The Claw

Katherine Meiszer
EEL 5666
Intelligent Machines Design Laboratory
Spring 1997
Instructor: Prof. Keith L. Doty
April 28, 1997

TABLE OF CONTENTS

ABSTRACT

This paper describes the work that I have completed on The Claw, the autonomous agent that I designed this semester in the Intelligent Machines Design Laboratory. My goal in designing The Claw was to design a mobile autonomous agent capable of independently locating, collecting, and delivering to a marked depository appropriately sized objects within its environment. In order to move about independently within its environment it had to be capable of performing collision avoidance behaviors. In order to be able to locate appropriately sized objects it had to be capable of some type rudimentary vision. To be able to pick up objects the robot needed to be equipped with a gripping arm. In order to be able to deliver the objects, it had to be able to "see" and recognize the beacon marking the deposit site. These properties of the robot were realized with infrared (IR) detectors and emitters for simple vision and a gripper arm design for object manipulation.

EXECUTIVE SUMMARY

The Claw uses Motorola's MC68HC11E9 EVBU universal evaluation board as its processor. Connected to the EVBU is a ME11 expansion board that contains 32K of RAM, motor drivers, output ports and other circuitry.

The Claw has a roughly rectangular body constructed on ¼ inch plywood. Two servos that were altered to act as DC motors and fitted with wheels provide the means for locomotion. They are situated at the front on the platform. A caster provides stability for the rear of the platform.

The robot performs collision avoidance with the use of IR emitters and detectors. The emitters (IR led's) emit IR light which will bounce off obstacles in the robot's path. The detectors detect this IR light and produce a signal voltage whose magnitude corresponds to the amount of IR detected. The higher the voltage, the closer the robot is to the obstacle.

The robot also has bump sensor that detects when the robot accident runs into an object. It consists of a bumper that sticks out beyond the tip of the claw. When that bumper is hit, it depresses a microswitch. The robot detects this and can be programmed to respond accordingly.

Locating objects is performed in a similar way to collision avoidance. The IR led's are collimated and lensed. The object detection system has three of these led's paired with their detectors, one pair mounted on each side of the claw and one in the middle. The lensing of the led's ensured that very little light from one led reaches another led's detector. The robot uses these emitter/detector pairs to center objects in front on its claw.

The gripper arm of the robot is mounted via a hinge on the front of the platform. The arm can be raised and lowered via a cord attached to the end that runs up through an eye bolt to a spindle that is attached to the end of a servo. A claw is located at the end of the arm, which can be opened and closed via a second servo. A break-beam sensor is mounted across the claw that alerts the robot when there is an object in its claw.

INTRODUCTION

At the beginning of this project, I was interested in experimenting with manipulation of objects by autonomous agents. I was specifically interested in enabling the robot to locate objects independently. I decided to create a sort of scavenger robot. My goal in this project was to create an autonomous agent capable of moving independently through its environment. Its purpose would be to acquire any objects it came across that were of a size that could fit in its claw. Once it acquired the object, it would search for the deposit site, which would be marked by a beacon, and deposit the item. Then the process would repeat itself. This report will describe how I realized these goals. It will describe the designs for the robot's platform, gripping arm, and sensors. It will describe the means of control used to move the robot and its gripping arm. It will also describe the controlling code used to enable to robot to make decisions based on its sensor data and to carry out those decisions.

