PROJECT: ROB

Manufacturing Robot

Robert Quiroz
University of Florida
Department of Electrical and Computer Engineering
EEL 5666
Intelligent Machine Design Laboratory
Dr. Antonio Arroyo

TABLE OF CONTENTS

ABSTRACT

Rob is a wheeled vehicle with cylindrical shape that has two independent wheels and a caster wheel for support purposes and is designed to attend a call from a manufacturing station, discriminating between who is calling. There are two manufacturing stations that emit a beacon signal to be detected by Rob. This beacon is followed increasing the magnitude of the signal from it's starting point until reaches it. Once it reaches one of the beacon emitters it will start looking for the other beacon emitter.

EXECUTIVE SUMMARY

The purpose of this project is to design and build a robot that can attends the calls from several manufacturing facilities having the capabilities of difference who is calling. The goal is an actuated two wheeled robot that detects a beacon signal and distinguish one of them from other signals present.

Rob is designed around the Mekatronix MRC11 and MRSX01 boards. The robot is designed to avoid obstacles in its way by the means of IR detection. I t will react to bumping into objects by the mean of push button switches, find the calling station by the use of sonar and IR, and then come back by the path it used to arrive to the calling station.

Rob was successful in avoiding obstacles while trying to find the calling station using the IR object avoidance. It could find the location of the manufacturing station with the use of the sonar and IR. But it was not able to realize always that it has reached one of the manufacturing stations. The manufacturing stations should have an array of IR emitters instead of one IR emitter to accurately realize it has reached the manufacturing station. However the project successfully demonstrated the use of sonar and IR beacons as a way to call and be localized by a moving robot.

INTRODUCTION

The purpose of this project is to design and build a robot that can attends the calls from several manufacturing facilities having the capabilities of difference who is calling. The goal is an actuated two wheeled robot that detects a beacon signal and distinguish one of them from other signals present.

Rob is designed around the Mekatronix MRC11 and MRSX01 boards. The robot is designed to avoid obstacles in its way by the means of IR detection. I t will react to bumping into objects by the mean of push button switches, find the calling station by the use of sonar and IR, and then come back by the path it used to arrive to the calling station.

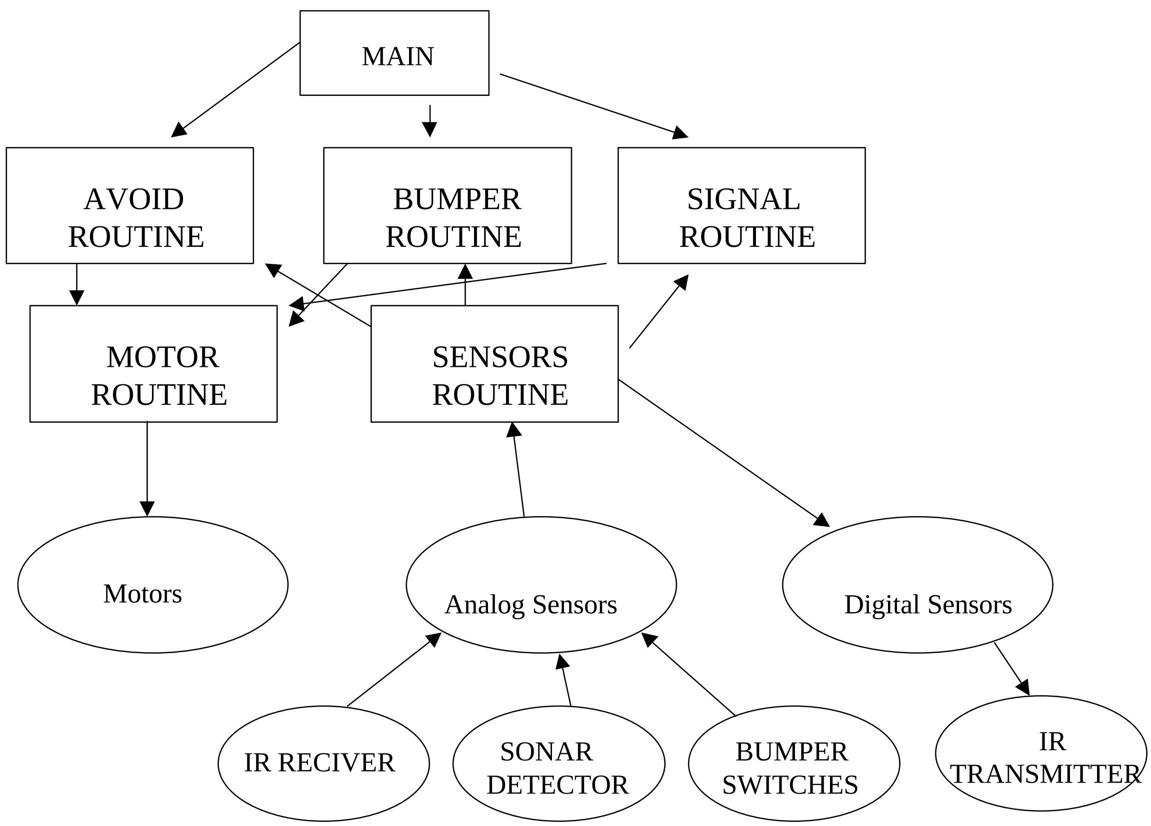
Rob was successful in avoiding obstacles while trying to find the calling station using the IR object avoidance. It could find the location of the manufacturing station with the use of the sonar and IR. But it was not able to realize always that it has reached one of the manufacturing stations. The manufacturing stations should have an array of IR emitters instead of one IR emitter to accurately realize it has reached the manufacturing station. However the project successfully demonstrated the use of sonar and IR beacons as a way to call and be localized by a moving robot.

