Kisa the Robotic Cat

Cynthia Manya

April 26, 2000

University of Florida
Intelligent Machines Design Laboratory

EEL 5666

TABLE OF CONTENTS

ABSTRACT

When I signed up for this class, I wanted to build something challenging but also interesting. Since I love animals, I decided upon building a small companion similar to that of the Sony Aibo dog. Kisa is designed to be a robotic pet that exhibits cat-like behaviors. For instance, it purrs when stroked at the neck (a favorite spot for most cats).

Other traits that I hoped to realize were enjoying attention (exhibited by a content meow when it hears its name) and obeying some commands (may sound like a typical trait to that of a dog, but there are cats out there that do obey commands). Of course, there are basic obstacle-avoidance characteristics to consider as well. I also have it look for a warm spot in a room and (ideally) take little cat-naps whenever it is bored. Perhaps later on I will incorporate behaviors that will make Kisa do tasks to ease the workload of humans, but at the moment it serves only as a robot that provides entertainment.

<u>Introduction</u>

Since this robot is designed to be autonomous, it needs various sensors to collect information on its surroundings and then use that information to make intelligent decisions. The sensors I used are for heat and contact. Also I incorporated light and IR proximity sensors to make Kisa's behaviors more realistic. Another feature that I hoped to realize is a taste for audio tones. If Kisa were to hear a soothing melody or an alarming noise, he would react accordingly. For this feature I would use a filter so that only certain frequencies exhibited in a song would generate a reaction. This is something that unfortunately was not realized in this semester, but I will leave it for further exploration in the future. Which brings me to one last feature that I realized: a limited form of communication. I wanted to have Kisa make sounds that tells people how he is currently feeling based on his surroundings. Currently he purrs and meows.

This report gives the documents the process that went into building Kisa. Therefore the reader will be aware of any mishaps as well as accomplishments that happened during the course of this project. All mechanical and electrical procedures is shown as well as all the programming. The programming code used is provided in the appendix.

<u>Integrated System</u>

All Kisa's sensor behaviors is actuated using a MC68HC11 microcontroller, which acts as his brain. The circuit board is the same one used in the TJPro robot, which was chosen because of its compact size. It also has 32 kbytes of SRAM, a total of 8 analog input channels, and I/O ports ready for use, making it convenient. For programming code, IC is used.

Since I wanted Kisa to walk, a total of three servos was used for each leg, making another microcontroller necessary for managing all twelve leg servos. This microcontroller is the single-chip board found in the TJ robot which can handle up to 16 servos. This microcontroller is used solely for the servos and is powered by its own battery pack so as not to drain the power needed for the sensor microprocessor.

The single-chip board was loaded with a program called superservo.sys written by Keith Doty. Once this program is loaded, all the other programming can be done on the TJPro (sensor) board. If the sensor board decides it wants a servo to be at a certain position, the proper command is sent via the SCI to the single-chip board, which understands the command and thus proceeds to set the servo to the desired position. Therefore the two boards are continually communicating with each other through the SCI port.

The cable used for this communication has only three wires: single-chip transmitter to sensor receiver (or vice versa), single-chip receiver to sensor transmitter (or vice versa), and single-chip ground to sensor ground. The other pins of the SCI port do not need to be connected so they are left out.

Mobile Platform

Kisa's body is designed on AutoCAD to give it an animal-like appearance. Much thanks goes to Megan Grimm for her AutoCAD drawings of the ROUS from which Kisa is based. Its head houses the IR detectors and emitters for basic obstacle detection. Also on the head are a precision thermistor with a Cadmium Sulfide (CdS) cell together acting as a nose. The body is sufficiently large enough to hold two microcontroller circuit boards (one for the servos, the other for the legs), two battery packs (again, one for the servos and the other for the sensor board), and the voice recognition and sound circuitry. This is a rather heavy load for the body, so therefore large-torque servos are used for the legs. The material used for all the body parts is wood due to its lightweight and sturdiness.

