First step towards a Humanoid

Vijay Subramanian

EEL5666
Intelligent Machines Design Laboratory

Department of Electrical and Computer Engineering

University of Florida

December 10, 2002

CONTENTS

ABSTRACT

Wouldn't it be very comfortable if somebody took waste papers and cans from you and put them in the trash without a fuss? 'H' is very good at that and he doesn't complain.

In the near future, when robots become a very important part of our lives, we will feel the need for robots that look like us, work like us, talk like us…… a Humanoid. This is the first step.

'H-First step towards a Humanoid' is a robot that was created for the Intelligent Machine Designs Laboratory course in the University of Florida in 2002.

H takes any waste like papers, cans etc and puts them in the trash for you. H is a an autonomous mobile robot with an arm and sensors. 'H' moves around, stops before you, takes anything that you give , takes it to the trash, puts it there and continues doing this. 'H' can also act as a messenger for people by carrying papers and objects.

EXECUTIVE SUMMARY

The project began with the idea of developing a humanoid-'H'. H has an arm mounted on a platform. Its brain is a micro controller development board, which works with the help of a group of sensors.

H moves around avoiding obstacles until it finds the line it needs to follow. Then it follows the line. When a human needs its help, he stands in front of it and H stops. On tapping H on the head it raises its arm and opens the gripper to receive the object. It takes the object to the trash, drops it and continues to do line following until there is an obstacle in its path or a human needs its help.

The platform is a circular wooden disc with an arm mounted on it and runs with two toy wheels and two caster wheels. The arm has one degree of freedom. At the end of the arm is a gripper, which is used for grasping objects.

The sensors on H are: A pair of IR emitter reflector sensors for avoiding obstacles in the path. Three bump sensors (front, right and left of the disc) for moving away from obstacles, if H hits any. Three infrared emitter-receivers, which guide H to follow a line. A shaft encoder for precision motion. An IR emitter-receiver serves as eyes for H to stop when humans need its help. A tap switch on top of H tells it that the human it saw needs it to do work for him and an IR emitter detector sensor on the gripper tells H when it has been given an object for it to carry.

INTRODUCTION

For years, roboteers around the world have felt the need for humanoid robots for mankind of the future. There are lots of companies making industrial robots. But not many/any are making humanoids that can do domestic as well as industrial jobs. 'Honda' has been trying to make a humanoid called 'Asimo'. Waseda University and MIT have a research lab just for humanoids. I share with them a vision for robots to coexist and cooperate with mankind.

Objective:

The objective of this project was to build a humanoid robot that will do some simple job for humans like take things and put them in trash. This project is the first step towards making a perfect humanoid.

This project:

The first step in the project was to read books to know a lot about robots so that I get ideas to design them and start working on it. This took a long time.

The next step was to get the parts, for an imaginary robot. Then design the robot body in AutoCAD and cut the wood using the T-Tech machine.

Then I had to write codes to test the board and each of the sensors. The sensors had to be calibrated for proper operation.

The wood had to be cut and the whole body mounted on the platform base.

The final stage was to get the robot to work perfectly. This was done by tweaking the electronics; modifying the codes and making the robot behave just as it is expected to.

INTEGRATED SYSTEM

The main modules:

1. The brain this is the most important component in robots too. This coordinates the operation of the other modules.
2. Mobile base: The robot is a circular disc moving on wheels with servos for motion.
3. The arm: This is the major part of the robot which is going to perform the operation it was intended to do.

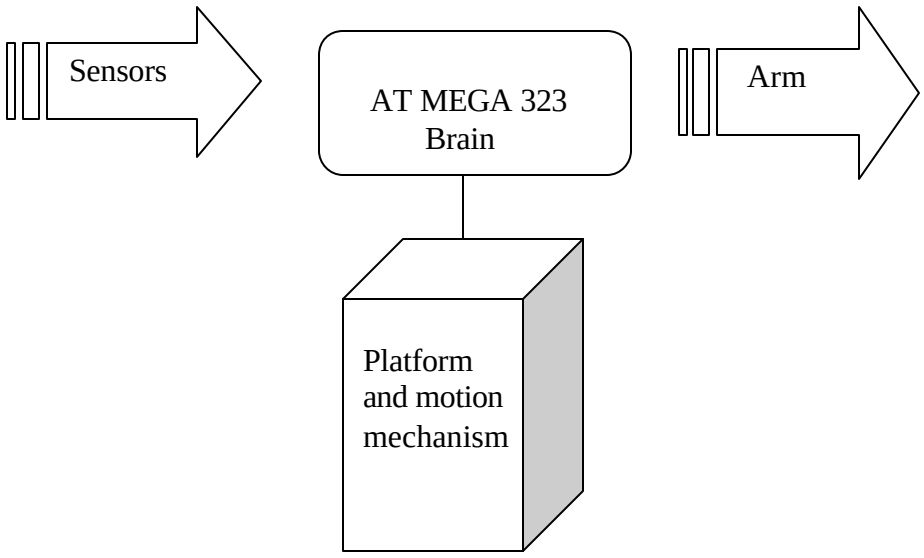


Fig.1.Modular diagram of H

4. Sensors: A group of sensors that help H in sensing the environment and act accordingly.
5. Power: Radio Shack Ni-MH AA rechargeable batteries

BRAIN

H’s brain is the Mega AVR-Dev -Development Board. It uses the ATMEL ATMega323-8PC Processor.

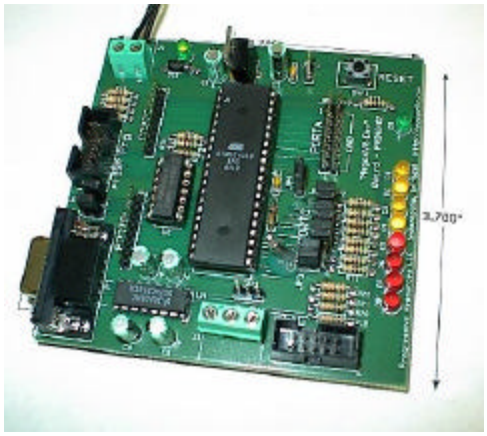


FIG.2. Development board

The important features of the board are:

- 32 I/O lines
- 1K EEPROM, 2K RAM
- 4 PWMs
- 8 Channel 10 bit ADC
- 32Kb of in-system programmable flash.
- Two 8-bit timer/counters
-One 16 bit timer/counter with input capture
- Programmable serial USART
- On chip oscillator.

MOBILE PLATFORM

BASE:

Objective:

The platform should be able to move well using wheels . The entire arm shall be mounted on the platform.

