Final Competition
CPE 470

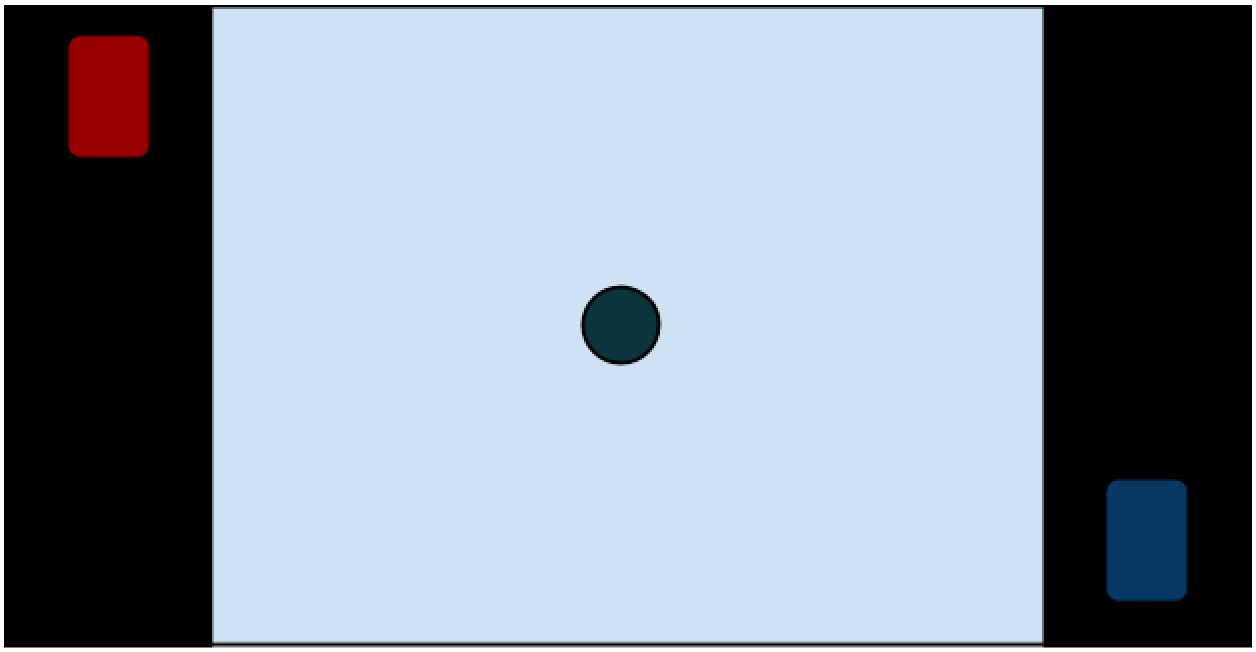
Jessica Smith, Luke Fraser

18 December, 2013

Introduction

This lab was the final competition. It consisted of a competition between two robots which started on either side of the field as shown in Figure 1. The two robots begin on opposite sides of the field and the goal of the competition is to kick the ball in the center of the field into another opponent's black goal area. The robot with the number of points at the end of two minutes was the winner. In the case of a tie, the robots would have five opportunities to score a penalty kick on the other goal. The kicking robot could only touch the ball once before stopping (making it a kick and not a push) and the defending robot needed to wait until the ball moved until it started defending. The robot with the most points moved on in the bracket.

