EEL 5666 IMDL
ALF

Barry Rodgers

TABLE OF CONTENTS

ABSTRACT

The purpose of this project was to create a robot suitable for studying various machine intelligence algorithms. The robot senses the world through several types of sensors. Bump sensors detect collisions with obstacles. IR receivers are used to detect obstacles at a distance to avoid. Photocells are used determine which direction to turn to move toward or away from a light source. Another photocell mounted on top of the robot determines if the robot is in a shadow or not. The robot's behaviors are collision detection, dynamic collision avoidance, light following, and hiding in shadows. These behaviors are used to demonstrate habituation to stimuli, and adaptation to the environment.

EXECUTIVE SUMMARY

The purpose of this project was to create a robot suitable for studying various machine intelligence algorithms. The motivation for this project came from taking EEL5840 last semester. Because I did not have a robot at the time, I had to use a TJ to do the Q-Learning project for that course. Although the TJ is a nice robot, it was too limited in memory and sensors to be very successful at this task. Therefore, I wished to create a robot which would be suitable for machine intelligence research. I named my robot ALF which stands for Artificial Life Form.

ALF uses the 68HC11 EVBU board coupled with the ME11 expansion board. ALF has 32 kB of RAM. The robot uses the same body as a TALRIK. Three bump switches are mounted on the front of the robot to detect collisions in the frontal hemisphere. Four IR receivers are used to detect obstacles to the front and sides of the robot. Two photocells measure light intensities to the front left and front right of the robot. Finally, a third photocell detects the light level directly above the robot. All eight analog inputs are utilized. For locomotion, two servos drive the wheels.

The robots behaviors are collision detection, collision avoidance, light source hunting, light source avoiding, hiding in shadows. These behaviors are used to demonstrate habituation to stimuli, and adaptation to the environment.

INTRODUCTION

The name of my robot is ALF. The purpose of the ALF project was to create a robot suitable for studying various machine intelligence algorithms. The motivation for this came after taking EEL5840 last semester. Because I did not have a robot at the time, I had to use a TJ to do the Q-Learning project for that course. Although the TJ is a nice robot, it was too limited in memory and sensors to be very successful at this task.

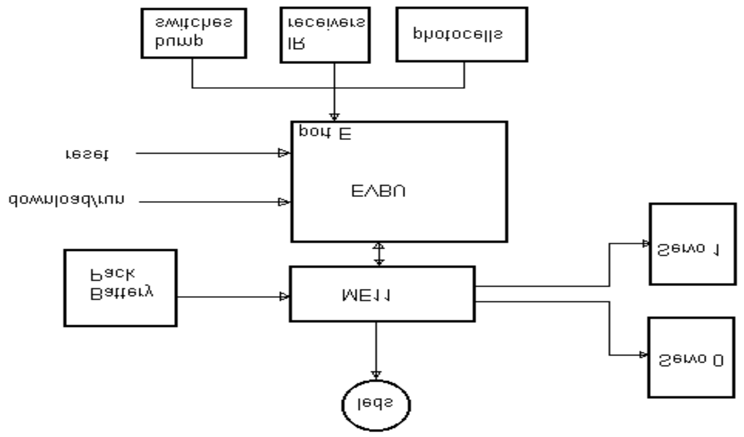
The following were the initial objectives for this project:

- Collision Avoidance
- Light Avoidance/Attraction
- Sound Avoidance/Attraction
- Learning (includes habituation, Q-learning, adaptation to environment).

This paper will discuss the integrated system of the robot. The platform, actuation, sensors, and behaviors will also be discussed in detail. The results from the project will also be presented.