As noted before, each leg consists of three servos. Two 139 oz. in. FMA Direct S400 servos are used to simulate motion similar to what a knee or elbow would do, and a 40 oz. in. servo sits on top of these servos to act as a ball and socket joint (providing only one degree of motion around the z-axis). The FMA Direct S400 servos were chosen since it was known in advance that the weight of the body will altogether be on the heavy side, thus possibly making smaller-torque servos peel out unexpectedly. The "elbow" and "knee" servos are the ones that support the entire weight, so for the "ball and socket" pivot joint a relatively small 40 oz. in. servo can be used. The FMA Direct servos can only provide a maximum of 270 degrees of motion, but it is more than enough for this robot. To make the robot walk, extensive programming is needed to control the center of gravity so that balance is maintained.

Each leg has three degrees of freedom due to the three servos each holds—under 180 degrees of motion around the x-y plane and about 270 degrees of motion in the z-axis.

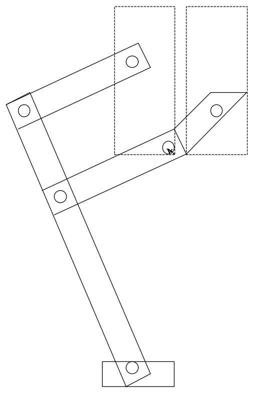

Figure 1 – Drawing of a leg. The dashed rectangles are meant to show how the leg is connected to the servos.

The body is given a sort of X-shape so that most of the robot's weight can be concentrated at the center. With all the circuitry and battery packs loaded on the body, the total weight comes to around 4.5 pounds.

Sensors

Below is a list of the sensors that are used (and some that were not able to be implemented) and how they fit into Kisa's behavior scheme.

IR detectors: Sharp cans are used for basic obstacle detection. IR emitters are placed as eyes on the head, and the detectors are directly underneath them. Once an obstacle is "seen" by these detectors, the sensor microcontroller will tell the servo microcontroller to move left or right accordingly. Port E pins 4 and 5 control the detectors.

Heat: A precision thermistor is used so that Kisa can (theoretically) roam around a room and stop when it finds a warm spot so it can take a nap. This behavior would occur when there is nothing for Kisa to do (when it is bored). The nose was decidedly the best spot to place the sensor. A voltage divider circuit (see figure 2) was needed for this sensor so that analog values can be outputted to the microprocessor. R2 in this case is 1.5 kΩ.

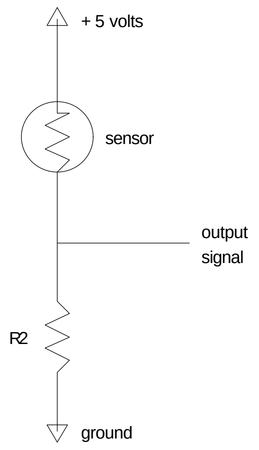

Figure 2 - Voltage divider circuit

Light: A CdS cell works in conjunction with the temperature sensor so that a "sunny" place can be seen more quickly. Since a sunny spot is brighter than the rest of the room, Kisa will know that spot is an opportune place to nap. The CdS cell is wrapped in a small tube (to give it a sense of direction) and placed in a socket right next to the thermistor. This sensor also utilizes a voltage divider circuit with R2 being 1.5 kΩ.

Touch: A pressure sensor is placed on the neck so that whenever a certain threshold is reached, the sound circuitry is activated by a relay and the robot purrs. Again a voltage-divider circuit is needed for this sensor with R2 at 47 kΩ.

Sensors not implemented-
Voice recognition: This is the sensor that is made using the HM 2007 chip bought from Images Co. This chip can learn forty 1.92-second words or phrases. A rather complicated algorithm is used to train the chip the new words, and it is described in the data booklet for this chip. This

chip uses a few pins of an input and output port of a processor to communicate. When the chip recognizes a word, a value is placed on the chip's output pins. This value can be read by the microprocessor using a latch and therefore different actions can be realized depending on that value. The words that would ideally be taught are:

- Stop – robot stops for an uncertain amount of time before resuming whatever it was doing
- Go – used after the stop command
- Kisa – robot understands its name and meows

Bump: Whiskers would be used to avoid objects that were not detected by the IR detectors. They will consist of a simple idea that another robot, Critter, utilizes. A metal wire is inserted into a long metal coil so that if these two meet, a short occurs and that data is decoded by the microprocessor so it knows an obstacle was hit. It will then turn left or right depending on which whisker was hit.

