A 3-D Distributed Mobile Sensor Network

NSF Summer Undergraduate Fellowship in Sensor Technologies
Yao Hua Ooi (Electrical Engineering) – University of Pennsylvania
Advisor: Dr. Daniel D. Lee

ABSTRACT

Biological organisms employ binocular visual and binaural hearing systems, as well as movement, to gather sensory information from multiple viewpoints for accurate sensory perception. In contrast, artificial sensory systems typically use either a multitude of sensors in a static array, or employ motion from a single mobile robotic platform.

The goal of this project is to build a small prototype of an adaptive, distributed network consisting of small, modular sensors and actuating components that will accurately position sensors at multiple 3-dimensional spatial locations and yield sensory information from multiple viewpoints. A working prototype of a single sensor node system was built, using a simple motorized spool design and a Motorola HC11 microcontroller. Control of the system was implemented in C and assembly to position a single sensor node in a 2-dimensional space. A user interface, allowing input via infra-red remote control was designed.

1. INTRODUCTION

Biological organisms make use of multiple viewpoints in sensing to accurately perceive the environment. Designers of artificial sensory systems have used some of the same techniques that biological systems employ to gain sensory information from multiple viewpoints. For example, in auditory sensing, a large number of microphones are used in an acoustic sensing array. Employing beamforming techniques on a static microphone array can greatly amplify sound sources along certain directions while decreasing the array's sensitivity to noise in other directions, allowing a more accurate detection of the direction of the sound source [1]. In the visual domain, multiple cameras are used to create binocular viewing systems to allow stereo depth perception.

However, a number of constraints limit the sensing capability of most artificial sensory systems. Most systems use a static arrangement of multiple sensors which prevents accurate sensing of objects that are not exactly positioned in a designated target space. Even systems that incorporate active motor systems to overcome this problem are limited to a single mobile platform or a small number of platforms. These platforms are typically wheeled and constrained to move along the floor in two dimensions, even though recent work is being done on using unmanned autonomous flying machines for three dimensional positioning [2].

To replace the paradigm of static sensor arrays and isolated mobile robots, we propose to build an adaptive, distributed sensor network that will allow the accurate positioning of sensors at multiple 3-dimensional spatial locations to gather sensory information from multiple viewpoints. The system will consist of small, modular sensors and actuating components that will allow active mobility of visual, audio and olfactory sensors in 3 dimensions to yield the maximum amount of information about the surrounding environment. The goal of this project is to build a small prototype of such a system to demonstrate the underlying concepts and yield insights about the feasibility of constructing such a system.

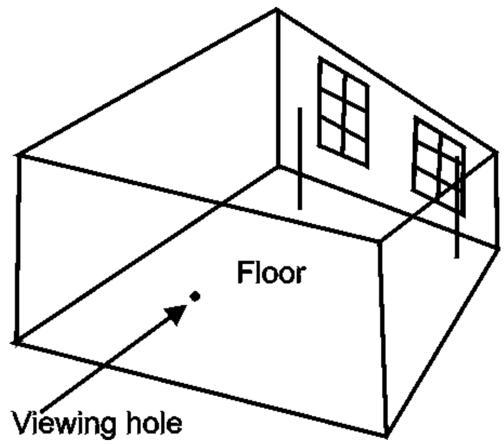

Figure 1: Actual shape and structure of an Ames room.

Figure 2: The Ames room illusion: Seen through the viewing hole, the identical twins appear to be drastically different in size.

2. BACKGROUND

2.1 Biological Motivation

The viewpoint of an observer crucially determines his perception. The dependence of an observer's perception on his viewpoint can be illustrated with the Ames room illusion. Figure 1 shows the actual construction of an Ames room. If an observer looks through the viewing hole, limiting himself to a single viewpoint, a pair of identical twins, standing on opposite sides of the room, will be perceived as being drastically different in size, as shown in Figure 2. Looking through the viewing hole removes any depth cues and makes the room appear normal and cubic although its shape is actually trapezoidal, the floor is actually on an incline and the walls are slanted outward. However, when the room is viewed from a different perspective, it becomes immediately clear to the observer that the room is not regularly shaped, and that the twins appear different in size merely because they are at different distances from the observer [3]. This simple illustration demonstrates the role that the location of an observer plays in sensory perception and highlights the importance of gathering information from multiple viewpoints for accurate perception.