Design:

1. The robot should not get stuck in tight spaces, so a circular shape is the best option for the robot.
2. The aero plane wood given in the laboratory is quite strong to hold the entire robot with the arm.
3. Since it is a humanoid with a fairly long arm, the robot should be fairly tall. So the platform has to be at a good height above the floor.
4. The platform supports an arm as well as the electronics. So it should be wide enough to accommodate both.

The final design was this: A 11 inch diameter Disc.

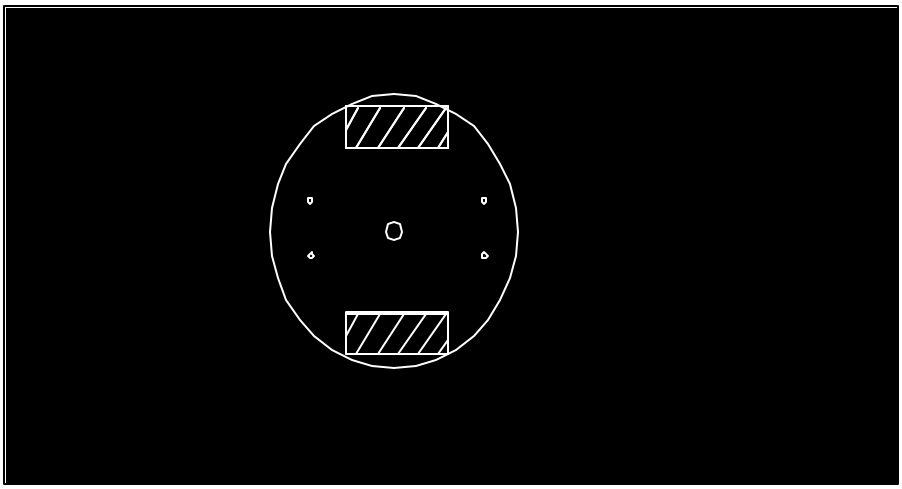


Fig.3. Plan of the platform without the arm.

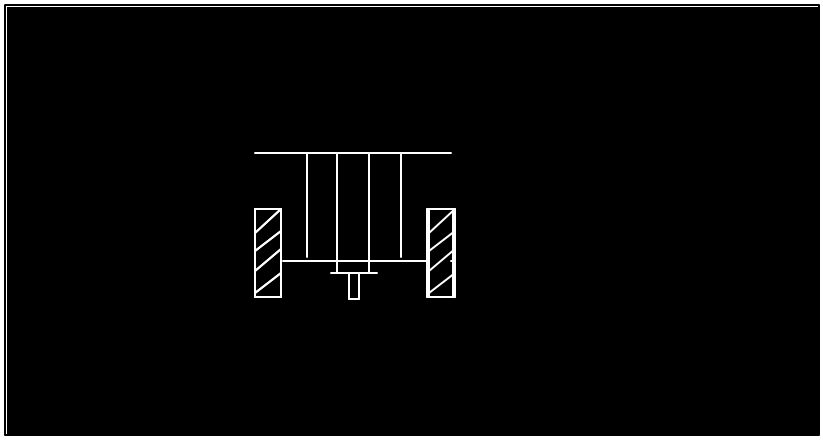


FIG.4.Elevation of the platform

A circular disc of wood was cut with the T-TECH machine available in the laboratory. The lower deck of the wheel is held with the platform using 8 inch spacers.

Hacking and calibrating the servo motor:

1. The servo motors have to be opened carefully.
2. The gear tabs have to be cut and removed.
3. Calibrate the servos with a program- Run the motor and find the stop value of the motor by trial and error.

The servo motors were glued on to the wheels. The servo motor was stuck to the lower deck of the platform using thread and tape (they stick well)

Lesson:

Fix all the screws and spacers to the platform well because the platform shakes a lot when the robot moves and all the loose screws come off.

THE ARM:

The arm has one degree of freedom. It has a gripper at one end and a servo motor at other. The shoulder of the robot is in between the motor and gripper at the center of gravity. The shoulder rotation is done using a servo motor. The gripper uses a 4-bar mechanism. This mechanism was chosen because it keeps the gripper fingers always parallel to each other for holding the objects well.

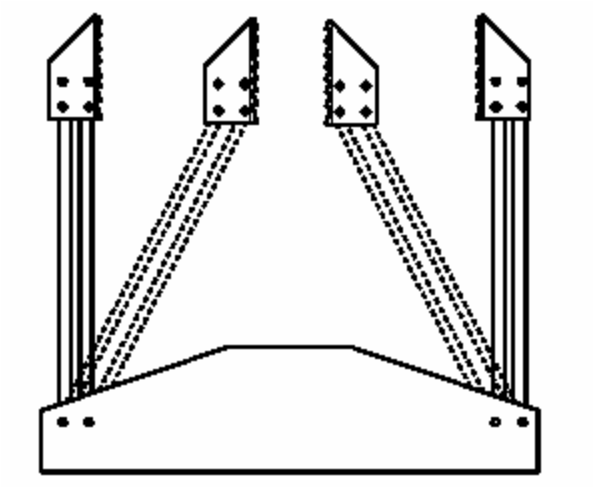


Fig.5. Gripper four-bar mechanism (Robot Builder's Bonanza, McComb, Gordon)

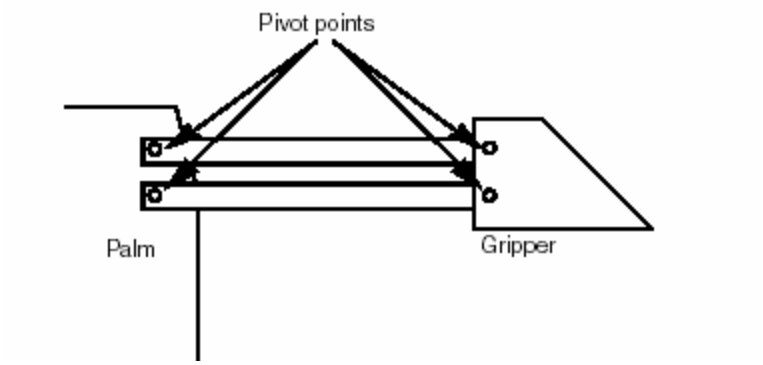


Fig.6. Finger system (Robot Builder's Bonanza, McComb, Gordon)

ACTUATION

MOTION:

Expected performance:

The robot has to do obstacle avoidance, line following, precise motion, arm rotation and gripper movement.

Driving:

Among the various methods of driving the motors, the Differential Steering is the simplest and the best. I used two wheels connected to two servo motors for moving and two caster wheels for supporting the platform. The idea of 4 wheels was dropped due to cost considerations. Contrary to the popular view in the laboratory, casters work very well. I had absolutely no problem with them.