INTEGRATED SYSTEM

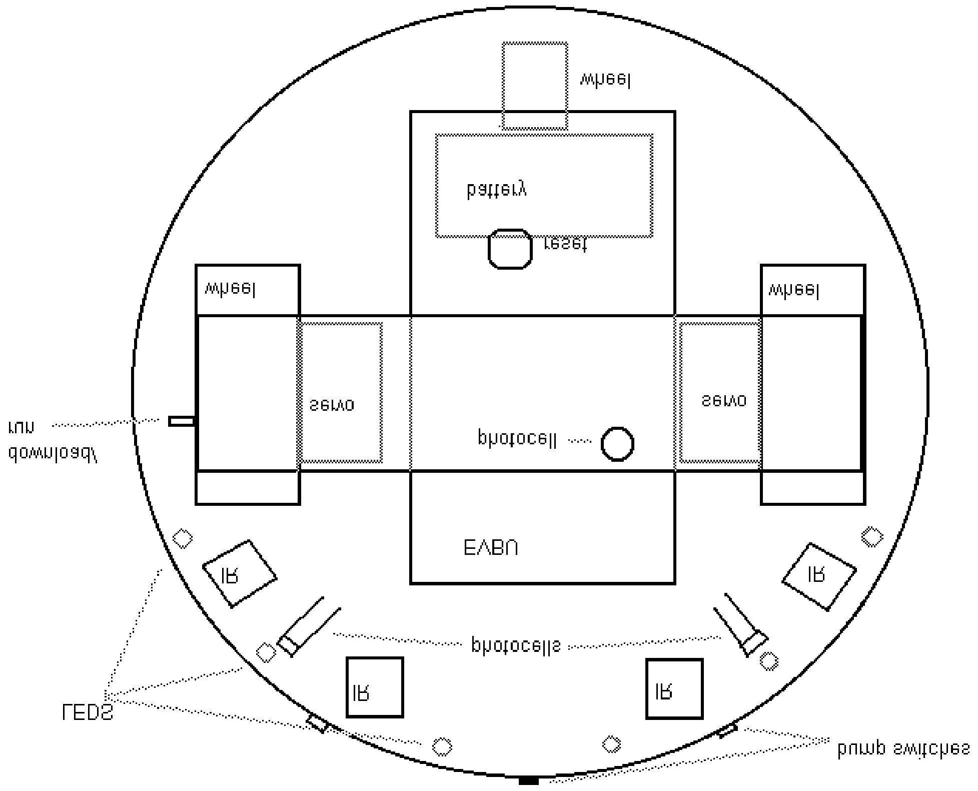
The following block diagram shows how ALF's systems interconnect.



Motion is controlled by the servos which are driven through the ME11. In software (IC is used in this project), they are controlled with the command *motor(n,F)*; where n is 0 or 1 and F is a floating point number between 0.0 and 100.0. The LED's are controlled by the port at 0x7000. These signals are modulated by the ME11 so that the Sharp IR receivers can see the reflections. The command to turn on the IR LED's is *poke(0x7000,0xff)*;. The IR receivers, bump switches, and photocells are read in at port E on the EVBU board. To read a sensor value in software, the command *analog(n)*; is used where n is the pin of port E the sensor is wired to (n ranges from 0 - 7). The download/run switch enables the robot to be put in bootstrap mode so that the IC program can be loaded. The reset button resets the HC11 processor.

MOBILE PLATFORM

ALF's platform is the same platform as a TALRIK robot. The following figure shows a top view of the platform. Note that lighter lines indicate that the line is beneath an object.



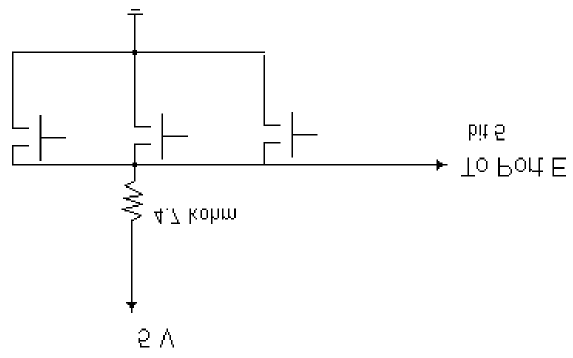
ACTUATION

The primary actuators on ALF are the two servos which drive the motors. The servos were hacked to allow full 360 degree movement. This involved removing the electronics, and cutting off a plastic tab that restricted movement.

