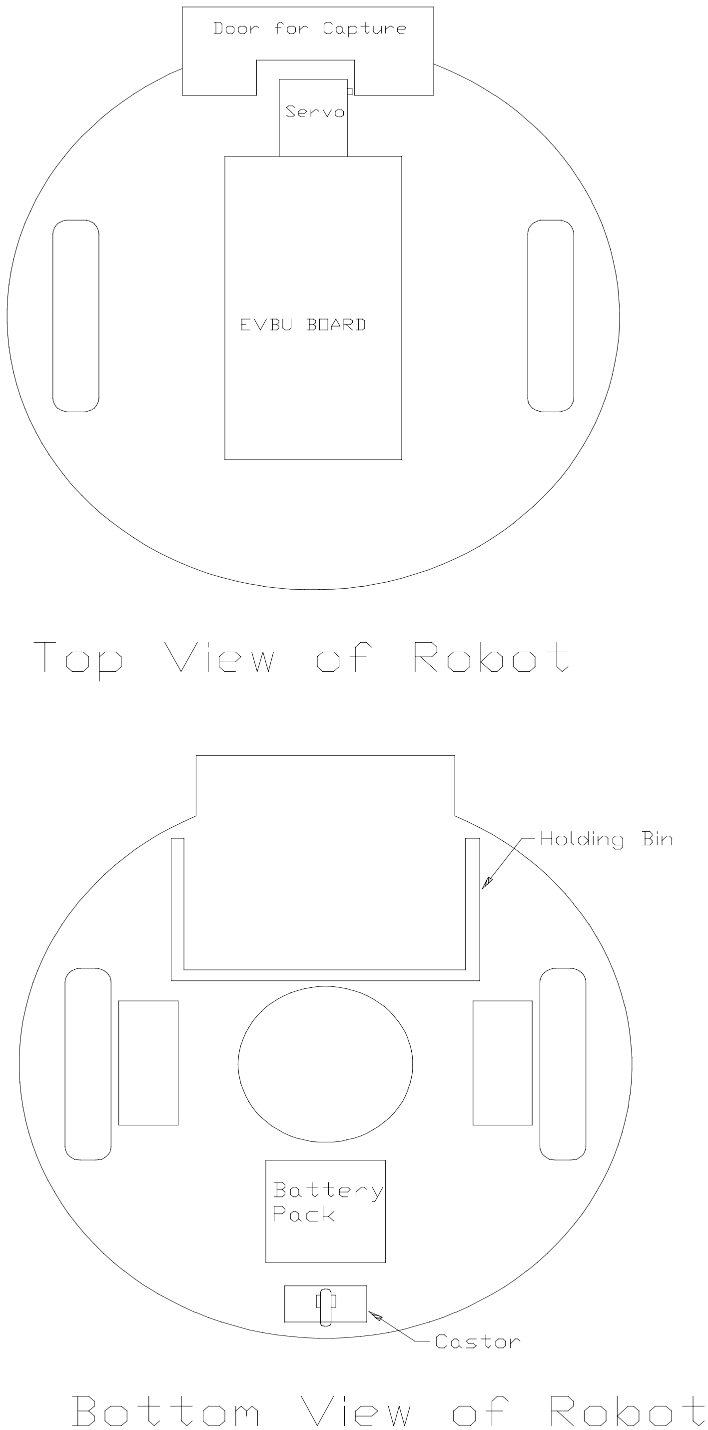

Figure 1 - Top and Bottom View of Robot Platform

used are hacked MS410 Servos. The servos were hacked by removing the circuit board in

them and cutting the stop on one of the gears. They were then mounted to the underside of the TALRIK board using the servo mounts that were supplied by Dr. Doty.

One of the problems that came about was construction of the wheel mechanism. To attach the wheels to the motors, the mounting mechanism had to be attached to the wheel using screws. Since a starter hole was needed for the screws, a Swiss-Army knife was used to produce them. Unfortunately, this is not a very accurate way of doing this, thus the wheels wobble on the robot. On other note, even though the wheels are supposed to be running at the same speed, I noticed that the left motor goes faster than the right motor. For the robot to go straight the left motor had to be run slightly slower than the right motor.