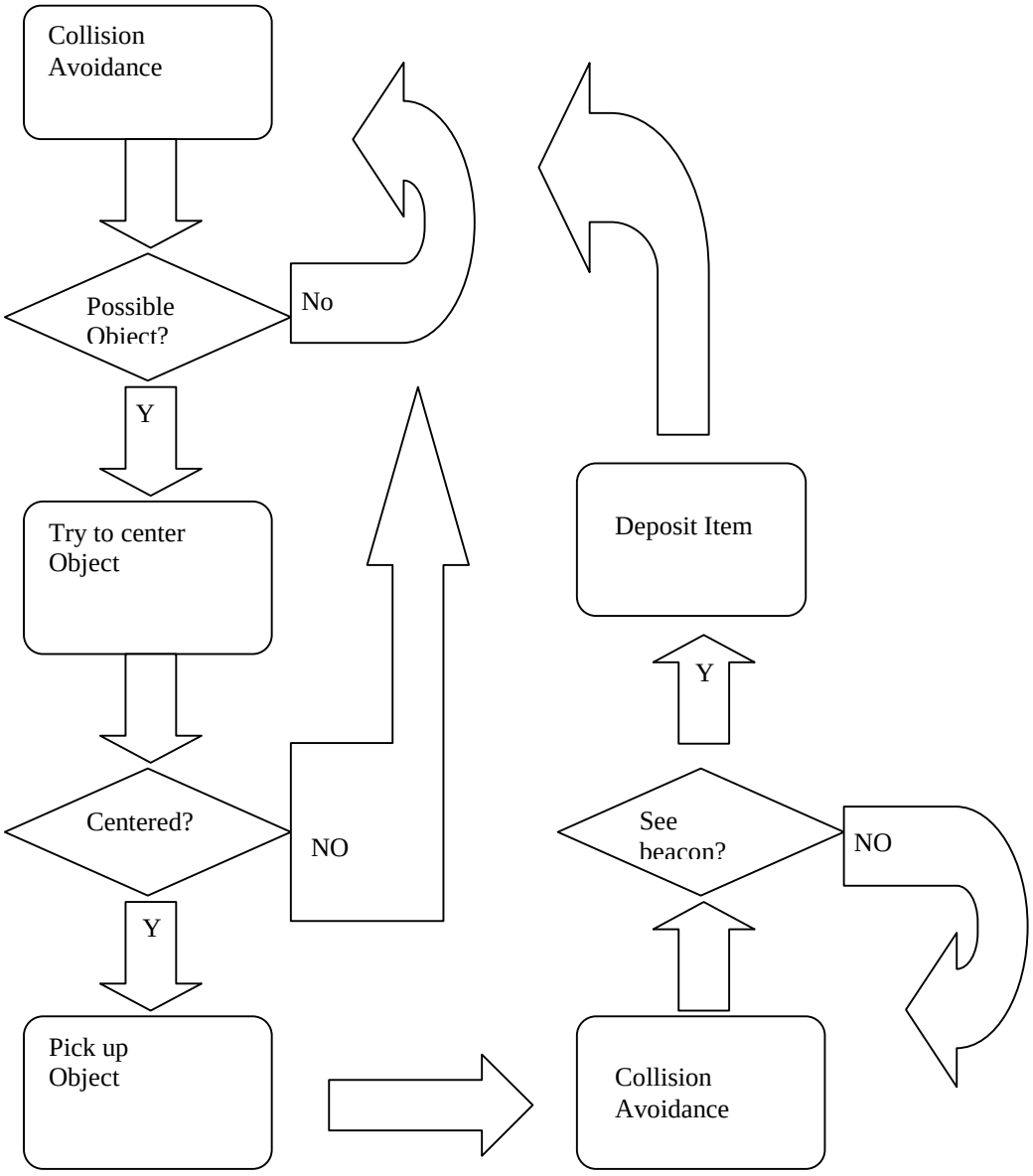
INTEGRATED SYSTEM

The robot is assembled on a mobile platform with a gripping arm and claw attached. The arm can be raised and lowered and the claw can be opened and closed via two servos. The two front wheels of the robot are mounted on servos hacked to act as dc motors. The direction and speed of these motors can be controlled directly through software.

Initially, the robot moves about its environment in a random pattern while performing obstacle avoidance. If its sensors detect an object that may be suitable for picking up, the robot turns towards the object and tries to center it in front of its claw. If

it is successful in centering it, it will move forward and pick the object up. If not, it will move on and continue doing collision avoidance until it finds another object. Once it picks the object up, it continues doing collision avoidance until it finds the beacon marking the deposit site. It then deposits the object. The process then starts all over. A flowchart showing the robots operation sequence is shown in figure 1.

Figure 1



MOBILE PLATFORM

The platform of the robot is constructed of ¼ inch plywood in is a roughly rectangular shape. Two wheels, which are driven by dc motors, are located at the front of the platform. A caster, centered in the rear, provides rear support. A large section was cut out of the middle of the front of the platform for the gripping arm to be mounted. A servo is mounted on the platform to raise and lower the arm. Another servo is mounted underneath the arm that opens and closes the claw. The EVBU board is mounted in the center of the platform. Two bump sensors extend in front of the claw on either side of it. The bump sensors allow the robot to detect when it has bumped into an object that the collision avoidance system missed. The structure is shown in figure 2.