Behaviors

Using the sensors described above, Kisa is able to find a warm place to nap and generate sounds when petted. Whenever it finds a bright warm spot, it meows in contentment. Theoretically, it can also avoid collisions with the IR detectors.

In order to generate sounds, a stuffed-animal toy ("Catz") that meows and purrs was hacked. The voltage powering the sound circuitry needed to be 3.5 V since anything above that may fry the electronics. A LM 317T regulator was used for this purpose. A resistor value of R1 = 270Ω was initially chosen so R2 can be solved for by hand calculations. For R2, a 10kΩ pot was used so adjustments can easily be made if needed.

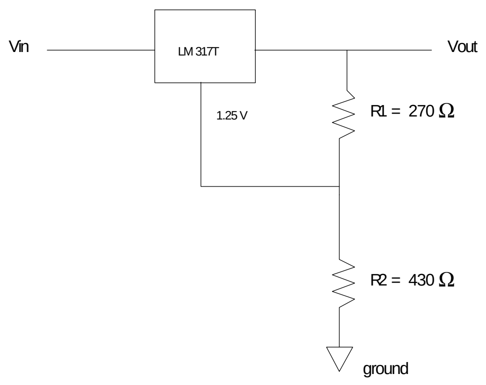

Figure 3 – Voltage regulator circuit.

For generating sounds, a SPST relay (bought at Radio Shack) was used so that the sound can be turned on or off by a digital port of the microprocessor.

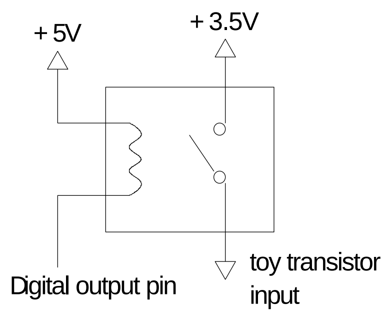

Figure 4 – Relay circuit.

Port D bits 4 and 5 acted as the switch that turns the sound on or off depending on the value that was in those bits (1 turns sound off, 0 turns it on). Below is a code segment that initializes port D to be a digital output (note that the hc11 and mil header files are needed in order for this code to work):

```
#include <hc11.h>
#include <mil.h>

void setdigport(void)
{
 SET_BIT(DDRD, 0x08);                    /* set PortD bits 3 & 4 as dig outputs */
 SET_BIT(DDRD, 0x10);

 SET_BIT(PORTD, 0x08);                   /* set PortD bits 3 & 4 to 1 (5 volts) */
 SET_BIT(PORTD, 0x10);
}
```

There are other various behaviors that I hope to look into for future work. I hope to have Kisa recognize frequencies with a filter to give it musical taste. This audio filter is a MAX266 chip, and it would give a clean signal once the original signal passes through the high- and low frequency bandpass filters, thus enabling Kisa to recognize melodies or loud sounds. Another idea is to use the whiskers and light detection sensors to find small dark spots to crawl into when it is afraid.

Experimental Layout and Results

These are the results for the IR detectors used in Kisa:

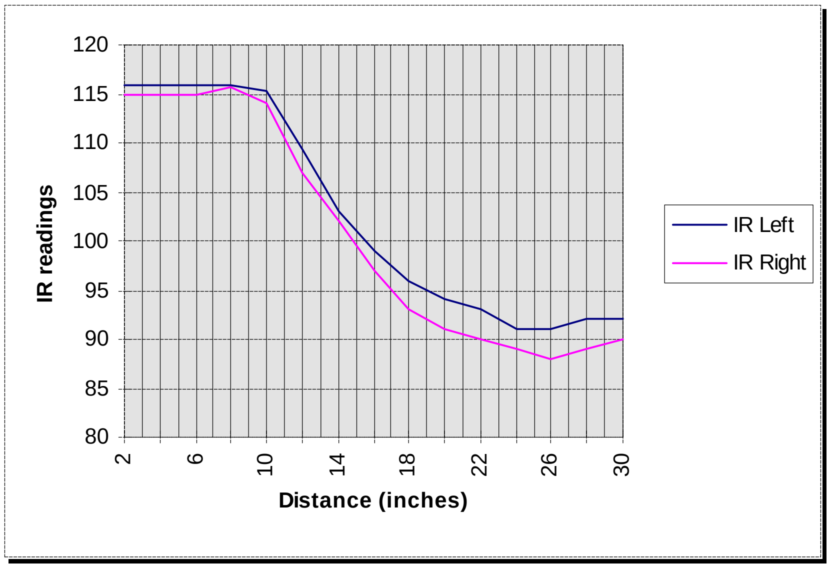

Figure 5 – IR readings based on distance.

As the IR detectors go beyond 30 inches, a linear pattern is seen. Therefore based on these readings, Kisa recognizes obstacles about 2 feet away and can then begin to decide which direction to turn. The direction of the turn would be determined randomly; also if there is a wall to the left as well as ahead of Kisa and it decides to turn left, its whiskers will let it know there is an obstacle there too.

For the touch sensor the range of values it gives are from about three to 250. A threshold of about 165 is used for Kisa to recognize someone touching it. A continuous value of 165 or higher means someone is petting it, so it purrs.

For the CdS cell and the thermistor, the range of values varied greatly depending on the resistor value used in the voltage divider circuit. Below is a table showing how the range of values changes with different resistors:

		CdS cell		Thermistor	
Resistor (Ω)	Black	White	Light	Room	Light
100	0	2	15	3	7
1k	3	13	100	24	30
6.8k	50	88	155	104	111
10k	50	111	170	126	132

For black, a black sheet of paper was placed underneath the sensor, and similarly for white. For light, a candle was placed in front of the sensors about 2 inches away.
The resistor used in both sensors' voltage divider circuits was 10kΩ since it had the greatest range in values.

<u>Conclusion and Future Work</u>

Many long hours of research and experimentation was a must, not to mention patience. Due to time and financial constraints, the robot was not completely realizable. It can stand and crouch but unfortunately not walk. Since the weight of the body is on the heavy side, the legs wobble far too much to be controlled. Knowing what I know now involving mechanics, I need to re-design the legs so that they are shorter and thicker.

I also noted that in order for Kisa to fully avoid obstacles, IR detectors or bump switches should be placed on the front paws so that objects under one foot high (the height at which the current IR detectors are placed) can be seen and avoided.

Even though I did not get to finish what I set out to accomplish, I will consider it an on-going project to continue working on as well as enhance in the future.

Documentation

Anita M. Flynn, Bruce A. Seiger, and Joseph L. Jones. *Mobile Robots: Inspiration to Implementation;* A.K. Peters, 1999.

Fred Martin. *The 6.270 Robot Builder's Guide*; 2nd edition, 1992.

Keith L. Doty. *TJPro Assembly Manual*; Mekatronix 1999.

Keith L. Doty. *TJ Assembly Manual*; Mekatronix 1999.

"HM 2007 Speech Recognition" data sheet, Images Co.

Acknowledgements

Along with the written documents I would like to thank everybody who gave their assistance:

-Megan Grimm for her AutoCAD drawings and insight

-The IMDL TA's, Scott Jantz and Ivan Zapata, for their suggestions and insight (not to mention patience)

-The students in MIL and IMDL who have helped and offered suggestions along the way