SENSORS

Bump Switches

Three bump switches are mounted on the front of the robot (see Figure 2). The way the bumper is designed, these three switches detect almost any collision in the robots front hemisphere. The three bump switches are wired to a single 4.7 kΩ resistor pulled up to 5 V. This signal is input to port E (**analog(5)**). When the bump switch is pressed, **analog(5)** returns 0. Otherwise, it returns 255. The following figure shows how the bump switches are wired.



IR Receivers

The robot uses four Sharp IR receivers to achieve collision avoidance. The IR’s originally gave only a digital output, so they were hacked to allow analog output. The receivers are wired to Port E. The leftmost IR (**LLIR**) is sent to bit 0. The left middle IR (**LIR**) is sent to bit 1. The right middle IR (**RIR**) is sent to bit 3. The rightmost IR (**RRIR**) is sent to bit 4.

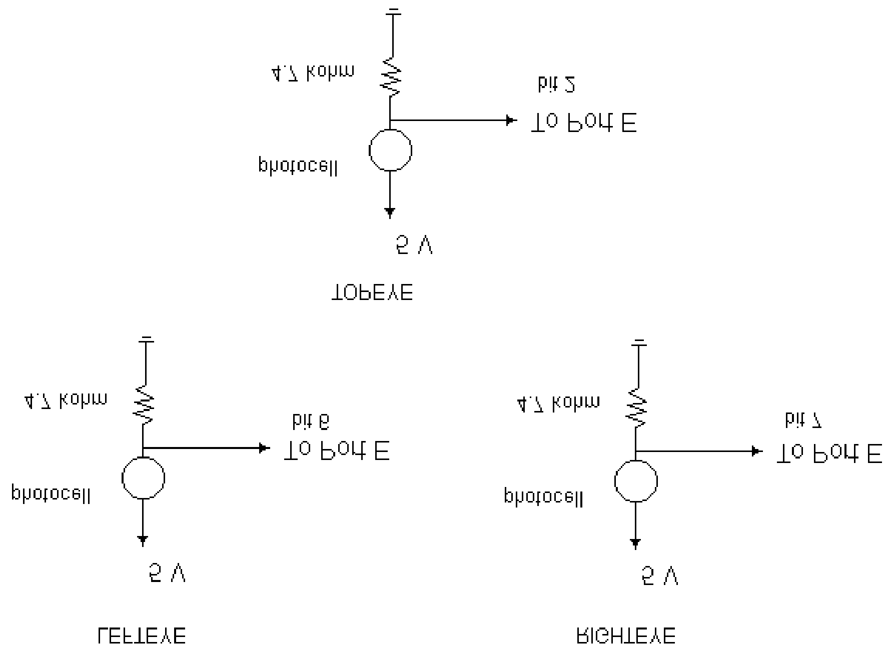
The greatest difficulty in dealing with the IR's, is that the type of surface reflecting the IR greatly affects its output. In general, the IR's performance can by can be characterized by the figure on the next page, and by the table below.

- No signal 85
- Object Far (> 3') 95
- Object Close (about 1')...........120
- Very Close (almost touching)...130

I experimented with a variety of positions for the IR's. A problem with some locations was that IR reflected off of the bumper which ruined the sensitivity of the receiver. The position shown if Figure 2 is the configuration that was finally settled on.

Photocells

The photocells are very easy to integrate with the system. Three photocells were used. One faces 45 degrees to the left and is referred to as LEFTEYE in software. The second faces 45 degrees to the right and is called RIGHTEYE. The third is mounted on the bridge on top of the robot and faces up. This photocell is called TOPEYE. The following figure shows the circuit diagram for the photocells.

