University of Florida

Department of Electrical Engineering

EEL 5666

Intelligent Machines Design Laboratory

KOCATKA: A Flying Robot

Paul J Taylor

Instructor: Dr. Antonio A. Arroyo

April 24, 1998

TABLE OF CONTENTS

ABSTRACT

Following is a discussion of the development of an obstacle avoidance and wall following system for an autonomous helium blimp using infrared emitters and sensors. The blimp reacts when the sensor determines there is an obstacle in the blimp's path. The program in ICC11 initiates changes in the propulsion system based on sensor readings. Directional control problems unique to blimps are presented. A possible sonar system is presented. A cost analysis is shown with a comparison to the average cost of terra-based robots. The closing discussion addresses future endeavors which appear likely as a result of the present experimentation.

EXECUTIVE SUMMARY

Kocatka (pronounced ka SOT ka), which is Russian for killer whale, is an autonomous blimp which is designed to navigate large areas at a human walking pace. The blimp is neutrally bouyant, so no lift is necessary from the motors. Two propulsion fans supply speed and directional control of the blimp. The component parts are minimal so as to make the blimp neutrally bouyant.

Kacotka's main navigation system is IR. A sonar system was studied, but because of weight restrictions and problems making the sensor stable, sonar was not used. The IR provides 2 feet of sensing information, which is sufficient for the blimp avoiding obstacles. A bump sensor indicates when the robot strikes an object head on.

A microprocessor provides a means of control for the system including the motors, IR emission and sensing, and bump sensor. ICC11 programmed functions for Kocatka to turn right, turn left, move forward, and move in a fast forward mode. The main program provides guidelines which determine's Kocatka's reactions to external stimuli. The program also implements a simple wall following behavior.

INTRODUCTION

My main purpose for taking Intelligent Machines Design Laboratory (IMDL) was the realization of an autonomous helium-filled blimp, name Kocatka. The physical design was of utmost importance because of the small amount of lift provided by the helium. The design also has to be flexible to permit for variations of air temperature and consistency. Kocatka's main behavior is obstacle avoidance. The original design was to use a sonar sensor. Since the sonar sensor was component rich and unstable, the basis was changed to infrared. The autonomous blimp also has the behavior of wall-following.

Following is a description of Kocatka separated into 9 sections. *Integrated System* discusses the overall relationships between individual components of the entire system. *Platform* discusses the physical construction of the gondola and blimp. Actuation reviews the power supply as well as the basic propulsion system. The section, Sensors, covers all the systems that were considered for use on Kocatka. Behaviors reviews each behavior. Cost Analysis covers the total cost of Kocatka and the comparison to average robots. Conclusion summarizes the successes and restrictions of this project along with possible future work. Documentation provides a list of references. Vendors details the vendors that supplied the various parts for Kocatka. Appendix A contains program code for Kocatka's behaviors, while Appendix B contains program code for testing the sensors and motors.

INTEGRATED SYSTEM

Any robot is a sum of its parts. The largest part of Kocatka is the blimp bag, shown in FIGURE 1. The helium filled bag provides the lift to offset the weight of the electronics and body weight. Several propulsion techniques exist. I implemented two motors placed 45^{o} from the center plane.

I used the MTJPRO11 board, produced by Novasoft, which contains a Motorola 68HC11 microprocessor, an extra 32K of ram, IR emitter modulation circuitry, bump circuitry, and voltage regulation. Some added circuitry was needed to run the motors. The devices controlled by the microprocessor were an IR emitter, IR detector, bumps sensor, and two motors with 3" propellers. The blimp, once power is applied, senses the environment and responds to stimuli through the programming.

MOBILE PLATFORM

Kocatka consists of two major parts, the blimp bag and the gondola. The blimps bag is 7' long and 2' round. The total amount of helium is approximately 28 cubic feet. This amount of helium allows for an additional 12 ounces of lift beyond the blimp bag weight. A picture of the blimp bag is shown in following photo.