Sources for Parts

Mekatronix - www.mekatronix.com

FMA Direct – www.fmadirect.com

Images Co – www.imagesco.com

Lowes Home Improvement – 3500 SW Archer Rd. 352-376-9900

Radio Shack – Archer Rd. 352-375-2426

The Home Depot – 7107 NW 4th Blvd.   352-332-7440

Electronics Plus – 2026 SW 34th St.   352-371-3223

Michaels Arts and Crafts – 3644 SW Archer Rd.   352-377-9797

Bearings and Drives – 2540 NW 74th Pl.   352-375-0568

Appendices

Touch sensor code (used for test purposes on a TJ robot):

```
/****************************************************************************
 *                      MEKATRONIX Copyright 1998                           *
 * Title                touchtj.c                                           *
 * Programmer           Keith L. Doty                                       *
 * Modified by          Cynthia Manya                                       *
 * Date                 February 2000                                       *
 * Version              1                                                   *
 *                                                                          *
 * Description                                                              *
 *    A very simple collision avoidance program. TJ will read each IR detectors,   *
 *    and turn away from any obstacles in its path. Also, if something touches TJ's *
 *    touch sensor, it will stop moving until whatever touched it has gone away.    *
 *    The back bumper is also checked for any collisions.                   *
 ****************************************************************************/

/***************************** Includes **********************************/
#include <tjbase.h>
/************************ End of includes ******************************/

/**************************** Constants *********************************/
#define FORWL  2000
#define FORWR  4000
#define BACKL  4000
#define BACKR  2000
#define STOP   0000
/************************ End of Constants ****************************/

/*************************** Prototypes *********************************/
void turn(void);
/************************ End of Prototypes ****************************/

/***************************** Globals ***********************************/
int rspeed, lspeed;
/************************** End of Globals *****************************/


void main(void)
/****************************** Main ***********************************/
{
  int irdr, irdl;
  int touch;
  int i;
  init_motortj();
  init_analog();
  init_ir();

  DDRC = 0x00;          /* Set PORTC (bumper) for input */

  while(1)
  {

/* The following block will read the IR Detector (ird) ports, and decide */
/* whether TJ needs to turn to avoid any obstacles.  It also reads the touch sensor */
/* port and determines whether soembody is touching it */
    irdr = ir_value(6);
    irdl = ir_value(7);
    touch = analog(2);

    if (irdr > 100)
      rspeed = BACKR;
    else
      rspeed = FORWR;
    if (irdl > 100)
      lspeed = BACKL;
    else
      lspeed = FORWL;
          if (touch > 65)           ; A threshold of 65 is used in this case
            {
            rspeed = STOP;         ; If somebody is touching TJ, it will stop
            lspeed = STOP;
            }

    motortj(RIGHT_MOTOR, rspeed);
    motortj(LEFT_MOTOR, lspeed);

/* This "if" statement checks the rear bumper. If the bumper is pressed, */
/* Tj will back up, and turn.                                    */

    if(PORTC & 0x01)
    {
      motortj(LEFT_MOTOR, FORWL);
      motortj(RIGHT_MOTOR, FORWR);
      for(i = 0; i < 25000; i++);
      turn();
    }
/* T= 27*(i-1)+24 cycles, i=2500, cycle = 0.5 microseconds */
  for(i = 1; i < 2500; i++);   /* T = 33748.5 microseconds */
  }
}
/*************************** End of Main ******************************/

void turn(void)
/*****************************************************************
 * Function:  Will turn in a random direction for a random amount of time          *
 * Returns:  None                                                                   *
 * Inputs                                                                           *
 *   Parameters: None                                                               *
 *   Globals:    None                                                               *
 *   Registers:  TCNT                                                               *
 * Outputs                                                                          *
 *   Parameters: None                                                               *
 *   Globals:    None                                                               *
 *   Registers:  None                                                               *
 * Functions called: None                                                           *
 *****************************************************************/
{
 int i;
 unsigned rand;

 rand = TCNT;

 if (rand & 0x0001)
 {
   motortj(RIGHT_MOTOR, FORWR);
   motortj(LEFT_MOTOR, BACKL);
 }
 else
 {
   motortj(RIGHT_MOTOR, BACKR);
   motortj(LEFT_MOTOR, FORWL);
 }
 for (i = 0; i < rand; i++);
}
```