INTEGRATED SYSTEM

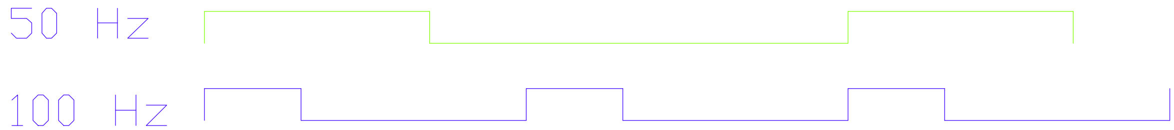
Rob uses the Mekatronix MRC11 and MRSX01 circuit boards. The MRC11 board is a board with that uses the microcontroller 68HC11 installed with 64k of RAM. The MRSXO1 is a sensor expansion board for the MRC11. The MRSX01 has all the circuitry used to control the instrumentation installed and used in Rob: 2 IR emitters, 2 IR detectors, 12 bump sensors, battery voltage sensor, 2 DC motor control and a sonar detector circuit.

Rob reads all the analog signals from the instruments installed and determines what action to take according to the readings. It determines if it has bumped with an object or detected an obstacle with IR. If it has found a signal from one of the manufacturing stations it control the motors to move in the needed direction. Once it's close to the manufacturing station it starts to follow the IR signal from the manufacturing station until it founds it. Figure 1 shows a block diagram of the algorithm implemented.

STRUCTURE OF THE SOFTWARE



Each manufacturing station has a sonar emitter and an IR emitter. The signals from each manufacturing station is emitted at a different frequency, in a way that Rob is able to determine the presence of the signal even though both manufacturing stations are transmitting at the same time. This is illustrated in the following figure.



MOBILE PLATFORM

Rob is a two-wheeled rounded platform. It is 10 inches in diameter with a caster for support. The wheels are 3 inches high, with a separation between wheels of 6 inches. The round shape allows Rob to move more freely and avoid in a better way to get stuck with an obstacle.

The platform has the circuits mounted on the top bottom of the robot and the batteries on the top center. Next to the batteries is the circuit board of the sonar detector. In the front of Rob there is located in the center the sonar transducer and next to it 2 IR emitters on the top and 2 IR receivers on the bottom of Rob.

Surrounding Rob there are 12 bump sensors with a wood bumper as a way to maximize the area of collision sensitivity. Next to the MRC11 are located the power and modes switches as well as the power LED. Rob is designed to rotate on its axis and by this way determine the origin of the sonar signal and move toward it.

ACTUATION

Two geared DC motors control the movement characteristics of Rob. The DC motors are actually servomotors transformed into 360 degrees motors. By the means of controlling which motor is running is possible to move towards the right or the left or go straight as well as go in reverse.

SENSORS

Rob uses 2 Sharp GP1U58Y 40 kHz IR Receivers in conjunction with 2 IR emitters modulated at 40 kHz to detect objects in the front of the way. The IR detectors are hacked to output an analog signal using a technique supplied by IMDL. It can be found in the Mekatronix Talrik Manual.

For tactile porpoises Rob has installed 12 miniature push buttons all around the body with a bumper as a way to increase the bumping area. These are connected with an array of resistors to one of the microcontroller analog ports. There is also installed a 40 kHz. Sonar receiver. This circuit was copied from a previous Robot called WobbleHead from the IMDL class of previous semesters. The circuit is detailed in the figure. The output from this circuit is a sinusoidal waveform whose Vp, depends on the distance from the transmitter.