Biological organisms have developed binocular vision systems and binaural auditory systems that allow them to take advantage of multiple viewpoints in perception. In addition, most biological organisms also employ movement in a process known as "active perception" to acquire new viewpoints and combine sensory information from multiple viewpoints to accurately perceive their surroundings [4]. For example, in olfactory sensing, humans actively move around an area while sniffing to detect the source of an odor. Since odors propagate slowly through wind transport and diffusion, an observer has little chance of detecting the source of an odor without obtaining observations of scents from multiple locations.

2.2 Intelligent Rooms

Significant advances in the fields of face recognition, gesture recognition, speech recognition and people tracking have brought about the concept of an intelligent room. An intelligent room is one in which the use of multiple audio and visual sensors, combined with algorithms that are capable of combining and processing multimodal sensory information, creates a system that autonomously captures and maintains awareness of objects and events in the room environment [5].

A 3-dimensional distributed mobile sensor network would fit the need to position sensors in 3-dimensional spatial locations in an intelligent room. With the appropriate control algorithms in place, it would be possible to create constellations of sensors that track and monitor individuals as they enter and wander about the surrounding space. The human computer interface in such an intelligent room would be revolutionary, as sensory information from a large number of relevant perspectives would be available in real time for information processing. For example, instead of a user having to walk up to a microphone to deliver a voice command, he will be able to give voice commands at any location in the room, since the audio sensors that follow him will be able to pick up the required audio information. The same principle can be applied to gesture recognition systems, in which visual sensors would continually capture multiple viewpoints of users in the room and respond to gesture commands [6].

2.3 Existing Sensor Network Projects

Currently, a number of radically different sensor network research projects are being conducted. The Smart Dust project at the University of California–Berkeley, is aimed at building sensor nodes that occupy less than 100 cubic millimeters and possess complete sensing and communication capabilities on a tiny mote. Researchers envision the sensor nodes eventually being small and light enough to be capable of floating around in a room, communicating sensory information to a base station for processing [7]. However, numerous challenges in power management, wireless transmission, and Micro-Electro-Mechanical Systems (MEMS), need to be overcome before such a complex system can be successfully implemented. Our sensor network project allows research into related control algorithms for tracking objects and positioning of sensor nodes in 3-dimensional space without delving into the problems associated with building a miniaturized, wireless system.

The Argus system, a distributed sensor network for real-time telepresence consists of a dense camera array, which is capable of capturing information from all angles around the imaging space [8]. The 3-dimensional distributed mobile sensor network we propose to build would enable added functionality to a static sensor network similar to the Argus system by allowing 3-dimensional positioning freedom of the camera array and enlarge the space enabled for real-time telepresence.

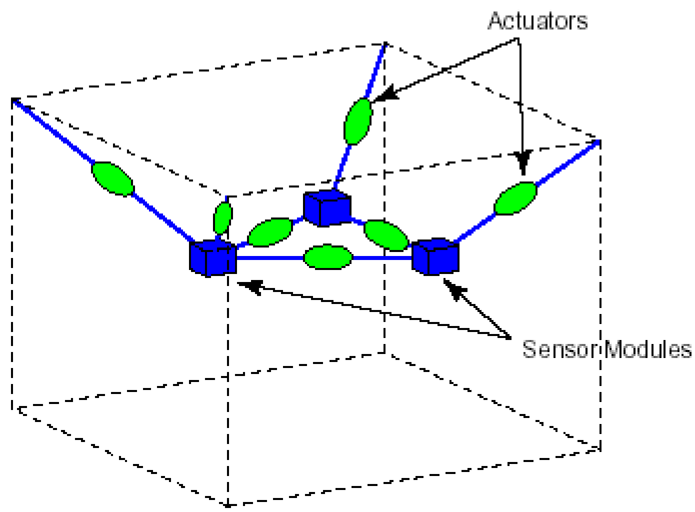

Figure 3: Design of a 3-D mobile sensor network prototype

2.4 Description of Proposed Prototype

The inspiration for the 3-dimensional positioning of the system comes from theatrical performances such as Peter Pan that incorporate “flying” actors and props that are suspended in the air by multiple cables. The designed of the proposed prototype is shown in Figure 3.

Each of the sensor modules is connected to a number of thin cables that provide 3-dimensional mechanical support, and also serve to distribute power and networking connectivity to the module. Using control software, small actuators will be controlled to wind or unwind connected cables to adjust the lengths of each of these cables. By changing the lengths of the supporting cables, the 3-dimensional positions of the suspended sensors can be quickly and freely adjusted.