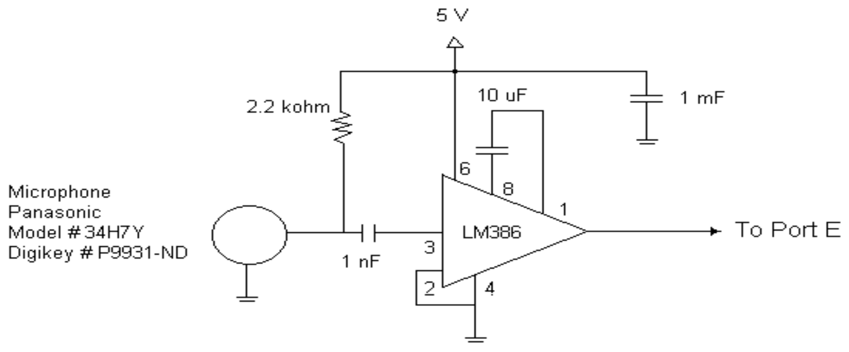


The way in which the photocells are wired makes the voltage proportional to the light level. The brighter the light, the higher the voltage. Here are some qualitative results.

Bright room	200
Not so bright	150
Dark (under a table)	90
Hand covering sensor	40

Microphone Sensor

This part of the project was very disappointing. I had intended to have the robot be able to move toward or away from sound sources. The idea for this sensor came from the book *Mobile Robots: Inspiration to Implementation*. The following figure shows the wiring diagram for this sensor.



As wired, the signal at the input to the LM386 should be amplified about 200 times. I thought that this would be enough. Unfortunately, this circuit did not work as expected. The circuit was wired up on a proto board for testing. The circuit showed no response at all to normal sounds in the room. Only by physically striking the microphone could I see any noticeable change in the output. While this might make for a good bump sensor, this was not what the sensor was designed for. This was at a time during the semester where I needed to have a stable platform to write code for. Therefore, I made the decision to discontinue development of this sensor. It would have taken a lot more work to get the sensor operational and there wasn't time.

BEHAVIORS

Collision Detection

When a bump is detected, the robot will back away from the obstacle. Before backing, the two center IR's are checked to estimate on which side of the robot the obstacle is on. If the obstacle is thought to be to the right, the robot turns to the left while backing. If the obstacle is probably on the robot's left side, the robot turns to the right while backing.

Collision Avoidance

This is what I spent most of the time coding for this project. Although I started out by using a rule base approach with a multitude of *if-then-else* statements, I wanted to do a more elegant solution. I was very interested in the presentation given last semester by Estela Bicho on the dynamic approach to collision avoidance. My collision avoidance program is based on the paper she presented called "The dynamic approach to autonomous robotics demonstrated on a low level platform."

The way the dynamic collision avoidance program works is as follows:

- Estimate the distance to obstacle in front of each IR (*d(n)*).
- Calculate repulsive force of each: *force(n) = b1* exp [-d(n)/b2]*.
- Weight the force of the middle IR's greater than the side IR's. Forces on opposing sides of the robot have opposite signs.
- Sum all of the weighted forces.
- Use the result to modify the turning rate.

The hard part is determining the values of *b1, b2,* and the weights. I determined these values through trial and error. There is a fine line between the robot ignoring its environment and the robot spinning uncontrollably in the middle of the room. Another problem area is estimating the distances. I used a linear assumption (see IR response figure) in doing this even though it is known that this is only an approximation. It made the calculations easier and seemed to work well. I was very pleased with the results when using the dynamic approach.

Light Following

The difference between the values returned by the left photocell and the right photocell is calculated. This number is then scaled and added to the calculations for the motor values. These values are added on top of the values previously calculated by the collision avoidance routine. The weights for these forces had to be chosen so that the light seeking behavior didn’t overwhelm the collision avoidance.

I had originally intended to have darkness seeking as a behavior. The problem is that darkness doesn't radiate. That is, these are no sources of darkness. Therefore, although the robot can detect a lamp on the opposite side of the room, it can't detect a shadow under a table from a distance. The only time this routine worked well was when the robot straddled a shadow so that one of the photocells was under the shadow and the other was not. When this happened, the robot would turn into the shadow. Another problem was that when the robot got close to a dark wall, the dark seeking behavior could cause the robot to turn toward the wall, risking a collision. Therefore, in the final program, when the robot is looking for a dark place, all it does is perform collision avoidance until the top sensor indicates it is in a dark area.

Shadow Hiding

The shadow hiding behavior is very simple. When the top photocell's value drops below a certain level (indicating that the robot is in a shadow), the robot stops. This behavior is mainly used to demonstrate the habituation.