Motors:

High torque motors (69 oz.in at 4.8V) were selected because the platform was required to carry the weight of the arm, the batteries and the electronics while in motion.

These motors were also used for the arm even though such high torque was not necessary because they were cheap.

Wheels:

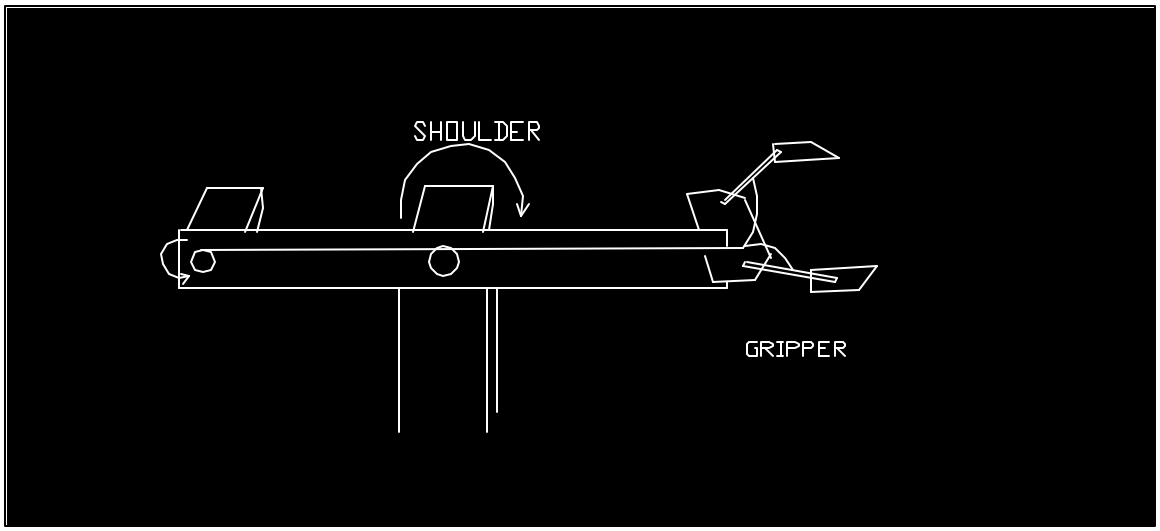
These were 5 inch in diameter. These were bought from Radio Shack from their toy spare parts. So they came with tires.

Motor connection:

Motors were directly connected to port D of the development board.

ARM:

The arm was actuated using two servo motors- one for the shoulder rotation and one for actuating the gripper.



(Fig.7.Design of the Arm)

Lesson learnt:

Make permanent connection for the robot and keep them connected. Use a switch to turn them on and off. If you connect them in reverse for a microsecond, you lose $10 and we are very much likely to connect them in reverse when working with so many wires all over the robot.

SENSORS

1. Obstacle Avoidance: Bump sensors and IR sensors.
2. Stopping before a human: IR sensors
3. Sensing an object in the gripper: IR sensors
4. Line following: IR sensors
5. Confirming that the human is going to give it a job: push button
6. Precision motion: shaft encoder, photo-reflectors.

1. **Obstacle avoidance**:

Bump Sensors:

On the circumference of the platform are fixed micro-switches at regular intervals. A bump skirt is fixed around the disc. When the robot hits an object, not sensed by the IR sensors or if the IR sensors fail to work or if the robot is hit from behind, the micro switches are pressed by the bump skirt which sends a signal to the robot informing it about a collision.

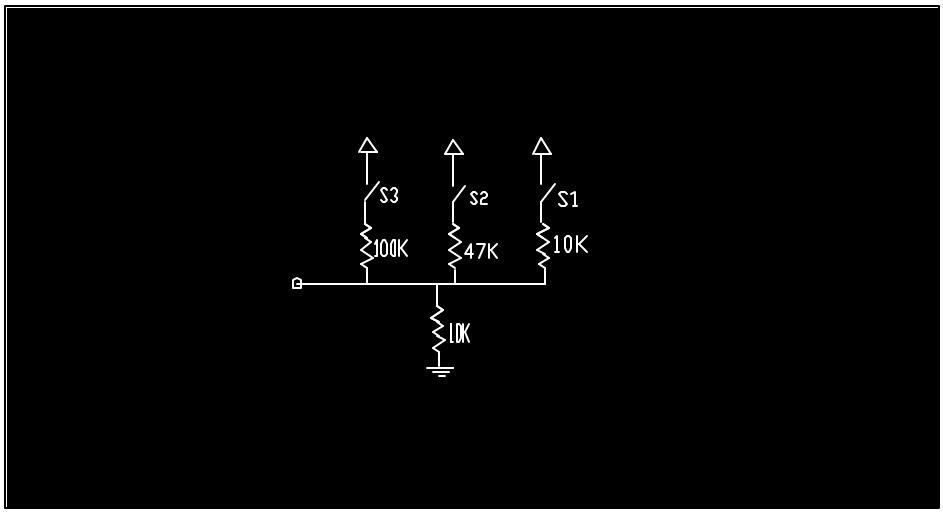


Fig.8. Circuit for bump sensor

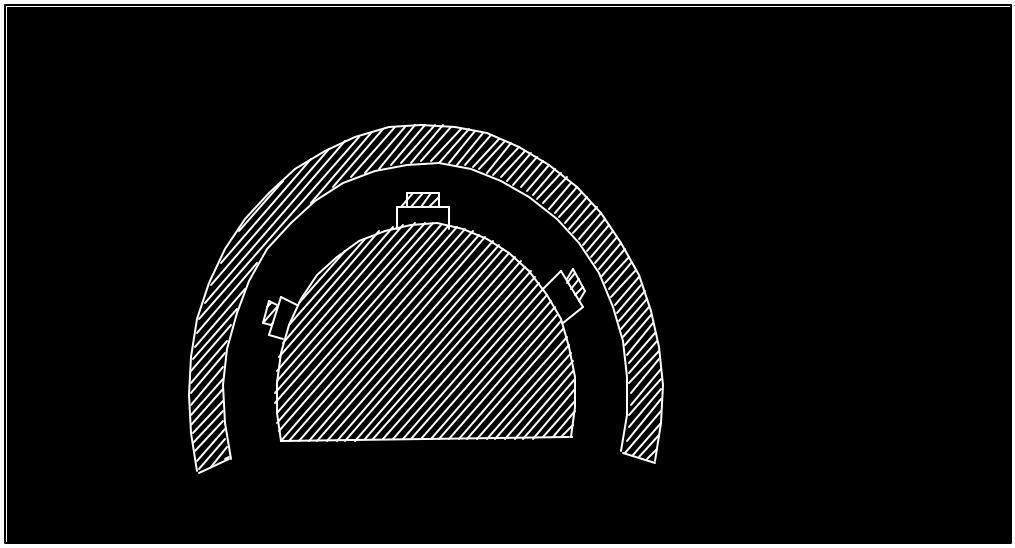


Fig.9 Bump sensor with bump skirts

Proximity sensor/IR sensor:

Pair of IR sensors SHARP GP2D12 is used for collision avoidance.