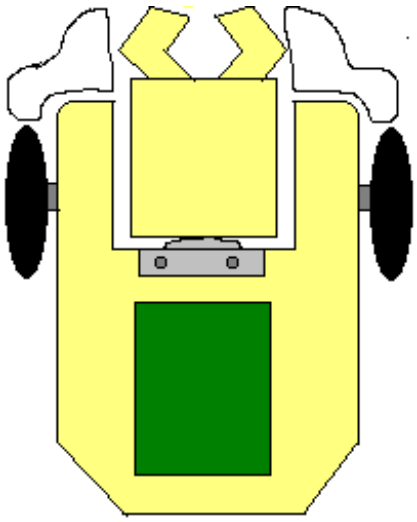


Figure 2

ACTUATION

Motors. The wheels are controlled by servos hacked to act as dc motors. The hacking process involves opening the servo case and removing the controlling circuitry and

potentiometer. Then power and ground wires are soldered to the motor. The gear with the stop must be removed, the stop clipped off, and the gear smoothed and replaced. Then replace the servo case. The servo now acts as a dc motor and its speed and direction can be controlled with PMW waveforms generated from the microprocessor.

Raising and Lowering the Arm. The arm is raised and lowered by a servo with a spindle attached. A cord is wrapped around the spindle. The cord runs up through an eyebolt and is attached to the end of the arm. The servo raises and lowers the arm by turning the spindle. The advantage of this is that by decreasing the radius of the spindle, the mechanical advantage of the system can be increased. This servo was also partially hacked. The potentiometer was pulled out of the servo and then reattached to the controlling circuitry by longer cable. The pot is mounted next to the gripping arm. A rod is attached to the pot and rests on top of the arm. As the arm moves up and down, it moves the rod, which turns the pot. Therefore the exact position of the arm can be controlled by the servo. This hack was developed and suggested to me by Kevin Harrelson, a teaching assistant for the IMDL course.

Opening and Closing the Claw. Each side of the claw is attached to the underside of the arm and is glued to a gear. The two gears interlock so that the two sides must turn together but one clockwise and the other counterclockwise. A servo with a gear attached is mounted on the underside of the arm. The gear interlocks with one of the two gears attached to the sides of the claw. This servo can therefore open and close the claw.

Actuation Algorithms. The robot is programmed using interactive C (IC), a language developed specifically for use with the EVBU board for robotics applications. The motor speed and control and servo position can be controlled directly with motor and servo commands. The motors and servos are programmed to react directly to sensor input. When sensors detect an obstacle or bump sensors detect a collision, the motors are programmed to respond to turn the robot away from the obstruction. When sensors detect and object that might be suitable for acquisition, the motors respond to try to turn the robot so that the object is centered in front of the claw. They then move the robot straight forward towards it. When the break-beam sensor is tripped, the servos respond by closing the claw and raising the arm. When the robot has an object and encounters the deposit site beacon, the servos respond by lowering the arm and opening the claw.

SENSORS

The Break-Beam Sensor The break-beam sensor is a Hamamatsu S4285-40. It drives its own infrared led and can distinguish the light from this led from any other infrared source that may be present. It consists of four pins, which are the output pin, the ground pin, the pin connected to the cathode of the led, and the power pin. The break-beam sensor and its led are mounted facing each other on opposite sides of the claw. The output pin is connected to an analog pin. Its voltage goes from zero to five volts when the sensor can no longer see its led. By monitoring the analog pin, the robot can detect when an object has moved into its claw. A circuit diagram of the break-beam sensor is presented in figure three.

