SUNFEST

(www.ee.upenn.edu/~sunfest)

SUMMER UNDERGRADUATE FELLOWSHIPS
IN
SENSOR TECHNOLOGIES

SUNFEST 2003

TECHNICAL REPORT

TR-CST17OCT03

CENTER FOR SENSOR TECHNOLOGIES
UNIVERSITY OF PENNSYLVANIA
PHILADELPHIA, PA 19104

SUNFEST 2003

SUMMER UNDERGRADUATE FELLOWSHIP
IN SENSOR TECHNOLOGIES
Sponsored by the National Science Foundation and Microsoft Corporation

PREFACE

This report is the result of 12 undergraduate students' research efforts during the summer of 2003. From May 27 through August 1, 2003, students from Penn and other colleges participated in the SUNFEST program, which is organized by the Center for Sensor Technologies of the School of Engineering and Applied Science at the University of Pennsylvania. This unique "Summer Experience for Undergraduates in Sensor Technologies" program was initiated in 1986 and has grown considerably in size. It is now recognized as one of the most successful summer programs for undergraduates in the country. I would like to express my sincere gratitude to the National Science Foundation for their continued support for this REU Site, as well as Microsoft Corporation for sponsoring two of our students.

The purpose of the SUNFEST program is to provide bright, motivated undergraduate students with the opportunity to become involved in active research projects under the supervision of a faculty member and his graduate student(s). The general area of research concentrates on sensor technologies and includes projects such as materials and technology for sensors, microstructures, smart imagers, and neural networks for sensory processing and robotics. By providing the students with hands-on experience and integrating them with a larger research group where they can work together with other students, the program intends to guide them in their career choices. By exposing the students to the world of research, we hope they will be more inclined to go on for advanced degrees in science and engineering.

The students participated in a variety of hands-on workshops in order to give them the tools to do first-rate research or enhance their communication skills. These included "Giving Effective Presentations", "Ethics in Science and Engineering", "Use of Electronic Databases" and "Writing Technical Reports". Students also had plenty of opportunity for social interactions among themselves or with faculty and graduate student advisors.

The success of such a program is best measured in terms of the number of students who have gone on to graduate school. To date a total of 156 students have participated in the SUNFEST program. Of the students who have since graduated, about 70% have gone on to graduate school or have received their masters and doctorate degrees. Several students also received awards based on their summer research.

This booklet contains reports from this year's projects, the quality of which testifies to the high level of research and commitment by these students and their supervisors. I would like to express my sincere thanks to the students for their enthusiastic participation; the help of the faculty members, graduate students and support staff is very much appreciated. I would also like to thank Ms. Lois Clearfield and the ESE staff for their invaluable help in making this program run smoothly.

Jan Van der Spiegel, Director
Center for Sensor Technologies

FINAL REPORT
2002 SUMMER UNDERGRADUATE FELLOWSHIP
IN SENSOR TECHNOLOGIES
Sponsored by the National Science Foundation
and Microsoft

TABLE OF CONTENTS

TABLE OF CONTENTS

(Continued)

LIST OF STUDENT PARTICIPANTS
IN THE SUNFEST PROGRAM SINCE 1986

Summer, 2003

Emily Blem	Swarthmore College
Brian Corwin	University of Pennsylvania
Vinayak Deshpande	University of Virginia
Nicole DiLello	Princeton University
Jennifer Geinzer	University of Pittsburgh
Jonathan Goulet	University of Pennsylvania
Mpitulo Kala-Lufulwabo	University of Pittsburgh
Emery Ku	Swarthmore College
Greg Kuperman	University of Pennsylvania
Linda Lamptey	University of Pennsylvania
Prasheel Lillaney	University of Pennsylvania
Enrique Rojas	University of Pennsylvania

Summer, 2002

Christopher Bremer	Colorado School of Mines
Aslan Ettehadien	Morgan State University
April Harper	Hampton University
Catherine Lachance	University of Pennsylvania
Adrian Lau	University of Pennsylvania
Cynthia Moreno	University of Miami
Yao Hua Ooi	University of Pennsylvania
Amber Sallerson	University of Maryland/Baltimore County
Jiong Shen	University of California-Berkeley
Kamela Watson	Cornell University
John Zelena	Wilkes University

Summer, 2001

Gregory Barlow	North Carolina State University
Yale Chang	University of Pennsylvania
Luo Chen	University of Rochester
Karla Conn	University of Kentucky
Charisma Edwards	Clark Atlanta University
EunSik Kim	University of Pennsylvania
Mary Kutteruf	Bryn Mawr College
Vito Sabella	University of Pennsylvania
William Sacks	Williams College
Santiago Serrano	Drexel University
Kiran Thadani	University of Pennsylvania
Dorci Lee Torres	University of Puerto Rico (Humacao)

LIST OF STUDENT PARTICIPANTS
IN THE SUNFEST PROGRAM SINCE 1986
(continued)

Summer, 2000

Lauren Berryman	University of Pennsylvania
Salme DeAnna Burns	University of Pennsylvania
Frederick Diaz	University of Pennsylvania (AMPS)
Hector Dimas	University of Pennsylvania (AMPS)
Xiomara Feliciano	University of Turabo (Puerto Rico)
Jason Gillman	University of Pennsylvania
Tamara Knutsen	Harvard University
Heather Marandola	Swarthmore College
Charlotte Martinez	University of Pennsylvania
Julie Neiling	University of Evansville, Indiana
Shiva Portonova	University of Pennsylvania

Summer, 1999

David Auerbach	Swarthmore
Darnel Degand	University of Pennsylvania
Hector E. Dimas	University of Pennsylvania
Ian Gelfand	University of Pennsylvania
Jason Gillman	University of Pennsylvania
Jolymar Gonzalez	University of Puerto Rico – Mayaguez
Kapil Kedia	University of Pennsylvania
Patrick Lu	Princeton University
Catherine Reynoso	Hampton University
Philip Schwartz	University of Pennsylvania

Summer, 1998

Tarem Ozair Ahmed	Middlebury University
Jeffrey Berman	University of Pennsylvania
Alexis Diaz	Turabo University, Puerto Rico
Clara E. Dimas	University of Pennsylvania
David Friedman	University of Pennsylvania
Christin Lundgren	Bucknell University
Heather Anne Lynch	Villanova University
Sancho Pinto	University of Pennsylvania
Andrew Utada	Emory University
Edain (Eddie) Velazquez	University of Pennsylvania

LIST OF STUDENT PARTICIPANTS
IN THE SUNFEST PROGRAM SINCE 1986
(continued)

Summer, 1997

Francis Chew	University of Pennsylvania
Gavin Haentjens	University of Pennsylvania
Ali Hussain	University of Pennsylvania
Timothy Moulton	University of Pennsylvania
Joseph Murray	Oklahoma University
O'Neil Palmer	University of Pennsylvania
Kelum Pinnaduwage	University of Pennsylvania
John Rieffel	Swarthmore College
Juan Carlos Saez	University of Puerto Rico, Cayey

Summer, 1996

Rachel Branson	Lincoln University
Corinne Bright	Swarthmore College
Alison Davis	Harvard University
Rachel Green	Lincoln University
George Koch	University of Pennsylvania
Sandro Molina	University of Puerto Rico-Cayey
Brian Tyrrell	University of Pennsylvania
Joshua Vatsky	University of Pennsylvania
Eric Ward	Lincoln University

Summer, 1995

Maya Lynne Avent	Lincoln University
Tyson S. Clark	Utah State University
Ryan Peter Di Sabella	University of Pittsburgh
Osvaldo L. Figueroa	University of Puerto Rico-Humacao
Colleen P. Halfpenny	Georgetown University
Brandeis Marquette	Johns Hopkins
Andreas Olofsson	University of Pennsylvania
Benjamin A. Santos	University of Puerto Rico-Mayaguez
Kwame Ulmer	Lincoln University

LIST OF STUDENT PARTICIPANTS
IN THE SUNFEST PROGRAM SINCE 1986
(continued)

Summer, 1994

Alyssa Apsel	Swarthmore College
Everton Gibson	Temple University
Jennifer Healy-McKinney	Widener University
Peter Jacobs	Swarthmore College
Sang Yoon Lee	University of Pennsylvania
Paul Longo	University of Pennsylvania
Laura Sivitz	Bryn Mawr College
Zachary Walton	Harvard University

Summer, 1993

Adam Cole	Swarthmore College
James Collins	University of Pennsylvania
Brandon Collings	Hamilton University
Alex Garcia	University of Puerto Rico
Todd Kerner	Haverford College
Naomi Takahashi	University of Pennsylvania
Christopher Rothey	University of Pennsylvania
Michael Thompson	University of Pennsylvania
Kara Ko	University of Pennsylvania
David Williams	Cornell University
Vassil Shtonov	University of Pennsylvania

Summer, 1992

James Collins	University of Pennsylvania
Tabbetha Dobbins	Lincoln University
Robert G. Hathaway	University of Pennsylvania
Jason Kinner	University of Pennsylvania
Brenelly Lozada	University of Puerto Rico
P. Mark Montana	University of Pennsylvania
Dominic Napolitano	University of Pennsylvania
Marie Rocelie Santiago	Cayey University College

LIST OF STUDENT PARTICIPANTS
IN THE SUNFEST PROGRAM SINCE 1986
(continued)

Summer, 1991

Gwendolyn Baretto	Swarthmore College
Jaimie Castro	University of Puerto Rico
James Collins	University of Pennsylvania
Philip Chen	University of Pennsylvania
Sanath Fernando	University of Pennsylvania
Zaven Kalayjian	University of Pennsylvania
Patrick Montana	University of Pennsylvania
Mahesh Prakriya	Temple University
Sean Slepner	University of Pennsylvania
Min Xiao	University of Pennsylvania

Summer, 1990

Angel Diaz	University of Puerto Rico
David Feenan	University of Pennsylvania
Jacques Ip Yam	University of Pennsylvania
Zaven Kalayjian	University of Pennsylvania
Jill Kawalec	University of Pennsylvania
Karl Kennedy	Geneva
Jinsoo Kim	University of Pennsylvania
Colleen McCloskey	Temple University
Faisal Mian	University of Pennsylvania
Elizabeth Penadés	University of Pennsylvania

Summer, 1989

Peter Kinget	Katholiek University of Leuven
Chris Gerdes	University of Pennsylvania
Zuhair Khan	University of Pennsylvania
Reuven Meth	Temple
Steven Powell	University of Pennsylvania
Aldo Salzberg	University of Puerto Rico
Ari M. Solow	University of Maryland
Arel Weisberg	University of Pennsylvania
Jane Xin	University of Pennsylvania

LIST OF STUDENT PARTICIPANTS
IN THE SUNFEST PROGRAM SINCE 1986
(continued)

Summer, 1988

Lixin Cao	University of Pennsylvania
Adnan Choudhury	University of Pennsylvania
D. Alicea-Rosario	University of Puerto Rico
Chris Donham	University of Pennsylvania
Angela Lee	University of Pennsylvania
Donald Smith	Geneva
Tracey Wolfsdorf	Northwestern University
Chai Wah Wu	Lehigh University
Lisa Jones	University of Pennsylvania

Summer, 1987

Salman Ahsan	University of Pennsylvania
Joseph Dao	University of Pennsylvania
Frank DiMeo	University of Pennsylvania
Brian Fletcher	University of Pennsylvania
Marc Loinaz	University of Pennsylvania
Rudy Rivera	University of Puerto Rico
Wolfram Urbanek	University of Pennsylvania
Philip Avelino	University of Pennsylvania
Lisa Jones	University of Pennsylvania

Summer, 1986

Lisa Yost	University of Pennsylvania
Greg Kreider	University of Pennsylvania
Mark Helsel	University of Pennsylvania

IMPLEMENTATION OF THE ENIAC ACCUMLATOR AND CYCLING UNIT ON A FIELD PROGRAMMABLE GATE ARRAY (FPGA)

NSF Summer Undergraduate Fellowship in Sensor Technologies
Emily Rose Blem (Mathematics and Engineering) – Swarthmore College
Advisor: Dr. Jan Van der Spiegel

ABSTRACT

An accumulator and cycling unit from the Electronic Numerical Integrator and Calculator (ENIAC) are implemented on a Field Programmable Gate Array (FPGA). The FPGA implementation is not architecturally identical to the original ENIAC, but the original architectural design is maintained where possible. The design maintains decimal number representation, ring counters to increment numbers, and ten different clocks to coordinate various accumulator functions. The implementation is a preliminary step in the design of a series of FPGAs that will include a constant transmitter, three accumulators, be connected to panels similar to those of the original ENIAC and be an exact replica in the programmer's experience.

Table of Contents

IMPLEMENTATION OF THE ENIAC ACCUMLATOR AND CYCLING UNIT ON A FIELD PROGRAMMABLE GATE ARRAY (FPGA)

Emily Rose Blem (Mathematics and Engineering) – Swarthmore College
Advisor: Dr. Jan Van der Spiegel

1. INTRODUCTION

Completed in 1946, the ENIAC (Electronic Numerical Integrator and Calculator) was the first general use electronic computer. It was re-created in 1997 on a silicon chip using CMOS technology at Moore School of Electrical Engineering at the University of Pennsylvania, and this paper explores implementation of the ENIAC on a series of Field Programmable Gate Arrays (FPGAs). The accumulator and cycling units have been implemented on an FPGA. The FPGA implementation is not architecturally identical to the original ENIAC, but the original architectural design is maintained where possible. The series of FPGAs will be connected to panels similar to those of the original ENIAC and be an exact replica in the programmer's experience. While the ENIAC-on-a-Chip demonstrated technological advances over the past 50 years, the FPGA implementation allows users to program as if they were interacting with the original ENIAC and serves as a valuable historical teaching tool.

The original ENIAC (Electronic Numerical Integrator and Calculator) was introduced as a tool for ballistics calculations as well as the first electronic general use computer. It was originally designed to calculate weapons firing tables for the Ballistics Research Laboratory during World War II. However, it was not completed in time for this purpose, and its first calculations were for nuclear research. This original ENIAC filled a room that was approximately 1800 square feet and contained thousands of components. The computer broke down an average of once every 5.6 hours, mostly because of its more than 40,000 vacuum tubes [1, 2]. The ENIAC was dismantled upon obsolescence, and the programming panels lent to various museums in celebration of the advance represented by the ENIAC.

In 1997, a team at Penn reconstructed the ENIAC, preserving the original architecture, onto a microchip as a tribute to the advances since the first computer. The microchip reproduction did not, however, preserve the original programming interface but instead required a software interface to program the chip. This paper describes a new implementation of the ENIAC on a series of FPGAs (Field Programmable Gate Arrays) to be controlled by panels similar to those used in the original ENIAC. FPGAs are logic devices that can be programmed using either graphical schematics of logic gates or VHDL code (described below). For this project, a Xilinx Spartan IIe FPGA was programmed using VHDL code and the ISE 5 software package. The FPGAs were interfaced with a custom-designed printed circuit board (PCB) for the input and output functions. The project goal is to preserve as much of the original structure as possible while maintaining the original user interface.

1.1 Basic ENIAC Architecture

A computer's architecture encompasses all the decisions made in the design process, such as number format, program structure, and data transfer system. The ENIAC had 30 individual units coordinated by digit and pulse trunks, equivalent to modern data and address buses. While modern architectures use a binary format to express numbers, the original ENIAC used decimal numbers. Numbers were transmitted

as pulses over the digit trunk, with one physical wire or line per digit. The digit trunk had 11 lines (wires), with 10 wires were for digits and a single wire or line for number sign information. Twenty of the units were accumulators; they could add or subtract a number from the number previously stored in the unit. Other units included a master programmer, initiating unit, function generators, multipliers, a divider and square rooter, a cycling unit, and input and output devices. The accumulator, cycling unit, and constant transmitter are discussed in detail in Section 2.

1.2 VHDL

VHDL is the acronym for Very High Speed Integrated Circuit (VHSIC) Hardware Description Language. It is a hardware description language supported by software packages such as ISE to design circuitry for implementation on logic devices. Logic devices are used to hardwire a program rather than implementing a software program on a general purpose platform. Logic circuit layouts were originally designed by hand and then built from individual components. The design process then shifted to drawing schematics with logic gates on a computer, and VHDL was eventually introduced to allow the user to program a logic device (FPGA in this case) using a low-level programming language. The VHDL compiler converts code into an arrangement of logic gates that the computer can then program onto a logic device. VHDL code is clean, portable, fast to design, and easy to simulate.

The top layer of a system designed using VHDL contains port statements for each entity that makes up the system. The output of one entity is connected to the input of another through the assignment of both the input and output to the same signal. Signals are declared within entities (or within the top-level port statement) and act like wires between components. Port statements allow entities to be connected to each other and to outside inputs and outputs.

Within an entity, there is a port statement and an architecture. The entity port statement declares inputs and outputs. The entity architecture contains the code that affects entity outputs. Since VHDL is a hardware description language, it acts like a circuit: all lines of code are run concurrently. Sequential statements, like those in most software programming languages, can be run within processes. Processes are updated each time that one of the variables upon which they are dependent changes, and generally contain a series of if statements. If statements can be dependent upon events in the variable value; they are run once when a change is detected in the variable. If a variable is used on the rising or falling edge, it is considered a clock. Each process can check for events in only one variable (although it can check the value of many variables), and variables can be modified within only one process.

1.3 Project Goals

The design constraints for the ENIAC reconstruction require maintenance of the spirit of the original architecture. Each of the original units of the ENIAC is designed in a separate VHDL file, to be implemented on a separate FPGA chip. FPGA chips have a

limited number of input and output lines, and to minimize the lines required, a cycling unit is implemented on each chip. The control unit produces a synchronizing clock and pulse, which are transmitted to each unit for pulse generation. The 10 different signals produced in the cycling unit are, as in the original ENIAC, used to control different processes. The project's final product will be a constant transmitter, cycling unit, initiating unit, and three accumulators implemented on FPGAs with an interface of switches and dials very similar to the original ENIAC. The paper below describes implementation of a clock divider, control unit (part of the initiating unit), cycling unit, and accumulator on a single FPGA as a preliminary step in the project.

2. IMPLEMENTATION

The design implemented on the FPGA has nine components. The clock divider slows down the oscillator included on the Digilent Digilab Board. The control unit starts and stops the clock provided to the cycling unit based on external operation mode settings. The cycling unit generates the pulse train. The start up unit reads in switches, saves their settings to RAM, and takes care of other output functions. The receiver updates digit values based upon received pulses and switch settings, which are displayed by the LED unit in real time. The sign unit keeps track of the current sign and the transmitter converts the values stored in digit registers to pulses on the digit trunks. Interaction between components is shown in Figure 1 and individual components are described in detail below.

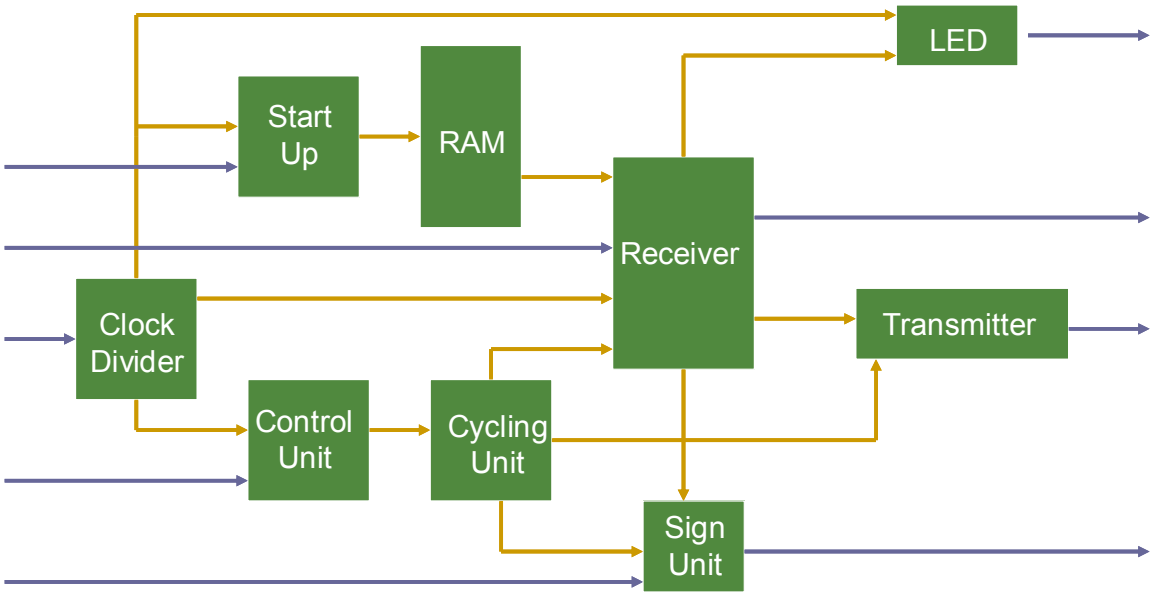


Figure 1: Interaction between units of the accumulator. Yellow lines are internal connections and blue lines are input and output lines.