Fig.10. SHARP GP2D12

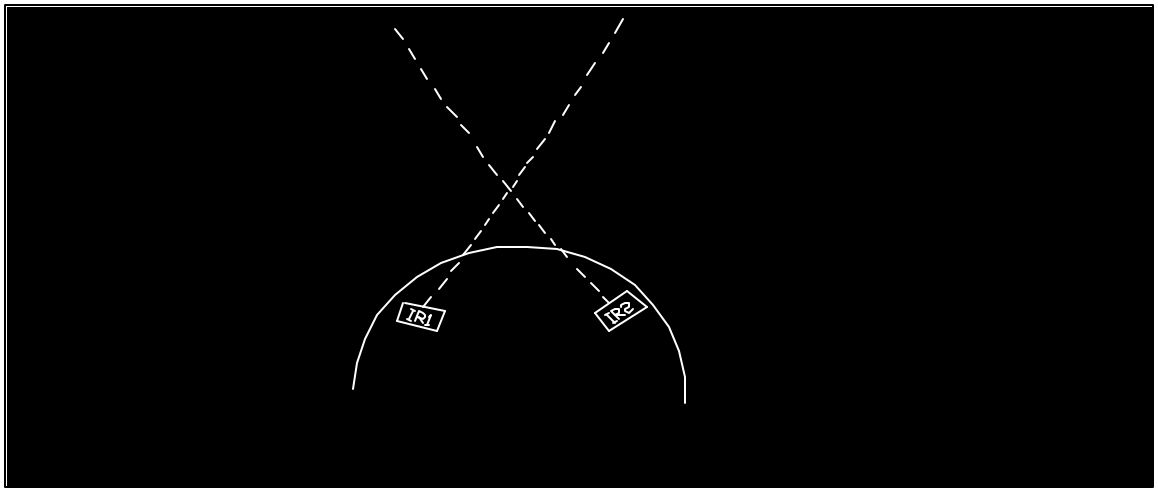


Fig.11 Arrangement of IR sensors for collision avoidance.

2. **Stopping before a Human: IR sensor**:

The IR sensor is placed on the top of the structure supporting the arm. That way, any tall object is classified as a human. This works well, though it sometimes stops for chairs and walls.

3**. Object sensor in the Gripper**:

The FAIRCHILD QRB1134 IR emitter detector with a range of 0.4 inches was used.

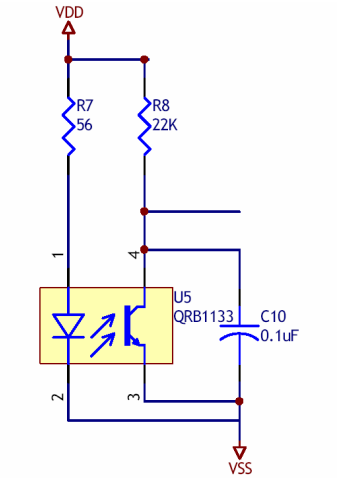


Fig.12. Circuit for the object sensor

When an object is placed in the gripper, it is sensed by this sensor and a signal is sent to the robot to close its gripper.

4. Line following:

The same FAIRCHILD QRB1134 IR emitter-detector was used for the line following behavior.

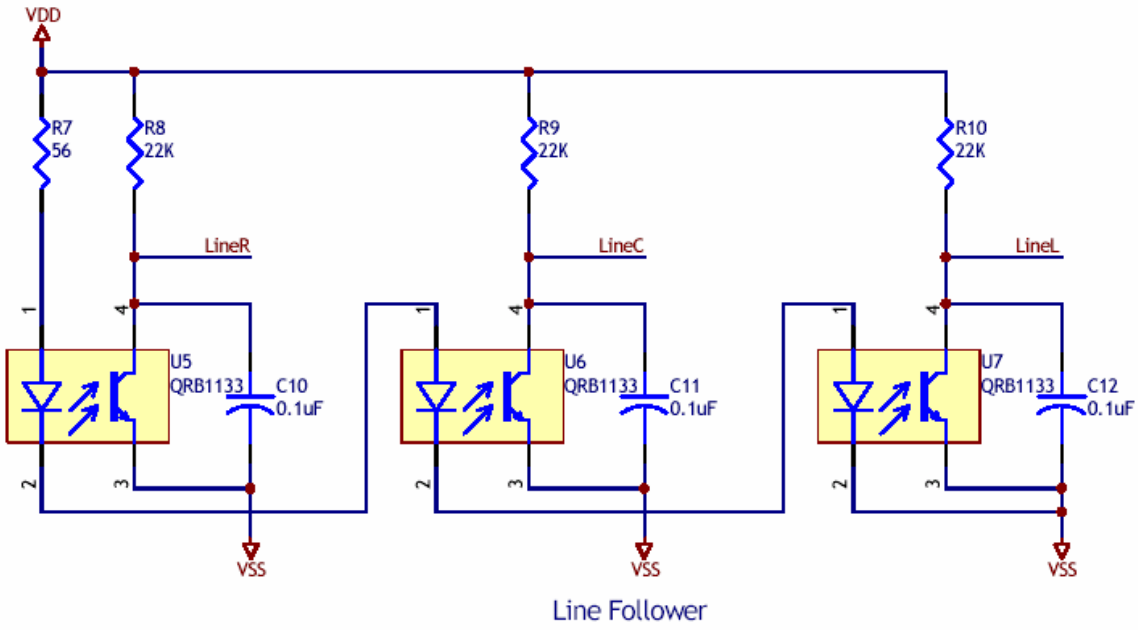


Fig.13. Circuit for line following.

A black line is drawn on the floor and the robot moves over it. The robot moves forward if the line is below the middle sensor, moves right if the line is below the left sensor and moves left if the line is below the right sensor and follows the line.

5. Sensing whether the human has a job for the robot:

A push button was used. The human had to tap the button to tell the robot that he needs its help.

6. Shaft encoder:

A shaft encoder was used for precise motion of the robot. This sensor is required when the robot is not required to do line following. The encoder helps H to find the location of the destination relative to the position of the human.