An individual sensor module will consist of some combination of a CCD camera to capture video information, microphones to monitor audio signals, and/ gas sensors to measure local chemical concentrations. An embedded microcontroller would be used to digitize the sensor readings, perform preliminary processing on the data, and to relay this information to external computers for further processing.

The advantage of this system is that it could easily be scaled up to include a large number of sensors and actuators. The supporting cables provide any necessary power and high-speed networking capability to distribute the sensory information, and each sensor can quickly and accurately be positioned by adjusting the cable lengths.

3. HARDWARE DESIGN OF A SINGLE-NODE SYSTEM

Since constructing a multiple-node system was well beyond the scope of a 10-week independent research project, and since the feasibility of the system design had yet to be ascertained, the goal for the summer was to construct a single-node system with a simpler design. By the end of the project, a system consisting of one sensor node and two actuators was successfully built, allowing the positioning of the sensor node in a 2-dimensional plane. With the design of the actuator system in place, it should be straightforward to extend the design to a system with four actuators that would allow 3-dimensional positioning of the sensor node.

3.1 Hardware Description

Figure 4: Assembled spool system

The spool system for controlling the length of the cables consists of a TS-53 Tower Hobbies servo motor that is modified to allow a full range of rotation, and custom made plastic and aluminum parts that were designed and machined. Fishing line is used in place of network cables to suspend the sensor node, as it allows a more compact spool design. Since the network has only one node, it is unnecessary to provide network connectivity to the prototype system, further simplifying the design.

A Technological Arts Adapt11C24DX board fitted with a Motorola 68HC11E0 microcontroller is used for software control of the system. The microcontroller board is plugged into a solderless breadboard to enable easy connections to external devices. The microcontroller used was equipped with 24K of external RAM and 32K of EEPROM. To allow user input via an infra-red (IR) remote control, an IR detector circuit was designed and connected to the microcontroller board. A regular laboratory power supply was used to power the system during test phases, while a 4 AA battery pack was used when the system was demonstrated outside the lab.

3.2 HC11 setup and IR receiver circuit

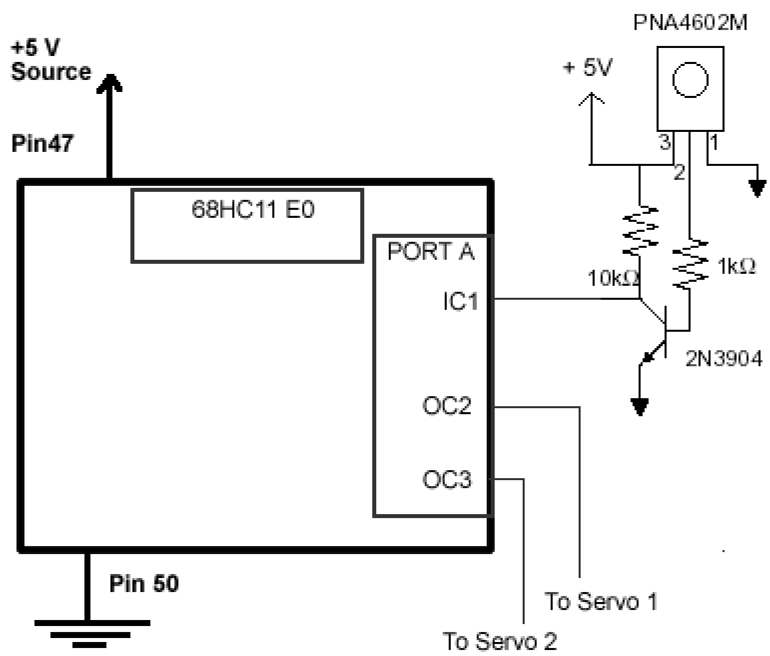

Figure 5: Pin connections to the HC11 microcontroller board

The Motorola 68HC11 is a multi-purpose, robust microcontroller chip used in many embedded systems for control and sensing applications, such as regulating temperature in a refrigerator and controlling the rate of combustion in a car engine. The Technological Arts Adapt11C24DX board combines a 68HC11 microcontroller, a voltage regulator, a port replacement unit, and a RS-232 serial communications interface in a single design that can be easily plugged into a solderless breadboard.

The output capture pins on the board were used to control the rotation of the motorized spool system. The layout of the pin connections is shown in Figure 5.