Door Servo

The servo used to lift the door was a simple unhacked MS410 servo. This servo was attached to the front of the robot using the same servo mounts that were used to attach the motors. The door was attached to the servo using one of the arms that were provided with the servo. The get the necessary power for the servo, six volts were tapped off the battery pack and sent to the servo. The servo was connected to the OC 1 of the EVBU board. The range that was used for the servo was 90°, the door is closed and the IR cans are parallel to the ground, to 155°, the door is open and the IR cans are perpendicular to the ground. These values were found through trial and error.

SENSOR SUITE

Overview

The sensors on the robot consist of five IR sensor cans, a bump sensor and a break beam sensor. The golf ball detecting sensor is made up of three of the IR sensors and the break beam sensor. The two outside IR sensors and the bump sensor are used for obstacle avoidance. The three sensors on the door are used for obstacle avoidance, golf ball detection and color determination. The break-beam sensor is used to stop the robot when it encounters a golf ball.

Break Beam

The break beam sensor that is being used is an IR LED and an IR photo-transistor. The photo-transistor used is an Archer SY-54PTR that can be found at Radio Shack. The photo-transistor and the LED are mounted 3 ¾" apart. If the photo-transistor and the IR LED are along the same line, Figure 2, than the reading off the photo transistor is in the range of 0-10. This reading is taken off analog port 5 on the EVBU board. To get this reading, a 10 k Ω resistor is used in series with the power. The analog reading is taken from the photo-transistor before the resistor. For the IR LED, two 330 Ω resistors were used in series. The circuit diagram for this is seen in figure 3.

As stated before, if the photo-transistor and the LED are perfectly aligned than the analog reading when the beam is intact is between zero and ten. Unfortunately, this is not the case. Since they are not perfectly aligned, the analog reading when the beam is intact is between 125-135. Although at first this seems to be a problem, it is not. When the beam is broken, the analog reading of the photo-transistor jumps to between 245 and 255. Since this is the case, than the high base reading is not a problem.

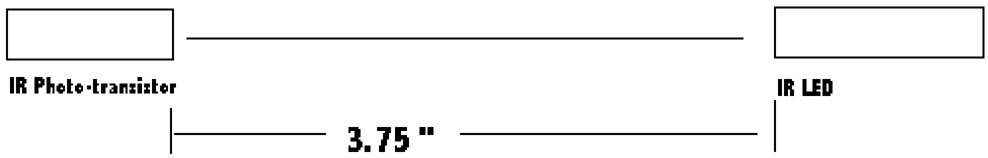

Figure 2 - Simple Drawing of Break Beam Sensor

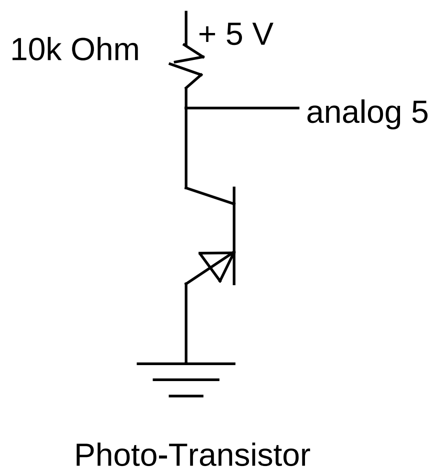

Figure 3 - Circuit Diagram of Break-Beam Sensor

Sharp IR Sensors

On the robot there are five Sharp IR sensors facing forward. There are two sensors mounted on the TALRIK platform. These are mounted at the front and to the side of the wheels to give a wide range of vision for the robot. These sensors are hacked so that an analog reading can be taken from them. On the underside of the robot, two IR

LEDs are attached and are pointing 45° from the center line of the robot. The other three Sharp IR cans and IR LEDs are attached to the door on the front of the robot. Below is a figure of how the sensors are arranged on the door. The LEDS are positioned 0.375 inches below the IR cans. These LEDs were collimated so that more accurate reading could be found and thus make golf ball detection and color determination easier.