2.1 Clock Divider

The ENIAC ran at 5000 addition cycles per second, or about 100 kHz, where one addition cycle is 20 pulses long. The Xilinx IIe FPGA is mounted on a Digilab II XL board with a 50 MHz oscillator. To make the LED display easier to follow, the clock

was slowed down to 10 pulses per second with a clock divider. The clock divider uses a counter to slow down the clock and preserves a 50% duty cycle.

2.2 Control Unit

The original ENIAC had one cycling unit that provided a pulse train to all other units. To minimize input and output connections between chips, the FPGA implementation generates a pulse train in each unit. The units are synchronized using a control unit that outputs the clock used by the cycling units in each accumulator, based upon the operation mode. The ENIAC has three operation modes: continuous, addition cycle, and pulse mode. Continuous mode is most commonly used for actual computations, while the other two modes are for debugging purposes. Pulse mode proceeds pulse by pulse, and the user must push an advance button between each pulse. Addition cycle mode runs for 20 pulses and then waits for the user to push the advance button before proceeding to the next 20-pulse cycle. The control unit starts and stops the output clock based upon these three modes and also outputs a synchronizing pulse once per addition cycle (at the same time as the cpp, central programming pulse) to ensure all units are at the same place in the cycle. The control unit uses a combination of "if" statements and counters to operate in the correct mode and output synchronization pulses at the correct time.

2.3 Cycling Unit

The ENIAC's cycling unit generates a pulse train with 10 individual signals. The pulse train is available to each unit of the system, and each signal acts as a clock to trigger internal operations. To minimize input and output connections in the FPGA implementation, each unit generates its own pulse train, using a synchronized clock, once per cycle pulse from the control unit (described above). The cycling unit is a state machine with 100 individual states, divided into 20 main states with five sub states each. The 20 main states are based upon the partition of the original ENIAC's pulse train into 20 time divisions, as shown in Figure 2 below. Pulses do not, however, necessarily occur at the beginning of one of these 20 divisions. The further division of each main state into five sub states provides the necessary resolution to produce a pulse train identical to the original. Pulses are generated by setting signals high during some sub states and low during others.

The cycling units on each unit are not identical to that of the original ENIAC. Since the operating modes are taken care of by the control unit, the individual control units do not have operating mode controls built into them. They do, however, look for the synchronizing pulse from the control unit. The synchronizing pulse arrives at the same time that the cycling unit is generating the cpp, and the cycling unit waits until the reception of the synchronizing pulse before proceeding. The accumulator does not require four of the clocks produced by the cycling unit (4P, 2P, 2'P, and 1P), and space constraints prompted their removal from the cycling units designed for the accumulators. The constant transmitter will, however, require these four clocks, and the cycling unit for

the constant transmitter contains those pulses. The clocks used in the accumulator are described below:

- 10P – The tenp clock cycles through each digit register during digit transmission.
- 9P – The ninep clock is used for digit transmission, reception, and sign unit operation. The ninep clock is gated to transmit the correct number of pulses in the transmitter, is monitored to check for incoming pulses in the receiver, cycles the sign unit when the number being received is negative, and is gated to transmit ninep pulses on the sign line of the digit trunk when a negative number is transmitted.
- 1'P – The onepprime clock is used to correct numbers sent in nine's complement. It is either added to the least significant digit of the number being transmitted by the transmitter, or to the unit's digit upon reception if the clear correct switch is set for the current program.
- cpp – The cpp, or central programming pulse, synchronizes program flow. It is generated by both the cycling unit and the control unit. Since the control unit is specific to the FPGA implementation, it was not used in the original ENIAC. The cpp generated by the control unit is called "synchpulse." The cycling unit on each accumulator waits for the synchpulse before generating its own cpp. This keeps all cycling units synchronized. The cpp generated by the cycling unit also waits for reception of a cpp from the last active unit. The line on which this cpp is received indicates which program to use, and ensures that each unit waits for the last operation to complete before beginning a new operation.
- ccg – The ccg, or clear correct gate, lasts seven pulse cycles and allows carries to occur after receiving pulses.
- rp – The rp, or reset pulse, is used to reset various registers and prepare for the next cycle.

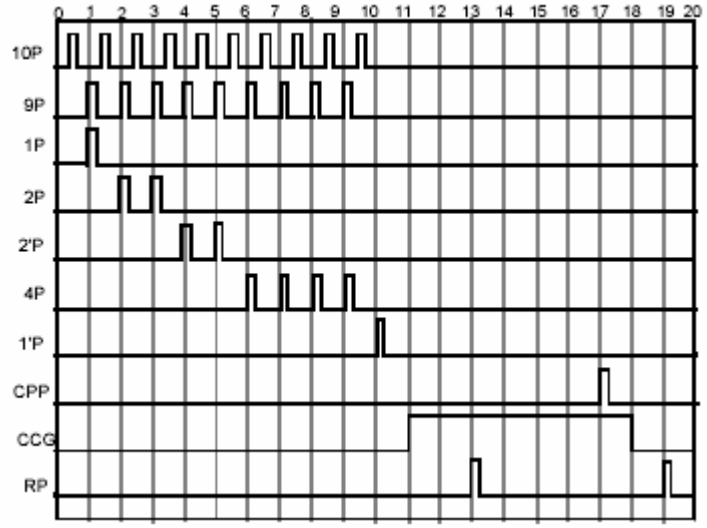


Figure 2: Pulse trains [1].

2.4 Accumulator

The original ENIAC used a set of 10 vacuum tubes for each of 10 digits. The unit is initialized with the first tube on and all others off, to represent zero. If a digit pulse is received, the first tube is turned off and the second is turned on, to represent a 1. The FPGA implementation uses a 10-bit register to represent these 10 vacuum tubes; zeros represent off and ones represent on. Numbers are transmitted between units on a digit trunk. The digit trunk has 11 wires: one for each of the 10 digits and one for the plus or minus setting. Digits are transmitted using the corresponding number of pulses; a 3 would be transmitted by sending three pulses. The plus or minus line transmits nothing if the number is positive and transmits 9 pulses if the number is negative.

The accumulator has 34 switches on a panel similar to that shown in Figure 3. The significant digit switch goes from 0 to 10 and allows the user to select how many significant digits the accumulator uses. If the selective clear switch is turned on and the accumulator receives a selective clear signal from the initiating unit, the accumulator resets its digit registers according to the significant digits setting. Twelve programs can be set on each accumulator. Each program has an operation switch, which can be set to receive from one of five ports, transmit from one of two ports (or both), or do nothing. Each program also has a clear correct switch. If the clear correct switch is set, the accumulator will send the 1'P pulse to the unit's digit if in the receiving mode and clear the accumulators at the end of cycle if in the transmitting mode. Eight of the programs can be set to repeat up to nine times.

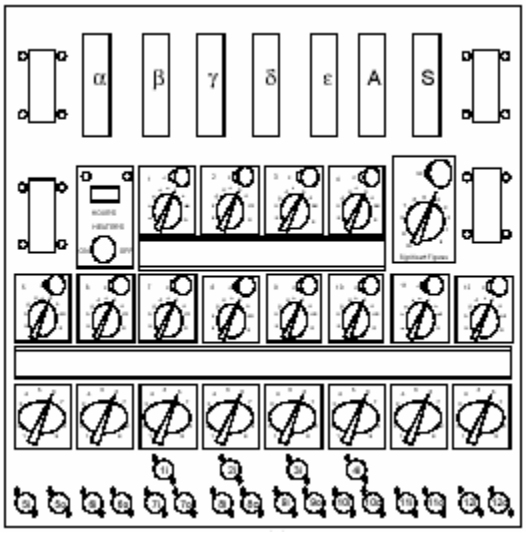


Figure 3: Accumulator panel [1].

The FPGA implementation of the accumulator has five components. The start-up component executes the initialization cycle by reading in switch settings, setting the RAM (random access memory), and resetting registers. The receiver shifts digit registers as digit pulses are received. The transmitter produces digit pulses according to the values stored in the digit registers. The PM Sign Unit keeps track of the sign of the stored number (plus or minus), and the LED component continuously updates the LEDs to

reflect the number currently stored in each digit register. Interaction between the units is shown in Figure 1.

2.4.1 Start-up

When the FPGA system is turned on, the start-up unit runs an initialization cycle. The initialization cycle stores the setting of each switch into memory. The start-up unit runs off the clock from the control unit (not one of the clocks generated by the cycling unit). To read in switch settings, switches are scanned using a state machine with two states per switch. The first state sends out the address of each switch and the second state reads in the corresponding setting. The switch addresses are encoded and pass through an off chip decoder, activating the correct switch. The setting is received through a data bus that is used as program pulse input lines after the start-up cycle.

The selective clear and significant digit settings are stored in registers available to other accumulator components. Program settings are saved in a 9-bit word. Operation switch and repeat switch settings are encoded before saving to decrease RAM size. The first bit represents the clear correct setting, the next four bits represent the operation switch setting, and the last four bits represent the repeat switch settings. The resultant word is saved into RAM at the address corresponding to the program number (for example, settings for program one are saved to RAM address “0001”).

The off chip decoder is also used to send program pulse outputs and select one of 4 digit trunks. If the current program has completed, the receiver sets the done flag high. The start up unit monitors the done flag, and emits the program pulse output on the correct line to the off chip decoder during the cpp. When the done flag is not high and the switches have been read in, the start up unit monitors the receive from register and sets the corresponding line of the off chip decoder high, allowing the selection of the correct digit trunk.

2.4.2 RAM

The random access memory (RAM) is dual port block RAM generated using the Core Generator feature of the ISE software package. Port A is used to write to RAM after setting the write enable bit high, and port B is used to read the RAM during accumulator operation. The RAM has 13 addresses and each word in RAM is nine bits long. The RAM address corresponds directly to the program being run. Words are described in 2.4.1: the first bit corresponds to the clear correct setting, the next four bits to the operation switch setting, and the last four bits to the repeat switch setting.

2.4.3 Receiver

The receiver performs a variety of actions based upon the current place in the pulse train and various register settings. The basic purpose of the receiver is to increment digit registers as pulses are received. When the reception of a number causes a carry (a digit is at 9 and a pulse is received), the receiver increments the appropriate digit. The

receiver saves the values in the digit registers so that they can be transmitted on the next cycle or a number can be added to or subtracted from them. Register clears according to significant digit settings can occur in several different cases in the receiver, and a few other housekeeping occur in the receiver. It is the largest component of the accumulator.

The receiver contains four processes. One process uses the cpp as a clock and is responsible for decrementing the repeat register and correctly setting the done flag. One process runs off the ninep clock and sets a flag whenever there are incoming digit pulses. The other two processes run off a faster clock, called ledclock, which was originally designed to run the LEDs. Since each process can monitor for events in only one clock, it was more convenient to run these two processes off the ledclock and use the other clocks as flags more than clocks. One of these processes sets flags to take care of carries generated when adding numbers in the accumulator. The other process contains the actual digit value registers and all other functionality. The process increments digit values when flags are set by either of the other two processes and runs various special functions.

The receiver uses the following signals as clocks: the cpp, registerreset, onepprime, ccg, rp, and ninep. The current value of each digit, receive flag, and transmit flag outputs to other units are updated independent of any clock.

The receiver stores the current value of each digit. The numbers are stored as 10-bit registers and numbers transmitted as pulses over a digit trunk as described above. The pulses are transmitted on the ninep clock. Each time the ninep clock goes high, one process in the receiver checks each line of the digit trunk. If the line is high, a pulse is being transmitted. The receiver sets “pshift” high, indicating that there is a shift due to a pulse. It also checks the value of the last bit of the corresponding digit register. If the last digit is high, the value currently stored is nine, and the pulse input should cause a carry to the next digit. The corresponding flag bit is set.

Carries, as necessitated by flags set while transmitting, are preformed while the ccg gate is high in their own process. The process is run on the rising edge of the ledclock and runs from the least significant digit to the most significant digit using a state machine. If the flag for a particular digit is high, “fshift” is set high. The last bit of the digit is checked, and if it is high, “cshift” is set high for the next digit. If “cshift” must be set high, the next digit’s last bit is checked, and if it is high, the next “cshift” is set. The process continues until no more carries are triggered. “fshift” is used for carries triggered while receiving pulses from the digit trunk, while “cshift” is used for carries triggered while implementing those carries. Flags from each state are reset in the next state.

Each bit of all three shift flag registers are “or”ed together outside of any process. If any of the bits are high, then the final shift register will have a high bit. The main process increments the value stored in the registers based upon this final shift register by shifting the first nine bits of the digit register to the right and moving the tenth bit around to the first bit. The shift simulates the ring counters used to store and

increment digits in the original ENIAC. The main process also controls all of the special functionality of the accumulator.

When the cpp is high, the receiver component resets flags, loads the next program from memory if necessary, and sets the registerreset flag as necessary. The registerreset flag is set if the current program is a transmit cycle and clear correct is set, the accumulator's selective clear switch is set and a selective clear signal is received from the initiating unit, or the initialization cycle is being run. If register reset is high, each digit is reset according to the accumulator's significant digit setting. These settings occur within the main receiver process. Program repetitions are monitored in a small separate process that uses the cpp as a clock. On a cpp event, the repeat register is decremented. When the register reaches 0, the repeat register resets and sets the done flag high.

If clear correct is set high and the unit is receiving, the unit's digit is incremented by one on the 1'P clock. Negative numbers are incremented using nine's complement, but the accumulator works in ten's complement. The transmitted signal is usually corrected to ten's complement according to the significant digits setting, but in some cases this will not occur and the clear correct switch is set preemptively. In this case, the unit's digit is incremented (no matter the significant digit setting) and so the option is ineffective unless the significant digit setting is 10.

When rp is high, the data loaded from memory during cpp is distributed as appropriate. The data is stored as a 9-bit word, which then must be split into three pieces (clear correct, operation mode, and repeat switch setting). The first bit of the word is the clear correct setting, the next four bits are the encoded operation mode setting, and the last four bits are the encoded repeat switch setting. Flags receive, transmit, aout, and sout are set according to the operation mode setting (aout and sout signal the output through the A or S ports, for addition or subtraction). Register receivefrom is also set according to the operation mode setting; if the accumulator is set to receive, then one of five receive ports (alpha, beta, gamma, delta, or epsilon) must be chosen. The null operation (O on the accumulator panel) is performed by setting both receive and transmit flags low. The encoded words for each operation mode are shown in appendix 1. [3]

2.4.4 Transmitter

The main function of the transmitter is to turn the zeros and ones in the digit registers into a set of pulses to transmit on the digit trunk. Each digit register is shifted to the right on the tenp clock until the '1' in the register is reached. At that point, a transmission flag is set and shifting continues to ensure that the number in the register returns to its original state. The transmission flag is combined with the ninep clock so that the number is correctly transmitted in nine's complement. If the number is positive, the nine's complement equivalent is just the number, so a 3 would be transmitted as three pulses. However, on transmission of a negative number, nine minus the digit value pulses are transmitted, with one pulse added to the least significant digit on the onepprime signal; it is thus transmitted as 9999999999 – the number + 1. In some cases, addition of the onepprime pulse is missed by the transmitter. Programmers set the clear

correct switch on the receiving accumulator in anticipation of this mistake. This the only function preformed by the transmitter.

2.4.5 PM Sign Unit

The plus-minus unit tracks the sign of the number stored in the accumulator. On transmission of a negative number, it sends nine pulses on the sign line of the digit trunk. The nine pulses change the sign of the receiving plus minus unit nine times, so the sign is flipped upon completion. The unit gates the ninep clock so that no pulses come out on the sign line if the number being transmitted is positive. If the most significant digit in the accumulator generates a carry, generating overflow, the sign unit flips. The overflow could be due to a true overflow, but is more likely the result of a subtraction using tens complement. In the case of a subtraction, the correct answer is produced by switching the sign of the number. If the sign of an operation is unexpected (for example, adding two numbers produces a negative number), a true overflow occurred. [3]

2.4.6 LEDs

The LED display serves no purpose in the actual accumulator function. It is, however, an important debugging tool, and a similar display, using lamps behind halved ping pong balls, was present in the original ENIAC. The LED display is updated in real time with the current value of each digit register. Each column of the display represents a digit, and each row represents a value from zero to nine. The LED unit scans through each column and determines the value of the corresponding digit. It then selects the row corresponding to that value. The column and row address are then output to light the correct LED. To prevent the row value being output before the column value, or vice versa, the column value is set to a dummy address, the row value is set, and then the column is set correctly. The LED unit runs at a clock 10 times fast than the cycling unit, so that all 10 digits (columns) can be updated during each cycle and the displayed digit is the value currently stored.

Two additional LEDs were added to the board, but are not controlled by the LED unit. One LED lights up when the number stored in the accumulator is negative, similar to a light on the original ENIAC. The other LED lights up when a program output pulse is emitted and is included for debugging and demonstration purposes. These LEDs direct outputs of the sign and start up components, respectively.

2.5 Constant Transmitter

The constant transmitter allows the programmer to load a constant value directly to an accumulator by setting a variety of switches. The value indicated by reading in the switches is then generated using a combination of the 1P, 2P, 2'P, and 4P pulses. The constant transmitter can be generated using combinational logic in VHDL.

The constant transmitter has two panels. One panel is used to choose the digit output terminal and the other to choose the constant. The constant is chosen by setting

between five and twenty dials. Each dial has the values zero through nine and represents a digit of the constant. The dials are arranged in groups of five, and the number to be transmitted can be composed from some combination of those groups. [1]

3. IMPLEMENTATION ON CHIP

The final implementation will be on a series of FPGAs with large panels similar to those of the original ENIAC. To test the work performed thus far, a PCB was designed to interface with a Xilinx IIe chip mounted on a Digilent Digilab II XL circuit board by Zheng Yang of the University of Pennsylvania. FPGA pins were assigned using the Digilent instruction manual and addresses for switches currently on the board are shown in appendix 2. [4]

The PCB board is designed for testing purposes, and does not include the functionality of the full accumulator panels. The board has four program mode switches, four clear correct switches, and two repeat switches, so it can run four different programs, two with repeat capabilities. It has two transmit ports (for each addition and subtraction, like the original ENIAC) and four receive ports (as opposed to the ENIAC's five). It has ports to transmit and receive coordinating cpps. There is a significant digits dial (with choices from 0 to 9) and a selective clear switch. Nine LED banks with 10 LEDs each display the value currently held in the accumulator. Two additional LEDs give the current sign of the accumulator value and the current status of the program pulse output line. The PCB board is shown connected to the Digilab board in Figure 4, and the I/O structure is shown in Figure 5.

The Spartan II FPGA has 96 input and output pins, but the full ENIAC accumulator panel requires over 100 input and output lines. Using encoded signals and a shared data bus drastically decreases the number of lines, and thus the number of pins, required to implement the design. The decoder shown in Figure 6 performs the switch selection, digit trunk input selection, and program pulse output depending upon the current cycle. The decoder takes a 4-bit binary address and converts it to one of 16 choices.

During the start-up cycle, the start-up unit scans through the addresses of all the switches and outputs them one at a time through the decoder. When a switch's address is output by the decoder, that switch becomes active. The switch's data then floods the data bus and is input to the FPGA. When the next switch address is selected, the last switch becomes inactive, the new switch becomes active, and the correct data is received. The significant digits switch, four operation mode switches, two repeat switches, and four clear correct switches are accessed in this manner. The significant digits switch has 10 options, the operation mode has eight options, and the repeat switches have 10 options; their outputs are encoded to four bits within each switch. Each clear correct switch has only two options (high or low), but all four are accessed at the same time and so require all four input lines. The selective clear switch does not use this system, but is directly input to the FPGA as it has only two options (high or low).