Habituation

The robot hides in the shadow until the light level increases. At this point, I enable light following behavior to simulate the robot being curious about the source of the light. After a certain amount of time, the robot will become uninterested in the light source, and will avoid obstacles until another shadow is found. This behavior is modified by habituation. If the robot is repeatedly exposed to the light, it will spend less time trying to find the light. After the robot becomes habituated, if the light is removed for an extended period of time, the robot will once again respond as if the light is a new experience. This simulates dehabituation. This habituation behavior is difficult to design because if it is subtle (as in nature) it is difficult to demonstrate in a timely manner.

Learning

The learning done by the robot is very simple. If the robot is in a collision, it adjusts several variables (affecting speed and force calculations) to make a collision less likely in the future. While this robot would also be ideal for Q learning, Q-learning would have not been practical with all the behaviors I am modeling. However, the robot is able to adapt to its environment.

CONCLUSION

What was accomplished?

ALF's dynamic collision avoidance routines allow it to navigate through its environment in a natural manner. The light following behavior was successfully integrated with the collision avoidance. The habituation works well although it is very deterministic at this point. I would like to add some randomness to the habituation to make the robot's responses less predictable.

Future Work

I would like to expand the analog port so that more sensors can be added. I would also like to experiment with Q-learning and Fuzzy Logic. I am most interested in getting the robot to behave like a simple biological organism.

Final Thoughts

This was a valuable project. It was satisfying to be able to implement in an actual robot, some of the concepts I had learned in the Machine Intelligence classes.

REFERENCES

Bicho, Estela. Gregor Schoner. “The dynamic approach to autonomous robotics demonstrated on a low-level vehicle platform.” Robotics and Autonomous Systems 21. (1997), p. 23-35.

Martin, Fred G. The 6.270 Robot Builder’s Guide. 2nd Ed. 1992.

Jones, Joseph L., Anita M. Flynn. Mobile Robots: Inspiration to Implementation. A K Peters, Massachusetts, 1993.