BEHAVIORS

Rob has the following behaviors: Avoid Obstacle, Object Detect, Find Manufacturing station by the sonar, Approach the manufacturing station via IR, memorize route used. The code of these behaviors is shown in the Appendix. The robot is designed to move forward and in the eventually of find an object in it's way to the sonar emitting station it will try to avoid it moving to the left or right for a period of time and then recovering the course it was following before. Avoid obstacle uses the IR receivers in conjunction with the IR emitters mounted on Rob. If Rob runs into an object the bump sensors detect it and try to avoid the object depending on the switch that was pressed or the combination of them.

To find the calling manufacturing station the robot turns around itself until it detects the signal, once the signal is detected the robot moves toward it, trying to maximize the amplitude of it. Once the robot has reached the maximum amplitude of the reading, and doesn't get a better signal. It changes to IR detection of the manufacturing facility approaching to it following the same pattern as in sonar detection, and does not use IR obstacle avoidance until it reaches the calling manufacturing facility. Once it has reached the manufacturing facility it will turn around following the path it used to reached the manufacturing facility and then start looking for the other sonar calling signal following the same procedure as for the first one.

CONCLUSION

Rob was successful avoiding obstacles in its way to find the calling station. It could find and differentiate the calling station by the means of sonar detection and approach to them by IR. But it couldn't determine when it has reached the calling station. The reason for this is that it was used only one IR LED emitter on the calling station instead of an array of IR LED's which would have made a more powerful signal that is easier to discriminate. However Rob demonstrated that it is possible to determine and identify a calling station by the means of Sonar and IR detection.

