Another Forking Robot

A.k.a. AFR

Matt Silverman

IMDL Summer 99

Table of Contents

Abstract	Page	3
Executive Summary		4
Introduction		5
Integrated System		5
Mobile Platform		6
Actuation		6
Sensors		6
Behaviors		8
Conclusion		8
Appendix		9

I. ABSTRACT

Another Forking Robot (AFR) is a mobile forklift robot. This forklift will differ from previous forklifts in that this forklift will have the forks on the side of the robot and extend out. This motion will be controlled by a servo. The forks will also extend up above the platform to reach bins in a screw box. The forks will not lower below the top of the robot. The special sensor idea is a barcode scanner that uses a phototransistor. This will let the robot find out what location it is at. The purpose is to be able to find a location, rise to a desired height and retrieve an object, on a palette. This idea came to me because of the item retrieval problems in the warehouse I work in. This robot is not intended to be a prototype of a larger model but rather a way to show how robotics can aide in solving my problems at work.

II. EXECUTIVE SUMMARY

AFR, a side loading forklift robot, uses the HC11 EVBU board together with a Mekatronix ME11 as the “brain” of the robot. This “brain” allow the robot to move using motors and wheels, it enables it to “see” using infared(IR) light. This is accomplished by shining IR in several directions and the with detectors record how much IR light was reflected back. The closer the robot is to an object the more IR light will be reflected. Once the robot gets within a certain distance to an object the detectors become saturated. Thus, they are detecting as much light as the possibly can. This is useful in avoiding obstacles or staying near a wall. In order to lift something up the robot will need a set of forks. As mentioned earlier these forks will function differently than previous forklifts. Instead of being in the front of the robot and going head to retrieve an object this robot will simply drive up next to the bin, raise the forks to the proper height and then extend the forks out into the bin. Spinning a threaded rod that goes through the platform with the forks will do the raising of the forks. A small servo will push the fork out and then pull them in from the bin. In order to retrieve the correct item, the robot must know what bin it is at. Therefore the floor will be marked with barcodes made from electrical tape. Using a phototransistor, the robot will move scan these barcodes as it moves over them. In order to get appropriate results a small light bulb needs to be placed under the platform. This is because the phototransistor returns a voltage based on the amount of light reflected in the base. When the transistor is over a stripe of black electrical tape the light reflected will be much less than when light is reflected off of blank paper.

III. INTRODUCTION

While working part-time at a warehouse I discovered a common problem. Every time an item need to be brought down from stock to the floor a human must walk the warehouse looking for the item. Then a forklift is driven to the necessary location and the item is “dropped”. Although this seems trivial, in a warehouse with approximately 50 double-sided aisles each with 15 bays and three levels of stock, it can be very time consuming. Now lets say the person searching for an item misses it on the first walk through. You can see that valuable time is wasted. This led me to an idea for a robot. Why not have the robot store the information where every item is? Better yet, why not have the robot go and get the item and eliminate the human’s busy work completely? The following descriptions show how all the parts, movements, and behaviors necessary to accomplish this.

IV. INTEGRATED SYSTEM

Using the Meketronix ME11 board and the HC11 EVBU board, I control two fully hacked servos that act as dc motors. The ME11 board is also used with the hacked sharp IR detectors and IR emitters. The emitters are pulsed at 40KHz at the output port 0x7000. I only used three pins of this port to power the IR LEDS. The detectors are given power and ground and then connected to the analog port. Six of these pins, PE0 – PE5, are for these detectors. PE6 is used for the Photo Transistor. In addition, PE7 could probably be used for an array of bump sensors. The servo for the extending of the forks is a standard the wire 48oz servo. It is given power, ground, and a pulse generated by OC3. Finaly, the 12V stepper motor is controlled by an Allegro 5804, which is given two inputs, a direction and a pulse.

IV. MOBILE PLATFORM

The platform is of an oval shape, 15 inches long by 10 inches wide, such that the forks will line up flush with the left edge of the platform. The forks will sit and a six inch wide platform above the circuitry and extend up. They will not lower below the top of the platform. The forks will also be retractable, thus they will sit directly above the platform and extend out.

V. ACTUATION

AFR will have the two hacked servos in the rear for movement, with two casters in the front for support. These servos can move forwards or backwards for any amount of time. The stepper motor will control the raising and lowering of the forks by spinning 1/4inch threaded rod. There is also a 3/16inch metal rod used for additional support that the fork platform will easily slide on. The forks will extend and retract using the servo from OC3. The forks are designed in this way so that they are not to be outside of the perimeter of the platform when the robot is moving. Also, an advantage to having the forks face the side is that no turning is necessary to use the forklift.

VI. SENSORS

<u>IR</u>

The robot will have IR emitters and detectors in the front, rear and along the left side. Each pair, a detector and an emitter, is placed strategically (Figure S-1) to accomplish different actuations. The pair in the rear and the 3 pairs in front are used in obstacle avoidance, thus ensuring that it will not hit any walls. If a sensor returns a value, A to D converted, that is larger than 110, then the robot will move in the opposite direction of where that value was returned from. The two pairs on the left side are used in a wall following behavior. Unlike obstacle avoidance, if the values returned are both above 110 then the robot will continue to move forward. If an obstacle floods the detectors in the front we assume that there is a corner and we should turn to the right.

Phototransistor

Locations on the floor will be marked by barcodes. They will be read by a phototransistor judging the amount of light reflected at several rows as the robot moves past them. For the scope of the class there will only be three bits on each barcode, a start bit and two data bits. Thus there can only be four locations. However, we can increase the number of data bits when there are more locations.