An IR detector circuit, consisting of a Panasonic PNA4602M IR detector chip and a 2N3904 transistor for signal amplification, was connected to the input capture pin of the HC11 board to allow for detection and processing of infra red signals. The design of the circuit is shown in Figure 6.

3.3 Modification of the TS-53 Tower Hobbies Servo Motor

Figure 6: A Tower Hobbes TS-53 servo motor

A servo motor, an active device often used to control motion in remote control cars and planes, consists of a DC motor, a gear box, and on-board position feedback electronics that control the motor and enable connections to be easily made to a microcontroller.

The TS-53 servo motor has three connector wires: power (usually +5V @200mA and capable of ~1A peak current), ground, and an input which accepts a pulse-width modulated (PWM) signal controlling the servos position. A picture of a Tower Hobbes TS-53 servo motor is shown in Figure 6.

There are a number of reasons a servo motor was chosen in the design of the motorized spool. Besides its compact size and low power consumption, the servo motor provides a high holding torque of 42 oz/inch, which is crucial in this application to hold sensor nodes firmly in position. A variety of possible end attachments makes it easy to mount wheels and different mechanisms to the servo motor. Furthermore, a servo motor has built in electronics that allow the DC motor to be rotated in both directions depending on the position of the shaft and the input signal, eliminating the need for a separate motor driver control board.

Figure 7: A pulse-width modulated (PWM) signal

The PWM signal required for control of the servo position is a square wave with the high time being variable as shown in Figure 7. The TS-53 servo motor design requires the PWM signal to be repeated at a rate of 20ms. By adjusting the pulse width, w, the position of the shaft can be controlled. A potentiometer connected to the shaft provides feedback to the system. When an input signal with a pulse width w is received, the position of the potentiometer determines the direction and speed at which the DC motor rotates to turn the shaft. When the shaft rotates, the position of the potentiometer changes, altering the input voltage to the control circuitry and changing the speed and direction of the rotation until the correct position for the shaft is reached. Usually a value of w = 1.5ms leaves the shaft in a neutral position (90 degrees), while a value of 0.5 ms will rotate the shaft to the extreme left (0 degrees) and a value of 2.5 ms will rotate the shaft to the extreme right (180 degrees).

However, the original design of a servomotor allows only for limited rotation of the shaft between 0 and 180 degrees. To be used in a motorized spool system, where continuous rotation of the motor is required to wind and unwind cable, the servo motor has to be modified [9].

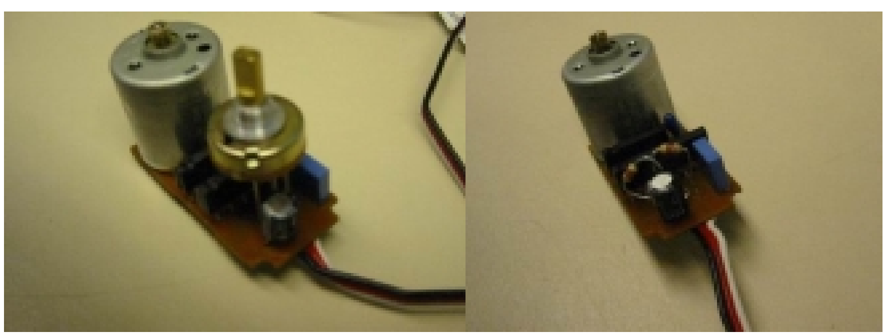

Figure 8: Modification of servo motor circuitry

Two changes were made to the original design of the servo. Firstly, the mechanical stopper that prevents full rotation of the shaft was removed using a wire cutter. Next, modification was done to the circuit to break the feedback loop that holds the shaft in a fixed position for a particular input signal. The potentiometer connected to the shaft was replaced with two 2.2KOhm resistors wound together to form an equivalent voltage divider circuit. After the potentiometer is removed, no the control circuitry receives no feedback of the shaft's position and is "fooled" into believing that the shaft is in a stationary position even if the shaft is rotating. Thus, applying an input signal with a pulse width, w, smaller than 1.5ms will cause the shaft to continuously rotate clockwise while an input signal with a pulse width, w, greater than 1.5ms will cause the shaft to continuously rotate counterclockwise. Figure 8 shows the original circuit and the modified circuit.