The Hamamatsu photo reflector was used for this sensor along with a disc with alternate white and black sectors. It is an IR emitter- photo transistor pair. The encoder disc was stuck to the wheel.

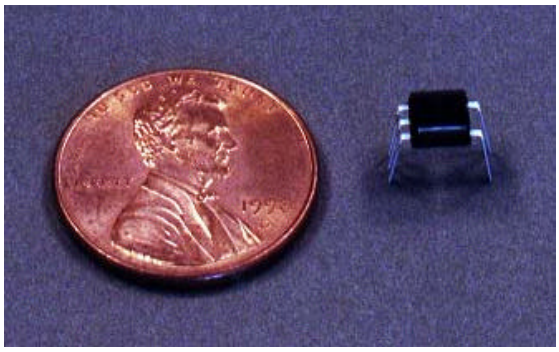


FIG.14 Hamamasu P5587

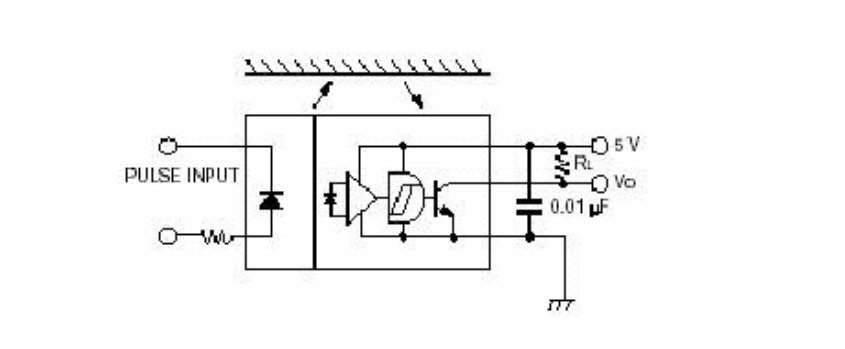


Fig.15. The Hamamatsu P5587 chip (www.acroname.com/robotics/parts/)

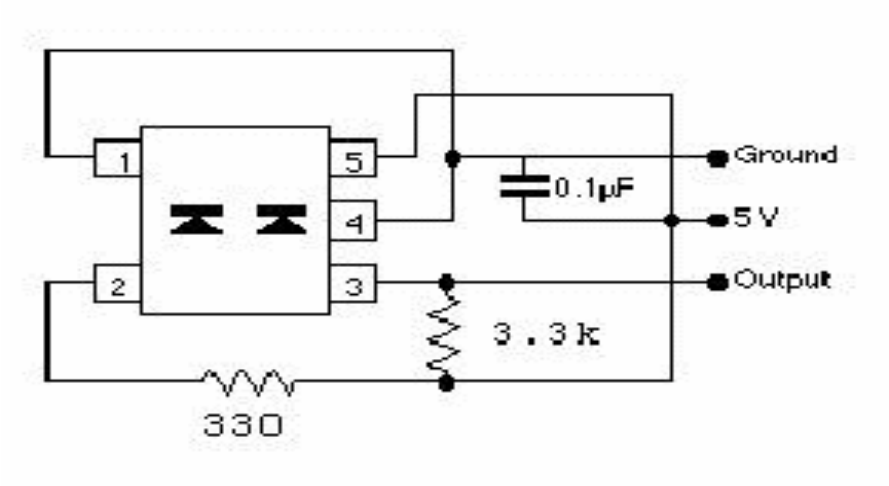


Fig.16. Circuit for the sensor (Courtesy: Louis Brandy, IMDL, Spring 2002)

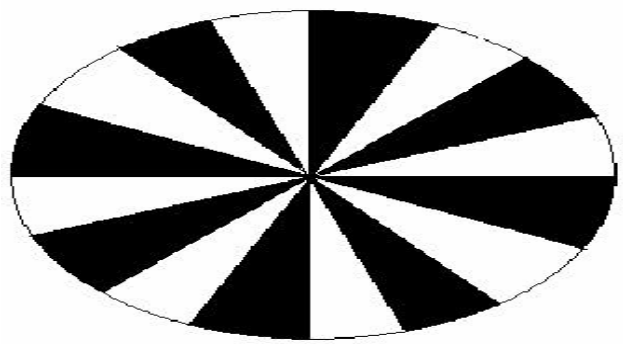


FIG.17. Encoder Disc.

Lessons learnt:

1. When you accidentally connect the sensor in reverse, you can throw it in the trash immediately.
2. Smaller the chip, the less likely it is to work for long. The Hamamatsu photo reflective sensors suddenly stopped responding one day. It was the size of a thumbnail and was difficult to connect and handle.

BEHAVIORS

H has three possible behaviors. It can be programmed to do any one or a combination. Small modifications in the code make these behaviors possible.

1. **Line Following**:

H stands on a line. When switched on, it starts following the line. When a human steps in front of it or waves his hand in front of H, it stops. The human taps H's head saying "Good boy H". H raises its arm, opens the gripper and waits till the human gives it any object to hold. H closes the gripper, lowers the arm and continues following the line. When H reaches a trash/ box, it stops again, raises its arm, opens the gripper and drops the object in the trash/box. Then it turns around and follows the line again.

If H senses a human and stops, but the human does not tap it on the head, H decides that the human does not want to give it anything and starts following the line again.

When H reaches the box/trash and does not have anything to drop, it turns around and continues following the line.

2. Precise Motion:

H starts from the trash/box, moves N steps (N can be varied in the code) forward and N steps backward. When it senses a human in its path, it stops as in the line following behavior takes the object and drops it in the trash.

The shaft encoder is used to detect the number of steps after which H stopped so that it can go back the same number of steps to the trash. This can also be done for different directions by moving H in right angles to the right or to the left.

3. Messenger:

H can also act as a messenger in which case it wanders around avoiding obstacles. When a human steps in front of H, it stops, raises its arm on tapping, opens the gripper and takes the object. Then lowers the arm and continues to do obstacle avoidance till it finds the next human. It then stops, raises its arm, drops the object and takes whatever is given to it. This way, H can act as a messenger in a place like an office.

EXPERIMENTS

1. Shaft Encoder:

When black sector is in front of the chip, the Hamamatsu photo reflective sensor gives 5V and 0V if white. The resolution improves with increase in number of sectors but the error in reading the number also increases. The optimal number was found to be 8. The input capture pin of the microprocessor was used to get the digital input. The count of the sensor was displayed in the AVR HyperTerminal. The robot was run for 40 inches and the counts of the sensor were compared with the calculated values.

Number of sectors	8	16	32
Resolution (inches)	1.25	0.625	0.3125
Error (inches)	0	1	3

Table.1. Change in error and resolution with change in number of sectors

As the number of sectors is increased, the sensor is unable to distinguish between black and white very quickly. So there is increase in error with increase in sectors.