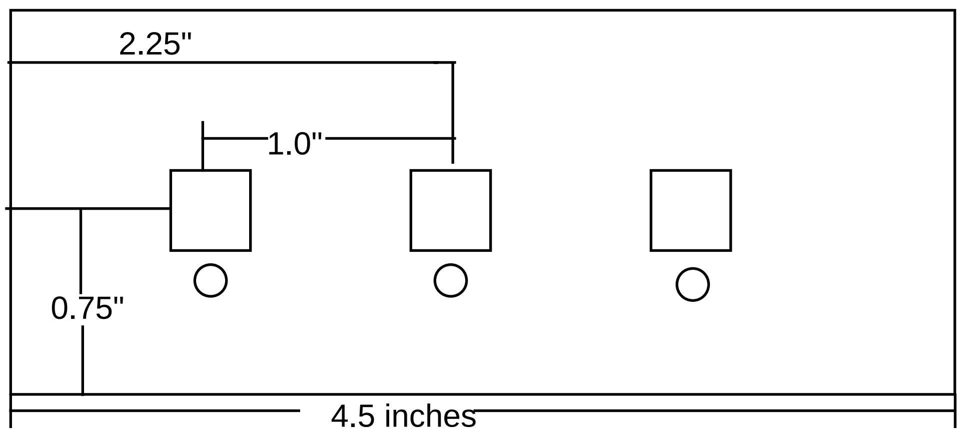

Figure 4 - Positioning of IR cans and IR LEDs

The IR cans and the LEDs were mounted on the back of the door so that when a golf ball is found, it will not hit either the cans or the LEDs. When the golf balls are captured, a wood beam is in between the golf balls and the IR cans and LEDs in the holding bin.

Bump Sensor

This sensor consists of three micro-switches and a piece if plastic that is wrapped around the side of the robot. Since only the back side of the robot has a bump sensor, it was rigidly attached. I decided to rigidly attach it instead of using holders because the force on the holders and the rigid attachment would be approximately the same.

PROGRAMMED BEHAVIORS

Overview

The robot's basic behavior is golf ball finding and color determination. Added to this is obstacle avoidance. An obstacle is defined as anything that is not a golf ball. How the robot determines if an obstacle is a golf ball or not is explained later in this report. Also the robot has a golf ball capture behavior which is triggered if the color of the golf ball is the one it is looking for.

The robot starts out in obstacle avoidance mode. If it then senses an object that might be a golf ball it goes towards that object. When the golf ball is found, the robot than determines the color and if the color is the one it is looking for than it collects the golf ball. Once it has collected three golf balls it deposits them next to the nearest obstacle. Otherwise it turns away from the golf ball and continues on its trek. Below is a block diagram describing the behaviors

Object Avoidance

Object avoidance is an inherent behavior for this robot. Object avoidance is accomplished using four IR sensors. The robot will avoid objects if either of the outside readings are above the threshold readings. If both outer sensor readings are below the threshold reading, than the robot interprets the readings from the front three. Again, these readings are check against a threshold reading and if all three are above and the robot backs up avoiding the object. Otherwise, the robot checks to see if it should turn left or right. Object avoidance is also accomplished through the bump sensor. If the bump

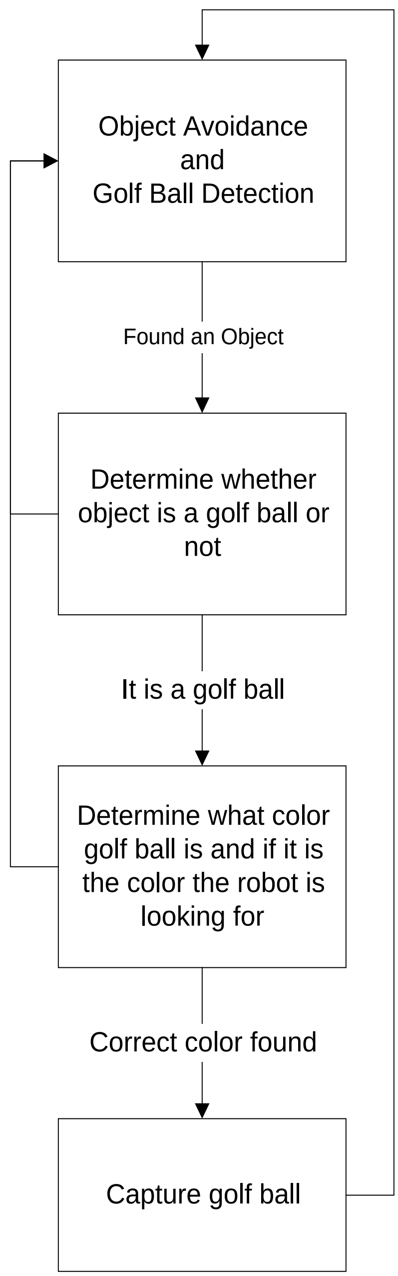

Figure 5 - Block Diagram of Behaviors

sensor is tripped, than the robot has backed into on object. The robot than moves forward and turns either left or right.