APPENDICES

Software in ICC11

```
#include <vectors.h>
#include <serial.h>
#include <hc11.h>
#include <mil.h>
#include <analog.h>


# define SENSORS *(unsigned char *) 0xffb8

# define IR *(unsigned char *) 0xffb9

void sensor_registor(int,int);
void read_sensors();
void wait();
void bumper();
void speed(int,int);
void detect();
void masignal();


int irdt[14];
int cds[6], battery, r_bump,f_bump,charge;
int i,j,k;
int sensor_mirror;
int sonread[100]; /* 100 readings from the sonar */
int irinitial, soninitial; /* initial values according to room*/
int speedl=0;
int speedr=0;
int condition=1;  /* if battery charge is low stop everything*/
int close=0;  /* if close to station stop ir avoidance*/
int small=0;  /* variable for small board */
int big=0;                          /* variable for big board */
int n;      /* number of time units the signal is on*/
int signals[2];  /* strength of the signal */
void read_sensors(void)
{

if (close=0)
            IR=0xff; /* turn on all ir emitters */

wait();  /* wait for ir emitters to emit */

INTR_OFF();

 sensor_registor(0x1f, 0x10);
 charge = analog(0);
 r_bump = analog(1);

sensor_registor(0x1f, 0x11);
```

```
 battery = analog(0);
 f_bump = analog(1);

sensor_registor(0x1f, 0x0f);
 irdt[1] = analog(3); /* ir detector of the left */

 irdt[2] = analog(4); /* ir detector of the right */

 irdt[3] = analog(5); /* sonar detector */

 for (i=0; i<100; i++){

        sonread[i] = analog(5);

        }

if (close=0)
        IR=0x00;  /* turn off ir emitters until next time */

INTR_ON();
}

void sensor_registor(int bit_clear, int bit_set) {

INTR_OFF();
CLEAR_BIT(sensor_mirror, bit_clear);
SET_BIT(sensor_mirror, bit_set);
SENSORS = sensor_mirror;
INTR_ON();
}

void wait(void) {

j=0;
for (i=1; i<4000; i++)
        {j= j+1;}
}

void speed (int speed_left, int speed_right){

int val, insl, insr;
        val=1;

        if (speed_left < 0) insl = -10;
        else insl = 10;

        if (speed_right < 0) insr = -10;
        else insr = 10;

        while(val){

                if (speedl!=speed_left) {
                speedl=speedl+insl;
                motor(0,speedl);
                }
```

```
                    if (speedr!=speed_right) {
                    speedr=speedr+insr;
                    motor(1,speedr);
                    }
                    if ((speedl == speed_left) && (speedr == speed_right))
                              val =0;
          }

}

/* detects the presence of some signal */

void detect(void){

int cont=1;

i=0;
n=0;

while ((cont=1)&& (i<99)){

          if (((sonread[i] < 126) || (sonread[i] > 134)) &&
                    ((sonread[i+1] < 126) || (sonread[i+1] > 134)))
                    {
                              cont=0;
                              n=1;
                              i++;
                    }
          i++;
}
cont=1;

while ((cont=1) && (i<99)){

          if (((sonread[i] > 126) && (sonread[i] < 134)) &&
                     ((sonread[i+1] > 126) && (sonread[i+1] < 134)))
          {
                    cont=0;
          }
          n++;
          i++;
}
}

/* detects the max and minimun on the signal*/

void masignal(void)
{
int max;
int min;

i=0;
max = sonread[i];
min = sonread[i];
i++
while (i<99){
```

```
		if (sonread[i+1] > max)
			{ max = sonread[i+1];}
		if (sonread[i+1] < min)
			{min = sonread[i+1];}
		i++;

	}
	signals[k] = max-min;
	}
	void bumper(void)
	{
							if ((f_bump > 40)||(f_bump < 55))
							{
									speedl = -50;
									speedr = -50;
									speed(0,speedl);
									speed(1,speedr);
									for( i=1;i<10;i++)
											wait();
									speedl = 0;
									speedr = 0;
									speed(0,speedl);
									speed(1,speedr);
							 }

							if ((f_bump < 75 )||(f_bump > 85))
							{
									speedl = -50;
									speedr = 0;
									speed(0,speedl);
									speed(1,speedr);
									for( i=1;i<10;i++)
											wait();
									speedl = 0;
									speedr = 0;
									speed(0,speedl);
									speed(1,speedr);
							 }

							if (f_bump > 135)
							{
									speedl = -50;
									speedr = 0;
									speed(0,speedl);
									speed(1,speedr);
									for( i=1;i<10;i++)
											wait();
									speedl = 0;
									speedr = 0;
									speed(0,speedl);
									speed(1,speedr);
							 }

							if ((f_bump < 135 )||(f_bump > 115))
							{
									speedl = -50;
```

```
                                speedr = 0;
                                speed(0,speedl);
                                speed(1,speedr);
                                for( i=1;i<10;i++)
                                        wait();
                                speedl = 0;
                                speedr = 0;
                                speed(0,speedl);
                                speed(1,speedr);
                }

                if ((f_bump < 115 )||(f_bump > 95))
                {
                                speedl = -50;
                                speedr = 0;
                                speed(0,speedl);
                                speed(1,speedr);
                                for( i=1;i<10;i++)
                                        wait();
                                speedl = 0;
                                speedr = 0;
                                speed(0,speedl);
                                speed(1,speedr);
                }

                if ((f_bump < 135 )||(f_bump > 115))
                {
                                speedl = 0;
                                speedr = -50;
                                speed(0,speedl);
                                speed(1,speedr);
                                for( i=1;i<10;i++)
                                        wait();
                                speedl = 0;
                                speedr = 0;
                                speed(0,speedl);
                                speed(1,speedr);
                }

                if ((f_bump < 135 )||(f_bump > 115))
                {
                                speedl = 0;
                                speedr = -50;
                                speed(0,speedl);
                                speed(1,speedr);
                                for( i=1;i<10;i++)
                                        wait();
                                speedl = 0;
                                speedr = 0;
                                speed(0,speedl);
                                speed(1,speedr);
                }
}


main()
```

```
{

init_serial();
init_analog();
init_motors();

read_sensors();
        soninitial = irdt[3];        /* initialize the value according*/
        irinitial = irdt[2] + irdt[1] ; /* the noise of the room */
        irinitial = irinitial / 2 ;

        motor(0,speedl); /* motor on the left side */
        motor(1,speedr); /* motor on the right side*/

while(condition)
{
                        if (charge < 40)  /* if the battery charge is less than*/
                                condition = 0; /* 6 volts stop everything */

                        read_sensors();
                        bumper();
                        k=0;
                        detect();
                        speedl=80;
                        speed(0,speedl);
                        wait();
                        speedl=0;
                        speed(0,speedl);
                        read_sensors();
                        k=1;
                        detect();
                        if (signals[0] > signals[1])
                                {
                                speedl=50;
                                speedr=50;
                                speed(0,speedl);
                                speed(1,speedr);
                                for (i=0; i<40; i++)
                                        wait();
                                speedl=0;
                                speedr=0;
                                speed(0,speedl);
                                speed(1,speedr);
                                }
                        if (signals[0] < signals[1])
                                {
                                speedl=0;
                                speedr=80;
                                speed(0,speedl);
                                speed(1,speedr);
                                wait();
                                wait();
                                speedl=0;
                                speedr=0;
                                speed(0,speedl);
                                speed(1,speedr);
```

```
                    }

                write("  sonar= ");
                put_int(r_bump);

                write("  sonar= ");
                put_int(f_bump);


        }

    }
```