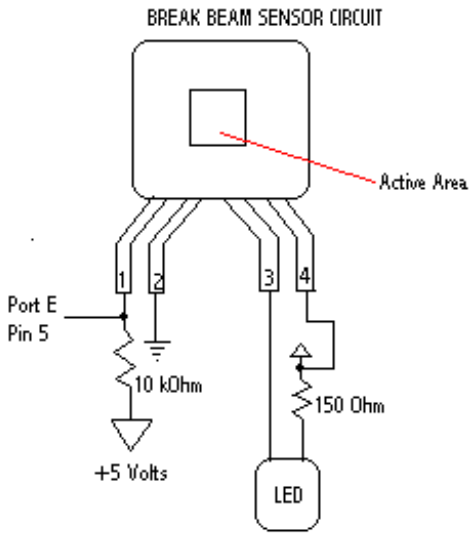


Figure 3

Collision Avoidance The Claw has five sets of IR detectors and emitters. The IR leds emit 40 kHz IR light and the detectors are sensitive to 40 kHz IR and output a voltage on their output pin corresponding to the strength of the IR that they detect. Four of the five sets are used for collision avoidance: the two on the outside and the two on either side of the claw (see fig. 5). The IR emitted from the leds bounces off of obstacles in the robot's path and is picked up by the detectors. Each detector output pin is connected to an analog pin on the EVBU. The robot can thus be programmed to turn away from the direction a certain detector is pointing towards when the voltage on that detectors output pin rises above a certain threshold.

Collision Detection There are some obstacles that the collision avoidance sensors will miss. That is where that bump sensors come in. Each bump sensor is a microswitch. They are attached to the body of the robot as shown in fig. five. All the switches are attached to one analog pin. The three switches on the left side of the robot are connected

in parallel between +5 volts and a 10kOhm resistor. The resistor is connected to the analog pin. The three switches on the right are connected in the same way but with a one kOhm resistor. This resistor also goes to the analog pin. The analog pin is also connected through a 4.7kOhm pull-down resistor to ground. The result is a voltage-divider circuit. When a switch on the left side on the robot is depressed it produces a different analog voltage than if a switch on the right side was depressed. A wire is mounted in front of the switch so that when the robot collides with something one or more of the switches on that side is depressed. By monitoring the voltage on the analog pin, the robot can detect a collision and determining on which side it occurred. The bump sensor circuit is shown in Figure 4.

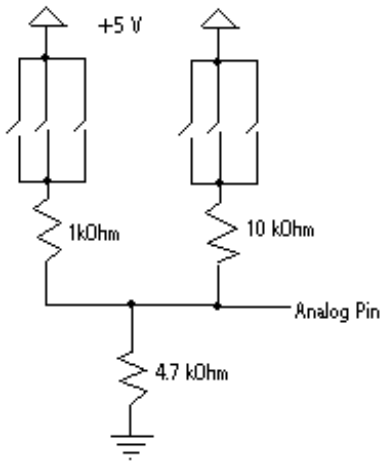


Figure 4

Object Detection. The robot must be able to locate objects it can pick up. This is done with the three IR detectors and collimated emitters mounted on the claw of the robot with lenses mounted in front of the led's to obtain as narrow a beam as possible. When one or two of the IR detector output pins goes above a certain threshold, but at least one does

not, then the object being detected is possibly an object the claw can pick up. The robot is programmed to try to maneuver itself so that the center detector's output is significantly higher than the other two. If it succeeds in doing this it can move forward until its break-beam sensor is tripped and then pick up the object. The robot also has one IR detector that detects 32.7 KHz IR, the frequency of the beacon that marks the deposit site. When the robot see this beacon, it deposits the object it is holding.

Figure 5 shows the layout of sensors on the body of the robot:

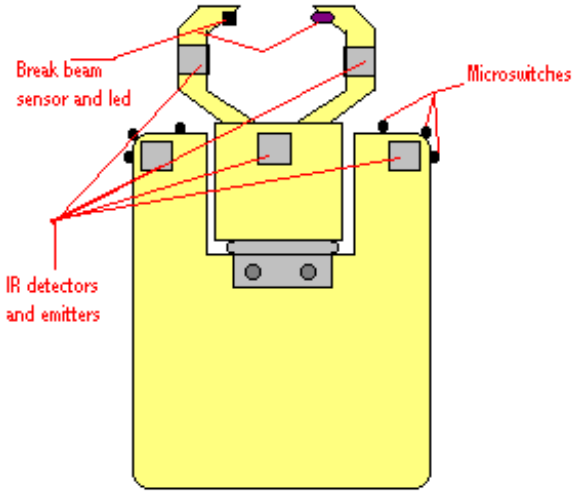


figure 5

BEHAVIORS

The robot has five behaviors: collision avoidance, collision detection, searching for objects, picking up objects, and object depositing.

Collision Avoidance. The robot uses the two IR detectors on the outside plus the two on either side of the claw for obstacle avoidance. If the output voltage on one of the two outside detectors goes above a certain threshold, the robot turns away from that direction.