3.4 Design of Mechanical Spool System

Figure 9: Components of spool system

The mechanical spool used in the design is a simple cylindrical piece of plastic that is firmly glued to the servo motor shaft. Fishing line is threaded through a small hole on the side of the cylinder and wound around it tightly. The end of the line is tied to a weight to provide tension to the string to prevent the windings on the spool from loosening. In addition, an aluminum strip is bent as shown in Figure 9 to provide support for the spool. The existing design of the spool allows for 5-pound weights to be suspended without strain on the system.

4. SOFTWARE DESIGN FOR SYSTEM CONTROL

4.1 Description of software

Control software for the system was written in C and compiled using the Imagecraft ICC11 compiler that assembles code for the 68HC11. Technological Arts' Microload was used to download assembled code onto the EEPROM on the microcontroller. A file containing interrupt addresses called vectors.c had to be included at the end of each program written to enable the microcontroller to identify the starting memory address of the program upon reset. The control software consists of two major parts: code to decode input from the IR remote control and code to control the servo motor.

4.2 Remote control decoding

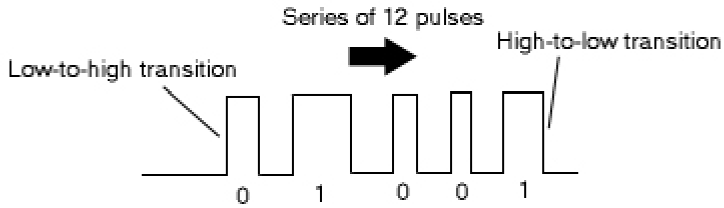

Figure 10: Decoding of pulse sequences

To allow easy control of the system, remote control functionality was added to the system. A Hauppauge remote control was used as an input device. Whenever a key is pressed, the remote control transmits 2 IR pulse trains, each containing 12 pulses. The IR receiver circuit connected to the input capture pin of the microcontroller triggers an interrupt service routine each time a pulse is received. The interrupt service routine records the value of the time counter (TCNT) on the microcontroller each time a high-to-low or low-to-high transition is detected. For each button pushed on the remote control, the 48 time counter values corresponding to the low-to-high and high-to-low transitions for the 24 transmitted pulses are recorded and stored in an array. The difference between every 2 time counter values is then calculated to get the width of each received pulse. Pulse widths larger than 3000 clock cycles are assigned a value of 1, while shorter pulse widths are assigned a value of 0.

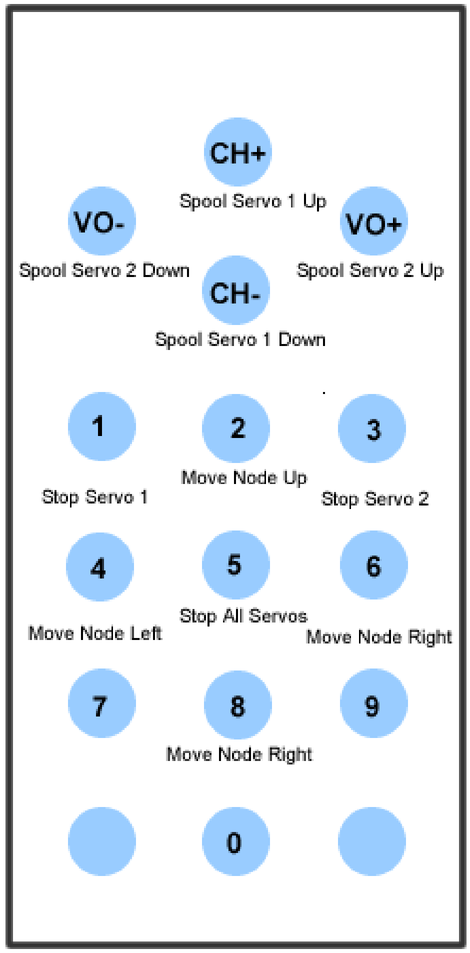

Figure 11: Layout of remote control keys

Since each key on the remote control transmits a pulse sequence that has a unique binary bit pattern, the control software is capable of identifying the desired user input by matching the received bit pattern to the matching key on the remote control. Once the desired user input is identified, the control software responds by generating the appropriate PWM signals to control the servo motor. Specific actions were programmed to correspond to different keys on the remote control. Figure 11 shows the layout of the remote control and the functions programmed for the different buttons.