2. IR Sensor:

An experiment was performed to calibrate the sensor. The results are presented in the form of a graph. The output of the sensor with variation in distance was plotted and was found that the range is 4 inches to 50 inches.

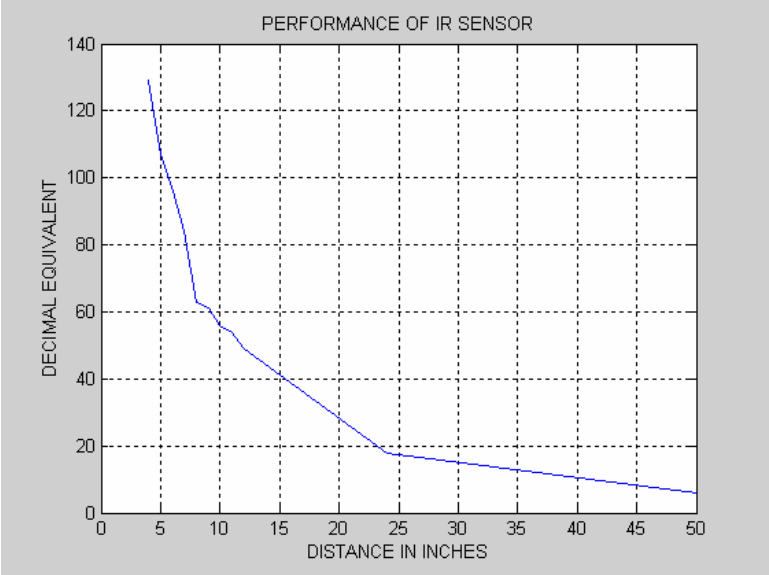


Fig.18 Performance of the sensor with change in object distance.

3. Testing the Development Board:

The board was tested using a simple program to turn on the LEDS in port C.

CONCLUSION

This project was started with the idea of this robot being the first step in developing a perfect humanoid. It had to be fairly tall to be one and so the platform was designed very tall compared to other robots. A simple arm with a gripper was designed and mounted on the platform. The robot now stands 3 feet tall. It has an arm for manipulating objects. A very simple vision system of IR sensors was provided. It was made intelligent with programs put into the micro controller and some behaviors were tested. It performed those functions well.

Since H was the first step towards a humanoid, had to be completed in a semester and had cost considerations, it had its limitations. H did not have a very good vision system- it could not tell that all tall objects were humans and it could not identify objects to be picked up. The arm had one degree of freedom and lacked the ability to manipulate objects well. Precision motion is not good enough. And the most important thing- it could not interact with humans effectively.

In spite of these limitations, H worked well in its environment. Vision though not good, achieved what it was supposed to do-stop before humans. We do not care whether the robot stops before the walls or not. We only get angry if it does not stop when we are in front of it. Don't we? The fact that the first arm I designed worked well assures me that this is definitely the first step towards a humanoid.

The designer of H, me, also learnt a lot building his first autonomous robot. I played and learnt real electronics. I learnt to design and create something. I made lots of mistakes and burnt chips, sensors, motors and also my hand. I learnt that temporary

connections are bad and permanent connections are temporary, that it takes a lot more time to get things working than I expect and that I learn a lot by actually making things. I was also assured that the human body is very good in its performance.

And thus, paving the way for a future of humanoids, was born **H.**

FUTURE WORK

The next step would be a humanoid with a better vision system capable of locating objects which the arm could pick up without a human's help, a more dexterous hand with the ability to feel, more intelligence that makes it capable of doing much more. The last step in humanoids is the humanoid 'programmed to love'.

REFERENCES

[1] Joseph L. Jones, Anita M. Flynn, Bruce A. Seeger, *Mobile Robots: Inspiration to Implementation*: Second Edition, A.K.Peters Publishers, Natick, MA 1990

[2] Fred Martin, The 6.270 *Robot Builder's Guide,* MIT, Cambridge, MA, 1992

[3] Newton C.Braga, *Robotics, Mechatronics, and Artificial Intelligence*: *Experimental Circuit Blocks for Designers*, Newnes, ISBN: 0750673893, 1st Edition, Nov 2001.

Acknowledgement

Mr. Amit Jayakaran helped me a lot in making this robot- in designing in AutoCAD, with the electronics and with the code.

The TA's were also of great help.

All my classmates who shared their ideas also helped me learn a lot.

And thanks to Dr.Arroyo for his pep talk.

APPENDIX

Sources for Parts

Development Board:

Mega AVR-Dev -Development Board;

www.prllc.com $56

Wheels:

Radio Shack $3.00/wheel.

Servo motors :

MARK III Robot Store.

Torque: 4.8V: 69 oz.in 6V: 86 oz.in

46 gms. , BP148X $ 10.25

IR emitter-detector: (obstacle avoidance)

MARK III Robot Store.

SHARP GP2D15 $ 8.25

Range: 10cm and 80cm.

IR emitter-detector :(To sense object in gripper and line following)

MARK III Robot Store.

FAIRCHILD QRB1134

Up to: 0.4 inch $ 1.75

Programs:

// The FINAL CODE which was demonstrated- line following, stopping before //humans, taking things, dropping in box behavior.

//HEADER FILE

ROBOT.H

```
void ADC_init(void)
{
    DDRA=0;                                                    //set A as input
    outp((1<<ADEN) | (1<<ADPS2) | (ADPS1), ADCSR);
                                        //Initialize to use 8bit resolution for all channels
}

u08 ADC_getreading(u08 channel)
 {
     u08 temp_valueH;
     ADMUX=ADMUX & 0xF8;

u08 bump_SW(void)
{
     return (ADC_getreading(2));
```

```
}

u08 obsense(void)
{
     return (ADC_getreading(2));
}

u08 tap(void)
{
     return (ADC_getreading(3));
}

u08 Top_IR(void)
{
     return (ADC_getreading(4));
}

u08 Line_Left(void)
{
    return (ADC_getreading(5));
}

u08 Line_Center(vo id)
{
    u08 led=ADC_getreading(6);
    PORTC=~led;
    return (led);
}

u08 Line_Right(void)
{
```

```
    return (ADC_getreading(7));
}
```