Figure 1: Field positioning

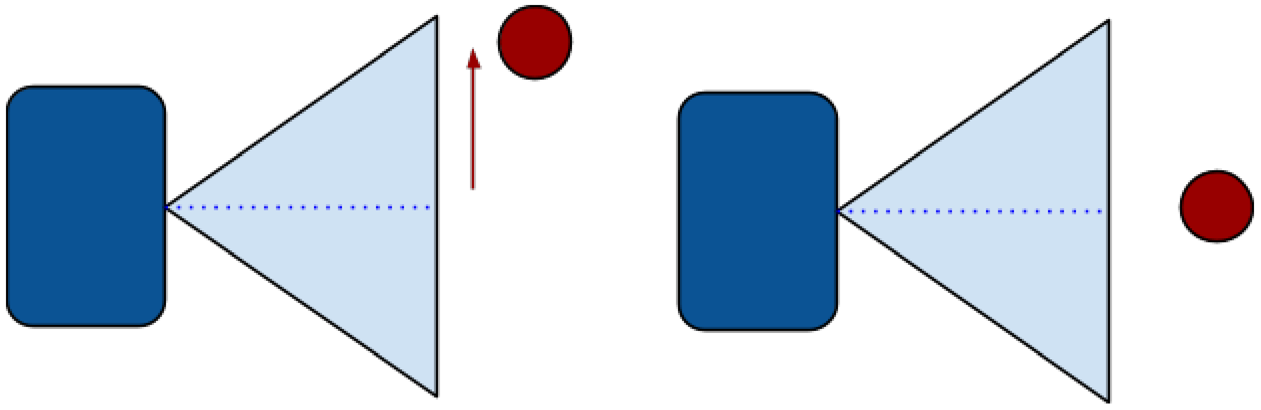


Hardware and Design

Our robot was designed very simply. It had a touch sensor to detect a kick for the penalty kick, a v-shaped pushing bar for normal competition, and the infrared sensor for detecting the ball. The idea behind our design was to be simple and fast. We wanted to get to the ball first and push it into the other robot's goal. The robot would turn until the infrared sensor detected the ball in front of it and then it would move towards it. It was a very simple design, and the goal was to keep it simple enough that the NXT would perform consistently.

Our goal kicking program did much the same, except when it detected the touch sensor being pushed (the ball being kicked), it stopped the robot. The defending program turned the robot left and then kept the ball at a certain point on the infrared sensor, to the left of the robot. This behavior is outlined in Figure 2.

Figure 2: Defense program



Discussion

The competition was average for us. We lost the first round due to losing the penalty kick stage. Our defense program was a little buggy. The problem was that we had it turn for a set amount of time. We should have had it turn until the ball faced a certain direction. The robot turned too far in the competition and ended up off-center and missed a defense. The other problem we encountered was that the touch sensor was stiff and it wouldn't always trigger on the kick part of the program. To try to compensate we added a small piece to give more surface area to the touch sensor, but it still did not always activate when it should have.

The main program worked well, but sometimes when it got close to the ball, the sensor readings wouldn't work correctly and it would turn too sharp or not sharp enough. This meant that we occasionally missed the ball and usually resulted in getting tangled up in the other robot. We did not implement a random-turn into our robot because we did not think it was necessary. However, the robot did get stuck against a wall a couple of times.

Code

MAIN SOCCER PROGRAM:

```
// Soccer Player Bot

#define PORT2 IN_2 // Compass Sensor
#define INFR IN_1 // Infared sensor
mutex moveMutex;

int compVal;
int compScore;
byte dir, s1, s2, s3, s4, s5;
float direction;

// take "other goal" compass measurement
void initCompass(){
   compScore = SensorHTCompass(IN_2);
   compScore -= 90;
}

// search for ball
task navigate(){

  initCompass();

  while( true ){
          ClearScreen();
          compVal = SensorHTCompass(IN_2);
          NumOut(0, LCD_LINE2, compVal);
          NumOut(0, LCD_LINE3, compScore);
          ReadSensorHTIRSeeker2AC(INFR, dir, s1, s2, s2, s4, s5);
          direction = dir;
          NumOut(0, LCD_LINE4, direction);

          Wait(100);
  }
}

void move_IR( float x){
          int A,B;
          if(x >= 0){
```

```
        B = 100.0 - 100.0*x;
        A = 100;
        }
        else{
        B = 100;
        A = 100.0 + 100.0*x;
        }

        OnFwd(OUT_A, A);
        OnFwd(OUT_B, B);
        Wait(100);
}

task move(){
        float norm_dir = 0.0;
        float strength;

        while(true){
        strength = (s1 + s2 + s3 + s4 + s5);
        NumOut(0, LCD_LINE5, strength);
        if(strength < 300.0){
        norm_dir = (dir - 5)/4;
        }
        Acquire(moveMutex);
        norm_dir = (direction - 5.0)/4.0;
        move_IR(norm_dir);
        Release(moveMutex);
        }
}

task main(){
        SetSensorLowspeed(INFR); // infared
        SetSensorLowspeed(IN_2); // compass

        Precedes(navigate, move);
}
```