4.3 Servo control

As described in section 3.3, pulse-width modulated (PWM) signals are used to control the rotational direction and speed of the modified servo motor. Precisely timed PWM signals are easily generated with the output capture function of the HC11 microcontroller. Using the programmable timer on the HC11, high and low voltages are produced on the output capture pins to generate appropriate pulse sequences to control the servo motors.

Because the modification of the servo motors produced slight variances in the resistances seen by the electronic control circuitry of each servo, calibration had to be performed to obtain the exact pulse widths that held each servo in a stopping position. Once these values had been determined, it was relatively straightforward to generate signals that produced clockwise and anticlockwise rotations of the servo motors. For example, if a pulse width value of w = 1.3 ms is found to correspond to the stopping position of a servo after calibration, applying a PWM signal with $w < 1.3$ ms would produce a clockwise rotation of the servo, while a PWM signal with $w > 1.3$ ms would produce a counterclockwise rotation of the servo. In addition, a greater difference between the pulse width applied and the pulse width corresponding to a stopping position produces a higher the rotation speed of the servo motor.

In addition to receiving user input via remote control, the control program was supplemented with code that allows user input via a keyboard connected to the serial communications interface of the microcontroller board. The additional input function allows for convenient inputting of pulse width values for each of the servo motors and proved to be very useful in calibration and testing of the system.

5. DISCUSSION AND CONCLUSIONS

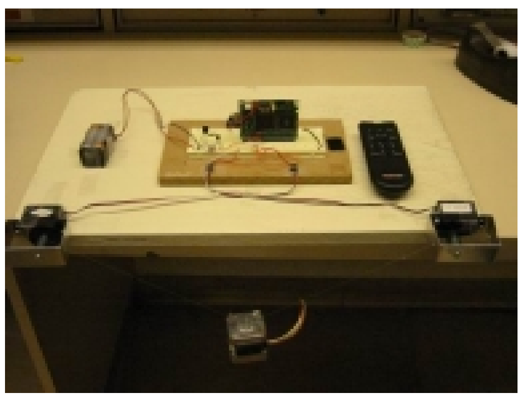

Figure 12: Picture of completed single-node, 2 actuator system

Figure 12 shows a functioning 2 actuator, single node system that was constructed by the end of the project. The completed system enables the weight suspended by the two motorized spools to be positioned anywhere in a 2 dimensional plan within the horizontal boundaries determined by the positioning of the two servo motors.

In addition to implementing the system, a significant amount of time was spent researching the modification of the servo motor to include an encoder that would allow detection of the servo motor’s position and enable precise lengths to be wound and unwound on the motorized spool. One of the attempted designs was to modify the potentiometer circuit on the servo motor control board to provide an external voltage that varies with the position of the motor shaft. This was done by connecting a wire leading out from the potentiometer to the analog-to-digital converter on the HC11 microcontroller. By detecting the change in voltage as the motor shaft makes a full rotation, the HC11 can detect the position of the shaft as well as count the number of rotations made. However, there were several difficulties faced with modifying the original potentiometer design of the servo to provide no feedback to the DC motor control circuitry while providing a feedback voltage to the A-to-D converter. Work is still being done to correct some mechanical and control problems related to the design of the encoder.

6. RECOMMENDATIONS

As this project is still in its initial stages, there are many more interesting research problems to tackle in building a complete distributed 3-D mobile sensor network system. In the design of a single node system, actuators can be fixed in 3-dimensional space to provide a convenient reference point for the positioning of the node, making the forward and inverse kinematics problem for node positioning a relatively straightforward one to solve. However, in a multiple node system, the position of each actuator would vary along with the position of each sensor node for any given configuration. To control such a system, it would be necessary to solve the forward and inverse kinematics problem related to the positioning of multiple nodes and actuators given different cables lengths.

In addition, a host of other mechanical difficulties would also have to be overcome to construct a reliable and robust multiple node system. If a working prototype can be successfully completed, it will open up the possibility of many other interesting research fields related to distributed processing of sensor data, algorithms for subject tracking, as well as command recognition based on multimodal sensory input. As a student at the University of Pennsylvania, I hope to continue work on this project in the coming school year.

7. ACKNOWLEDGMENTS

I would like to thank the NSF for their support of the REU program and SUNFEST. I am also grateful for Microsoft's sponsorship that allowed me to be a part of this research program. Special thanks goes to Dr. Daniel Lee, who has been a great mentor, advisor and encouragement throughout this project. In addition, I'd like to thank Lois Clearfield for taking care of the many administrative details and Sid Deliwala for his invaluable assistance in the EE lab. Last but not least, I would also like to thank Dr. Jan Van der Spiegel for making SUNFEST possible, and my fellow SUNFEST researchers for making the program an enjoyable one.