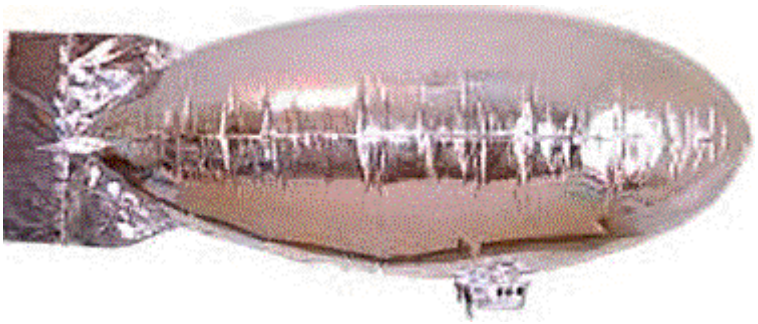

Fig 1: Blimp Bag

I made several versions of the gondola. The first one was too heavy. The final version of was made of balsa wood and styrofoam. A picture is shown below.

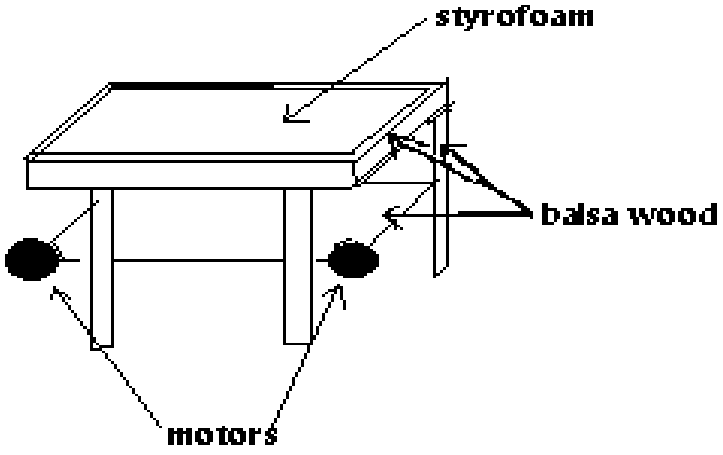

Fig 2: Gondola Construction

The motors were placed at a 45 degree angle from the center line of the gondola. One motor is on the left and one on the right. Originally a third motor was placed in the back and run in reverse. I removed this motor because of weight restrictions.

Components such as the microprocessor, batteries, and components are placed on top of the styrofoam. These components were held in place by velcro.

I attached the gondola to the blimp with string rope. The gondola had eyelet loops to hold the string. On the blimp, I placed cut stripes of a drinking straw attached to the blimp with standard masking tape. The string rope went through the drinking straws and connected to the other side of the gondola.

ACTUATION

Battery Power

Kocatka originally had an 8-cell, 9.6 V NiCd battery pack for power. This was changed because of weight restrictions to a 6-cell, 7.2 V NiCd battery pack. Voltage was regulated on the MTJPRO11 board for 5V for the microprocessor. The run time is estimated at over twenty minutes.

Motors

Kocatka used two ¼ Oz. motors with 3" propeller. Both motors operated in the forward position. Typical current draw for these motors is 1 amp at 4.8 volts.

Motor Drivers

To interface between the motors and the microprocesser, some circuitry was necessary. A TIP 120 was used. The schematic is shown below.