BALL KICKING PROGRAM

```
// Soccer Player Bot

#define PORT2 IN_2 // Compass Sensor
#define INFR IN_1 // Infared sensor
#define TOUCH IN_3
mutex moveMutex;

int compVal;
int compScore;
byte dir, s1, s2, s3, s4, s5;
float direction;
bool kicked = false;

// take "other goal" compass measurement
void initCompass(){
   compScore = SensorHTCompass(IN_2);
   compScore -= 90;
}

// search for ball
task navigate(){

  initCompass();

  while( true ){
          ClearScreen();
          compVal = SensorHTCompass(IN_2);
          //NumOut(0, LCD_LINE2, compVal);
          //NumOut(0, LCD_LINE3, compScore);
          ReadSensorHTIRSeeker2AC(INFR, dir, s1, s2, s2, s4, s5);
          direction = dir;

          Wait(100);
  }
}

void move_IR( float x){
          int A,B;
          if(x >= 0){
          B = 100.0 - 100.0*x;
```

```
        A = 100;
        }
        else{
        B = 100;
        A = 100.0 + 100.0*x;
        }

        OnFwd(OUT_A, A);
        OnFwd(OUT_B, B);
        Wait(100);
}

task move(){
        float norm_dir = 0.0;
        float strength;
        while(!kicked){
        strength = (s1 + s2 + s3 + s4 + s5);
        if(strength < 300.0){
        norm_dir = (dir - 5)/4;
        }
        Acquire(moveMutex);
        norm_dir = (direction - 5.0)/4.0;
        move_IR(norm_dir);
        Release(moveMutex);

        if(SENSOR_3 == 1){
        kicked = true;
        NumOut(0,LCD_LINE3, 1);
        }
        }
        Off(OUT_AB
        );
}

task main(){
        SetSensorLowspeed(INFR); // infared
        SetSensorLowspeed(IN_2);  // compass
        SetSensorTouch( IN_3 );  // TOUCH

        Precedes(navigate, move);
}
```

KICK DEFENSE PROGRAM

```
// Soccer Player Bot

#define PORT2 IN_2 // Compass Sensor
#define INFR IN_1 // Infared sensor
#define TOUCH IN_3
mutex moveMutex;

int compVal;
int compScore;
byte dir, s1, s2, s3, s4, s5;
float direction;
bool kicked = false;

// take "other goal" compass measurement
void initCompass(){
   compScore = SensorHTCompass(IN_2);
   compScore -= 90;
}

// search for ball
task navigate(){

  initCompass();

  while( true ){
          ClearScreen();
          compVal = SensorHTCompass(IN_2);
          //NumOut(0, LCD_LINE2, compVal);
          //NumOut(0, LCD_LINE3, compScore);
          ReadSensorHTIRSeeker2AC(INFR, dir, s1, s2, s2, s4, s5);
          direction = dir;

          Wait(100);
  }
}

void move_IR( float x){
          int A,B;
          if(x <= -1 || x >=1){
          B = -100;
```

```
        A = -100;
        }
        else if(x == 8){
        B = 0;
        A = 0;
        }
        else{
        B = 100;
        A = 100;
        }

        OnFwd(OUT_A, A);
        OnFwd(OUT_B, B);
        Wait(100);
}

task move(){
        float norm_dir = 0.0;
        float strength;
        while(true){
        strength = (s1 + s2 + s3 + s4 + s5);
        if(strength < 300.0){
        norm_dir = (dir - 5)/4;
        }
        Acquire(moveMutex);
        norm_dir = (direction - 5.0)/4.0;
        move_IR(norm_dir);
        Release(moveMutex);
        }
}

task main(){
        SetSensorLowspeed(INFR); // infared
        SetSensorLowspeed(IN_2);  // compass
        SetSensorTouch( IN_3 );  // TOUCH

        Precedes(navigate, move);
}
```