During receive cycles, the receiver unit of the accumulator pulls down the current program's settings from RAM and determines the digit trunk from which to receive digit pulses. It then outputs the address of that digit trunk to the decoder. The decoder sends an output enable signal to the correct digit trunk based upon that address. The output enable signal allows the data from that digit trunk to reach the digit trunk data bus. All four receive digit trunks use the same pins into the FPGA. However, only one output enable can be active at a time, and thus only one digit trunk's data arrives per cycle. Two pins output program output pulses at the end of a program to activate the next program; there are only two program output pins, as only programs with repeat switches send the program output pulses. The program pulse output pin for program four has been connected to an LED for debugging and demonstration. The decoder also has two empty output lines; one of these lines must be selected when none of the other functions are desired.

The data bus for the switches serves an additional purpose. After the switches have been scanned and their values stored, the buffer shown in the figure is activated, allowing the program pulse input lines to use the data bus. When a program pulse input is received on one of the four lines, it uses the data bus. The LEDs have eight output lines: four for column selection and four for row selection. There is also an LED which is on when the accumulator value is negative and off when it is positive. The additive and subtractive digit trunk outputs each have 10 lines, 9 for digits and 1 for the sign.

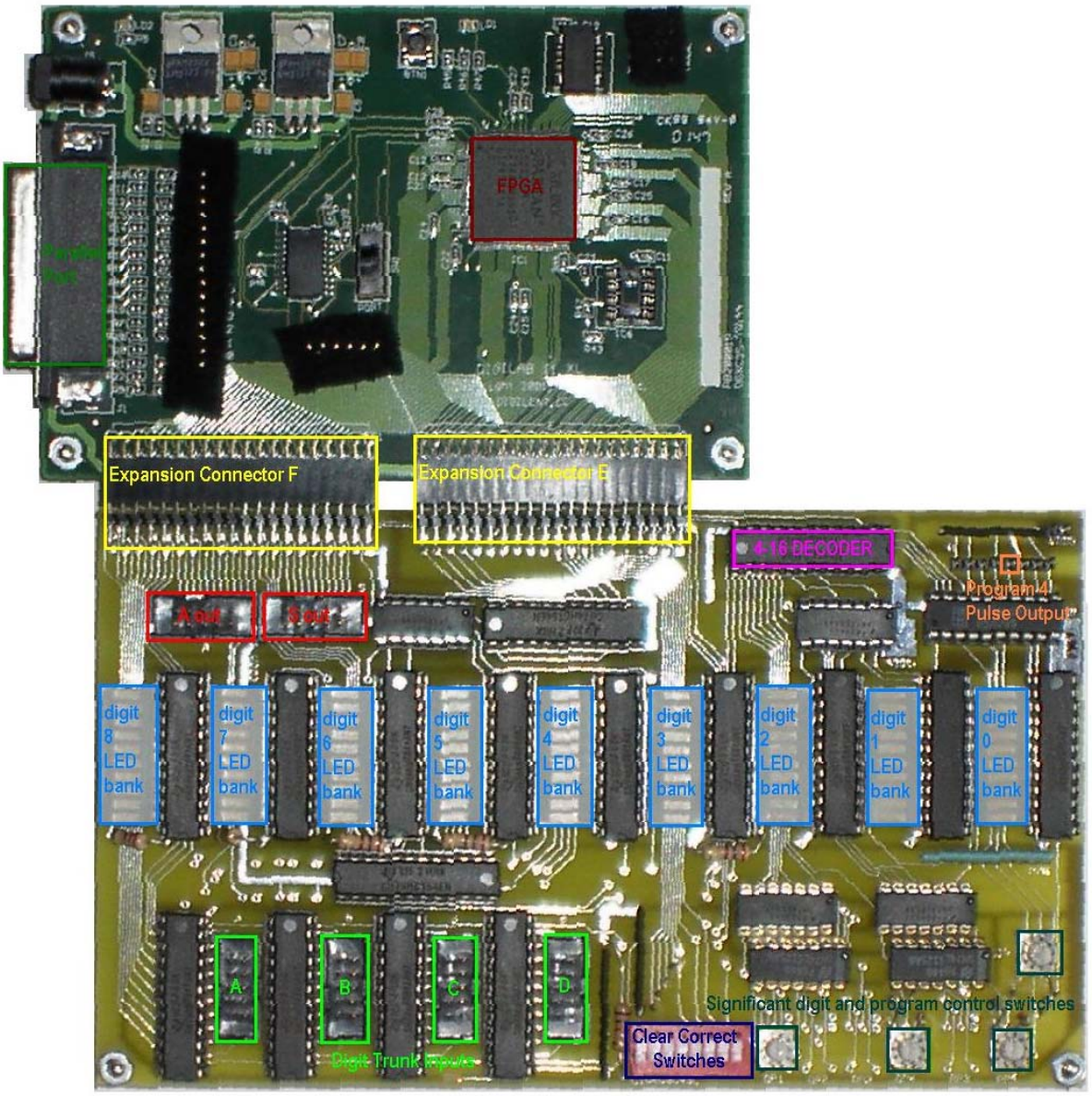


Figure 4: The board upon which the accumulator and cycling unit were implemented.

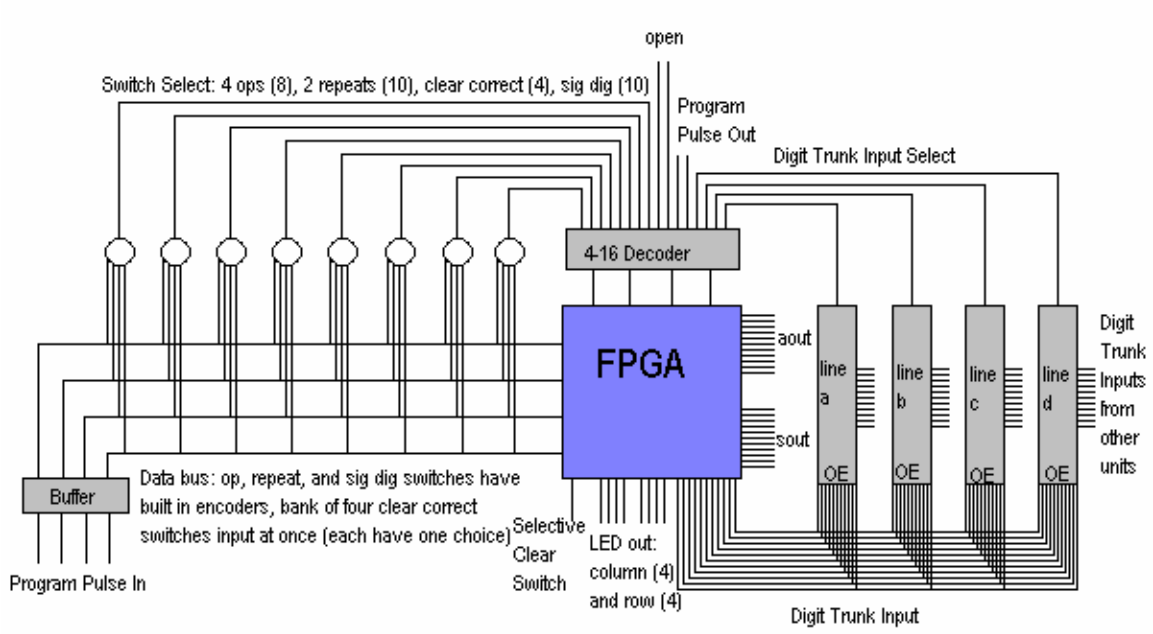


Figure 5: Connections between the FPGA and other devices.

4. RESULTS

The accumulator is fully functional in logical simulations, as shown in Figure 6. Timing simulations and actual implementation upon the board are less successful. The first units implemented on the board were the LED unit and clock divider. The LEDs can be directly loaded with a number by replacing the connection with the receiving unit by a constant value. During this implementation, the clock divider was used to decrease the FPGA device. Slowing down the clock is very useful for debugging.

The cycling unit and receiver were implemented next. The cycling unit is required for operation of all units other than the LED unit. The accumulator was first set to display a certain value and then reset itself at the end of each addition cycle, to ensure the unit was functioning. The accumulator was then set to receive nine pulses per addition cycle in the unit's digit by tying the digit trunk to "00000000001," where the first bit represents the sign line, and the next 10 bits represent the 10 digits from digit 9 down to digit 0. The accumulator was thus expected to count by nines from zero to 9999999999.

Counting on the unit's digit was relatively simple to debug. The carry operation was not as successful. The original design used a second set of ring counters to increment digits when flags were high during the ccg. This design was very similar to that of the original ENIAC, which simply let the carries run through the digits until the value settled. On the FPGA, however, the design created race conditions, and values did not have time to settle before they were tested. A single flag tended to cause the next digit to increment twice, and carries that triggered additional carries were not recognized. The current receiver layout presented the solution to this flaw. When counting by nines, the accumulator functions smoothly up to the transition between 999 and 1000. Each

time that a carry is triggered in the thousand's digit, the accumulator stops for approximately seven minutes. It then resumes and runs smoothly, with the correct answer for the last operation. The accumulator was run for a long period of time to observe the transition from 9999 to 10000. The transition again took approximately seven minutes.

The start-up unit was tested by reading the value of the switches into memory and setting the LEDs to display values corresponding to the contents of RAM. The original design led to race conditions between selecting a switch and reading the corresponding input on the data bus, but using a state machine to step between choosing switches and reading inputs eliminated the problem. The sign indicator LED and program pulse output LED were added to the PCB to check for further functionality. Subtraction was simulated tying the digit trunk input to "10000000001." The first bit is '1,' so the sign of the number is negative. The number being received is thus -9 (since the pulse in the unit's place is received 9 times per addition cycle). The LEDs correctly displayed the value in the accumulator as 9 and the sign as negative. The clear correct switches were tested by setting the digit trunk to "10000000000," such that the unit was receiving "negative zero," while the clear correct switch was high. The accumulator incremented the unit's digit once per cycle on the onepprime pulse, as expected. The program pulse output functionality was tested by setting the repeat switch to various values and monitoring the program pulse output LED; it flashed after the correct number of program repetitions.

The sign LED and program pulse output LED provided unexpected debugging information. A negative sign is represented by nine pulses during the first half of the addition cycle (on the ninep clock), with the sign flipping each time a pulse is received. The sign thus flips on the ninep, and the sign LED flashes on the ninep. When the accumulator was allowed to run to the point that the seven minute delay occurs, the digit LEDs again stopped. The sign LED, however, continued flashing. The program pulse output LED continued flashing at the correct times. The accumulator, then, continues functioning while the logic delay occurs.

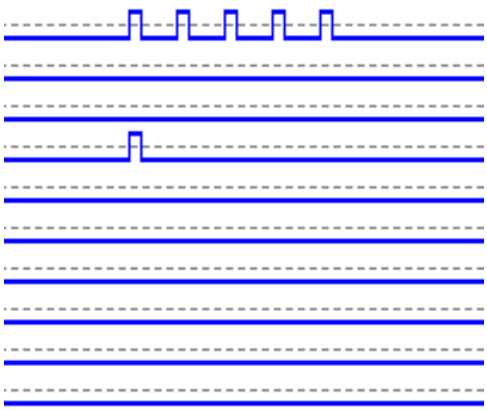


Figure 6: Transmitter outputting pulses in a logic simulation. The value in the accumulator was obtained by adding 6 to 999, and the output of 1005 on the digit trunk is shown, with the least significant digit line shown on top.

5. DISCUSSION AND FUTURE WORK

Difficulties experienced in the project stemmed from the adherence to the original pulse train and resultant timing issues. Running off the pulse train increased program complexity and decreased productivity. The original ENIAC was more efficient and fail proof than this implementation. The FPGA chip used for implementation was a tight fit, and a larger chip would improve system performance.

Space constraints were a constant limitation. The FPGA used has 30,000 logic gates; devices with up to 500,000 gates are commonly available. The code was optimized several times to fit onto the chip. The optimization included converting behavioral statements (if…else statements) into combinational logic statements and reducing the number of registers used in the code. The carry operation was originally implemented in a separate entity than the receiver, but the two entities were combined to reduce size. The optimization significantly reduced program size, but the chip still did not contain sufficient space. After changing compiler preferences to emphasize optimization during the translation and mapping processes, the entire implementation fit onto the desired FPGA.

The problems experienced during the counting trial suggest timing will be an issue. The reason for the seven minute delay after 110 addition cycles is unclear. The sign and program pulse output LEDs show that the cycling unit, sign component, and even some of the receiver continue functioning. When the significant digit setting is set so that the second least significant bit is five upon start up, the seven minute delay occurs after 110 cycles, not at the value 993. The delay appears unrelated to the cycling unit or carry operation. The delay will cause major synchronization problems with the other units as the program pulse output continues during the delay.

Implementation with the full input and output capabilities will require several changes to the program. The start-up unit, in which the switches are read, needs several changes for the full implementation. It will need to be redesigned for the encoders, decoders, and I/O structure of the full panels. The words stored to RAM are designed for the full implementation, so RAM and the word convention associated with it will not need to be modified. The receiver decodes information from RAM, and assumes the full data format that will be needed for the full unit. Addresses will change with the full implementation, but the current format should be sufficient for the required increased input and output.

The final product will include a system of three accumulators and a constant transmitter. The constant transmitter was not designed for this report. The cycling unit designed for the accumulators does not contain the 4P, 2P, 2'P, and 1P pulses that are unique to the constant transmitter. However, the first cycling unit was designed to work for any unit and thus includes these pulses. The constant transmitter should be a relatively simple piece to design and implement. The interaction between three accumulators will need to be thoroughly tested and debugged, as it has been performed only in simulation for this report.

6. CONCLUSIONS

The work described in this paper achieved the original goals for the summer project. Two accumulators were not hooked together, preventing full testing, and debugging may still be necessary for the completed units. The project proved more difficult than expected due to timing issues and unfamiliarity with the Xilinx platform. Learning the architecture of the original ENIAC took a significant amount of time, but the architecture is relatively straightforward. The architecture was designed for the hardware available at the time. Implementing it on modern hardware was difficult, as the concept of a system running off a set of 10 clocks is unusual and the FPGA is not designed to run on more than one clock. Significant progress toward the final product was made on this project. The final product will provide an excellent learning opportunity for people to experience how the first general purpose computer was programmed.

7. ACKNOWLEDGMENTS

I would like to take this opportunity to thank my advisor, Professor Jan Van der Spiegel, and Zheng Yang, the graduate student who helped me with this project, including designing and building the PCB on which I implemented my work. I would also like to thank the National Science Foundation for providing funding for my project through the SUNFEST REU program. Finally, I would like to thank my fellow SUNFEST student, Nicole Dilello, and my lab partner from previous projects, Seth Jacobson, for their advice in various stages of the project.

8. REFERENCES

1. J. Van der Spiegel, J. Tau, T. Ala'ilima, and L. P. Ang, The ENIAC: History, Operation, and Reconstruction in VLSI, *The First Computers—History and Architectures*, MIT Press, eds. R. Rojas, 2000, p.121-178.

2. J. Tau, ENIAC on a Chip: The Monolithic ENIAC, Master's Thesis, Philadelphia, PA, University of Pennsylvania School of Engineering and Applied Science, December 1996.

3. A. Goldstine, Technical Description of the ENIAC, Part I, Philadelphia, PA, University of Pennsylvania Moore School of Electrical Engineering, June 1, 1946.

4. Digilab 2 XL Reference Manual, Pullman, WA, Digilent, Inc., May 7, 2002, p. 6.

9. APPENDICES

APPENDIX 1: OPERATING MODE ENCODING

Switch Value	Port	Mode	Encoded
0	None	Null	0000
1	Alpha	Receive	0001
2	Beta	Receive	0010
3	Gamma	Receive	0011
4	Delta	Receive	0100
5	Add	Transmit	0101
6	Subtract	Transmit	0110
7	Add and Subtract	Transmit	0111

APPENDIX 2: DECODER SWITCH ADDRESSES

Address	Function
0100	Clear Correct Switches
0101	Significant Digits Switch
1011	Program 4 Repeat Switch
1110	Program 4 Operation Mode Switch

SONY AIBO MOTION CALIBRATION AND MONITORING/CONTROL SYSTEM

NSF Summer Undergraduate Fellowship in Sensor Technologies
Microsoft Fellowship Recipient
Brian Corwin (Computer and Telecommunications Engineering) – University of Pennsylvania
Advisor: Dr. Daniel Lee

ABSTRACT

The Sony Aibo are robotic dogs used to test new ideas in artificial intelligence, computer vision, and robotic motion through the application of having the dogs play soccer. An important part of building the system the dogs use is debugging. There are two key parts to this process: calibrating the dog's motions, and monitoring the dog's sensory outputs and control input. Both parts of the process help the developer to understand how the dog perceives the world and acts in it. Each of these questions was addressed with a different system. A magnetic positional sensor device was used (along with dog control and data processing programs) to obtain calibration information about the dog's various walks and kicks. A client/server system was developed to allow multiple users to access different sensory outputs, the dog's command input (only one user at a time), and the dog's standard output in order to facilitate system development and debugging.

Table of Contents

1. INTRODUCTION

The Sony Aibos, like any intelligent agent, interact with the world through a sequence of perception, evaluation, and action. In this project, the Aibo's actions were calibrated, and a system to monitor the Aibo's perceptions (and also control its actions directly) was created. Calibrating motions determines how commanded motions (and their expected results) compare with the actual motions performed. Information about this can be used to adjust the system, so the dogs' expected actuations are carried out as accurately as possible. Controlling the dog (or letting the dog control itself) while monitoring its different sensory and system outputs is an important step in understanding and debugging the dog's sensor, information processing, and self-control systems. In other words, in order to understand what the dog is doing, one must "see" the world as the dog does by viewing its sensory outputs. A system using a sensor device (for precise positional information), associated program, and calibration scripts was used for motion calibration, and a client/server program was created to allow for dog control and observation of dog outputs by multiple users at once. In the future, it is hoped, these systems will be integrated to give the developer more tools.

This paper outlines the design of these two systems as well as the results from two iterations of motion calibration. Section 2 provides background on the hardware and operating system of the Aibo, the software systems developed by Dr. Daniel Lee for the Aibo to use for soccer, and the principles, hardware, and operating system for the Nest of Birds (NOB) sensor device used in the motion calibration. Section 3 discusses the creation of a control program for the NOB as well as calibration scripts. The section also covers the results of some of the motion calibrations done for the dogs. Section 4 covers the design, implementation, and use of the monitoring and control server for the Aibo. Section 5 covers the design and use of the first prototype client created to interact with the server. Discussion and conclusions are presented in Section 6 and future work and recommendations in Section 7.

2. BACKGROUND

2.1 Sony Aibo and Robocup Soccer

The Sony Aibo (see Figure 1) is a robotic dog used by universities for research, particularly to participate in Robocup Soccer. This is an event sponsored by the Robocup Federation in which robots of different types (including the Sony Aibo) play soccer; in the Aibo case the game is played with teams of four [1, pp. 1-4]. All perception, decision, and actuation activities are performed by the Aibos themselves, although communication and coordination is allowed (with strict rules) between dogs through a wireless network [1, pp. 1-4].

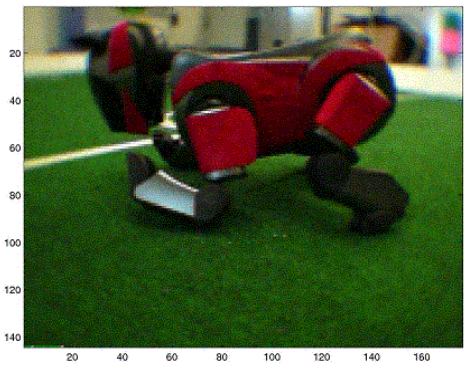


Figure1: The Sony Aibo.

These tournaments create an activity through which new ideas about computer vision, robotic motion, and artificial intelligence are developed and tested. After the annual tournament, each team writes a report describing their system, so that groups learn from one another and advance scientific development.

The Sony Aibos are themselves a powerful platform on which to work. Each dog has four legs, each with a motor in the hip, knee, and ankle that allow for three degrees of freedom in each leg [2, pp. 7-12]. The dog's head and neck has three motors that give it three degrees of freedom (pan, tilt, and roll) [2, pp. 6-12]. A camera in the dog's head has a focal length of 161 pixels (2.18 mm), operates at 25 frames per second, and takes images that are 352 x 288 pixels in size [2, p. 20]. The Aibo also has a short-range distance sensor, two-channel microphone, and speaker [2, pp. 19-21]. Each dog also has a slot for an IEEE 802.11 wireless Ethernet card. The Aibo operating system also has a built in Application Programming Interface (API) using Sony's OPEN-R technology. The technology allows custom programs to be written that operate the dog [3], running off memory sticks inserted into a slot on the dog. A TCP/IP networking stack allows the Aibo to communicate with other Aibos and computers over a wireless Ethernet [4].