Ball Detection

Again, the IR sensors are used in ball detection. When a object is in front of the robot, two of the three IR sensors will record above threshold readings. When this happens, the robot goes toward the object. The robot continuously checks these IR readings and as the object which it thinks is a ball gets closer, it will interpret the IR readings it has to determine if it is a golf ball. For a more in-depth look at how a golf is determined, see the code in Appendix A. If the robot determines that the object is not a golf ball, than it will avoid it and go on.

Color Determination

Once the robot has found a golf ball it goes towards it until the break-beam sensor is tripped and the robot stops. The robot than backs up for ¾ of a second and find the maximum IR reading from the front three IR sensor pairs. This reading is than compared to stored reading for white, orange and black balls. If the correct color golf ball is found than the robot opens the door and “eats” the golf ball.

Golf Ball Depositing

Once the robot has collected three golf balls, it then looks for a place to deposit them. The robot looks for the nearest obstacle and deposits the golf balls at this obstacle. When it gets there, the mouth opens up and the robot backs away from the obstacle. It then turns right or left and close its mouth and looks for more balls.

EXPERIMENTS

To find out if the robot could differentiate between colors, I performed an experiment. An IR LED was collimated and measurements were taken at one inch intervals and there values were compared. As you can see from the graphs below, it is possible to differentiate between orange white golf balls, but not yellow golf balls. The black golf balls have a lower maximum IR reading after backing up for ¾ of a second as compared to the orange golf balls.