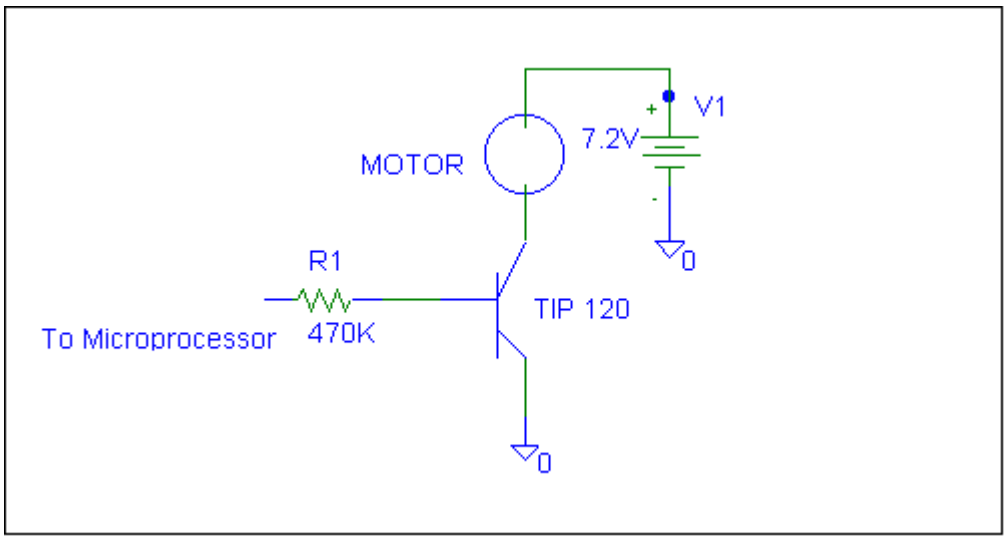

Fig 3: Motor Driver

SENSORS

Sonar

Originally Kocatka was to have a sonar system for sensing. Sonar has good distance capabilities, up to 25 feet. The schematic is shown below.

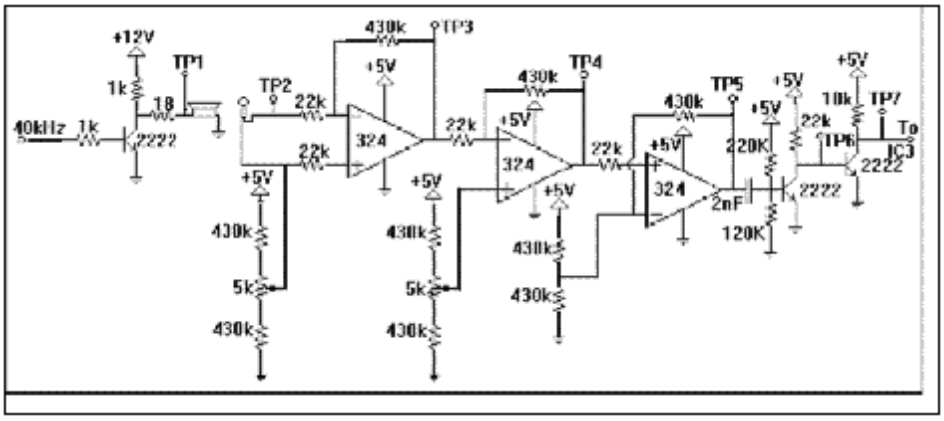

Fig 4. Sonar Schematic

This schematic was taken from Kelly Thomas robot Thomas. I could not achieve the same results as Mr. Thomas nor could I make the system stable. Finally, because of weight restrictions, I chose to change to infrared.

Infrared

I used the standard IR configuration used in the MIL lab. I could only implement one IR because of weight restrictions. The leads to the IR emitter and detector were both 3' feet long so they could be attached to the front of the blimp bag with velcro. Below is some data taken to characterized the IR sensor.

Distance	Analog Reading
0.2	129
0.28	122
0.36	111
0.44	105
0.52	103
0.6	101
0.68	100
0.76	99
0.84	98
0.92	98
1	97
>1	97