8. REFERENCES

1. B. D. Van Veen and K. M. Buckley. Beamforming: a versatile approach to spatial filtering. *ASSP Magazine, IEEE*, 5(2):4–24, 1988.

2. 29 Palms Fixed/Mobile Experiment Tracking vehicles with a UAV-delivered sensor network. UC Berkeley and MLB Co,

http://www.eecs.berkeley.edu/~pister/29Palms0103/

3. R. Gregory. *Eye and brain: the psychology of seeing*. McGraw-Hill, 1990.

4. R. Bajcsy. Active perception. *Proc. IEEE*, 76(8):996–1005, 1988.

5. Brooks, R. A. with contributions from M. Coen, D. Dang, J. DeBonet, J. Kramer, T. Lozano-Perez, J. Mellor, P. Pook, C. Stauffer, L. Stein, M. Torrance and M. Wessler, The Intelligent Room Project. *Proceedings of the Second International Cognitive Technology Conference (CT'97)*, Aizu, Japan, August 1997.

6. Michael Coen. The Future Of Human-Computer Interaction or How I learned to stop worrying and love my Intelligent Room. *IEEE Intelligent Systems* March/April 1999.

7. Smart Dust project website. UC Berkeley, http://robotics.eecs.berkeley.edu/~pister/SmartDust/

8. Steve Feller. Argus: A Distributed Sensor Network for Real-Time Telepresence, http://www.disp.duke.edu/~sfeller/Publications/

9. Kam Leang. Modifying the Tower Hobbies TS-53 Servomotor for Continuous Rotation, http://www.leang.com/robotics/info/articles/servo/servomod.pdf