2.2 Aibo Software

Dr. Daniel Lee and colleagues have created an extensive software system that runs the Aibo team. At the lowest level, code written in C++ interacts directly with the Aibo's API to handle some of the basic functionalities of the dog, such as its vision system. On top of this is the core high-level decision-making program consisting of a state machine [5, p. 1]. Previously this layer was implemented in C/C++, but a more flexible solution was found in running a PERL script on a PERL interpreter embedded in the Aibo [5, p. 1]. PERL had several advantages including the ability to recover from errors, ease of development, and extensibility that allowed the low-level C++ functionalities to be called through a limited interface, allowing for maximum implementation encapsulation [5, p. 2].

2.3 Important Aibo Inputs/Outputs

Although the Aibos perform autonomously during competition, the system in Section 2.2 does have inputs and outputs to which other computers can connect, assisting in debugging and development (communication with the Aibo is described in Section 2.1). The system connects specific outputs and inputs to different ports in the dog's networking stack; for example, port 59000 is the standard output for the dog (where system information and error messages are shown), so if another computer connects to a dog's IP address on port 59000 it can retrieve these messages. In addition, 1001 is the input port, 6006 is the "blob vision" port (see Section 2.4), and 6000 is the camera port. A special format for the packets sent to these ports is specified by the OPEN-R protocol. Each packet has a total size (integer) value telling the total number of bytes in the packet and a number of data (integer) value that is the number of data elements in the packet, in addition to the data itself. For the input, the data is a string that represents a command to

be executed by the dog. This takes advantage of a useful feature of PERL: Since PERL is interpreted commands can be executed on the fly [5, p. 2].

2.4 Aibo Vision System

Each image from the camera comes out in YUV pixel format. Existing computer vision systems cannot deal with the complexity of even the simplest photograph, so in order for the Aibo to extract relevant information from the images, the photos must be simplified into a form that the Aibo's processing power and known computer vision techniques can handle. The simplification system is based on the needs of the Robocup soccer field; on the field important visual cues for the dog are distinct colors (the ball is bright orange, the goals are yellow and blue, etc.). The distinct colors give the dogs specific visual cues to look for; i.e., they only need to recognize this particular subset of visual stimuli.

Each pixel of the image is mapped to one of the important cue colors (orange, blue, green, etc.) or to nothing, in a method similar to that used by Carnegie Mellon University's Aibo team [6, p. 2]. In essence, every pixel that is close to green is mapped to green, every pixel that is close to orange is mapped to orange, etc. and everything else is ignored. Although the number of colors has been limited there are still thousands of individual pixels. The image is processed again, so that an area with a large concentration of a particular color pixel is united into one box.

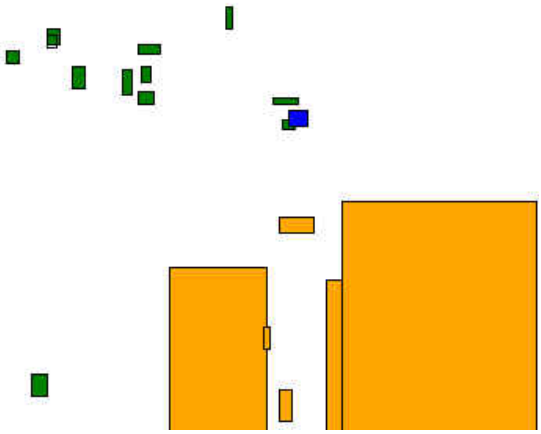


Figure 2: Blob view image.

Each box (which is also called a "blob") has certain characteristics: the position of each corner and of the centroid (the center of mass of the pixels). After processing, the image becomes an array of bounding box structures that stores the information on all "blobs" in the image. This is simple enough for the dog to process; for example, in order to find the ball the Aibo simply looks for an orange blob. Figure 2 shows a blob view image.

2.5 Nest of Birds (NOB) Design, Operating System, and API

The NOB (see Figure 3) is a magnetic sensor device with a central unit, a transmitter, and four sensors. The sensors and transmitters are attached to the central unit by cables. The device is connected to a computer via a serial port in its central unit. The NOB works by magnetic field emission and electromagnetic induction. The transmitter and the sensors all have electrical coils in the x, y, and z directions [7, pp. 2-3]. The

transmitter turns on each coil one at a time, and the magnetic field emitted by the coils induces an electric current in the coils of the sensors [7, pp. 3-4]. The amount of induction in each coil is inversely proportional to the distance to the transmitter and orthogonality of the angle of the coil to the field. Thus by comparing the induced voltages in each sensor coil (x, y, and z) when each of the transmitter coils (x, y, and z) is on, the device can tell the relative position (x, y, and z) of the sensor to the transmitter and the azimuth, elevation, and roll of the sensor relative in the coordinate system of the transmitter. The NOB has a range of 3 feet from the transmitter [8, p. 8].

Figure 3: Nest of Birds device

The API of the NOB consists mainly of commands represented by single-byte positive integers. Sending a certain byte to the NOB via the serial port tells the NOB to perform a particular action; for example, writing number 66 asks for a data point from a sensor [8, p. 88]. Many functions in API are superfluous to the simple system needed for motion calibration; so only key features will be discussed.

The NOB has certain parameters that help control its actions (all can be examined and some can be changed): the error status of each bird, the address of the transmitter, etc. The examine/change functions are used to manipulate these parameters, and thus control the actions of the NOB [8]. In this way, for example, the transmitter mode, bird hemisphere (direction of the coordinate system of the sensor), and bird data mode are all set. Each bird (sensor) has an address; these addresses are from 1 to the number of birds in the device (here, 4). The status of each bird can also be obtained, which will tell much about its state: is it master or slave, running or not, error or none, etc. The auto-configuration function tells a bird to be the master bird, through which all communications to other birds must go, and how many birds there will be. Without the auto-configuration, only one bird can be used at a time [8, pp. 131-134].

3. AIBO MOTION CALIBRATION SYSTEM

The motion calibration system was designed around the NOB magnetic sensor device in two parts. First, a C++ program was designed to initiate and configure the NOB and to obtain data from the device when asked. Second, Matlab scripts were written to obtain data from the NOB program and send motion commands to the dogs to get raw data, to process the data, and to plot it. Both of these programs were designed to work on a Linux computer. The C++ program was a completely separate program from

the Matlab scripts. The Matlab script simply executed the C++ program and redirected the program's output so that it could be parsed and used by the Matlab script.

3.1 Nest of Birds Control Program

The NOB consists of four sensors and a central unit (including the transmitter), so this kind of configuration was modeled using C++ classes, which is consistent with the principles of good object-oriented design [9, p. 85]. A class called Bird was created that contained data pertinent to an individual sensor, such as the bird address, hemisphere, data mode, and status. The class has methods that are important to a sensor. There is an auto-configure method, set data mode, set hemisphere, get status, get error, and get data packet. The system is made to work only in the Point/Angle data mode.

A class called Nest is a model for the whole device. It contains an array for all the Bird objects in the Nest as well as data members and methods of its own that are important for the NOB as a whole. It contains the transmitter address and mode (as well as methods to get and set these). This class constructor initializes the whole system by connecting to the NOB, configuring it, and creating a certain number of Bird objects based on the number used in the NOB (4 was always used).

The main function of the program creates a Nest object with 4 Birds and then sets up the Birds (all have upper hemisphere and position/angle data mode) and the transmitter (pulsed mode and address is bird 4). Then the program then simply reads points from the device. The positions and angles are 2-byte integers (± 32767). The data bytes read from the NOB are first processed (bit manipulation) and then converted to the appropriate units (centimeters and degrees). The positional integers are relative to 3 feet. Thus to convert to centimeters:

$$(pos/32767)*(36\text{ in}/3\text{ ft})*(2.54\text{ cm}/1\text{ in}) = 0.0027906125*pos$$

Angle integers are relative to 180°, so to convert to degrees:

$$(angle/32767)*(180\text{ degrees}) = 0.0054933332*angle$$

3.2 Aibo Motion Calibration Matlab Scripts

There are three basic steps to the motion calibration: 1) Obtain calibration data and store it to a file, 2) process the data and store it in another file, and 3) graph the results. The dog can be commanded through a Matlab interface. For this reason and for ease of development, Matlab was used to create these parts of the calibration software. Similar Matlab programs were developed to calibrate the dog's kicks.

The first program runs the experiment. One of the sensors is attached to the dog and the dog is placed close to the transmitter (about a foot away, to keep the sensor within transmitter range). The NOB is started and an initial position reading is taken.

The dog then walks in the way specified on the command line, x, y, and θ (in millimeters and degrees). Another reading is taken after the dog is done walking, and both the start and end readings are written to a file. The dog then tries, through a simple negative feedback mechanism, to get to the start position, so that the dog does not eventually drift out of sensor range. The dog repeats this process many times.

The data must be processed for two reasons. First, the motions that the dog is commanded to do are all relative to the end of his head and not where the sensor is (taped to a ruler on the dog's side to keep the sensor away from the magnetic interference of the dog's servo motors). So the sensor positional information and the known distance from the end of the head to the sensor must be used to get the head position. Figure 4 shows a simple diagram of the dog; the distances d1 and d2 have been measured for the Aibo and Xsen, Ysen, and θ are obtained from the NOB. The transformation from sensor to head coordinates is:

$$Xhead = Xsen - d1*Cos(\theta) - d2*Sin(\theta)$$
$$Yhead = Ysen - d1*Sin(\theta) + d2*Cos(\theta)$$

The second reason is to get the data in the right coordinate system. The x, y, and θ displacements should be in the coordinate system of the dog's start position, with the end of its head as the origin. This has two implications. First, x and y translations must done to get the dog's head to be the origin rather than the transmitter. Second, the dog's coordinate system maybe at an angle with the transmitters, so a coordinate system rotation is needed (the θ change is the same for all reference frames: $\theta_{FINAL} - \theta_{INIT.}$):

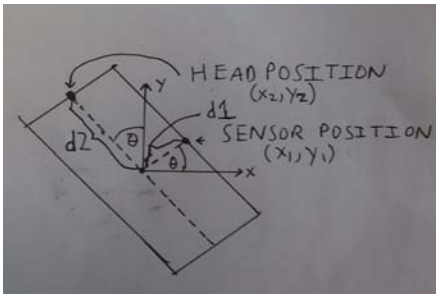


Figure 4: Coordinate system diagram.

$$Xmove = + (X_{FINAL} - X_{INIT})*Cos(\theta_{INIT}) + (X_{FINAL} - X_{INIT})*Sin(\theta_{INIT})$$
$$Ymove = - (X_{FINAL} - X_{INIT})*Sin(\theta_{INIT}) + (Y_{FINAL} - Y_{INIT})*Cos(\theta_{INIT})$$

3.3 Aibo Motion Calibration results

The C++/Matlab system was used to calibrate the x, y, and θ of the dog. Only one direction was done in a trial (e.g., the dog would never do a forward and then walk left). The x and y motions were tested from -20 to +20 cm in 2 cm increments and θ from -100º to +100º, in 20º increments, with 10 to 20 data points for each walk.

Two important pieces of information needed to be obtained from the data: the ratio between the value of the real action and the commanded action, and whether there were any significant aberrant movements (e.g., if the dog was walking forward, did it drift right or left, or turn). These types of conclusions are much harder to quantify since there are no expectations for the pattern they should follow; the ratios of

command to actual movement are expected to have a linear relationship. To find these ratios for each movement type (x, y, and θ) the average actual displacement for a particular distance command was found, and a linear regression was performed on these data (see Table 1):

Motion	Linear Regression
Y (Forward/Back)	0.7757
X (Right/Left)	0.7172
Theta (Counterclockwise/Clockwise)	0.9107

Table 1: Results from linear regressions on calibration data.

These regressions were not sufficient, since the graphs clearly show that for all motion types the results for positive and negative motions were very different (see Appendix A). Therefore, separate regressions were done on the positive and negative motion data (see Table 2):

Motion	Negative Reg.	Positive Reg.
Y (Forward/Back)	0.7111	0.8404
X (Right/Left)	0.6946	0.7398
Theta (Counterclockwise/Clockwise)	0.8987	0.9249

Table 2: Results from linear regressions on positive and negative sections of calibration data.

For the forward and side motions the drifts were small (averages were all less than 5° or 2 cm in magnitude). Also, the linear regressions for each were all less than 0.1, so they showed no trend. The turns actually caused large negative x displacements that increased with the turn's magnitude. Also, the right turn caused a positive x (right) displacement, and the left turn caused a negative x (left) displacement (see Figure 5):

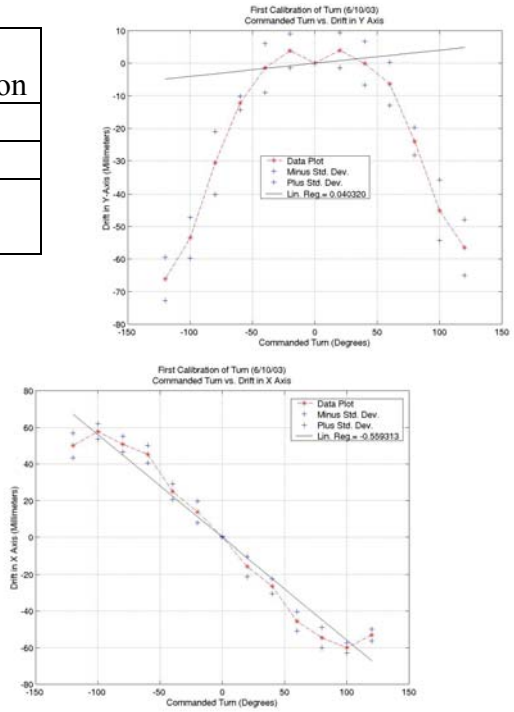


Figure 5: Graphs of x and y displacement during turn calibration.

These results show that the dog's turns, which were supposed to be performed around the tip of the dog's head, were actually being performed at a point approximately 5.8 cm behind this. This was later confirmed visually.

Three kicks the paw, and right and left kicks were calibrated. The x, y, and θ displacements for 30 kick trials were obtained for each kick type. For the paw kick, as expected, the dog never turned more than 6° and never moved more than 2.5 cm forward, back, left, or right. For the left and right kicks the dog would move to the left and right on

average 15-16 cm and 2-4 cm backwards. The dog also changed its angle by 50°-60° each time. This again was expected; the dog must turn about its center to kick the ball to the side, which explains the left and right displacements.

4. DOG CONTROL AND MONITORING SYSTEM: SERVER

4.1 Purpose and Basic Design

The purpose of this system is to create an integrated monitoring and control system for the dogs that can be used for development and debugging purposes. The system is designed to have an intermediate system between the dogs and the client-- a server – for several reasons. First, the dog's networking system is such that only one person may connect to a TCP port at a time. Having to handle many connections would add a heavy load on the dog's systems. Second, the server can control and regulate access to information. Third, the server can also do some data processing. For instance sometimes more information will be sent out from the dog than is needed. The server can eliminate this unneeded data before sending it to the client. Lastly, for high-volume, error-flexible outputs such as video the server can take the data from the dog via TCP and then use UDP to multicast it out, which reduces the load on the server and network [10, pp. 469-470, 528].

The system has a Linux server that consists, of four servers, written in C++ and a Microsoft Windows client written in C# for the .NET framework. Each server is for one type of input or output. One is for clients to send commands for the dog to execute, one is for dog text outputs, another is for the dog's "blob vision" (see Section 2.4), and one is the camera server.

4.2 General Server Libraries

A key feature of any server is its ability to handle the myriad errors that can occur in a network. The most important errors occur during reading, writing, and connect. Reads and connects can potentially block forever if the end point cannot be reached, since they will just keep trying. For writing, if a write occurs on a closed socket in Unix a signal will be sent that will shutdown the program unless it is caught. In order to deal with these and other issues, a class of static wrapper functions for regular socket systems calls (bind, read, close, etc.) was created. These wrappers handle the important read, connect, and write problems with timeouts and signal handlers and also throw exceptions when these problems occur or if a system call returns an important error.

These functions were then used to create classes for sockets. A base class was created called Socket{}. This class lays out the basic interface implemented by all the later derived socket class types (TCP client socket, etc.), by declaring an important set of virtual functions. This interface consists of a constructor that creates the socket and, if required by the socket type (as for a TCP listening socket), binds it to a port and IP address. The destructor of the class closes the socket. The only other functions are read and write (the read function requires that a timeout be specified). Several types of socket classes are derived from the base socket

class: TCP client, TCP listening, TCP server, and multicast server sockets. This interface makes any socket class match the simple theoretical model of a socket: a connection to another computer that is created to send and receive information and then be destroyed. This is one of the guidelines for good object-oriented design [9, pp. 76-101].

4.3 General Server Functions and Control Flow

Although each server is made separately because each has different requirements, large parts of the control flow of each server are similar. Each server works on the "one thread per client" design paradigm (also, for some, one thread per dog), rather than a "one process per client" paradigm [10, pp. 752-754]. This is done because threads are faster to create, and sharing data between threads in global variables is easier than sharing data between processes by IPC. Each server also has the same main loop (see Figure 6).

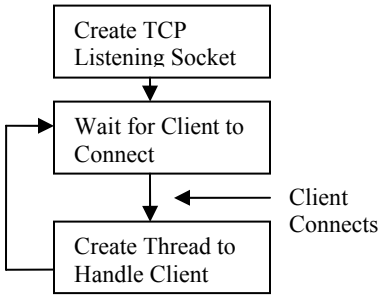


Figure 6: Flow chart for server main loop.

This loop is ubiquitous for all the servers, so it is a universal function. Then each server enters a per-client thread to handle that client's needs. The resulting control flow is shown in Figure 7:

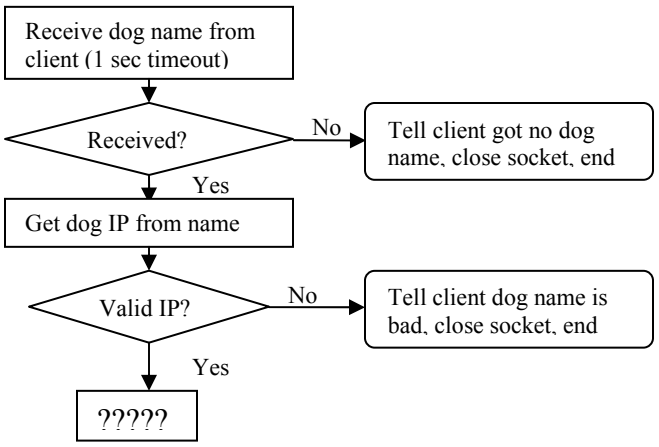


Figure 9: Flow chart of beginning of per-client thread, where the request dog domain name is retrieved and processed.

After this the servers act differently. Each server has to remember what dogs are being used, to know what to do with a new request for a dog. Three of the servers have to associate other information with any dog that is being used (the multicast port for that dog, the number of clients using it, etc.). So each server has a list of dogs that is

implemented using the C++ Standard Template Library map container class. The map stores these dog-specific objects, with each object mapped (stored and looked up) by the IP address of the dog [11, pp. 466-473].

The servers next look for the requested dog in the map. For the input server, if the dog is found then the client is told that the dog is being used, closes the socket, and ends. The output type servers allow for multiple clients to access the same dog, since this does not cause a problem. For these servers the flow of control is shown in Figure 8:

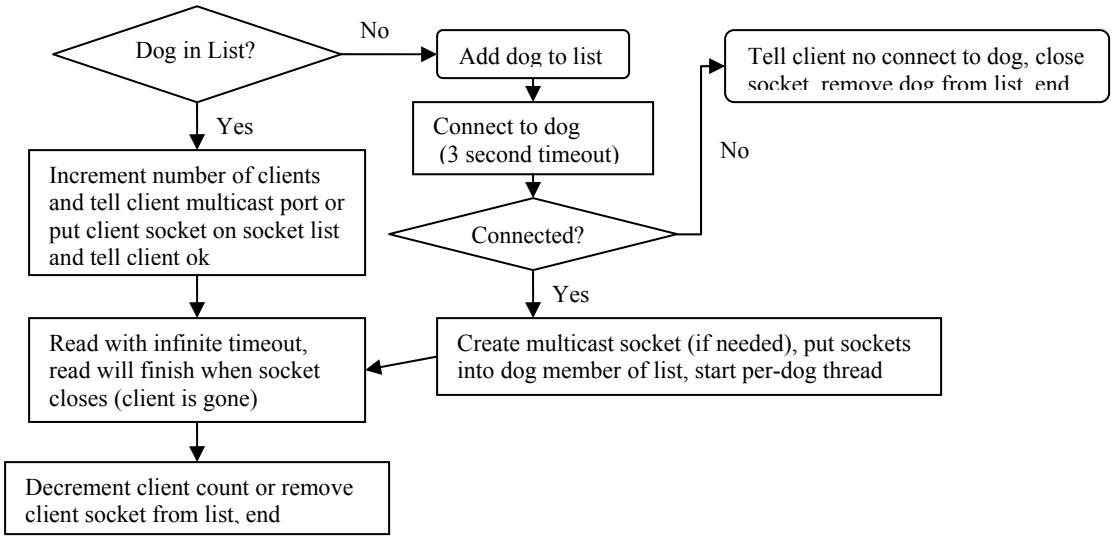


Figure 8: Flow chart of per-client thread (for output type servers).