APPENDIX   Program Code

```
/*          The ALF program. Features dynamic collision avoidance, shadow hiding, light following
/*          and habituation.  */

float LEFT_M,LEFT_MB,LEFT_MP,LEFT_MI= 0.0;
float RIGHT_M,RIGHT_MB,RIGHT_MP,RIGHT_MI,SPEED,SLIMIT;
int TOPEYE,LEFTEYE,RIGHTEYE,LLIR,LIR,RIR,RRIR,BUMPER,URGENT,HIDE=0,HITS;
float K,SPEEDL;
float D0,D1,D3,D4,BETA2,L0,L1,L3,L4;
int SHADOWTHRESH;
int REACTIONLIGHT=255,LLOFFSET,LOFFSET;
int SAFE=1;

/* This routine assigns variables values based on sensor readings. */
void read_sensors()
{
            while(1){
                        poke(0x7000,0xff);
                        LLIR = analog(0)-85;
                        LIR = analog(1)-85;
                        TOPEYE = analog(2);
                         RIR = analog(3)-85+LOFFSET;
                         RRIR = analog(4)-85+LLOFFSET;
                        BUMPER = analog(5);
                        RIGHTEYE = analog(6);
                        LEFTEYE = analog(7);
                        poke(0x7000,0x00);
          defer();
            }
}

/*          This routine describes the robots light behavior.  The robot performs collision avoidance
/*          until it finds a shadow.  Once in the shadow, it becomes still.  When light is put on
/*          the robot, it will move toward the light.  The strength of this reaction is determined
/*          by simulated habituation. */

void light_behavior()
{
      while(1){
          if(HIDE == 0){

              light_follow();
              if(seconds() > (45.0/((float)HITS + 1.0))){
                    HIDE=1;
                                    }
              SAFE=0;
                        }

          else if(SAFE == 1){
              if((TOPEYE > SHADOWTHRESH) | | (LEFTEYE > 200) | | (RIGHTEYE > 200)){
                                                HITS=HITS+1;
                                                reset_system_time();
                                                HIDE = 0;
                                                SAFE = 0;
                                    }
                                    else if (seconds() > 10.0){
                                                reset_system_time();
                                                HITS=HITS-1;
                                                if(HITS < 0){HITS=0;}
                                    }
                        }
                        else{
                        LEFT_MP = 0.0;
                        RIGHT_MP = 0.0;
                                    SPEEDL=100.0;
              if (TOPEYE < SHADOWTHRESH) {
```

```
                                        SAFE=1;

                                }
                                else {

                                        SAFE=0;
                                }

                        }
            defer();
        }
}

void light_follow()
{
        SPEEDL=100.0 -(float)RIGHTEYE/5.0 - (float)LEFTEYE/5.0;
        LEFT_MP = (float)(RIGHTEYE-LEFTEYE)/2.0;
        RIGHT_MP = -LEFT_MP;
}

void collision_avoid()
{
                while(1){
                        if (BUMPER < 100){
                        SLIMIT = SLIMIT - 1.0;
                        BETA2 = BETA2 + 1.0;
                        URGENT=50;
                        if(LIR > RIR){
                                LEFT_MB = -40.0;
                                RIGHT_MB = -90.0;
                                }
                        else {
                                LEFT_MB = -90.0;
                                RIGHT_MB = -40.0;
                                }
                        }
                  else {

            /* Estimate distance in front of each IR */
                   D0 = 16.0 -  (float)(LLIR)/3.0;
                   D1 = 16.0 - (float)(LIR)/3.0;
                   D3 = 16.0 - (float)(RIR)/3.0;
                   D4 = 16.0 - (float)(RRIR)/3.0;

           /* Calculate repulsive force generated by each IR */
                   L0 = 20.0*10.0*exp(-D0/BETA2);
                   L1 = 60.0*10.0*exp(-D1/BETA2);
                   L3 = 60.0*10.0*exp(-D3/BETA2);
                   L4 = 20.0*10.0*exp(-D4/BETA2);

           /* Sum forces to alter direction */
                    LEFT_MI = L0 + L1 - L3 - L4;
                   RIGHT_MI = -LEFT_MI;

            }
              defer();
              }
}

void drive_motors()
{
                float OML=0.0, OMR=0.0;
                while(1)
                {
                   if(URGENT==0){
                        SPEED = SLIMIT - (float)LIR - (float)RIR;
                        if(SPEEDL < SPEED){
                                SPEED = SPEEDL; }
                  LEFT_M = SPEED + LEFT_MI;
                  RIGHT_M = SPEED + RIGHT_MI;

                  LEFT_M = LEFT_M + LEFT_MP;
                                    if(LEFT_M>100.0){LEFT_M=100.0;}
                                    if(LEFT_M< -100.0){LEFT_M = -100.0;}

                  RIGHT_M = RIGHT_M + RIGHT_MP;
                                    if(RIGHT_M>100.0){RIGHT_M=100.0;}
                                    if(RIGHT_M<-100.0){RIGHT_M=-100.0;}

                  if(SAFE == 1 && HIDE==1){
                                               LEFT_M=0.0;
                                               RIGHT_M=0.0;
                                               }
                                    OML = (LEFT_M + (K * OML))/(K+1.0);
                                    OMR = (RIGHT_M + (K * OMR))/(K+1.0);
                                    motor(0,OML);
                                    motor(1,OMR);
                        }
                        else{
                                    motor(0,LEFT_MB);
                                    motor(1,RIGHT_MB);
                                    URGENT=URGENT - 1;
                                    }
                        defer();
              }
}


void main()
{
              HITS=0;
      K=3.0;
              SAFE=0;
      SLIMIT=80.0;
      BETA2=30.0;
      HIDE=1;
      LEFT_MP = 0.0;
      RIGHT_MP = 0.0;

      poke(0x7000,0xff);
      SHADOWTHRESH=analog(2)-50;
      LLOFFSET=analog(0)-analog(4);
      LOFFSET=analog(1)-analog(3);
      poke(0x7000,0x00);

              start_process(read_sensors());
              start_process(collision_avoid());
              start_process(drive_motors());
      start_process(light_behavior());

}
```