//**MAIN PROGRAM**

ROBOT.C

```
#include <io.h>
typedef unsigned short u16;
typedef unsigned char u08;

#include "adc.h"                    // converting the analog port signals to digital.
void delay(u16 delay_time) {
   do {
     u08 i=0;
     do {
     asm volatile("nop\n\t"          //volatile can be changed by any routine
         "nop\n\t"
         "nop\n\t"
         "nop\n\t"
         ::);
     } while(--i);                  //0 to ff then decrement till 0
   } while(--delay_time);           // till delay time goes to 0.
}
void motor_init(void)               // motor initializing routine
{
  outw(OCR1AL,512);                 // set PWM to stop speed.
  outw(OCR1BL,512);                // set PWM to for the other wheel to stop.
  outp((1<<COM1A1) | (1<<COM1B1) | (1<<PWM10) | (1<<PWM11), TCCR1A);
                                  // set output compare OC1A/OC1B clear on compare match,
  outp( (1<<CS11), TCCR1B);             // store prescaler Clk (I/O)/1024.
  sbi(DDRD, PD4);                    // port D pin D4 is for pwm-timer 1
  sbi(DDRD, PD5);                   //port D pin D5 is also pwm with timer 1.
```

```
//SERVO LEFT+RIGHT 7-28 center=19
    outp(0, TCNT2);                    /* start value of T/C2 */
    outp(0x66, TCCR2);                 /* init the counter */
//  outp(7, OCR2);                     /* value of OCR2 */
        //SERVO UP+DOWN 7+28
   outp(0, TCNT0);                     /* start value of T/C2 */
   outp(0x64, TCCR0);                  /* init the counter */
//  outp(7, OCR0);                     /* value of OCR2 */
}
void move_forward(void)
    {
      outp( (1<<CS11), TCCR1B);
      outw(OCR1AL,551);                //both wheels move forward
      outw(OCR1BL,386);
     }

void move_right(void)
        {
        outp( (1<<CS11), TCCR1B);
        outw(OCR1AL,1000);
        outw(OCR1BL,1000);
        }
void move_left(void)
   {
        outp( (1<<CS11), TCCR1B);
        outw(OCR1AL,1);                //stop one wheel and move other wheel only
        outw(OCR1BL,1);
    }
void move_back(void)
        {
        outp( (1<<CS11), TCCR1B);
```

```
        outw(OCR1AL,1);                 //both wheels move back
        outw(OCR1BL,1000);
        }
void stop(void)
 {
  outp( 0, TCCR1B);
  }

void lift_arm(void)
     {
        u08 i;
        for (i=8;i<=20;i++)
        {
        outp(i,OCR2);                   // value of OCR2    PB7
        delay(0xfFF);
        }
}
void lift_arm_half(void)
   {
        u08 i;
        for (i=8;i<=14;i++)
        {
        outp(i,OCR2);                   // value of OCR2     PB7
        delay(0xfFF);
        }
    }
void close_gripper(void)
{
        u08 j;
        for (j=16;j<=24;j++)
        {
```

```
        outp(j,OCR0);                   // value of OCR0     PB3
        delay(0x8FF);
      }
}
void lower_arm(void)
{
        u08 i;
      for (i=20;i>=8;i--)
        {
        outp(i, OCR2);                              // value of OCR2
        delay(0xfFF);
        }
}
void lower_arm_half(void)
{
        u08 i;
      for (i=14;i>=8;i--)
        {
        outp(i, OCR2);                              // value of OCR2
        delay(0xfFF);
        }
}

void open_gripper(void)
{
        u08 j;
       for (j=24;j>=16;j--)
        {
        outp(j,OCR0);                           // value of OCR0
        delay(0x8FF);
        }
```

```
}

void obstacle(void)
{
            if (Left_IR()>120)
             {
                move_left();
                delay(0x8ff);
             }
             else if (Right_IR()>120)
             {
                move_right();
                delay(0x8ff);
              }
}
u08 check_switch(void)
{
u16 delay_time=0xff;
   do {
     u08 i=0;
     do {
        if (tap()<100)
        return(1);
     } while(--i);                    //0 to ff then decrement till 0
   } while(--delay_time);             // till delay time is 0.
     return(0);
}

void line_follow(void)
{
        u08 threshold=200;
```

```
        while (Line_Left()>threshold)
        {
                move_right();
        }
        while (Line_Right()>threshold)
          {
                move_left();
          }
        while (Line_Center()>threshold)
        {
               if (Line_Left()<threshold && Line_Right()<threshold)
                 {
                     move_forward();
                 }
                 else if (Line_Left()>=threshold)
                 {
                    move_right();
                 }
                 else if (Line_Right()>=threshold)
                 {
                    move_left();
                 }
if (Top_IR()<120)
                    {
                         return;
                    }
                 if (Left_IR()>105 || Right_IR()>105 )
                       {
                         stop();
                         if(obsense()<240)
                         {
```

```
                    lift_arm_half();
                    open_gripper();
                    lower_arm_half();
                }
                move_left();
                delay(0xFff);
                return;
                }
}
}

int main (void)
{
DDRC=0xff;
DDRD=0xff;              //set D as output
DDRB=0xff;              //set B as output
u08 flag;
ADC_init();             // initialising analog to digital

u08 i,j,temp,bump;
motor_init();
//      open_gripper();

while(1)
        {
                move_forward();
                while(Top_IR()<60)
                {
                        if (Line_Left()>200 || Line_Right()>200 || Line_Center()>200)
                                line_follow();
                        if (Left_IR()>105 || Right_IR()>105 )
```

```
                {
                    stop();
                    if(obsense()<240)
                    {
                            lift_arm_half();
                            open_gripper();
                            lower_arm_half();
                    }

                    move_left();
                    delay(0xFff);

                    }
                    obstacle();
                    move_forward();
}
        stop();
        flag=check_switch();
        if (flag==1)
        {
                lift_arm();
                if(obsense()<240)
                {
                    flag=check_switch();
                    if (flag==1)
                    {
                        open_gripper();
                        delay(0x2Fff);
                    }
                }
                else
```

```
                    {
                        open_gripper();
                        while(obsense()>240);
                    }

                    close_gripper();
                    lower_arm();
                }

            }

}
    //  bump=bump_SW();
    else if (bump>205 && bump<225)
    {
        move_back();
        delay(0x1Fff);
        move_right();
        delay(0xFff);
    }
    else if (bump>120 && bump<140)
    {
        move_left();
        delay(0xFff);
    }
        else if (bump>12 && bump<25)
    {
        move_right();
        delay(0xFff);
    }
   }
```

```
}
```