APPENDIX A

C program code:

```
/* Servo Motor Remote Control Program (Calibrated for 2 Servos) */
/* Yao Hua Ooi */
/* July 31st, 2002 */
#include <hc11.h>

#pragma interrupt_handler IC1ISR
#pragma interrupt_handler OC2ISR
#pragma interrupt_handler OC3ISR
#pragma interrupt_handler SCIISR

void IC1ISR();
void OC2ISR();
void OC3ISR();
void SCIISR();

int input;
int countflag;
int count;

unsigned int t2;
unsigned int t3;

unsigned int t_high2; /*pulse width for OC2 */
unsigned int t_low2;

unsigned int t_high3; /*pulse width for OC3 */
unsigned int t_low3;

unsigned int t_high4; /*pulse width for OC4 */
unsigned int t_low4;

void main() {

t2 = 60, t3 = 61, t4= 58;

t_high2 = 2600; /* stationary pulse width value for OC1 */
t_low2 = 37400;

t_high3 = 2610; /* stationary pulse width value for OC2 */
t_low3 = 37390;

t_high4 = 2580; /* stationary pulse width value for OC3 */
t_low4 = 37520;

/* IR input setttings */

TCTL2|=0x30;   //capture high-to-low and low-to-high transition
TMSK1|=0x04;  //enable IC1 interrupt
```

```
/* SCI settings */

BAUD = 0x30;        /* 9600 Baud */
SCCR1 = 0x00;
SCCR2 = 0x0c;       /* Enable SCI transmitter and receiver */
SCCR2 |= 0x20; // Receiver interrupt enable

/* OC2 settings */
TOC2 = 0x7FFF;
TCTL1 |= 0xC0;                  /* Set output pin high for OC2 */
TMSK1 |= 0x40;          /* Enable OC2 interrupt locally */
TFLG1 &= 0x40;          /* Clear OC2 flag */

/* OC3 settings */
TOC3 = 0x7FFF;
TCTL1 |= 0x30;          /* Set output pin high for OC3 */
TMSK1 |= 0x20;          /* Enable OC3 interrupt locally */
TFLG1 &= 0x20;          /* Clear OC3 flag */

asm ("CLI");            /* Enable interrupts */
}

void changetime() {
        printf("\nPlease enter desired duty cycle percentage for Servo 1 (0 - 100): ");
        t2 = scanchar();
        t_high2 = 2000 + t2 * 10;
        t_low2 = 10 * (3800 - t2);
        printf("\nPlease enter desired duty cycle percentage for Servo 2 (0 - 100): ");
        t3 = scanchar();
        t_high3 = 2000 + t3 * 10;
        t_low3 = 10 * (3800 - t3);
}

int scanchar() {
        int val=0;
        char c;
        c=getchar();
        while (isdigit(c))  {
        val = 10*val + (c-'0');
        putchar(c);
        c= getchar();
        }
        return val;
}

void SCIISR(){
   unsigned char scsrval,scdrval;
   scsrval=SCSR;
   scdrval=SCDR;
   changetime();
}

void OC2ISR() {
        countflag++;
        TFLG1 &= 0x40;
        if (TCTL1 & 0x40) {
```

```
			TOC2 += t_high2;
			TCTL1 &= 0xBF;
			}
		else {
			TOC2 += t_low2;
			TCTL1 |= 0x40;
			}
}

void OC3ISR() {
		TFLG1 &= 0x20;
		if (TCTL1 & 0x10) {
			TOC3 += t_high3;
			TCTL1 &= 0xEF;
			}
		else {
			TOC3 += t_low3;
			TCTL1 |= 0x10;
			}
}

void IC1ISR() {
		static int i;
		static unsigned int period[48];
		static int diff[47];
		int count;

		if (countflag > 20) {
		i = 0;
		countflag = 0;
		}

		TFLG1 &= 0x04;
		period[i] = TIC1;
		i++;
		if (i==48) {
			for (count=0; count<47; count++) {
				diff[count]=period[count+1]-period[count];
				if (diff[count]>3000) {diff[count]=1;}
				else {diff[count]=0;}
			}
			if (diff[13]==1 && diff[14]==1 && diff[19]==1 && diff[20]==1) {printf("Source");

			/* Spool Servo 1 Down */
			else if (diff[13]==1 && diff[14]==1 && diff[21]==1) {
			printf("CH -");
			t2 = 20;
			}
			else if (diff[13]==1 && diff[14]==1 && diff[15]==1) {printf("Full Screen");}
			else if (diff[13]==1 && diff[14]==1 && diff[17]==1) {printf("Minimize");}

			/* Spool Servo 1 Up */
			else if (diff[13]==1 && diff[14]==1) {
			printf("CH +");
			t2 = 100;
			}
```

```
/* Spool Servo 2 Down (to the left) */
else if (diff[15]==1 && diff[16]==1 && diff[21]==1) {
printf("VOL -");
t3 = 21;
}

/* Spool Servo 2 Up (to the right) */
else if (diff[15]==1 && diff[16]==1) {
printf("VOL +");
t3 = 101;
}

else if (diff[17]==1 && diff[18]==1 && diff[21]==1) {
printf("9");
}

/* Downward Movement (Servo 1 & 2) */
else if (diff[17]==1 && diff[18]==1) {
printf("8");
t2 = 20;
t3 = 21;
}

else if (diff[17]==1 && diff[20]==1 && diff[21]==1) {
printf("Mute");
}

else if (diff[17]==1 && diff[20]==1) {printf("Radio");}
else if (diff[17]==1) {printf("TV");}

/* Stop all Servos */
else if (diff[20]==1 && diff[21]==1 && diff[19]==1) {
printf("5");
t2 = 60;
t3 = 61;
}

/* Move Node Left */
else if (diff[20]==1 && diff[19]==1) {
printf("4");
t2 = 100;
t3 = 20;
}

/* Move Node Right */
else if (diff[19]==1 && diff[22]==1) {
printf("6");
t2 = 20;
t3 = 100;
}

else if (diff[19]==1) {
printf("7");
}
```

```
		else if (diff[15]==1 && diff[22]==1) {printf("Reserve");}

		/* Upward Movement (Servo 1 & 2) */
		else if (diff[22]==1 && diff[21]==1) {
		printf("2");
		t2 = 100;
		t3 = 101;
		}

		/* Stop Servo 2 */
		else if (diff[21]==1) {
		printf("3");
		t3 = 61;
		}

		/* Stop Servo 1 */
		else if (diff[23]==1) {
		printf("1");
		t2 = 60;
		}
		else {printf("0");}

		printf("\n");

		i=0;

		t_high2 = 2000 + t2 * 10;
		t_low2 = 10 * (3800 - t2);

		t_high3 = 2000 + t3 * 10;
		t_low3 = 10 * (3800 - t3);

		i=0;
		countflag = 0;
	}
}

#include "vectors.c"
```