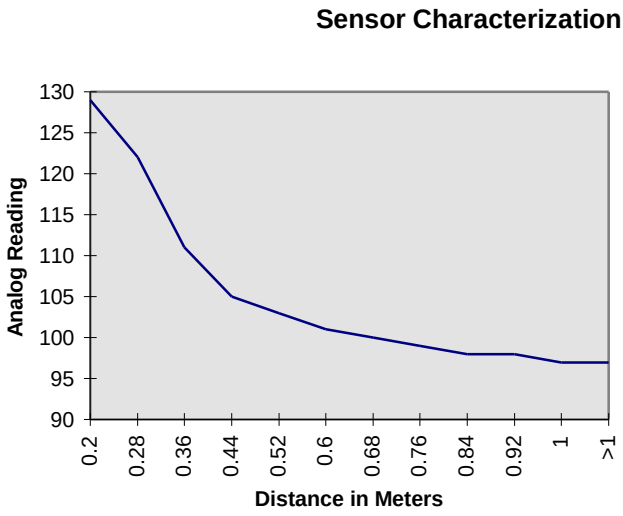

Fig 5. IR Data

Bump Sensor

I implemented a bump sensor on Kocatka. The bump sensor consisted of a microphone and some circuitry to make an output reasonable for analog port of the microprocessor. The circuit was normally high (5V). When a strong breadth or hard bump hit the microphone, the output level went to 0V. The circuit is as below.

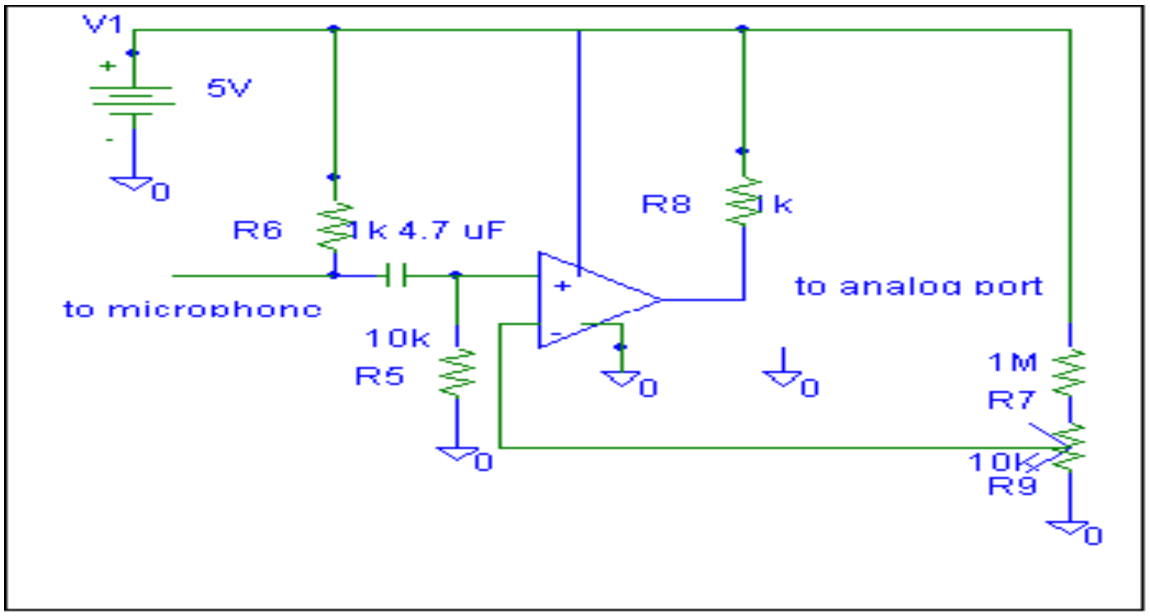

Fig 6. Bump Sensor Schematic

BEHAVIORS

Collision Avoidance

Kocatka's main behavior is collision avoidance with monitors the infrared sensor and uses basic manuevering to avoid collisions with objects. The microprocessor can call functions to turn left, right, go forward, and go fast forward.

Wall Following

I tried to implement a simple wall following routine in Kocatka's program. This was done be modifying some of the collision avoidance code. Also, the air currents in the New Engineering Buidling helped with this problem also.

COST ANALYSIS

I wanted to prepare a cost analysis as cost is probably the most important factor in any real project in the real world. All the receipts for anything purchased for the robot were kept and cataloged in an Excel spreadsheet. Below are the total cost for Kocatka.