// SHAFT ENCODER

```
#include <io.h>
#include <interrupt.h>
#include <sig-avr.h>
typedef unsigned char  uint8_t;
#define F_CPU           6000000                 /* 6Mhz */
#define UART_BAUD_RATE     19200                /* 19200 baud */

#define UART_BAUD_SELECT (F_CPU/(UART_BAUD_RATE*16l)-1)
typedef unsigned char  u08;
typedef  char  s08;
typedef unsigned short volatile u16;
typedef short s16;

uint8_t lo_val, hi_val,flag;
u16 value1=0,value=0,temp0,temp1;

void delay(u16 delay_time) {
   do {
     u08 i=0;
     do {
     asm volatile("nop\n\t"
        "nop\n\t"
        "nop\n\t"
        "nop\n\t"
        ::);
     } while(--i);
   } while(--delay_time);
}
```

```
SIGNAL (SIG_INPUT_CAPTURE1)
{
temp0++;
flag=PINA;
PORTC=PINA;
if(bit_is_set(flag,1))
        {
        value++;
        }
if(bit_is_set(flag,0))
        {
        value1++;
        }

}

/* uart globals */
volatile char* uart_data_ptr;
static volatile u08 uart_counter;

SIGNAL(SIG_UART_DATA)
/* signal handler for uart txd ready interrupt */
{
   uart_data_ptr++;

   if (--uart_counter)
      UDR = *uart_data_ptr; //outb(UDR, *uart_data_ptr);
      /* write byte to data buffer */
   else
      cbi(UCSRB,UDRIE);             //Have to disable IRQ, or it will repeat
```

```
}

SIGNAL(SIG_UART_RECV)
/* signal handler for receive complete interrupt */
{
   register char led;

   led = UDR;       /* read byte for UART data buffer */
//    PORTC = ~led;     /* output received byte to PortB (LEDs) */
   if(bit_is_set(UCSRA,UDRE)) {
      UDR = led;
   }
}

void uart_send(u08 *buf, u08 size)
/* send buffer <buf> to uart */
{
   if (!uart_counter) {
      /* write first byte to data buffer */
      uart_data_ptr  = buf;
      uart_counter   = size;
      UDR = *buf;
      sbi(UCSRB,UDRIE);      //Enable the send IRQ
   }
}

void uart_init(void)
/* initialize uart */
{
   UBRRH = 0x00;
   UBRRL = UART_BAUD_SELECT;
```

```
   UCSRB = _BV(RXCIE)|_BV(RXEN)|_BV(TXEN);
   UCSRC = _BV(URSEL)|_BV(UCSZ1)|_BV(UCSZ0);
   UCSRA = 0x00;
}

int main(void)
{
   DDRC  = 0xff;                        /* PortC output */
   PORTC = 0x00;                        /* switch LEDs on */
   DDRD  = 0;                           // Port B output for D9 LED
   DDRA=0;
   delay(0x01FF);
/* Enables ADC and start conversion in free running interrupt mode */
flag=0;
outp(32,TIMSK);                  //Enables the T1C0 overflow interrupt
outp(0x00,TCNT1H);               //start value of timer variable
outp(0,TCNT1L);

outp(0,TCCR1A);                   //No compare/capture/PWM
outp(1,TCCR1B);                      //prescale
value=0;
char buffer [10]="         ";
   uart_init();
   sei();                            /* enable interrupts */
DDRA=0;
   for (;;) {                       /* loop forever */
      itoa(value,buffer,10);
      uart_send(buffer, 3);
      delay(0x0FF);
      uart_send("+", 1);
      delay(0x05F);
```

```
        itoa(value1,buffer,10);
        uart_send(buffer, 3);
        delay(0x0FF);
        uart_send("=", 1);
        delay(0x05F);
        itoa(temp0,buffer,10);
        uart_send(buffer, 3);
        delay(0x0FF);
        uart_send(", ", 2);
        delay(0x05F);
        PORTA=0xFF;
         sbi(PORTD,1);
           /*
        if(bit_is_set(PINA,0))
           {
           PORTC=0;
           }
           if(bit_is_set(PINA,1))
           {
           PORTC=0xff;
           }*/
    }
}
```

// **OBSTACLE AVOIDANCE**

```
#include <io.h>

typedef unsigned short u16;
```

```
typedef unsigned char u08;
#include "adc.h"
void delay(u16 delay_time) {
   do {
     u08 i=0;
     do {
     asm volatile("nop\n\t"                  //volatile can be changed by any routine
        "nop\n\t"
       "nop\n\t"
        "nop\n\t"
        ::);
     } while(--i);                            //0 to ff then decrement till 0
   } while(--delay_time);                     // till delay time goes to 0.
}

void motor_init(void)
{
          outw(OCR1AL,512);                 //at 512 motor stops rotating(trial and error //
          outw(OCR1BL,512);
          outp((1<<COM1A1) | (1<<COM1B1) | (1<<PWM10) | (1<<PWM11),
  TCCR1A);               // set output compare OC1A/OC1B clear on compare match,
 outp( (1<<CS11), TCCR1B);                                  // store prescaler Clk(I/O)/1024.
sbi(DDRD, PD4);                    // port D pin D4 is for pwm-timer 1
sbi(DDRD, PD5);                    //port D pin D5 is also pwm with timer 1.
          //SERVO LEFT+RIGHT 7-28 center=19
outp(0, TCNT2);                      /* start value of T/C2 */
outp(0x66, TCCR2);                   /* init the counter */
//          outp(7, OCR2);                        /* value of OCR2 */
          //SERVO UP+DOWN 7+28
outp(0, TCNT0);                     /* start value of T/C2 */
outp(0x64, TCCR0);                  /* init the counter */
```

```
//        outp(7, OCR0);                    /* value of OCR2 */
}

void move_forward(void)
        {
        outw(OCR1AL,1000);                          //both wheels move forward
        outw(OCR1BL,1);
        }
void move_right(void)
        {
        outw(OCR1AL,1000);
        outw(OCR1BL,1000);
        }
void move_left(void)
        {
        outw(OCR1AL,1);
        outw(OCR1BL,1);
        }
void move_back(void)
        {
        outw(OCR1AL,1);                     //both wheels move forward
        outw(OCR1BL,1000);
        }

int main (void)
{
DDRD=0xff;                  //set D as output
DDRB=0xff;                  //set B as output
ADC_init();                 // initialising analog to digital
u08 i,j,temp,bump;
motor_init();
```

```
while(1)
        {
        move_forward();
        bump=bump_SW();
    if (Left_IR()>80)
    {
        move_left();
        delay(0xFff);
    }
    else if (Right_IR()>80)
    {
        move_right();
        delay(0xFff);
    }
    else if (bump>205 && bump<225)
    {
        move_back();
        delay(0x1Fff);
        move_right();
        delay(0xFff);
    }
    else if (bump>120 && bump<140)
    {
        move_left();
        delay(0xFff);
    }
        else if (bump>12 && bump<25)
    {
        move_right();
        delay(0xFff);
    } }}
```