If the output voltage of both of the two outside detectors on the claw goes above a certain threshold, the object in the robot's path is too large to be picked up and is therefore an obstacle. The robot backs up and then turns away.

Collision Detection. If the collision avoidance system fails to detect an object, the robot will collide with the object and the bump sensors will be activated. The robot will back up and turn away from the side on which the bump was detected.

Searching for Objects. The three IR emitter and detector pairs on the gripping arm are used for object detection. When one of the outside pairs detects an object but the other does not, the object is possibly of a size that will fit in the robot's claw. The robot will turn itself to try to position itself so that it see the object with the center IR pair but not so much with the two side pairs. If it succeeds in doing this it moves forward until the break beam sensor is tripped. If it fails it moves on continuing to do collision avoidance.

Picking Up Objects. When the break-beam sensor is tripped, the robot responds by closing its claw and raising its arm. It then resumes collision avoidance until it finds the deposit beacon.

Object Depositing. When the robot sees the beacon marking the deposit site while it is holding an object, it will stop, lower its arm, and open its claw. After it has released the object, it will back up and move away. It will then resume collision avoidance behaviors until it finds another potential object to pick up.

EXPERIMENTS

I performed two experiments to determine the effectiveness of the lensed and collimated leds and their detectors for detecting objects in the robot's path. In the experiments I placed an object in front of the claw and took IR reading from all three detectors. I repeated this step for distances from 0 to 7 inches from the claw in increments of ½ inches. In the first experiment, the object was placed in front of the center detector (IR2). In the second experiment, the object was places in front of the right side detector (IR3). IR4 is the left side detector. The results of the two experiments are shown in figures 6 and 7.

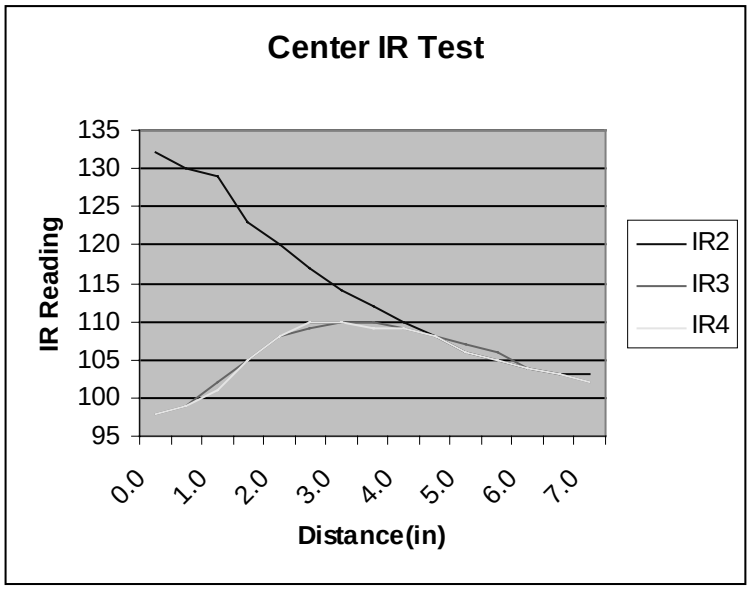
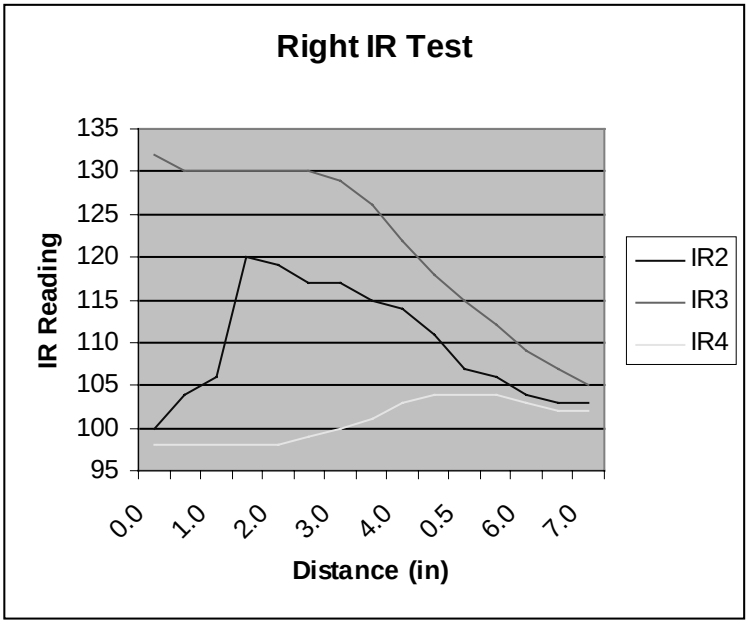