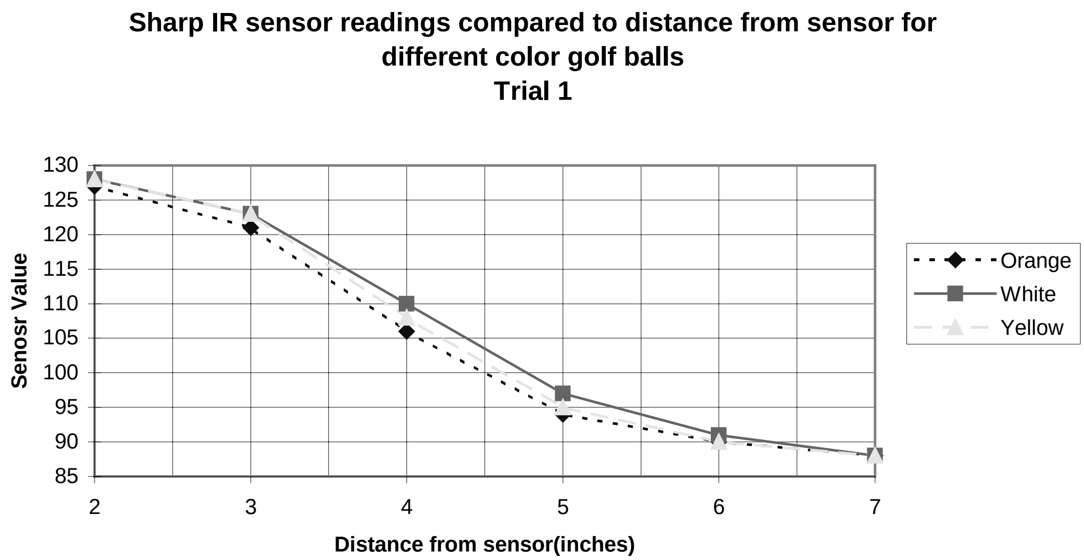

Figure 6 - First Test to See if Robot Can Differentiate colors

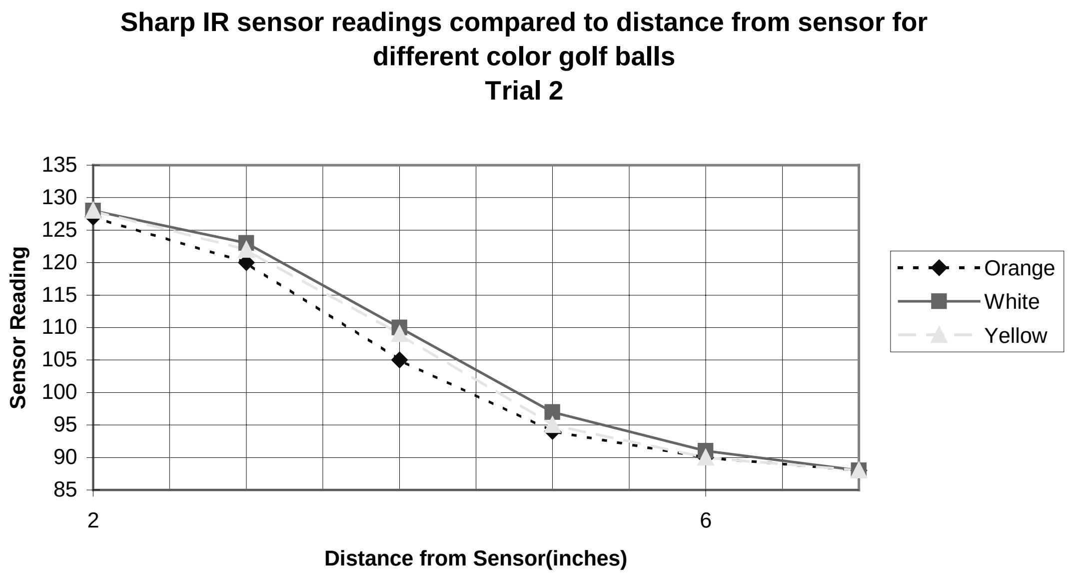

Figure 7 - Second Test For Color Differentiation

The next experiment that was performed was to determine how long it took for the robot to reach the threshold reading on the white, orange and black golf ball. To do this a simple program was written. A golf ball was placed in front of the robot and the robot would go forward, the beam would be tripped and the robot would stop. The robot then backs up until the threshold reading is attained. The robot stores this in a array and takes a total of fifteen readings. The results from this experiment are inconclusive and thus it was decided to back up for a set amount of time and take an IR reading. This method provided better results and was used in the programming of the robot. By inspecting the graphs above, it was determined that the difference in the IR reading was a maximum at 4”. By trial and error, this time was determined to be ¾ of a second.

CONCLUSION

The robot objectives were completed as I had thought. The robot is able to maneuver around a room avoiding objects and is able to determine if an object is an obstacle or a golf ball. When a golf ball is detected, the robot can determine whether the color is orange, white or black and collect the golf ball is necessary. Unfortunately, the depositing behavior was working at the time of the demonstration.

At the present moment, the robot can only differentiate between white, orange and black golf balls. To make this robot more useful, experimentation to determine if the robot can see blue or red golf balls should be performed. Also, installation of another sensor that will allow the robot to see other colors would be useful. Yellow golf balls are seen as white and this is another limitation of the robot. Probably the largest limitation of the robot is that it will not run on thick carpeting or grass. Different wheels need to used to allow the robot to run on grass and thick carpeting.

If this project were to be started over, I would probably design my own platform to fit the wheels better and allow for a better holding bin underneath the robot. Also, the three IR sensors would be arranged differently on the door. One would be placed at a level above the golf ball and the other two would be moved closer to the center approximately 1.5 inches apart. This would allow the robot to better determine if an object is a golf ball or not. This project would also move quicker the second time around because I would not have to learn all the basics of the circuitry, soldering and sensors.