The main difference here between servers is between the output server and blob and camera servers (see Section 4.2). Finally, the per-dog thread for these servers is shown in Figure 9:

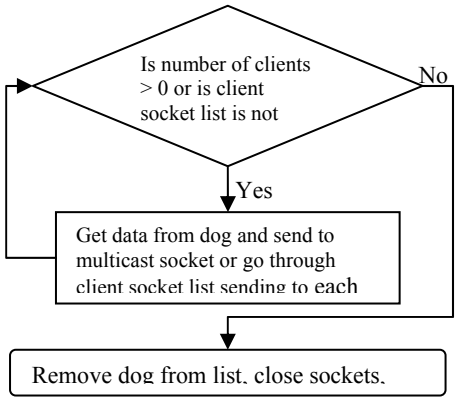


Figure 9: Flow chart of per-dog thread (for output type servers).

4.4 Server-Specific Details

4.4.1 Input and Output Servers

The input server does not need to have per-dog threads, since each dog can only be used by one client. After the dog is connected to the server loops infinitely, waiting for commands with a 5-minute timeout. If a read error or timeout occurs, the server assumes the client is done with the dog, closes the socket, removes the dog from the list, and ends the thread. The strings (commands) that the server gets from the client must be packaged into an OPEN-R packet (see Section 2.3) before being sent to the dog.

The output server's dog list contains objects with the TCP socket connected to the dog and another map of TCP sockets, each connected to a client for that dog. The output data from the dog are sent to each client one at a time by TCP sockets and not through a multicast, for reliability; the client cannot miss error messages from the dog, and multicast uses UDP, which does not have guaranteed delivery. But since a new message must be sent to each client one at a time, if a new message comes in before the last is sent, messages might be missed. Two assumptions mitigate this problem. First, it is assumed that the number of clients will be very low (around 5 per dog at most). Second, it is assumed that all clients will be on the same local Ethernet as the server. Thus there will be few clients to write to, and each write will be fast, so messages will not be missed.

4.4.2 Blob Server

This server works via multicast. Each object in the dog map has the TCP socket connected to the dog, the multicast socket, and the number of clients using the multicast. A data set is requested, and the dog then sends the server an OPEN-R package. After the total size and number of data comes a special header that specifies the size of the two-dimensional array that holds the blob information (the rows are integer values for the blob description structure, and the columns are each individual blobs). A two-dimensional array of this size is created, and the data are then read from the dog into this array. The important data for displaying blobs are extracted from the array (color, lower left corner, and upper right corner x and y) from each blob structure and are then sent to the multicast. The server then waits for 0.2 seconds before getting and sending the next set of blobs.

4.4.3 Camera Server

This server works almost identically to the blob server except that there is no special header. The dog sends the OPEN-R packet header (total size and number of data) and the compressed JPEG version of the last image taken by the camera. This file (array holding the JPEG) is so large (between 2000 and 10000 bytes) that it must be written to the multicast socket in pieces. Retrieving the JPEG from the dog and writing it to the multicast takes so long that no pause is done. In fact, the size of the JPEG completely controls the speed at which images are sent.

JPEG is an image compression format, and thus the size of the image is related to its complexity (simple pictures can be compressed more than complicated ones). Thus the more complex the image the dog sees, the longer it takes to receive and multicast, and

the fewer frames per second (FPS) the camera achieved. A test was done to quantify the relationship between FPS and JPEG size. The FPS at an instant was calculated as FPS = 1/(time of last receive and multicast). As the camera server transmitted, the dog's image was made more and less complex by putting a piece of white paper in front of the camera. The graph made using the 860 data points collected is shown in Figure 10:

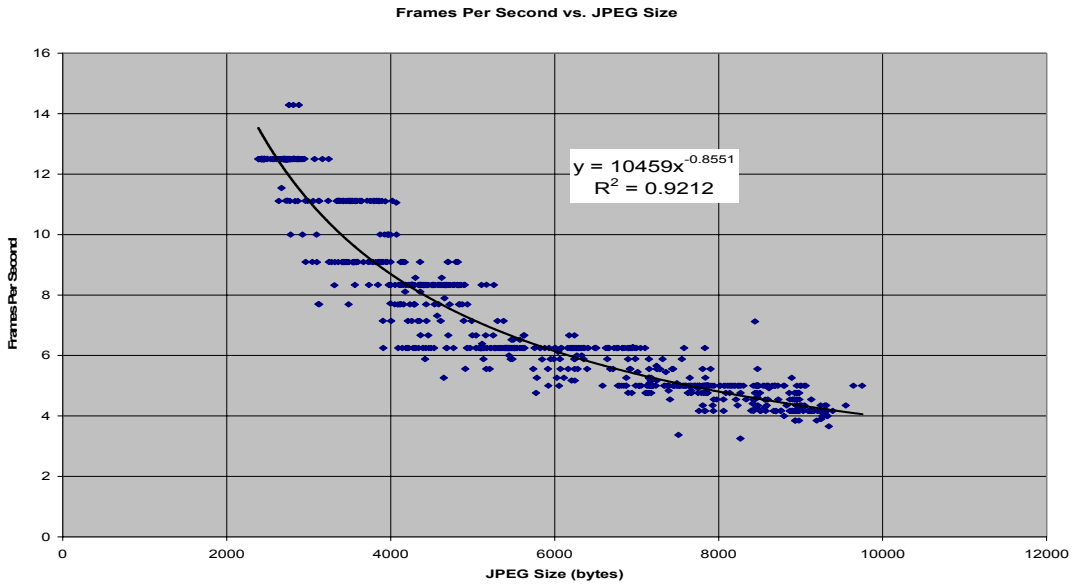


Figure 10: FPS vs. JPEG size for camera server

The relationship of this graph was hypothesized to be: FPS α 1/JPEG size. The regression of the graph supports this, since its estimated function is close to FPS=1/(JPEG size), and the R^2 is above 0.9. The graph also shows the limits of the FPS: the camera works between approximately 4 and 13 frames per second. Four FPS is not very fast, and certainly does not produce streaming video, but it is much better than the 1 to 2 FPS achieved in the previous dog camera system design.

5. DOG CONTROL AND MONITORING SYSTEM: CLIENT

5.1 Client Basic Design

Most of the design and features of the client program come directly as a consequence of the design and control flow of the server, but several important features and design concepts should be emphasized. As mentioned earlier, the client is a Microsoft Windows program written in C# (with Visual Studio .NET) for the new .NET framework. The client has a main window where the user can specify the dog and function (blob vision, etc.) desired, and a new window for this will be created. The client is multithreaded, spawning new threads for each output type service (e.g., a new thread is created to handle the continuous receiving and displaying of data from a blob observer, camera, or output server socket), so the main thread is not over-burdened. When exceptions are caught or error messages are sent from the server, a message box showing

that error is displayed, and the window it came from is shut down (along with any threads). Screen shots of the main window and some other windows are in Appendix B.

The input window has a text box for writing commands and buttons to send commands to the server and clear the text box. The input window can save and load scripts stored in .txt files. The output window has a read-only text box that shows the output text from the dog. It can also save the output of the dog to a .txt file.

The blob observer, camera, and remote all have user controls created to handle these particular graphics displays. Each one uses the .NET framework's System.Drawing library and its Graphics class. For the blob observer, for each blob a FillRectangle and a Rectangle (for black outline) are drawn in the user control window (see Section 2.4). For the camera observer, the JPEG data are read into an array, which is read into a Stream object, which is read into an Image object, which is then displayed. The network transmissions and these steps account for the almost one-second delay between the time an image is captured by the camera and when it shows up on the client window.

5.2 Remote Control Client

The fifth client window is the remote control (see Appendix B). This program connects to both the input server and camera server, allowing the user to navigate the dog around a room. Certain keys are bound to sending specific commands to the dog (using callbacks). Thus there are keys to make the dog go forward, backward, right, and left, and to turn right and left. Also, if the user clicks on a certain point in the camera image, the program takes the position of that click and computes what command it should send to the dog to have it point the camera in that direction.

To compute this value requires understanding how the dog is commanded to move its head. A command called PointHead has three parameters: the x, y and z (in mm) coordinates (body's coordinate system) of the object that the dog should try to look at. For the remote, the dog is commanded to look at an object 1 meter away at the azimuth and elevation where the screen was clicked. Calculating the point head parameters requires two steps. First, the x, y, and z coordinates of the camera vector are calculated for the dog's head's coordinate system. Second, these coordinates are transformed into the dog's body's coordinate system.

The distance in u and v (pixels) from the center point of the camera to the click point are found. The focal length of the camera is 161 pixels. Thus a vector from the camera center to the click point is created (see Figure 11). The azimuth (Φ) and elevation (Θ) of this vector are calculated from u, v, and the focal length.

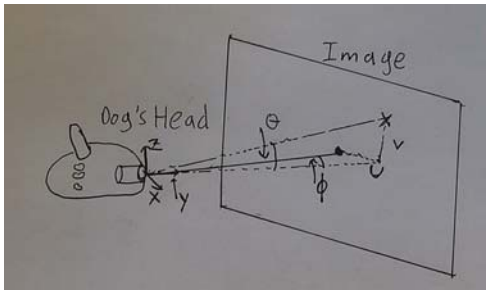


Figure 11: "Click" vector from camera to image diagram

$\Phi = \text{ArcTan}(u/161)$
$\Theta = \text{ArcTan}(v/(u^2 + 161^2)^{1/2})$

Another vector in this same direction, only 1 meter in length, is the vector of the new point where the dog should look. The azimuth and elevation are used to find the x, y, and z coordinates of the end of this vector in the head's coordinate system.

$\text{Xhead} = 1000*\text{Cos}(\Theta)*\text{Sin}(\Phi)$
$\text{Yhead} = 1000*\text{Cos}(\Theta)*\text{Cos}(\Phi)$
$\text{Zhead} = 1000*\text{Sin}(\Theta)$

The head is at a certain azimuth and elevation relative to the dog's body coordinate system. This azimuth and elevation can easily be determined by knowing the x, y, and z coordinates in the body's coordinate system of the last PointHead command. These angles are used to translate the new PointHead coordinates from the head's coordinate system to the body's coordinate system.

```
Xbody = Xhead*Cos(Az.) + Yhead*Sin(Az.)*Cos(El.) - Xhead*Sin(Az.)*Sin(El.)
Ybody = -Xhead*Sin(Az.) + Yhead*Cos(Az.)*Cos(El.) – Zhead*Cos(Az.)*Sin(El.)
Zbody = Yhead*Sin(El.) + Zhead*Cos(El.)
```

The only factor unaccounted for is the pitch (dog twists its head). The previously stated algorithm works on the assumption that in order to look at a point the dog turns and raises or lowers its head, but does not twist it. This is only true for small azimuths and elevations. For large angles the dog's head twists and the algorithm breaks down. Two measures correct this: the angles to which the dog can be commanded to move its head can are limited, and if a problem occurs the "look front" button can be pressed, which will reset the dog to look forward.

6. DISCUSSIONS AND CONCLUSIONS

The motion calibration system developed here may not be precise enough and easy enough to use for the required needs. The system worked well only for short-range, repetitive motions such as walks and kicks. One problem is that the NOB itself does not have a large range. This could be fixed by adding an extended range transmitter, but then there is the problem of wires. Even if a large area can be covered the dogs cannot be

allowed to move anyway they wish, because they will become tangled in the cords. Wireless sensors can be obtained, but then the magnetic interference from the dog's motors will probably interfere with the transmissions.

The motion calibration system does do a good job at discovering major trends in the dog's movements, such as discovering that the dog did not turn precisely around the end of its head. It can also tell if the calibration factor for a motion is greatly off, but between natural inconsistencies in movement, uncertainty, interference from the dog's motors, and the somewhat crude system (using the ruler) to counteract this, precise results about the dog's motions are probably not obtainable.

The server/client system does offer some valuable tools. This system is a good basic platform for debugging the Aibos. The existing client already has many features that make it a valuable for system debugging, and these tools can easily be added to.

7. FUTURE WORK AND RECOMMENDATIONS

The motion calibration system has produced some good results, but has not reached its full potential and may never do so. Given the facts of limited system range and motor interference, the NOB does not seem to be the best choice for a positioning system. Given the time and effort required to set up the extended range transmitter and perhaps get wireless sensors, it may be better to try another positioning system. It is also now clear though, that there are great benefits to having a positional system especially if it is integrated with the server system.

There is clearly great potential in the server system. The infrastructure and libraries are there to easily add servers for other types of sensory outputs (such as sound or positional information from some external device). There is also great potential for creating more complex and useful clients (such as the remote control) that interact with multiple servers. Examples include a better color mapping system that could interact with the input, camera, and blob servers to allow for easier creation of color maps, and creation of a program for teaching the dogs where they are on the field that would interact with the positional, input, and output servers.

8. ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Daniel Lee of the University of Pennsylvania, for all of his assistance, expertise, and patience. I would also like to thank Paul Vernaza and David Cohen for their help with the Aibos, and Janice Fisher for her assistance with this paper. Finally, I would like to thank the Microsoft Corporation for their generous support of my project and the SUNFEST program.

9. REFERENCES

1. Sony, *Sony Four Legged Robot Football League Rule Book*, Sony Corporation, 2003.

2. Sony, *OPEN-R SDK: Model Information for ERS-210*, Sony Corporation, 2002.

3. Sony, *OPEN-R SDK: Programmer's Guide*, Sony Corporation, 2002.

4. Sony, *OPEN-R SDK: OPEN-R Internet Protocol Version4*, Sony Corporation, 2002.

5. Dr. Daniel Lee, David Cohen, and Paul Vernaza, Autonomous Robot Soccer Team Implementation for Robocup 2003 Legged League*, Unpublished,* 2003.

6. Manuela Veloso et al., CMPack-02: CMU's Legged Robot Soccer Team, *Unpublished*, 2002.

7. Ascension Technology, *Flock of Birds: Technical Description of DC Magnetic Trackers*, Ascension Technology Corporation, Burlington, VT, 2002.

8. Ascension Technology, *Flock of Birds: Installation and Operation Guide*, Ascension Technology Corporation, Burlington, VT, 2002.

9. Cay S. Horstmann, *Mastering Object-Oriented Design in C++*, John Wiley & Sons, Inc., New York, NY, 1995.

10. W. Richard Stevens, *Unix Network Programming*, Vol. 1, Prentice-Hall, Upper Saddle River, NJ, 1998.

11. Matthew H. Austern, *Generic Programming and the STL*, Addison, Wesley, and Longman, Inc., Reading, MA, 1999.

10. APPENDIX A: CALIBRATION GRAPHS

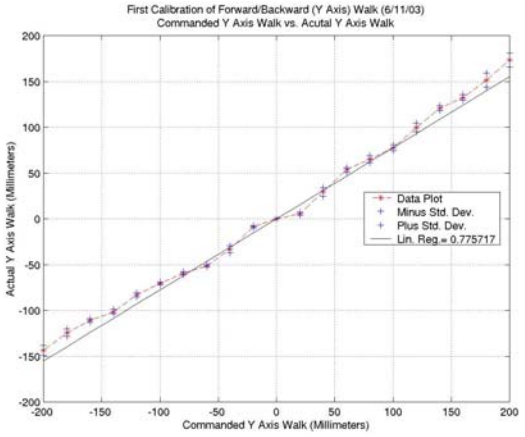


Figure 1: Forward motion calibration (single linear regression).

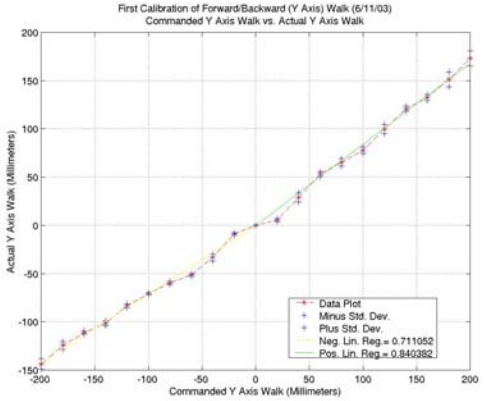


Figure 4: Forward motion calibration (positive and negative linear regressions).

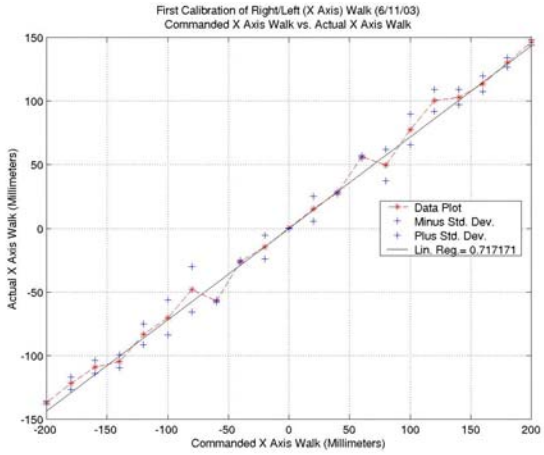


Figure 2: Side motion calibration (single linear regression).

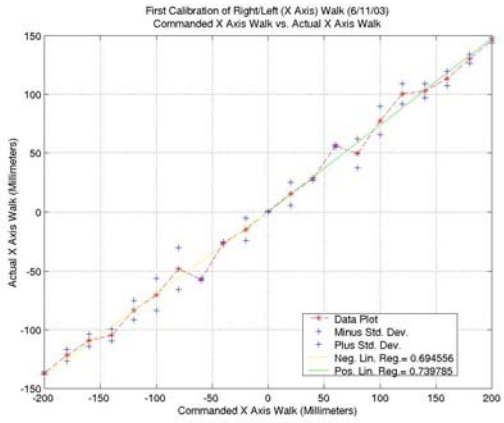


Figure 5: Side motion calibration (positive and negative linear regressions).

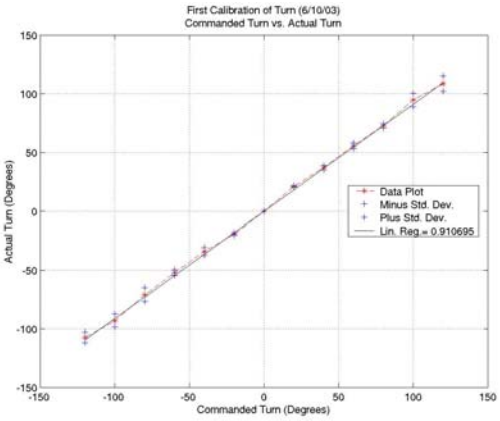


Figure 3: Turn calibration (single linear regression).

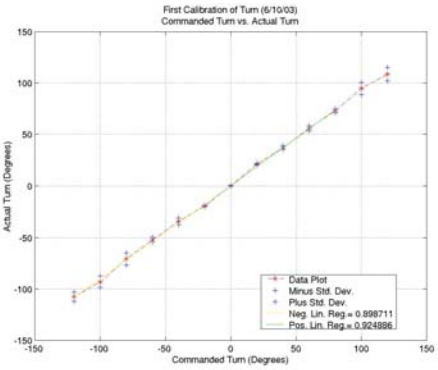


Figure 5: Side motion calibration (positive and negative linear regressions).