Figure 6

Figure 7



CONCLUSION

The robot is very successful at locating objects in its path as long they are not located outside of the line of sight of the two outside lensed leds. If the object falls within their line of sight the robot is very successful at turning towards then and maneuvering them into its grip. However the range of the robot's vision is limited. It cannot discern possible objects from a distance further than about 3 ½ or 4 inches. As a future project it would be interesting to employ a different type of robot vision that might have better range and make searching for objects more efficient. It would also be useful to program the robot to do some sort of search pattern instead of relying on random luck.

APPENDIX A

Program code:

```
int beam,left_ir, right_ir, center_ir1, center_ir2, center_ir3, thresh;
int bump,bump_l,bump_r,bump_prio;
int signal,beacon_l,beacon_r,beacon_prio;
int a_l, d_l, a_r, d_r;
int avoid_r,avoid_l;
int grip_r,grip_l;
int search_l,search_r;
int avoid_prio,grip_prio,search_prio,prev_prio,drop_prio;
float serv1,serv2;
float value1=70.;
float value2=65.;
float drop_val1,drop_val2,val1,val2;
float time;
int set;

void main()
{
  poke(0x7000, 0xff);
  thresh = 110;
  d_l=50;
  d_r=50;
  a_l=50;
  a_r=50;
  beacon_prio=0;
  servo_on();
  servo_deg1(value1);
  servo_deg2(value2);
  start_process(grip());
  start_process(raise());
  start_process(servo_cont());
  start_process(sensor());
  start_process(avoidance());
  start_process(arbitrate());
  start_process(motors());
  start_process(search());
  start_process(bump_cont());
  start_process(beacon());
}


void sensor()
{
```

```
  while(1)
  {
    left_ir=analog(6);
    right_ir=analog(7);
    center_ir1=analog(2);
    center_ir2=analog(3);
    center_ir3=analog(4);
    beam=analog(5);
    bump=analog(0);
    signal=analog(1);
    defer();
  }
}


void arbitrate()
{
  while(1)
  {
    if(bump_prio==100)
    {
     d_l=bump_l;
     d_r=bump_r;
    }

    else if(drop_prio==100)
    {
     defer();
    }

    else if((grip_prio == 100) && (beacon_prio==0))
    {
     d_l = grip_l;
     d_r = grip_r;
    }

    else if((search_prio==100) && (beacon_prio==0))
    {
     d_l = search_l;
     d_r = search_r;
    }

    else if(avoid_prio==100){
     d_l = avoid_l;
     d_r = avoid_r;
    }
```

```
    defer();
  }
}

void motors()
{
  while(1)
  {
    if(((grip_prio == 0) && (search_prio == 0) && (bump_prio == 0))
       || ((beacon_prio == 100) && (drop_prio == 0)))
    {
      a_l = ((9*a_l) + d_l)/10;
      a_r = ((9*a_r) + d_r)/10;
    }
    else {
      a_l = d_l;
      a_r = d_r;
    }
    if(drop_prio==100)
      defer();
    else
    {
      motor(0,a_l);
      motor(1,a_r);
    }
    defer();
  }
}

void avoidance()
{
  while(1)
  {
   if((signal>thresh) && (beacon_prio==0))
    {
     avoid_l=-50;
     avoid_r=-50;
     wait(500);
     avoid_l=0;
     avoid_r=-50;
     wait(1000);
     avoid_l=50;
     avoid_r=50;
    }

    else if (center_ir2>thresh && center_ir3>thresh)
```

```
    {
      avoid_prio=100;
      avoid_l=-50;
      avoid_r=-50;
      wait(500);
      avoid_r=0;
      wait(1000);
      a_l=0;
      a_r=0;
    }
    else if ((left_ir<thresh)&&(right_ir < thresh))
    {
       avoid_prio=100;
       avoid_l = 50;
       avoid_r = 50;
    } else if ((left_ir>thresh)&&(right_ir<thresh))