APPENDIX A - Code

```
int side_max_ir = 100;
int fm_ir = 110; /*front max ir*/
int ball_ir = 100;

int right_side, left_side,right_front, left_front, center, beam, bump;

int right_motor = 1;
int left_motor = 0;
float initial_angle = 90.0;
float open_angle = 165.0;

int ir_move;
int id_move;
int ball_move;
int robot_move;
int bump_move;
int deposit_move;

int its_a_ball=0;
int found_ball = 0;
int beam_tripped=0;
int id_check=0;
int bumped = 0;
int deposit = 0;

int yes=2;
int maybe = 1;
int no = 0;

int STOP = 0;
int FORWARD = 1;
int BACKWARD = 2;
int LEFT = 3;
int RIGHT = 4;
int RIGHT_SLOW = 5;
int LEFT_SLOW = 6;

int max_ir;

long return_time;
int balls_captured=0;
int ball_color;
int capture = 0;

int WHITE = 0;
int BLACK = 1;
int ORANGE = 2;
int white_ir = 108;
int orange_ir = 100;

int max;
int ir;
int deposited = 0;


/*********************************************************/
```

```
/****  turning function, left, right back and forward ****/
/********************************************************/

void turn_left(float speed)
{
 motor(right_motor,speed);
 motor(left_motor,-speed);
}

void turn_right(float speed)
{
 motor(right_motor,-speed);
 motor(left_motor,speed);
}

void go_straight()
{
 motor(right_motor,25.0);
 motor(left_motor,25.0);
}

void go_back()
{
 motor(right_motor,-25.0);
 motor(left_motor,-25.0);
}

/***********************************************************/
/*********  move function, takes direction from arbitrator **/
/***********************************************************/

void move(int direction)
{
 if(direction == FORWARD)
   go_straight();
 else if (direction == BACKWARD)
   go_back();
 else if (direction == LEFT)
   turn_left(25.0);
 else if (direction == RIGHT)
   turn_right(25.0);
 else if (direction == RIGHT_SLOW)
   turn_right(10.0);
 else if (direction == LEFT_SLOW)
   turn_left(10.0);
 else if(direction == STOP)
   stop();
}

/**********************************************************
** The following routines up to the obstacle routine are ***
** used to capture the golf ball ***************************
**********************************************************/

void find_max_ir()
{
```

```
  max_ir=analog(2);
  if(analog(3) > max_ir)
    max_ir = analog(3);
  if(analog(4) > max_ir)
    max_ir = analog(4);

}


/*********************************************************
***  this routine identifies the color of the golf ball*****
***  by looking at the max ir reading after backing up******
***  for three quarters of a second***********************/

void identify_color()
{
   long time_s;
   find_max_ir();
   time_s = mseconds() + (long)750;
   while(time_s > mseconds())
    {
      id_move = BACKWARD;
      find_max_ir();
   }
   find_max_ir();
   id_move = STOP;
   wait(500);
   if (max_ir >= white_ir)
     ball_color = WHITE;
   else if(max_ir >= orange_ir && max_ir < white_ir)
     ball_color = ORANGE;
   else
     ball_color = BLACK;
}

/***  this routing checks to see if this is the color the robot wants*/

void check_color()
{
  int ball_check;
  ball_check = balls_captured/3;

  if(deposit == 1)
    capture = 0;
  else if(ball_color == WHITE && ball_check == 0)
   capture = 1;
  else if(ball_color == BLACK && ball_check == 2)
   capture = 1;
  else if(ball_color == ORANGE && ball_check == 1)
   capture = 1;
  else
   capture = 0;
}

/* this routine has the robot "eat" the ball*/
```

```
void eat_ball()
{
 long timer;


 servo_deg(open_angle);
 timer = mseconds() + (long)1000;

 while(mseconds() < timer)
  id_move = FORWARD;

 id_move = STOP;
 servo_deg(initial_angle);
 wait(1000);

}

/* this rountine contorls the color id and ball capture behavior*/

void ball_capture()
{
 while(1){
   if(beam_tripped)
    {
     its_a_ball = 0;
     id_check = 1;
     identify_color();
     check_color();
     if (capture)
      {
        eat_ball();
        ++balls_captured;
        depsoited = 0;
      }
     else
      { id_move = RIGHT;
        wait(500);
      }
     id_check = 0;
    }
  }
}
/***********************************************************
*****  This is the obstacle avoidance routine.  It also ****
***** sends to another rountine if it thinks it encounters**
***** a ball and also trips the its_a_ball *****************
***********************************************************/

void obstacle()