11. APPENDIX B: CLIENT PROGRAM SCREEN SHOTS

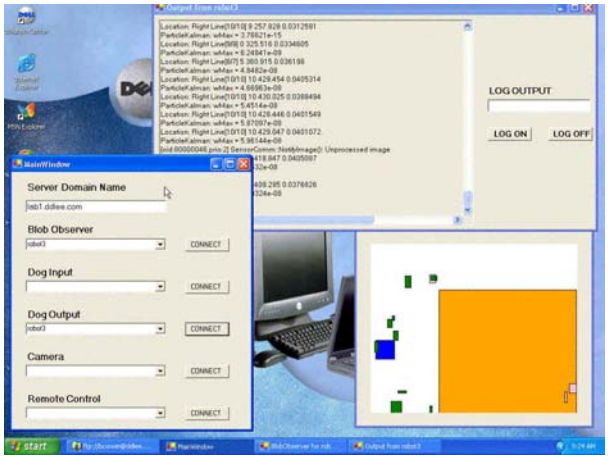


Figure 1: Picture of main window and sub-windows for client program.

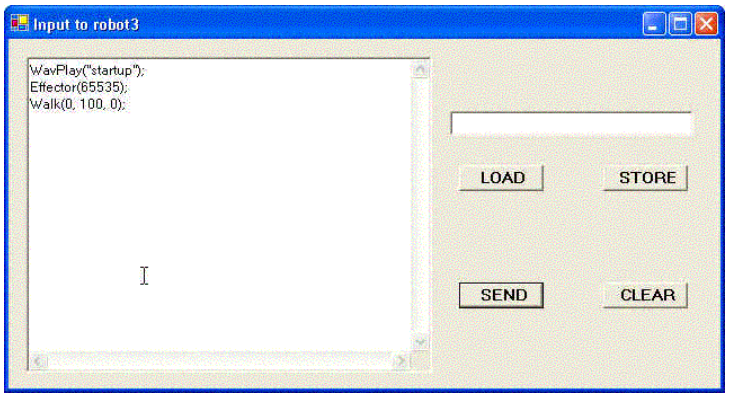


Figure 2: Window of input client program.

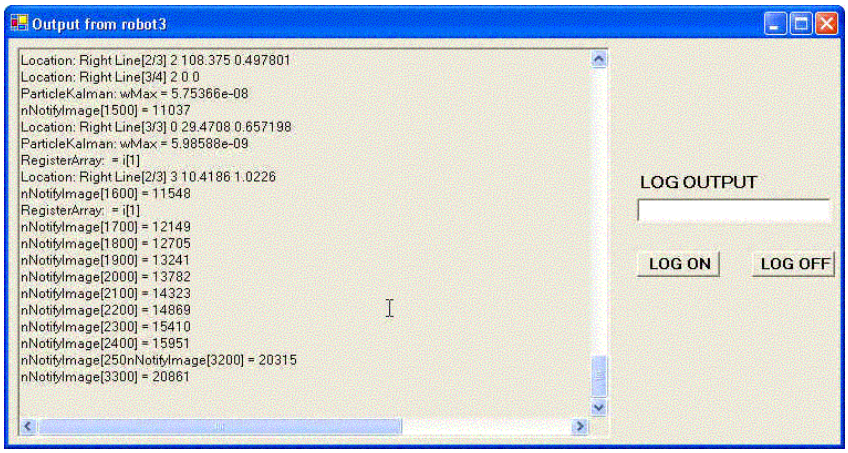


Figure 3: Window of output client program.

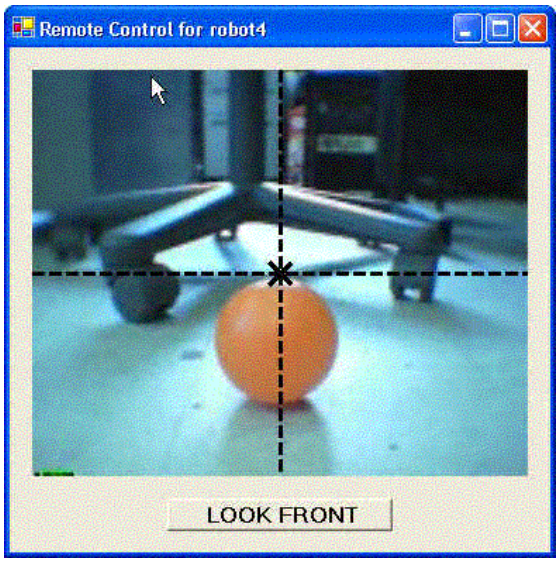


Figure 4: Window of remote control client program.

DIELECTROPHORETIC ASSEMBLY, INTEGRATION, AND CHARACTERIZATION OF FUNCTIONAL NANOSTRUCTURES

NSF Summer Undergraduate Fellowship in Sensor Technologies
Vinayak Deshpande (Electrical Engineering) – University of Virginia
Nicole DiLello (Electrical Engineering) – Princeton University
Advisor: Dr. Stephane Evoy

ABSTRACT

Nano-electro-mechanical systems (NEMS) are pivotal to integrated systems research. The ability to integrate sensors and processors on a single wafer has the potential to create highly sensitive and complex systems that respond to changing environmental stimuli. This project focuses on the assembly, integration, and characterization of both nanowires and nanotubes used in NEMS devices. Using dielectrophoretic assembly, the project assembled and electrically characterized Rh and GaAs nanowires as well as multi-walled carbon nanotubes (MWNTs). It also found optimum parameters for assembly for each of these nanostructures for a specific circuit design. The optimum conditions always involved applying a sinusoidal field of 100 kHz at an amplitude of 10 to 35 V. The duration of assembly varied from 1 minute to 5 minutes. The capacitive properties of contact pads play a critical role in assembly yields, and resistivity of MWNTs is inversely related to temperature. The research also raised questions about the physical structure of these nanostructures and how it may be affected by changing temperature. It appears that a critical temperature is reached at which the resistivity of the nanotube increases drastically. Finally, the project produced clamped MWNTs for future mechanical testing.

Table of Contents

1. INTRODUCTION

Micro-electro-mechanical systems (MEMS) are perhaps at the forefront of integrated systems research. MEMS devices hold the possibility of integrating processing power and sensing power to create powerful micrometer-sized systems [1]. Thus MEMS devices have the potential to both sense and act upon environmental stimuli. These small-scale, fully functional devices could solve problems ranging from detecting diseases within the human body to exploring remote worlds. Furthermore, because their manufacturing processes are similar to integrated circuit techniques, MEMS devices can achieve a high level of durability, reliability, and complexity with relatively low costs.

MEMS research has the potential to revolutionize integrated systems research even further with the introduction of nanotechnology. Nanotechnology, or the manipulation and ordering of particles on the nanoscale, has numerous uses in a wide variety of fields. Specifically, for MEMS research nanotechnology has the potential to create even more advanced and smaller "laboratories on a chip." Nanoscale MEMS devices, known as NEMS (nano-electro-mechanical systems) devices, would be able to integrate numerous sensors, actuators, and processing units on a single silicon wafer.

Recent advances in the synthesis of nanowires (wires of some nanometer diameter) have given rise to the notion of creating rudimentary nanowire-based MEMS devices. To build these nanoscale devices, however, the physical, chemical, and electrical properties of the nanowires must be observed. Furthermore, their integration with micrometer-sized devices is pivotal to their commercial use. Hence, current research revolves around the characterization of these nanowires and their integration with their micrometer-sized counterparts (nanowires serve as interconnects between micrometer-sized features).

This paper is concerned with the assembly and characterization of certain nanoscale devices. Specifically, this paper reports on Rh nanowires, GaAs nanowires, and multi-walled carbon nanotubes (MWNTs). The rhodium nanowires are metallic structures that should act as conductors. The GaAs nanowires are semiconducting structures that are expected to have a specific band gap, one that could lead to applications in optics [2, 3]. The MWNTs can be used as a mechanical device to transport fluid. They also have applications as a field emitter, sensor, and battery electrode [4, 5].

This paper will first give a brief literature review, describing how these nanostructures are synthesized and assembled. The review discusses the laser-assisted catalytic growth process, the most popular method for synthesizing nanowires; various methods for synthesizing the carbon nanotubes, including chemical vapor deposition, the most popular method; and various ways to assemble these nanostuctures, culminating in dielectrophoresis. Dielectrophoresis was the process used in this research. This report details the assembly of these three nanostructures, giving the optimal conditions that should be used to assemble them on metallic electrodes. The equipment and process used to trap these structures in places where they were easy to characterize are described.

The next section of the report discusses how the nanostructures were characterized — specifically, current-voltage relationships and how they change with temperature. This information is important so that these structures can be characterized as sensors. The paper concludes by discussing ways in which the research could have been improved, ideas for future research, and the broad outlook for the future of NEMS devices.

2. LITERATURE REVIEW

2.1 Nanowire Synthesis

Nanowires and nanotubes can be synthesized and assembled in a variety of ways. Scientists and engineers are continually trying to develop new methods to give people more control over the outcome, including how long and wide these nanostructures should be and where they assemble on very large scale integrated (VLSI) circuits.

A popular way to synthesize nanowires has been developed by scientists at Harvard University [6, 7]. This method, called laser-assisted catalytic growth (LCG), uses a vapor-liquid-solid process. The process was first used to synthesize Si nanowires. In this instance, the process began with a solid piece of $Si_{0.9}Fe_{0.1}$. Laser ablation of the Si-Fe combination created a vapor of Si and Fe that quickly condensed to form liquid Si-Fe nanoclusters. These nanoclusters acted as the catalyst upon which the nanowire was grown. The nanoclusters were then supersaturated in Si, which then precipitated and crystallized as nanowires.

The success of this process requires consulting a binary phase diagram for Fe-Si. This phase diagram can show where FeSi liquid can coexist with a Si solid. The process should then be operated at this temperature and pressure so that the FeSi liquid catalysts can start the crystallization of Si nanowires. For this reason, to create Si nanowires on a FeSi catalyst, the experiment had to be run above 1150°C. While this was not impractical, it was desirable to find a lower temperature [6, 7].

Gold and silver have been studied as possibilities for creating a nanocluster catalyst. Inspection of the Au-Si binary phase diagram indicated that creating Si nanowires from an Au catalyst could be done nearly 800°C lower than when using Fe. In this case, however, the nanoclusters would consist of pure Au and not an Au-Si combination as when using Fe. Experiments demonstrated that Si nanowires grew on Au catalysts at temperatures between 370°C and 500°C. Furthermore, the nanoclusters that terminated the wires were indeed Au [6].

While this method was easily extended to create Ge nanowires, creating compound semiconducting nanowires proved to be more complicated. Compound semiconductors, such as GaAs, create a more complicated phase diagram because there are now three elements to consider: Ga, As, and a metal used for the catalyst (usually Au) [8, 9]. The phase diagram was simplified, however, by looking at the pseudobinary phase

diagram of the metal and the semiconductor. In this case, the compound semiconductor was treated as a single element. By using this method, researchers could synthesize GaAs nanowires on Au nanoclusters at temperatures above 630°C.

Finally, the length of the nanowires can be controlled by the time spent at higher temperatures. As long as the nanowires are in the furnace, they will continue to grow. They will leave the furnace when the gas flow from the laser ablation carries the wires out of the furnace. The laser ablation can be controlled; thus, the length of the nanowires can be controlled and predicted [6].

Thus, the laser-assisted catalytic growth method for synthesizing nanowires is quite useful. Known phase diagrams can be used to accurately predict optimal conditions for synthesizing the wires, and laser ablation can be used on a variety of metals to create catalysts.

2.2 Carbon Nanotube Synthesis

Because carbon nanotubes are different from nanowires, a different process of synthesis is needed. Nanotubes are hollow structures through which liquid could flow. Nanowires, on the other hand, are solid structures that will conduct current and can otherwise act as a wire. Synthesizing a tube proved to be more difficult, though the process was similar.

2.2.1 Carbon Arc Discharge Method

Carbon nanotubes were first discovered by accident by Sumio Iijima. The accidental conditions were then reproduced in an effort to purposely synthesize the nanotubes [10, 11]. A DC voltage of approximately 20 V was applied to pure graphite electrodes. The current was kept between 40 and 100 A. These conditions caused an electric discharge between the anode and the cathode. This discharge creates a carbon plasma, and the anode is constantly evaporated and deposited on the cathode.

The deposition on the cathode forms a cylindrical shape that can then be broken off and opened. Inside are numerous columns that contain nanotubes. These tubes form only when there is a large amount of current (40 A or higher). If there is not enough current, the deposits will simply be amorphous carbon.

This process is not very accurate and leaves little control over the size and length of the nanotubes. It also gives little control over where they are synthesized. A new method, known as chemical vapor deposition (CVD), uses the same basic idea of depositing a carbon plasma, but attempts to control the outcome more than the electric discharge method.

2.2.2 Chemical Vapor Deposition

A method employed by chemists at Nanjing University in China uses a process called microwave plasma-enhanced chemical vapor deposition (MW-PECVD) [4, 5]. This process is similar to LCG because it uses a catalyst to grow the nanotubes. In this specific experiment, the Fe was used as a catalyst to grow the carbon nanotubes. The catalysts were placed in a quartz tube, which was then placed in a vacuum system. Argon and hydrogen were first used to reduce the Fe catalysts, and then benzene vapor was introduced into the system. The benzene vapor then crystallized on the Fe catalysts, forming nanotubes.

The MW-PECVD system was improved in South Korea at Chungnam National University. The deposition of metal films in a vacuum was replaced by a spin coating of a magnetic field solution. The magnetic fluid was created by dispersing magnetic nanoparticles in solution. The solution used was a dispersion of Fe_3O_4 nanoparticles dissolved in ammonia hydroxide. It was coated on a Si wafer and then placed in a furnace under low pressure. NH_3 gas was introduced into the furnace followed by C_2H_2. Carbon nanotubes were formed on the Si substrate.

It was found that the diameter of the nanotubes was directly proportional to the diameter of the nanoparticles in the magnetic fluid. Larger particles yielded larger nanotubes. The researchers could control the size of the nanoparticles by the amount of ammonia hydroxide. A larger amount of the ammonia hydroxide yielded smaller nanoparticles, which then yielded smaller nanotubes. Since the diameter of the particles could be controlled easily, so could the diameter of the tubes.

2.2.3 Template-Synthesis Method

A completely different method for synthesizing carbon nanotubes uses templates. Scientists from Tsinghua University in Beijing have developed a technique that uses templates to control the diameter of the nanotubes [12].

They use highly ordered anodic aluminum oxide (AAO), which is a nanoporous material. The size of the pores can be controlled by adjusting the oxidizing voltage of the aluminum. The AAO was placed in a vacuum chamber, and CVD was used to synthesize the nanotubes within the pores of the AAO template. Iron was used as a catalyst and was electrodeposited into the pore bottom of the template. The Fe was then reduced, using hydrogen gas, and an ethylene-argon gas mixture was introduced into the system. The nanotubes then grew on the Fe catalysts that had taken the shape of the pores in the AAO template.

The diameters of the tubes were directly proportional to the diameters of the pores. The average diameters of the tubes were generally a few nanometers larger than the diameters of the pores, but this is thought to be due to thermal expansion of the pores during nanotube growth.

2.3 Assembly Technologies

This section describes previous methods used to assemble nanowires. In this section, three specific methods will be described: directed growth, chemical templating, and dielectrofluidic assembly. It will review the works of Dr. Hongjie Dai (Stanford University) on directed growth procedures, Dr. Chad Mirkin (Northwestern University) on chemical templating, and finally Dr. Thomas Malluok (Penn State) and Dr. Theresa Mayer (Penn State) on dielectrofluidic assembly.

2.3.1 Directed Growth

Directed growth is the method by which nanowires are grown at specific locations to integrate them with some other circuitry. This gives rise to the possibility of creating nanowire based electronic devices. Specifically this involves knowledge of nanowire synthesis using chemical vapor deposition. Chemical vapor deposition (CVD) is the process by which elements in gaseous form can be condensed to thin films of solid form. These thin films can then be deposited on a wafer. Currently, it is possible to grow Single Walled Nanotubes (SWNT) using a process known as catalyst patterning. Catalyst patterning is similar to standard photolithography procedures for wafer fabrication, except that an additional catalyst layer is placed on top of these electrodes [13, 14, 15, 16]. This catalyst layer contains special chemical properties that allow CVD processes to actually orient and grow SWNTs across these electrodes.

After initial wafer fabrication is complete (using standard photolithography / thermal oxidation / sputtering procedures), "a poly(dimethylsiloxane) (PDMS) elastomer stamp"[pg. 7975, 14] is covered with a catalyst solution. This PDMS stamp is then placed on top of the fabricated wafer, hence transferring the catalyst solution to the electrodes on the wafer. After calcinations, the wafer is run through standard CVD processes. This causes the growth of the SWNTs across the electrodes on the wafer.

Using this catalyst / CVD method (catalyst patterning), growth of rudimentary nanostructures is possible. The key to growth is the catalyst material used and the conditions of the CVD processes. Furthermore, wafer fabrication also plays an important role in SWNT growth. Gaps between adjacent electrodes affect lengths of the grown SWNTs [15,16]. Also, using directed growth, a strong bond between electrode and SWNT is present. This is useful when doing mechanical testing on the grown SWNT.

2.3.2 Chemical Templating (DNA Templating)

Chemical templating is a process by which nanoparticles can be assembled using the recognition patterns of DNA. By using thiol modified nanoparticles, a DNA pattern can be added within them. Hence, when another thiol modified nanoparticle is introduced that complements the first one, assembly occurs. This section will examine the work of Dr. Chad Mirkin in this field.

DNA templating, as a method for nanoparticle assembly, is employed by doing the following. First, nanometer sized particles (such as Au) have attached on their surfaces DNA oligonucleotides and thiol [17]. Thiol is used as an adhesive between the Au particles and the oligonucleotides. When nanoparticles with complementary DNA oligonucleotides are introduced, assembly occurrs. Thermal denaturation is the process by which one can disassemble these particles.

Furthermore, two component nanoparticle systems can also assemble by DNA templating [18]. A component is a reference to a specific type of particle; Au specifically was used in Dr. Mirkin's one component system. Similar thiol based particles are used once again for templating in the two component system. This shows that DNA templating based assembly is not necessarily subjected to a particle's size and chemical composition. Hence, it gives rise to the possibility of constructing "anatural" bondings and having different types of particles assemble (e.g. Au and Rh).

2.3.3 Dielectrofluidic Assembly

Dielectrofluidic assembly is the method used by this project to assemble nanowires. Dielectrofluidic assembly involves the introduction of electric fields to assemble nanowires suspended in some medium (such as acetone or isopropanol). This assembly procedure involves the optimization of three parameters, voltage, time, and frequency, to maximize nanowire assembly yields. This section will look at the papers of Dr. Thomas Malluok and Dr. Theresa Mayer.

This method begins with lithographically defined electrodes where nanowires can assemble. For electric field assisted assembly to occur, metallic nanowires must polarize [19]. The most convenient way for polarization is through the introduction of an AC signal source. This is due to the source's frequency. Usually, a higher frequency allows the nanowires to polarize more quickly. The unfortunate fact of this is that too high of a frequency actually causes no polarization and consequently no assembly. It should also be noted that once a nanowire assembles, the possibility that another nanowire will assemble near it is low. This is due to the assembled nanowire's own electric field impeding further assembly.

Another parameter that decides assembly yields for this method is the peak to peak voltage applied by the AC signal sources [19,20]. A higher peak to peak voltage causes a stronger electric field. Hence, with stronger electric fields, there is greater likelihood that a nanowire will assemble. However, like the frequency parameter, too high of a voltage causes the assembly electrodes to actually burnout.

Lastly, the time the AC signal source is present is also important in determining assembly yields. Much like the voltage parameter, the time the AC signal source is present has a direct effect on the chance for assembly. However, again like the voltage parameter, too great of a time can actually cause the assembly electrode to burnout. This is due to the fact that the assembly electrodes are just capacitors, and hence, when the dielectric medium between them (air) becomes conductive, a short is produced.

3. EXPERIMENTAL METHODS

3.1 Wafer Design

The wafer design process involved the development of two distinct designs for different uses. The first design, the experimental wafer design, was created primarily for the University of Pennsylvania's NEMS Laboratory. It served as a vehicle for both nanowire assembly and testing. The second design, the VT / Experimental wafer design, had dual purposes. Because the design was outsourced to locations with greater precision than the University of Pennsylvania's Microfabrication Laboratory, more sensitive versions of the first design were produced. Along with the more precise version of the first design, the VT / Experimental design also had a replication of a VLSI Virginia Tech design, which at a future date, could serve as another testing vehicle for the NEMS Laboratory. The experimental wafer design was actually fabricated, while the VT / Experimental design was completed only up to its design stage.