    {
       avoid_prio=100;
       avoid_l = 50;
       avoid_r = 0;
    } else if ((left_ir<thresh)&&(right_ir>thresh))
    {
       avoid_prio=100;
       avoid_l = 0;
       avoid_r = 50;
    }
    else {
      avoid_prio=0;
      avoid_l=50;
      avoid_r=50;
    }
    defer();
  }
}

void grip()
{
  while(1)
  {
    if(analog(5) < 200)
    {  val2=65.;
       grip_l=50;
       grip_r=50;
       grip_prio=0;
    }
```

```
    else if(analog(5) > 200)
    {
      grip_l=0;
      grip_r=0;
      grip_prio=100;
    }

    if((d_l==0) && (d_r==0) && (grip_prio==100))
      val2=90.;

    defer();
  }
}

void raise()
{
  while(1)
  {
    if (val2==65.)
      val1=70.;
    else
      val1=60.;
    defer();
  }
}

void servo_cont()
{
  while(1)
  {
    if(drop_prio==100)
    {
      serv1=drop_val1;
      serv2=drop_val2;
    }
    else
    {
      serv1=val1;
      serv2=val2;
    }

    if(serv1 < value1)
      value1--;
    else if(serv1 > value1)
      value1++;
    if(serv2 < value2)
```

```
      value2=serv2;
    else if(serv2 > value2)
      value2=serv2;
    servo_deg1(value1);
    servo_deg2(value2);
    defer();
  }
}

void search()
{
 while(1)
 {
  if(center_ir2>120 && center_ir3<107)
  {
   search_prio=100;
   motor(0,0);
   motor(1,0);
   time=0.;
   reset_system_time();
   while(center_ir2>110 && time<3.)
   {
    time=seconds();
    search_r=25;
    search_l=0;
   }
   search_prio=0;
  }
  else if(center_ir3>120 && center_ir2<107)
  {
   search_prio=100;
   motor(0,0);
   motor(1,0);
   time=0.;
   reset_system_time();
   while(center_ir3>110 && time<3.)
   {
    time=seconds();
    search_l=25;
    search_r=0;
   }
   search_prio=0;
  }
  else if(center_ir1>110 && center_ir2>110 && center_ir3<100)
  {
   search_prio=100;
```

```
      motor(0,0);
      motor(1,0);
      time=0.;
      reset_system_time();
      while(center_ir2>110 && time<3.)
      {
        time=seconds();
        search_r=25;
        search_l=0;
      }
      search_prio=0;
    }
    else if(center_ir1>110 && center_ir3>110 && center_ir2<100)
    {
      time=0.;
      reset_system_time();
      while(center_ir3>110 && time<3.)
      {
        time=seconds();
        search_l=25;
        search_r=0;
      }
      search_prio=0;
    }
    else
    {
      search_prio=0;
    }
  }
}

void bump_cont()
{
  while(1)
  {
   if((bump >= 80) && (bump <= 82))
   {
     bump_prio=100;
     bump_l=-50;
     bump_r=-50;
     wait(500);
     bump_r=0;
     bump_l=-50;
     wait(500);
   }
   else if(((bump>=210)&&(bump <= 211)) || ((bump >= 214)&&(bump<=216)))
```

```
   {
    bump_prio=100;
    bump_l=-50;
    bump_r=-50;
    wait(500);
    bump_l=0;
    bump_r=-50;
    wait(500);
   }
   bump_prio=0;
   defer();
  }
}

void beacon()
{
  while(1)
  {
    if(grip_prio==0)
    {
     beacon_prio=0;
     prev_prio=0;
    }
    else if(grip_prio==100)
    {
     if(prev_prio==0)
     {
      wait(1000);
      prev_prio=100;
     }
     beacon_prio=100;
     if((signal>130) && (signal< 140))
     {
      drop_prio=100;
      beacon_l=0;
      beacon_r=0;
      drop_val1=70.;
      drop_val2=65.;
      motor(0,-50);
      motor(1,-50);
      wait(500);
      motor(0,0);
      motor(1,-50);
      wait(1000);
      motor(1,50);
      motor(0,50);
```

```
        wait(100);
        drop_prio=0;
      }
    }
    defer();
  }
}
```