{
 while(1)
 {
  if(left_side >= side_max_ir)
    if(right_front >= ball_ir || center >= ball_ir)
     {
      ir_move = BACKWARD;
```

```
      wait(1000);
      ir_move = RIGHT;
     }
   else
     ir_move = RIGHT;
  else if(right_side >= side_max_ir)
   if(left_front >= ball_ir || center >= ball_ir)
     {
      ir_move = BACKWARD;
      wait(1000);
      ir_move = LEFT;
     }
   else
     ir_move = LEFT;
  else if(center >= ball_ir)
    {
     ir_move = BACKWARD;
     wait(1000);
     ir_move = RIGHT;
    }
  else
   ir_move = FORWARD;
 }

}

/*****************************************************
***** this determines if its a ball or not ***********
*****************************************************/

void is_it_a_ball()
{
  while(1)
  {
   if(left_side > ball_ir || right_side > ball_ir)
     found_ball=0;

   else if(!found_ball)
    {
     if(center >= ball_ir)
       if(right_front >= ball_ir && left_front <= ball_ir)
        found_ball = 1;
       else if(right_front <= ball_ir && left_front >= ball_ir)
        found_ball = 1;
       else
        found_ball = 0;

     }
   else if(found_ball)
    if(center > ball_ir|| right_front> ball_ir || left_front >ball_ir)
     if(center > right_front && center > left_front)
       ball_move = FORWARD;
     else if(right_front > left_front)
       ball_move = RIGHT;
     else
       ball_move = LEFT;
```

```
    else
     found_ball = 0;
  }
}

/****************************************************
** This function reads the bump sensor**************
****************************************************/

void bump_sensor()
{
 while(1)
  {
   if(bump==0)
    {  bumped = 1;
       bump_move = FORWARD;
       wait(500);
       bumped = 0;
    }
  }
}

/****************************************************
***** The function that reads the IR values for******
***** the 5 sensors and the beam sensor *************
*****************************************************/

void read_ir()
{
 while(1)
  {
   /** read the IR can results and the beam sensor **/
   right_side = analog(0);
   left_side = analog(1);
   right_front = analog(2);
   left_front = analog(3);
   center = analog(4);
   beam = analog(5);
   bump = analog(7);

   /** check to see if beam is tripped or not **/
   if(beam > 200 && !id_check)
    beam_tripped = 1;
   else
    beam_tripped = 0;
  }
}

/*****  this is the ball deposit rountine *****/

void deposit_balls()
{


 while(1)
  {
```

```
    if(balls_captured > 0 && balls_captured%3 == 0 && !deposited)
     deposit = 1;
    else
     deposit = 0;

    if(right_front > center &&  right_front>left_front)
     {
      ir = right_front;
      max = RIGHT;
     }
    else if(left_front > center)
     {
      ir = left_front;
      max = LEFT;
     }
    else
     {
      ir = center;
      max=FORWARD;
     }

    if(ir > 125 && deposit)
     {
      deposit_move = STOP;
      servo_deg(open_angle);
      deposit_move = BACKWARD;
      wait(1000);
      servo_deg(initial_angle);
      deposit_move = RIGHT;
      wait(500);
      deposit =0;
      deposited = 1;
     }
    else
     deposit_move = max;
   }
}

/****************************************************
******** initialization of robot ********************
****************************************************/

void initialize()
{
 servo_on();
 servo_deg(initial_angle);
 poke(0x7000,0xff);
}


void wait(int milli_seconds)
{
  long timer_a;
  timer_a = mseconds() +(long)milli_seconds;
  while(timer_a > mseconds() )
    defer();
```

```
}

/*****************************************************
****** THE ARBITRATOR ********************************
*****************************************************/

void arbitrator()
{
  while(1)
   {
    if(beam_tripped)
     robot_move = STOP;
    else if(id_check)
     robot_move = id_move;
    else if(bumped)
     robot_move = bump_move;
    else if(deposit)
     robot_move = deposit_move;
    else if(found_ball)
     robot_move = ball_move;
    else
     robot_move = ir_move;

    move(robot_move);
   }
}

void main()
{
 initialize();
 start_process(bump_sensor());
 start_process(is_it_a_ball());
 start_process(read_ir());
 start_process(obstacle());
 start_process(ball_capture());
 start_process(deposit_balls());
 start_process(arbitrator());
}
```