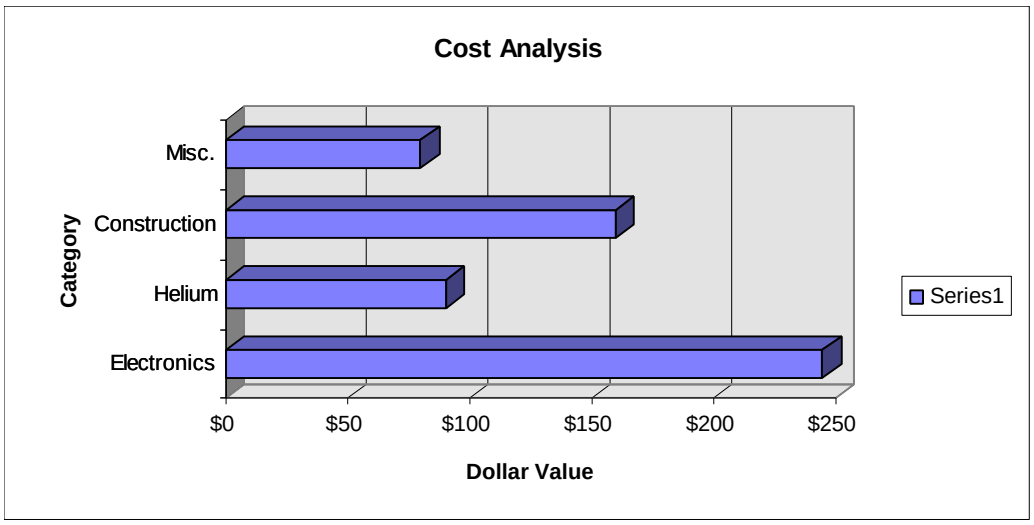

Fig7. Cost Analysis

The miscellaneous category includes tools. Some of the cost was due to parts not use, but this of course is part of any endeavor.

The total cost of Kocatka was approximately $575.00. This is nearly $300 more than the cost of a normal robot in IMDL. Part of the extra cost can be attributed to the blimp bag and helium. The other portion of the cost comes from the need for specialized parts (that is very low weight) for the blimp.

CONCLUSION

Kocatka is a fully autonomous blimp robot which successful avoids objects and follows walls. Propellers enable the robot to move left, right, and forward. The infrared sensor allows Kocatka to sense the environment and act according. The MTJPRO11 microprocessor allows for control of the robot and future expansion. The wall follow behavior gives Kocatka a peaceful and serene behavior.

A major consideration of an helium-filled vehicle is weight. Even with a fairly large bag, the largest the vendor could supply, the weight restriction was 12 ounces. This severely limited the amount of sensors that could be placed on Kocatka.

For future work, I would like to make a sensor better than IR yet light enough to place on the system. Next time, I would not make a gondola. It added extra weight. The gondola was helpful this time for testing. A good next step would be to implement an up and down function for Kocatka.

DOCUMENTATION

Doty, Keith L. TJ-PRO Assembly Manual, MIL webpage, January 29, 1998.

Doty, Keith L. TJ-PRO Users Manual, MIL webpage, January 29, 1998.

Jones, Joeseph & Flynn, Anita, Mobile Robots: Inspiration to Implementation, A.K. Peters Publishers, Wellesley, MA, 1993.

Miller, Bruce and Qaiyumi, Aamir, Holly: An Autonoumous Air Cushion Vehicle, MIL webpage, December 9, 1996.

Piri, John. Gondola Blimp Construction Plans, http://www.ridgecrest.ca.us/~jpiri, May 12, 1996.

Thomas, Kelly, Thomas, MIL webpage, 1995.