3.1.1 Experimental Wafer Design

The goal of this wafer design was to create a wafer that could assemble nanowires and test them for electrical and mechanical properties. Electrofluidic assembly was the method used for assembly. Electrofluidic assembly involves the introduction of electric fields to assemble nanowires suspended within some solution (usually acetone or isopropanol). The design used micrometer-sized conductive electrodes (see Figure 1) for AC signal source excitation. This created a uniform electric field between these electrodes. However, an electrically insulating SiO_2 layer was added on top of the now "buried" electrode layer, to create a weakened electric field above the SiO_2 layer. The design then added another conductive electrode layer on top of the SiO_2 to concentrate the weakened electric field between these "upper" electrodes (passive electrodes). This concentration of electric field makes it possible for nanowire assemblage to occur on this wafer, given the proper voltage, frequency, and time parameters for the AC signal source.

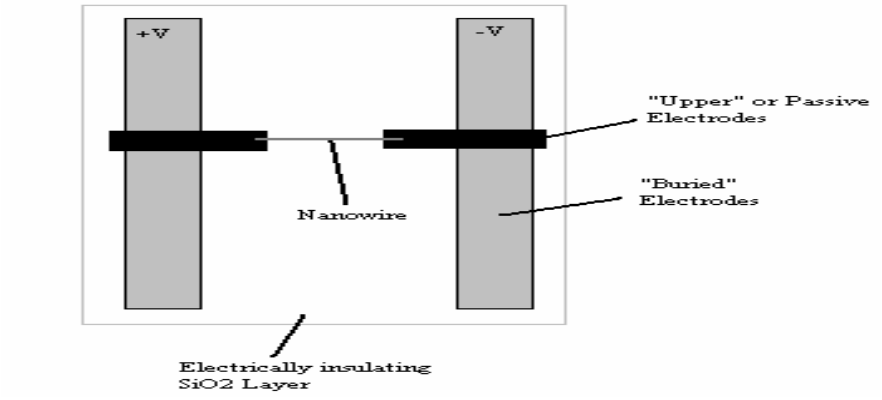


Figure 1. Experimental design setup.

All designs for the wafer were done using AutoCAD 2000 software. The wafer model was a 4-layer design with 100 assembly sites (places where nanowires could assemble) per circuit. Two different designs, "clamped" and "unclamped," were placed on the wafer, and this section will elaborate on their characteristics.

Circuit designs for both the "clamped" and "unclamped" versions began with the "buried" electrode layer. This layer was the site for the AC signal source introduction. The layer is considered "buried" because it is the first layer on the wafer and consequently underneath layers 2, 3, and 4 (see Figures 1 and 2). All electrodes within this layer were of 10 μm width. This layer provides 10 pairs of parallel electrodes, with one of the pairs connected to the positive signal source and the other to the negative signal source. Each of these parallel electrodes was 500 μm long. Additionally, this layer provides two electrodes that connect the paired electrodes to outside AC signal sources. Other lengths are specified in Figure 3.

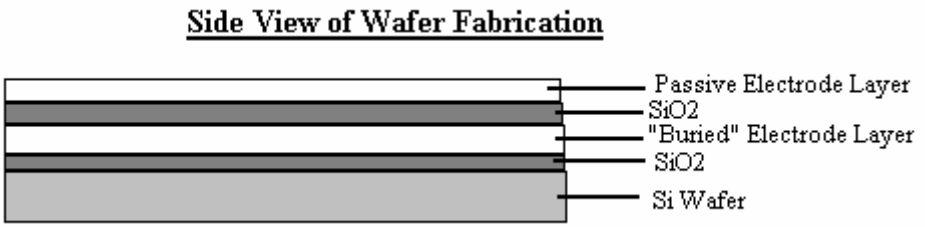


Figure 2. Side view of wafer fabrication (thickness not to scale).

The second layer for both "clamped" and "unclamped" versions provides contact points between the aforementioned "buried" electrode layer and the third layer. As stated before, after the "buried" electrode layer (layer 1) is added, layer 2 is the electrically insulating SiO_2 layer. Since layer 2 is electrically insulating, it is not possible for any outside AC signal source to induce an electric field on the "buried" electrode layer. Hence layer 2 of the AutoCAD model etches out 50 μm-sized squares from the SiO_2 layer so that contact can be made with the "buried" electrode layer. Two squares are etched out — one for the positive and one for the negative AC signal source contact point.

The third layer for both versions is the "upper" electrode or passive electrode layer. This layer produces concentrated electric fields between all passive electrodes as a result of their conductive properties. These passive electrodes, coming in pairs (because one will take positive and one will take negative signal sources), are also known as assembly sites (see Figure 4). Spacing between each passive electrode pair varies from 5 μm to 30 μm (5 μm, 6 μm, 7 μm, 8 μm, 10 μm, 12 μm, 15 μm, 20 μm, 25 μm, and 30 μm) and changes per column (see Figure 3). These varying lengths allow for a multitude of different-sized nanowires to be assembled on a single circuit. All passive electrodes have widths of 10 μm.

The third layer also provides for vias, or contact pads where electrical probes can be placed to excite AC signal sources across the wafer. Using the inlets provided by layer 2 (which removes 50 μm squares from the electrically insulating SiO_2 layer), the

third layer encloses those inlets (allowing for contact with "buried" electrode layer 1) and then connects them to relatively large 200 µm squares. These 200 µm squares in the third layer serve as contact pads for electrical probes.

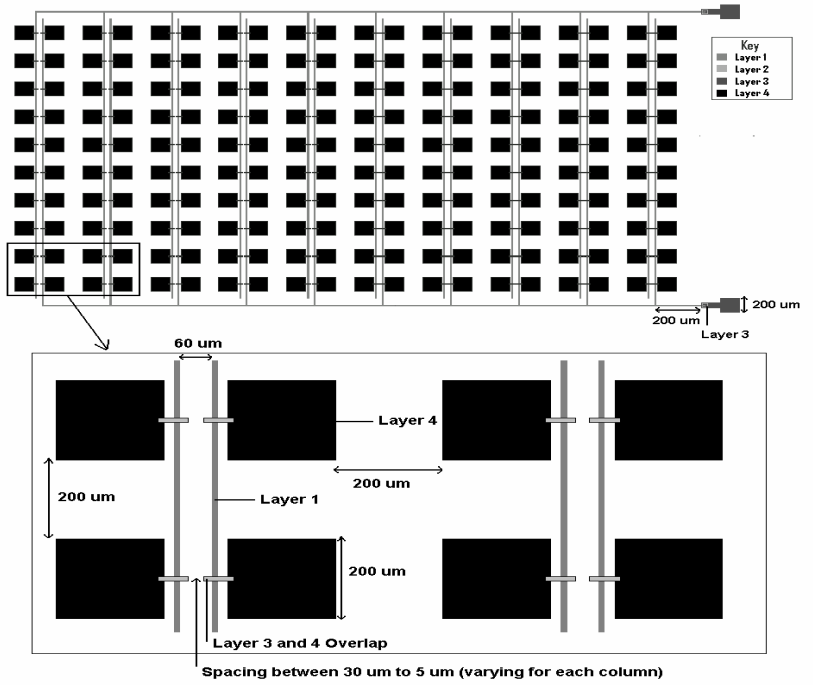


Figure 3. Experimental wafer design and dimensions.

Differences in the "clamped" and "unclamped" versions arise first in layer 3. Layer 3 is completed in the "clamped" version after all passive electrodes and vias are added. A "clamped" version is more useful in mechanical testing of assembled nanowires because this version essentially clamps down the nanowire after assemblage, and hence, provides for a more stable testing environment. The fourth layer for a "clamped" version circuit is introduced after electrofluidic nanowire assembly has occurred. This fourth layer goes on top of the passive electrodes and the assembled nanowire — so the nanowire is sandwiched with the third passive electrode layer on bottom and the fourth clamping electrode layer on top.

In the "unclamped" version, which is better used for electrical testing purposes, layer 3 also includes 200 µm-square contact pads, adjacent to and overlapping the passive electrode pairs. Measurements including I-V curves, resistance, and other properties can be obtained using these pads.

Layer 4 is applicable only to the "clamped" version of this circuit. As described before, layer 4 essentially sandwiches the assembled nanowire in order to create a more suitable environment for mechanical testing. Layer 4 also includes 200 µm-square contact pads for the same reasons as in the "unclamped" version. Dimensions and locations of layer 4's clamps are identical to those of the passive electrode layer in layer 3.

Several other factors were taken into account for this design. First, all points in the design where adjacent figures were to come into contact had to overlap. This took into account the possibility that fabrication mistakes could occur (e.g., electrical connections would hold even if there was a 1 μm mistake in fabrication). Second, all dimensions for all figures had to be revised to adhere to the University of Pennsylvania Microfabrication Laboratory's capabilities (i.e., smallest dimension of 3 μm). Finally, alignment marks and cutting marks were also present on the design to ensure proper layer alignment.

3.1.2 Virginia Tech / Experimental Wafer Design

Like the experimental wafer design described above, the VT design was created to assemble and test nanowires. All specifications in the first design could be scaled down because this second design will be outsourced to a location with more precise tools for fabrication. Additionally, a replication of a Virginia Tech VLSI design was added to this wafer. Both the smaller specification design and the VT VLSI design are discussed in this section.

Differences between the first wafer design and this wafer design are as follows. First, all assembly sites in the VT wafer have gaps ranging from 3 μm to 6 μm, with increments of approximately 0.3 μm, as opposed to gaps of 5 μm to 30 μm in the first design. Second, the VT wafer has a total of only three columns, as opposed to 10 columns in the first design. Hence there are only 30 possible assembly sites instead of 100. Finally, 16 of these circuits can be fit (due to their small size) on half of a 2 inch wafer.

The VT design is a VLSI design with a total of 17 assembly sites per circuit. Assembly sites on this design again vary from 3 to 6 μm, but with increments of 0.2 μm. There is a total of only one column and 17 rows in this design. The widths of all electrodes ("buried" and passive) are 3 μm.

There are three variations to the VT design. Design A of the VT design (see Figure 4) is a basic assembly circuit. Design B has its two 5 μm-wide electrodes (running parallel to the assembly sites) connected to the buried layer 1. These two parallel electrodes are connected to the 14 μm square but not to the assembly sites themselves. The parallel electrodes are used for testing a nanowire for electrical and mechanical properties after assembly. Lastly, Design C is similar to B except that it lacks the two 5 μm-wide parallel electrodes. This circuit is useful when connecting an assembled nanowire to a microprocessor for analysis.

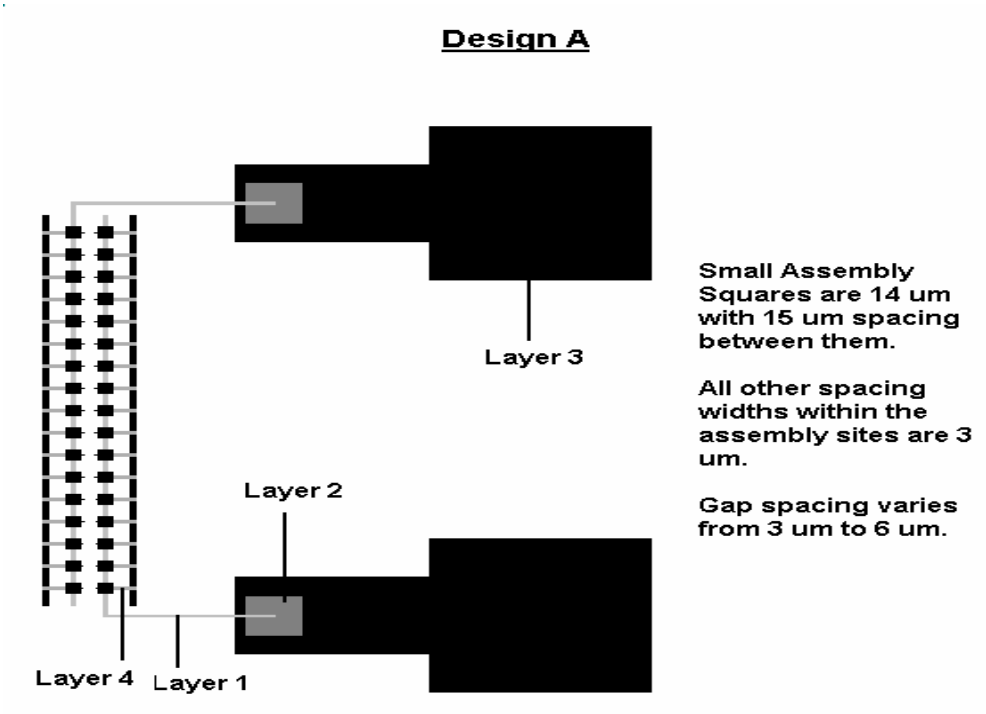


Figure 4. Design A of the VT design.

3.2 Wafer Fabrication

Fabrication of the experimental wafer involved three steps: mask fabrication, thermal oxidation / sputter deposition, and finally photolithography procedures. Each of these steps, along with the specifics of its application on these wafers, is explained in this section. This wafer fabrication involved only the experimental wafer design described in section 3.1.

3.2.1 Mask Fabrication / Photolithography

Mask fabrication involves the transfer of AutoCAD models into tangible glass plate patterns of a design. Masks serve as stencils for etching a design on a particular wafer . There are two types of masks: clear field and dark field. Each "paints" (adds) a layer on the glass plate that blocks UV light. Clear field masks "paint" the AutoCAD design on the glass plate. Dark field masks "paint" everything on the glass plate except the AutoCAD design.

These masks are important during photolithography procedures. Photolithography is the process of etching a design onto a particular layer, using UV light and photoresists. Photoresists, similar to photographic film, exhibit different chemical and physical properties when exposed to UV light. One of these properties, vulnerability to certain acids, is what allows a mask and photoresist to etch out a pattern within a layer (see Figure 5).

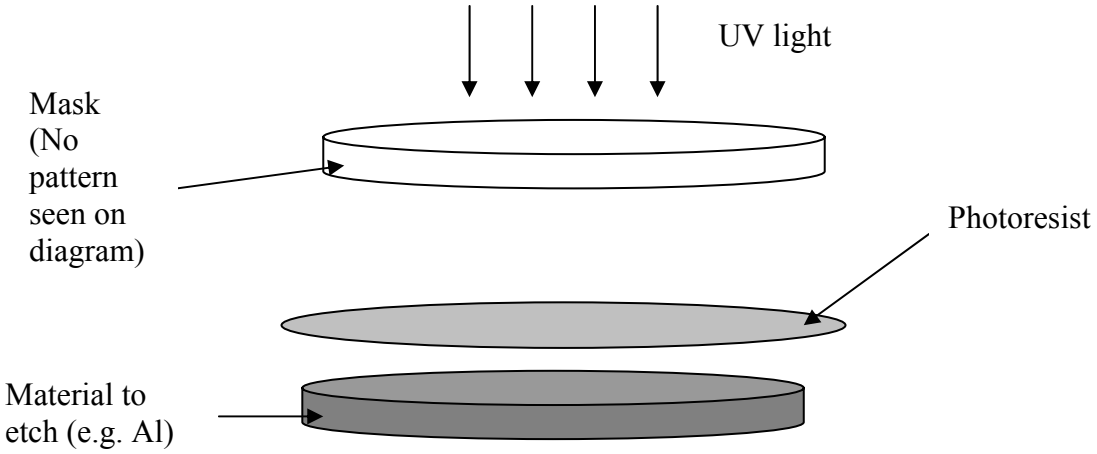


Figure 5. Photolithography setup.

Either negative or positive photoresist can be used. Positive photoresist is developed when it is exposed to UV light. Negative photoresist, on the other hand, is developed when it is not exposed to UV light. Hence, negative photoresist is usually used with a clear field mask and positive photoresist is usually used with a dark field mask (assuming that one is using the liftoff method which is described in next section).

Another important decision that must be made in photolithography is whether to use liftoff or etching methods. Etching involves the placement of a photoresist on top of the layer to etch. Then the mask is exposed to UV light, and because the mask blocks UV light where the design pattern occurs, the design is replicated on the photoresist. With the use of an appropriate acid, an imprint of the design is placed on the layer. Liftoff, on the other hand, uses sputter deposition to "grow" the design on the wafer. Liftoff is a more delicate procedure and hence is advisable in highly sensitive wafer production. However, etching allows for the addition of thicker layers.

3.2.2 Thermal Oxidation / Sputtering

The actual addition of new layers (layers to be later etched using photolithography) is done through two processes, thermal oxidation and sputtering. Thermal oxidation is most commonly used to grow the initial SiO_2 layer on top of the silicon wafer. Heating the silicon wafer to a very high temperature (usually above 900°C) grows this SiO_2 layer. However, as a result of the chemical reactions that bind the growth of this layer, only a relatively small amount can be added (approximately 500 nm). To add larger amounts of SiO_2 or any other type of layer (e.g., Al or Au layer) requires using the process known as sputter deposition. Sputter deposition excites an element (such as Al or Au) to a point that it vaporizes and then condenses on top of the silicon wafer. This creates a uniform layer of that element on top of the wafer However, sputter deposition, although allowing for greater thickness, is not as accurate as thermal oxidation.

3.2.3 Experimental Wafer Fabrication

To fabricate the specific wafer used for experimentation, the following methods were employed. A 2-inch silicon wafer was used for fabrication. The first layer added was an electrically insulating 500 nm SiO_2 layer. This layer was grown using thermal oxidation at a temperature of 950°C. The SiO_2 prevented the electrical properties of the silicon wafer from playing a role in the circuit.

The next layer added was a combination of a Cr and Au layer. First, a 10 nm Cr layer was sputtered on top of the SiO_2. Next, a 190 nm Au layer was sputtered on top of the Cr layer. Au has superior conductive properties, making it an excellent element for the first layer in the design. But Au easily peels off SiO_2. Since Cr has adhesive properties it was added to enable the Au layer to be more easily attached to the silicon wafer. After standard photolithography procedures were performed (clear field mask, 1.2 μm negative photoresist with liftoff method), the first layer of the AutoCAD model was imprinted on the Au layer.

After photolithography, an additional 150 nm SiO_2 layer was deposited on top. Unlike the first SiO_2 layer, this layer was deposited using sputtering. Layer 2 in the AutoCAD design (the layer that etches out contact points between the now "buried" first layer and an outside AC signal source) was imprinted on this SiO_2 layer using a clear field mask, positive photoresist, and the etching method of photolithography.

Next, the passive electrode layer (layer 3 in the AutoCAD model) was added by using again the liftoff method with clear field masks and a negative photoresist. This layer was made of a 100 nm Ni-Cr alloy, chosen for its Young's Modulus. This design required a material of high Young's Modulus so that it would not easily bend when a nanowire assembled on top of it. Although Cr would have been a more optimum layer, a Ni-Cr alloy was used instead because of constraints in the University of Pennsylvania's Microfabrication Facility.

The last layer added, specifically for the "clamped" version, was an additional 100 nm Ni-Cr alloy. Using liftoff with clear field masks and a negative photoresist, the fourth layer was added after electrofluidic assembly had occurred.

3.3 Assembly Parameters

Three properties decide whether a nanowire will assemble during electrofluidic assembly. These three properties are the voltage, frequency, and time for the applied electric field. The importance of each parameter is described below.

The peak-to-peak voltage is important for the AC signal source because a stronger electric field will have a greater chance to cause assembly. Although it may seem logical to consequently increase the magnitude of the voltage to the highest extent, maximizing the voltage has adverse effects because the passive electrodes may burn out. If too strong an electric field is induced across these passive electrodes, the capacitor-like electrodes,

will burn out (similar to a capacitor burning out when exposed to a very strong electric field). As with a capacitor, the larger the spacing between the two pairs of passive electrodes, the stronger the electric field required to burn them out. Hence, a 30 μm gap has a very small chance of a burnout compared to a 5 μm gap.

The second parameter deciding nanowire assembly yields is AC signal source frequency. AC fields allow for the polarization of nanowires so that they can properly assemble across the passive electrodes. A higher frequency will cause a faster polarization of the nanowires, and consequently, faster assembly. However, too high a frequency causes no polarization. In this case, the shifting electrical signals change too quickly for the nanowires to polarize. Hence, they stay as they were, in a nonpolarized state. Optimum frequency is heavily dependent on the types of nanowires used.

The last parameter that decides nanowire assembly yields is the length of time the AC signal source is applied. As with voltage, long time periods usually have adverse effects, such as causing greater electrode burnouts.

3.4 Assembly Setup

An alternating electric field is applied between arrays of adjacent 10-μm-wide metal electrodes separated by 60 μm. The field is obtained by applying a 100 kHz signal from a Topward 8110 function generator between a second pair of electrodes isolated from the top alignment electrodes by 500 nm of SiO_2. This signal is then amplified by a Bogen GA-6A amplifier and applied to the electrodes through an Electroglas Wafer Prober 131.