This sensor is used to measure reflecting light and since the platform blocks most of the light in the room a small bulb is positioned next to the transistor to flood the floor with white light. We then measure the reflection on white paper with strips of electrical tape. This bulb improved the readings unbelievably. Without the bulb the converted reading ranged from 0-5. With the bulb the values jumped to 25-35 for a stripe and 45-60 for blank white paper. The transistor emitter is connected to Vcc, 5V, and the collector is connect to a 1Kohm resistor to ground. The base is what the light is reflected into. The voltage across the 1K resistor is measured and converted on pin PE6.

VII. BEHAVIORS

The behaviors that are fully functional are wall following, obstacle avoidance, and barcode scanning. The incomplete behaviors are the movement of the forks. The hardware is designed but no software is written to control the motors. For the behavior algorithms see the code section of the appendix.

VIII. CONCLUSION

Although this project did not accomplish what I intended I learnt a lot about how robots operate. I completed the basic design of the robot, the movement, and special sensor behaviors of the robot. In the summer semester time is very valuable. There is little time to waste. I plan to complete my idea even after the semester is over. The things left to be done are the coding of the motors to actuate the forks, the addition of a communication system, and a receiving dock. If I could do the project over I would have worked harder in the beginning and made sure that I had all the parts I needed as soon as possible. I also realize now that my idea was too complex to finish in a short semester. I am pleased with my work but I am disappointed I did not meet my goals. I feel that there is still much more that can be done with this and as I said earlier I will continue to work.

IX. APPENDIX

Code

```
/***************INCLUDES**********************/

#include <analog.h>
#include <clocktjp.h>
#include <motorme.h>
#include <servome.h>
#include <serialtp.h>
#include <isrdecl.h>
#include <vectors.h>
```

```
#include <stdio.h>


/*************END INCLUDES*****************/

/**************CONSTANTS*********************/

#define LEFT_MOTOR 0
#define RIGHT_MOTOR 1
#define MAX_SPEED 100
#define MIN_SPEED 0
#define AVOID_THRESHOLD 120
#define SCAN_THRESHOLD 40
#define LEFT_IR analog(3)
#define MID_IR analog(4)
#define RIGHT_IR analog(5)
#define SCAN analog(6)
#define BACK_IR analog(0)
#define REAR_IR analog(1)
#define FRONT_IR analog(2)
#define IRE *(unsigned char *) (0x7000)
/***********END CONSTANTS**************/

void do_scan(void);

void main(void)
{

int irdr, irdl,irdb,irdsf, diff,irdsr, irdm, speedr, speedl;

init_analog();
init_motorme();
init_servome();
init_clocktjp();
init_serial();
IRE = 0x07; //Turns on IR Emitters



while(1)
{

        irdr= RIGHT_IR; // get all IR Readings
        irdl= LEFT_IR;
        irdm= MID_IR;
        irdb= BACK_IR;
        irdsr= REAR_IR;
        irdsf= FRONT_IR;
        diff = irdsf - irdsr; //Calculated difference for wall following

        if (diff== 0) // wall following
        {
                if (irdm < AVOID_THRESHOLD)
                {
                printf("diff is 0 %d\n", SCAN);
                if (SCAN < SCAN_THRESHOLD)
```

```
                        do_scan(); // scan function
                speedl = 70;
                speedr = 70;
                }
                else
                {
                speedl = 100;
                speedr = -100;
                }

        }
        else if ((irdm > AVOID_THRESHOLD)&&(irdl > AVOID_THRESHOLD)) // turn to follow
        {
        speedl = 100;
        speedr = -100;
        }else if ((irdsf > 110)&&(diff < 0)) // routines to even out wall follow
        {
        printf("front is is higher %d\n", SCAN);
        if (SCAN < SCAN_THRESHOLD)
                        do_scan();
        speedl = 70;
        speedr = 60;
        }
        else if ((irdsr > 110)&&(diff>-5))
        {
   printf("rear is higher %d\n", SCAN);
   if (SCAN < SCAN_THRESHOLD)
                        do_scan();
        speedl = 60;
        speedr = 70;
        }
        else //avoid obsticales
        {
                if (irdr < AVOID_THRESHOLD)
                        speedl = 100;
                        else
                        speedl = -100;
                 if (irdl < AVOID_THRESHOLD)
                        speedr = 100;
                        else
                        speedr = -100;
        }
        motorme(LEFT_MOTOR, speedl);
        motorme(RIGHT_MOTOR, speedr);
        wait(150);
        motorme(LEFT_MOTOR, 0);
        motorme(RIGHT_MOTOR, 0);


}
}

void do_scan()
{
 int sbone,sbtwo;
  motorme(RIGHT_MOTOR, 0);
```

```
motorme(LEFT_MOTOR, 0);
printf("Rolled backed to: %d\n", SCAN);
wait(3000);
motorme(RIGHT_MOTOR, 75); //position for first reading
motorme(LEFT_MOTOR, 50);
wait(215);
motorme(RIGHT_MOTOR, 0);
motorme(LEFT_MOTOR, 0);
wait(1000);
sbtwo = (SCAN < 35); //get reading
printf("Bit Two : %d\n", SCAN);
wait(4000);
motorme(RIGHT_MOTOR, 75);// position for second reading
motorme(LEFT_MOTOR, 50);
wait(210);
motorme(RIGHT_MOTOR, 0);
motorme(LEFT_MOTOR, 0);
wait(1000);
sbone = (SCAN <40); //get reading
printf("Bit One : %d\n", SCAN);
 wait(4000);
if ((sbone)&&(sbtwo))  // output to leds
        IRE = 0x1f;
else if (sbtwo)
        IRE = 0x17;
else if (sbone)
        IRE = 0x0f;
wait (700);
motorme(RIGHT_MOTOR, 40); /go a little to clear transistor
motorme(LEFT_MOTOR, 40);
wait(75);
}
```

Figure S-1