VENDORS

Radio Shack
3315 SW Archer Road
Gainesville, FL 32608
352-375-2426

West Coast Blimp & Electronics
713 Cottonwood Drive
Ridgecrest, CA 93555
http://www.ridgecrest.ca.us/~jpiri

Electronics Plus
2026 SW 34th Street
Gainesville, FL
352-371-3223

Wal-Mart
3570 SW Archer Road
Gainesville, FL
352-378-0619

Lowes
3500 SW Archer Rd
Gainesville, FL
352-376-9900

The Party Place (Helium)
3712 W. University Ave.
Gainesville, FL. 32601
352-371-4646

APPENDIX A

```
// This program will run the blimp robot
// KOCATKA
//
// VERSION 1.0
//
// Paul J Taylor
//
// April 24, 1998
//
//

#include <mil.h>
#include <hc11.h>
#include <servotjp.h>
#include <serial.h>
#include <analog.h>
#include <vectors.h>
#include <timux03.h>

#include <stdio.h>
```

```
////////////////////////////////////////////////
//Important defines
////////////////////////////////////////////////
#define IR *(unsigned char *) 0x7000 /* address of IR emitters */
#define  WAIT   for(inner_wait_loop =0 ; inner_wait_loop < 12000 ; inner_wait_loop++)
; //do nothing
#define DELTATIME 2 //wait one ~5th second between actions
#define BREAK for(inner_wait_loop = 0; inner_wait_loop< 150; inner_wait_loop++) ;
#define GO_FORWARD    SET_BIT(PORTA,136);
#define GO_FFORWARD   SET_BIT(PORTA, 168);
#define TURN_RIGHT    SET_BIT(PORTA,40);
#define TURN_HRIGHT   SET_BIT(PORTA,8);
#define TURN_LEFT     SET_BIT(PORTA,160);
#define TURN_HLEFT    SET_BIT(PORTA,128);
#define STOP          CLEAR_BIT(PORTA,65535);
#define EYE           center = analog(1);
////////////////////////////////////////////
// Global Variables
////////////////////////////////////////////

//variables associated with timux
unsigned mseconds, dseconds, seconds, minutes;
```

```
int center;
int counter;
int random;
int min;

////////////////////////////////////////////
//Prototypes
////////////////////////////////////////////

////////////////////////////////////////////
void events(void);
void dseconds_events(void);
void seconds_events(void);
void minutes_events(void);
void forward(void);
void left(void);
void right(void);

int
main(void)
{
        unsigned inner_wait_loop;
```

```
   init_servos();
   init_analog();
   setbaud(BAUD9600);
   init_timemux();
   srand(TCNT);

IR=0x89;

STOP;
min = 95;
WAIT;
center = 0;

   while(1)
            {

      EYE;

      if (center <  108){
                  forward();
                 }
      else if (center < 120) {
                  //do some random stuff here.
```

```
                    random = rand();
                    random = (random/327);
                    if (random > 15)
                        {while (center > 108)
right();
                          {right();
                           EYE;
                          }
                        }
                    else
                        { while (center > 108)
                         {left();
                          EYE;
                         }
                        }
                   }
            else
                {
                 while (center > 120)
                   {STOP;
                    EYE;
                   }
                }
```

```
            }


}


void forward(void)
{
  int inner_wait_loop;
  int i;
  for (i =0; i < 10 ; i++)
      {  GO_FORWARD;
         BREAK;
         STOP;
         BREAK;
      }
}
void left (void)
{
  int inner_wait_loop;
  int j;
```

```
  for (j = 0; j < 10; j++)
      {  TURN_HLEFT;
         BREAK;
         STOP;
         BREAK;
      }
}
void right (void)
{
  int inner_wait_loop;
  int k;
  for (k = 0; k <10; k++)
      {  TURN_HRIGHT;
         BREAK;
         STOP;
         BREAK;
      }
}
```