This is the program currently running in the robot's TJPro board. The single-chip board has the Super Servo program (servomod.sys) so no more programming is needed for it.

```
/**************************************************************
 * Program: kitty.c                                           *
 * Version: 1                                                 *
 * Programmer: Cynthia Manya                                  *
 * Date: 4/23/00                                              *
 * Description: cat test                                      *
 *       8 data bits, 1 start bit, 1 stop bit, no parity,     *
 *       9600 baud, no flow control                           *
 **************************************************************/
/**************************** Includes *************************/
#include <tjpbase.h>
#include <hc11.h>
#include <mil.h>
#include <stdio.h>
/************************* End of Includes ********************/
/**************************** Defines **************************/
#define DELAY 1500
#define PAUSE 3
/************************* End of Defines *********************/

void servoit(char sn1, char sn2, char pos1, char pos2, char pos3, char pos4);
void setdigport(void);
void setup(void);
void crouch(void);

void main(void)
{
 int touch, CdS;

 init_analog();
 init_clocktjp();
 setdigport();

 IRE_ON;                        /* turn on IR emitters */

 put_char('A');                 /* turn all servos on */
 put_char(' ');
 wait(PAUSE);

 setup();                       /* set cat in standing position */

 while(1)
 {
  while ((touch=analog(7)) > 140)
  {
   crouch();
   CLEAR_BIT(PORTD, 0x10);               /* turn sound on (set bit 4 to 0 volts) */
   wait(DELAY);
   SET_BIT(PORTD, 0x10);
  }
  wait(DELAY);
  setup();                               /* get back into standing position */

  if (CdS=analog(3) > 165)               /*check if CdS cell detected bright light*/
  {
   CLEAR_BIT(PORTD, 0x08);               /*if so, meow*/
   wait(125);
   SET_BIT(PORTD, 0x08);
  }
 }
}

void setdigport(void)
{
 SET_BIT(DDRD, 0x08);                    /* set PortD bits 3 & 4 as dig outputs */
 SET_BIT(DDRD, 0x10);

 SET_BIT(PORTD, 0x08);                   /* set PortD bits 3 & 4 to 1 (5 volts) */
 SET_BIT(PORTD, 0x10);
}

void setup(void)
{
 servoit('0', '0', '3', '0', '0', '0');  /* standing position */
 servoit('0', '1', '3', '0', '0', '0');
 servoit('0', '2', '3', '0', '0', '0');
 servoit('0', '3', '3', '0', '0', '0');
 servoit('0', '4', '3', '0', '0', '0');
 servoit('0', '5', '3', '0', '0', '0');
 servoit('0', '6', '3', '0', '0', '0');
 servoit('0', '7', '3', '0', '0', '0');
 servoit('0', '9', '3', '0', '0', '0');
 servoit('1', '0', '3', '0', '0', '0');
 servoit('1', '1', '3', '0', '0', '0');
 servoit('1', '2', '3', '0', '0', '0');
}
```

```
void servoit(char sn1, char sn2, char pos1, char pos2, char pos3, char pos4)
{
 put_char('W');                    /* command to control one servo */
 wait(PAUSE);
 put_char(' ');
 wait(PAUSE);

 if (sn1 != '0')                   /* sn1 & sn2 equals servo number (0-12) to be controlled */
 {
  put_char(sn1);
  wait(PAUSE);
 }

 put_char(sn2);
 wait(PAUSE);
 put_char(' ');
 put_char(pos1);                   /* position (2000-4000) to set servo */
 wait(PAUSE);
 put_char(pos2);
 wait(PAUSE);
 put_char(pos3);
 wait(PAUSE);
 put_char(pos4);
 wait(PAUSE);
 put_char(' ');
 wait(PAUSE);
}

void crouch(void)
{
 /* Front right leg */
 servoit ('0', '1', '3', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '0', '3', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '4', '3', '5', '0', '0');
 wait(PAUSE);

 /* Back left leg */
 servoit ('1', '1', '3', '5', '0', '0');
 wait(PAUSE);
 servoit ('1', '2', '3', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '7', '3', '0', '0', '0');
 wait(PAUSE);
 /* Front left leg */
 servoit ('1', '0', '2', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '9', '2', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '6', '2', '7', '0', '0');
 wait(PAUSE);

 /* Back right leg */
 servoit ('0', '2', '2', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '3', '2', '5', '0', '0');
 wait(PAUSE);
 servoit ('0', '5', '3', '0', '0', '0');
 wait(PAUSE);
}
```