APPENDIX B

TESTING CODE

```
//

// This program will run the blimp robot

// KOCATKA

// DIREC.C

// VERSION 1.0

//

// Paul J Taylor

//

// April 24, 1998

//

//


#include <mil.h>

#include <hc11.h>

#include <servotjp.h>

#include <serial.h>

#include <analog.h>

#include <vectors.h>

#include <timux03.h>
```

```
#include <stdio.h>

////////////////////////////////////////////////
//Important defines
////////////////////////////////////////////////
#define IR *(unsigned char *) 0x7000 /* address of IR emitters */
#define  WAIT   for(inner_wait_loop =0 ; inner_wait_loop < 12000 ; inner_wait_loop++)
; //do nothing
#define DELTATIME 2 //wait one ~5th second between actions
#define BREAK for(inner_wait_loop = 0; inner_wait_loop< 200; inner_wait_loop++) ;
#define GO_FORWARD    SET_BIT(PORTA,136);
#define GO_FFORWARD   SET_BIT(PORTA, 168);
#define TURN_RIGHT    SET_BIT(PORTA,40);
#define TURN_HRIGHT   SET_BIT(PORTA,8);
#define TURN_LEFT     SET_BIT(PORTA,160);
#define TURN_HLEFT    SET_BIT(PORTA,128);
#define STOP          CLEAR_BIT(PORTA,65535);
#define EYE           center = analog(1);
////////////////////////////////////////////
// Global Variables
////////////////////////////////////////////

//variables associated with timux
```

```
unsigned mseconds, dseconds, seconds, minutes;
int center;
int counter;
int random;
int min;

////////////////////////////////////////////////
//Prototypes
////////////////////////////////////////////////

////////////////////////////////////////////////
void events(void);
void dseconds_events(void);
void seconds_events(void);
void minutes_events(void);
void forward(void);
void left(void);
void right(void);
int
main(void)
{
        unsigned inner_wait_loop;
```

```
    init_servos();
    init_analog();
    setbaud(BAUD9600);
    init_timemux();
    srand(TCNT);

  IR=0x89;

  STOP;
  min = 95;
  WAIT;
  center = 0;
Int
main(void)
{
  unsigned inner_wait_loop;

  init_servos();
  init_analog();
  setbaud(BAUD9600);
  init_timemux();
  srand(TCNT);
```

```
        IR=0x89;


        STOP;
       WAIT;
        center = 0;
        while(1)
              {
                 forward();
               right();
               left();
              }


}



void forward(void)
{
  int inner_wait_loop;
  int i;
  for (i =0; i < 10; i++);
      {  GO_FORWARD;
         BREAK;
```

```
        STOP;
        BREAK;
      }
}

void left (void)
{
  int inner_wait_loop;
  int j;
  for (j = 0; j < 10; j++)
      {  TURN_HLEFT;
        BREAK;
        STOP;
        BREAK;
      }
}

void right (void)
{
  int inner_wait_loop;
  int k;
  for (k = 0; k <10; k++)
      {  TURN_HRIGHT;
```

```
        BREAK;
        STOP;
        BREAK;
      }
}

//ANALOG definitions
// Left IR is 2
// Right IR is 3
// Bumper is 0

//State is a function of IR

//-----------------------------------------
//Routines for ti-mux
//
//         variations on routines by E. Yim
//  use for timing routines....
//
//-----------------------------------------


//Main event handler
```

```
void events(void)
{
	/* Put all the msecond-level events here */
	if (++(mseconds) == 100)
	{
		mseconds = 0;
		dseconds_events();
	}
}
```

```
void dseconds_events(void)
{
	if (++(dseconds) == 10)
	{
		dseconds = 0;
		seconds_events();
	}
```

```
        /* Put all the dsecond-level events here */
}




void seconds_events(void)
{
        int i;


        if (++(seconds) == 60)
        {
                seconds = 0;
                minutes_events();
        }
        /* Put all the second-level events here */
        //blink_LED();
}




void minutes_events(void)
{
        if (++(minutes) == 60)
```

```
        {
                minutes = 0;
        }

        //change index in saving H-table delta

}
```