The optimal conditions for assembling Rh nanowires were applying an alternating field of 100 kHz – 10 V_{pp} for 5 minutes. The optimal conditions for assembling MWNTs were applying an alternating field of 100 kHz – 45 V_{pp}, also for 5 minutes. The optimal conditions for assembling GaAs nanowires were applying an alternating field of 100 kHz – 10 V_{pp} for 1 minute. Furthermore, the GaAs nanowires needed to be sonicated for 5 minutes prior to assembly. Without this sonication, the nanowires tended to clump together, making it difficult to isolate a single nanowire for testing.

An important factor of the assembly was the ability to reuse chips for multiple trials. Chips were cleaned by be being covered in acetone and sonicated for 5-10 minutes. The chips were observed during sonication to ensure that the top layer of electrodes was not being removed.

3.5 Electrical Characterization Setup

Electrical characterization was performed on Rh rods and MWNTs. Unfortunately, the dimensions of the GaAs nanowires made it impossible to run electrical tests on them. The length of these nanowires was approximately 2 μm, so they had to be assembled on chips with gap sizes of 2 μm or less. The chips used were fabricated 2 years ago without allowing for the capability to test nanowires electrically. A solution to

this problem was to design a final mask to act as the top layer, allowing electrical measurements of the assembled GaAs wires. Unfortunately, the drawings used to fabricate these chips did not include alignment marks. Thus, aligning this final layer would have been very difficult.

A final mask layer was designed and fabricated to make electrical measurements possible, but, unfortunately, it was not used for actual testing. There are plans for the mask and chips to be sent elsewhere to do electrical testing.

Electrical characterization of Rh rods and MWNTs was performed in ambient atmosphere. I-V relationships were measured at changing temperatures to calculate a trend in resistance. The devices were probed using a Micromanipulator 6000 probe station along with a Bausch & Lomb MicroZoom II microscope. A Hewlett Packard 4145B Semiconductor Parameter Analyzer was used to measure and record the I-V characteristic. The temperature of the device was raised using a Powerstat Variable Autotransformer (type 3PN116B), which heated up the metal platform on which the wafer rested. The I-V characteristic was then transferred from the Parameter Analyzer to a diskette via a program written for MS-DOS.

4. RESULTS AND DISCUSSION

4.1 "Clamping" Wafer Design

After electrofluidic assembly occurred on these fabricated wafers, the following observations were made. First, the optimum parameters for assembly for the "clamped" version were at 30 volts peak to peak at 100 kHz for 300 seconds. Second, the "unclamped" version had greater resistance to assembly than did the "clamped" version. This can be attributed to the "unclamped" version's extra capacitive properties. The "unclamped" version had two 200-μm-square contact pads in addition to the passive electrodes. These relatively large square pads probably caused greater capacitive properties to be exhibited by the assembly sites. To alleviate this property, it is advised that higher peak-to-peak voltages be introduced. Third, increases in time usually had a diminishing return after a certain point. After about 300 seconds, any increase in time had little effect in gaining higher yields for this circuit. In fact, increasing times to 600 seconds actually caused more electrode burnouts than it did additional nanowire assembly. Fourth, nanowire assembly was most likely between the gaps that had similar lengths as the nanowires. Finally, all nanowire assembly at gaps above 20 um was extremely unstable and easily breakable.

Only a quarter of the wafer design described in section 3.1.1 was used. This quarter wafer included four "clamped" version circuits with a total of 400 possible assembly sites (see Figure 6). After clamping occurred, the wafer was cut into four pieces with one circuit per new chip. Results of nanowire assembly yields after clamping are shown in Table 1. Furthermore, voltage, frequency, and time parameters are displayed below Table 1 as well as their effects.

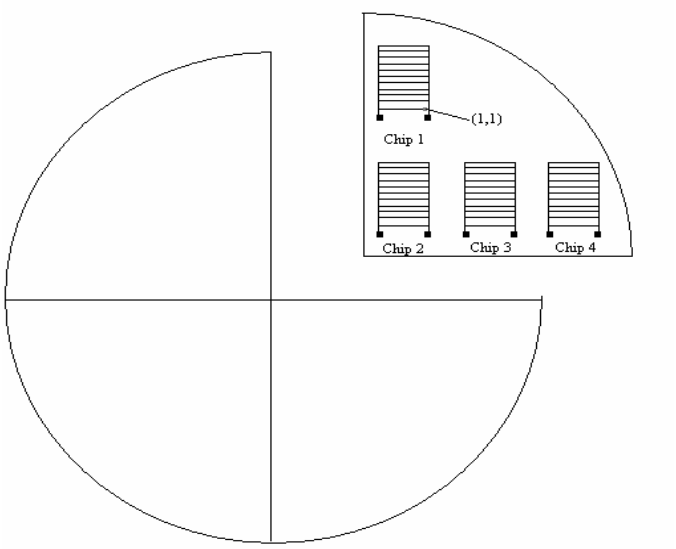


Figure 6. Chip location and details (figures not drawn to scale).

Chip 1	**Chip 2**	**Chip 3**	**Chip 4**
(1,2) (1,4) (1,7) (1,8) (2,5) (2,6) (2,7) (2,8) (3,1) (3,7) (4,5) (4,8) (4,9) (7,3)	(2,1) (2,6) (3,10) (3,8) (3,6) (3,5) (3,4) (7,10)	(1,9)	None

Table 1. Nanowires successfully "clamped"

Note: (1,2) and (1,8) had 2 and 3 nanowires, respectively, for Chip 1. (3,5) and (3,6) had 2 nanowires each for Chip 2. Parameters for assembly are 30 volts, 100 kHz for 300 seconds.

After the fourth or "clamping" layer had been placed, the chip was cut to create four pieces, each containing a circuit (see Figure 7). All nanowires that successfully clamped (as indicated in the table) remained clamped.

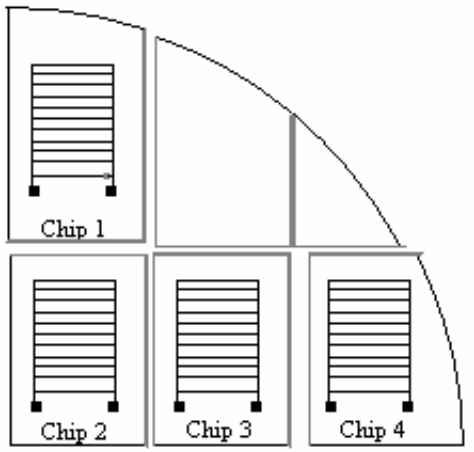


Figure 7. How the chips were cut.

The following are two pictures taken with a Scanning Electron Microscope (SEM) of the assembled nanowires for Chip 2:

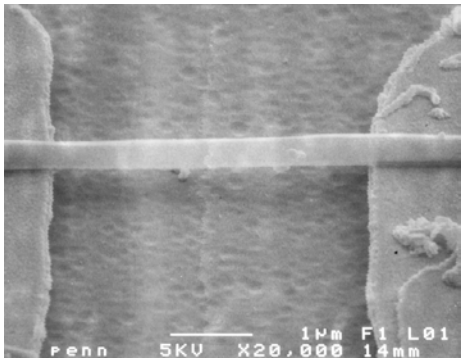


Figure 8. SEM of (2,6) on Chip 2.

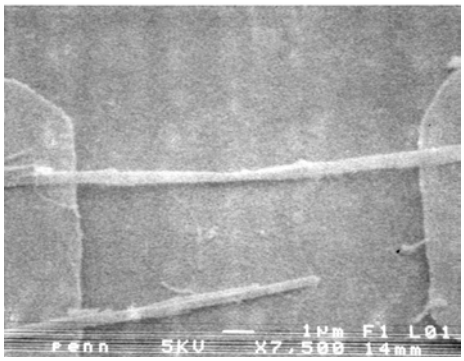


Figure 9. SEM of (7,10) on Chip 2.

4.2 GaAs Nanowires

GaAs nanowires were successfully assembled on chips with gap sizes of 1-3 µm. Figure 10 shows a picture taken with a SEM of a particular GaAs nanowire. Electrical testing could not be completed because of limitations of the chip. A final mask layer, however, was designed and fabricated to enable electrical testing. The lack of alignment marks on the original drawings will make this final layer difficult, although not impossible, to align.

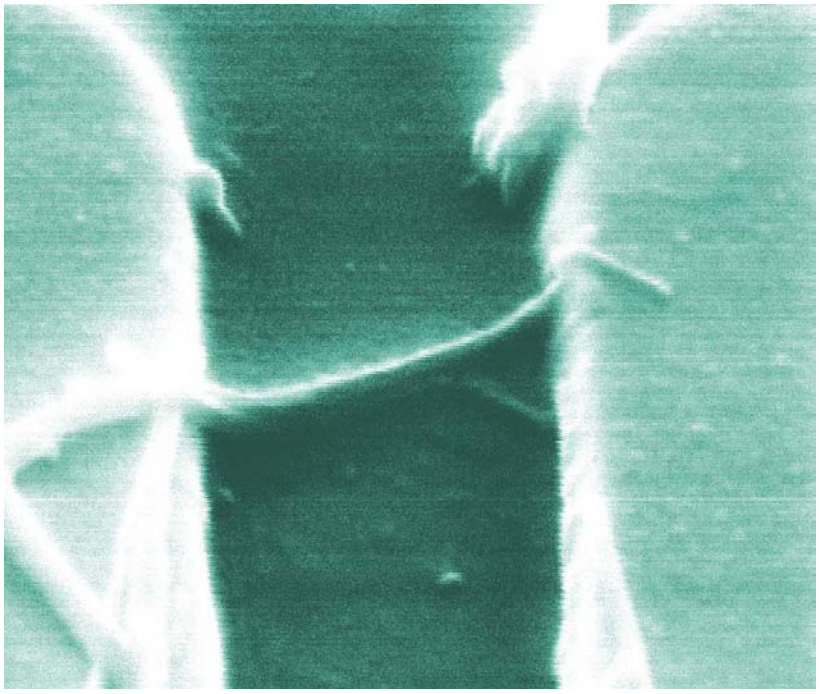


Figure 10. A picture of a GaAs nanowire taken with a SEM.

To overcome the problem of the lack of alignment marks, pre-existing features on the chip were used as alignment marks — specifically, the electrodes at sites (1,9) and (9,9). Their functionality will be sacrificed, however, because they will be exposed to UV light and thus destroyed. This was a tradeoff in designing the final layer.

A possible problem in making electrical measurements may be the large resistance of the GaAs nanowires. The resistivity of undoped GaAs at room temperature is 10^7 Ω·cm. The length of the wires is approximately 2 μm, and the wires are 20-80 nm in diameter. Assuming a diameter of 50 nm, this yields a cross-sectional area of 7.85e-11 cm^2. Using the equation $R = (\rho l)/A$, the resistance can be calculated to be 2.5e13 Ω. This is an extremely large, though expected, value for resistance. GaAs is a semiconductor, so the resistivity is expected to be higher than that of conductors. Furthermore, the wires are so thin that the ratio of l/A is quite large. Such a large resistance will complicate electrical measurements because large values are difficult to sense accurately. Methods of lowering the resistance include doping the wires or raising the temperature. Since GaAs is a semiconductor, both of these methods will decrease the resistance, thus making resistance a characteristic that can be measured by available instruments.

4.3 Rhodium Nanowires

Rh nanowires were successfully assembled on various chip designs. They were first successfully assembled on chips with a gap size of 1-3 μm. The optimal conditions for these chips were a signal at 100 kHz – 15 V_{pp} for 5 minutes. These wires could not be tested because of the limitations of the chips. A successfully assembled Rh nanowire on these chips is shown in Figure 11.

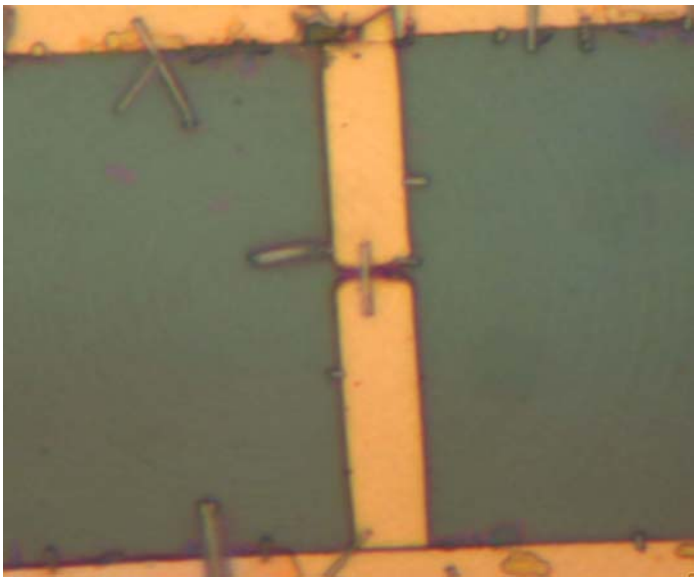


Figure 11. An assembled Rh nanowire.

The Rh nanowires were also successfully assembled on chips with gap sizes of 5-30 μm. Since the gap sizes are larger on these chips, a stronger field must be applied to the chips to achieve successful assembly. Thus, the optimal conditions for these chips were a signal at 100 kHz – 40 V_{pp} for 5 minutes. This stronger field did not damage the chips at all because the spacing of the features was relatively large.

Testing of the Rh nanowires was done on the second group of chips (with gap size of 5-30 μm). Figure 12 shows the I-V characteristic for a Rh nanowire at room temperature. The slope of this line is 0.0036 A/V, which yields a resistance of 277 Ω. The resistivity of Rh is known to be 4.3 μΩcm. The nanowire was bridging a gap of 5 μm, so the length will be assumed to be 5 μm. The diameter of the nanowires is approximately 200 nm, which yields a cross-sectional area of 3.14e-10 cm^2. These values give a theoretical resistance of 6.85 Ω. This indicates that the measured resistance values include a large contact resistance between the Rh rod and the Ni-Cr electrode on which it rested.

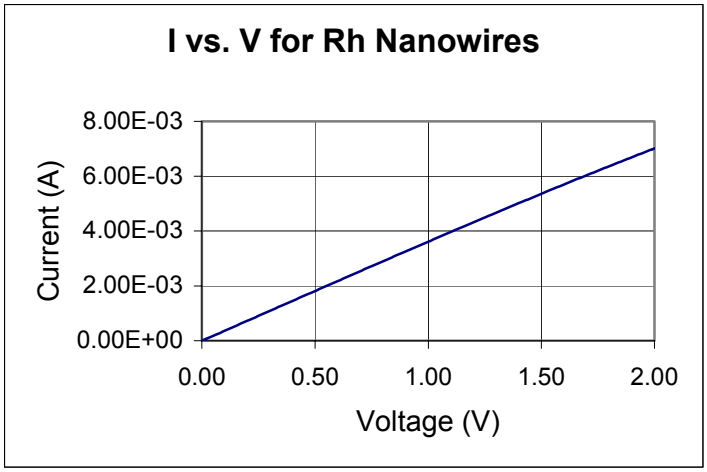


Figure 12. I-V characteristic for a Rh nanowire

The temperature was then raised to measure the I-V characteristic again. Unfortunately, once the temperature reached 50ºC, the I-V characteristic became noise. Inspection with an optical microscope showed that the Rh rod had burned at the end and had become disconnected from the electrode. A method of preventing this in the future may be to increase the temperature more slowly. Rapid changes in temperature may have been too strenuous on the materials used. Thus, slowing down the change in temperature may prevent the rod from burning and allow accurate data to be taken. This method was not implemented in this project because of both time and equipment constraints.

4.4 Multi-walled Nanotubes

Multi-walled carbon nanotubes (MWNTs), while more difficult to assemble than Rh nanowires, proved to be more easily measured while raising the temperature. Figure 13 shows the I-V characteristics for MWNTs at different temperatures. From these data, it is evident that as the temperature increases from room temperature to 125°C, the resistance of the MWNT decreases. At room temperature, this particular MWNT had a resistance of approximately 200 kΩ. At 125°C, the same MWNT had a resistance of approximately 100 kΩ.

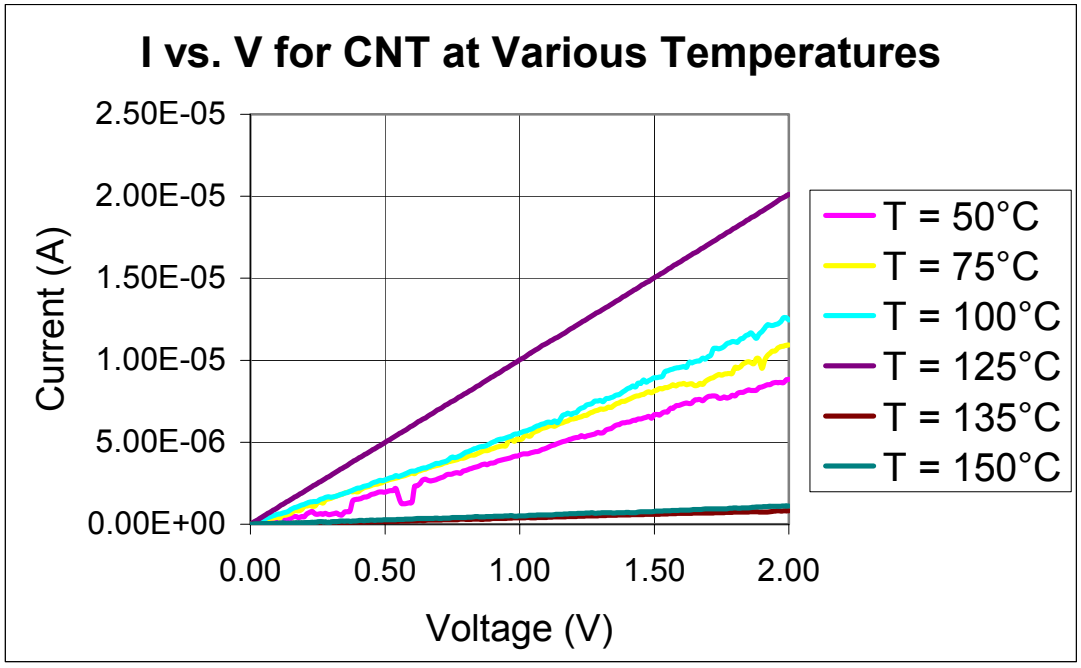


Figure 13: I-V characteristic for an MWNT at different temperatures.

These data are not unexpected. The MWNTs used are essentially sheets of a graphite lattice rolled into a cylinder. The MWNTs can be thought of as objects in three dimensions: *a*, *b*, and *c* (see Figure 14). The *a* and *b* dimensions represent the horizontal plane of the lattice, while the *c* direction represents the vertical plane. Electrically, graphite is a semi-metal, which means that it is a conductor in the *ab* plane and an insulator in the *c* plane. When conductors are exposed to heat, their resistivity will increase. When insulators are exposed to heat, their resistivity will decrease.

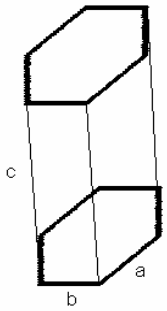


Figure 14. A structural representation of graphite.

The MWNTs were fabricated such that resistance was being measured across the insulating *c* plane. Resistivity is known to decrease with temperature along this plane, so decreasing resistance is normal.

The data in Figure 13 were then used to calculate an energy barrier that is common to semiconductors. For any wire, $R = (\rho l)/A$ or $R = l/(\sigma A)$, where ρ is the resistivity and σ is the conductivity. Furthermore, $\sigma = n\mu q$, where n is the carrier concentration, μ is the mobility of carriers, and q is the charge of an electron. Thus, it

follows that R = l/(nμqA). For semiconductors at high temperatures, resistance is dominated by the carrier concentration n. The equation relating carrier concentration and temperature is $n = (N_c N_v)^{1/2} e^{-E/(kT)}$, where n is the carrier concentration and E is the energy barrier. If ln(n) vs. 1/T is graphed, then the slope of that graph is equal to –E/k. Ln(n) vs. 1/T was graphed for this MWNT, and the slope was calculated to be $-1482\ K^{-1}$. This graph is seen in Figure 15. By using the fact that $-1482\ K^{-1} = -E/k$ where k is the Boltzmann constant, it is shown that E = 2.05e-20 J, or E = 0.13 eV.

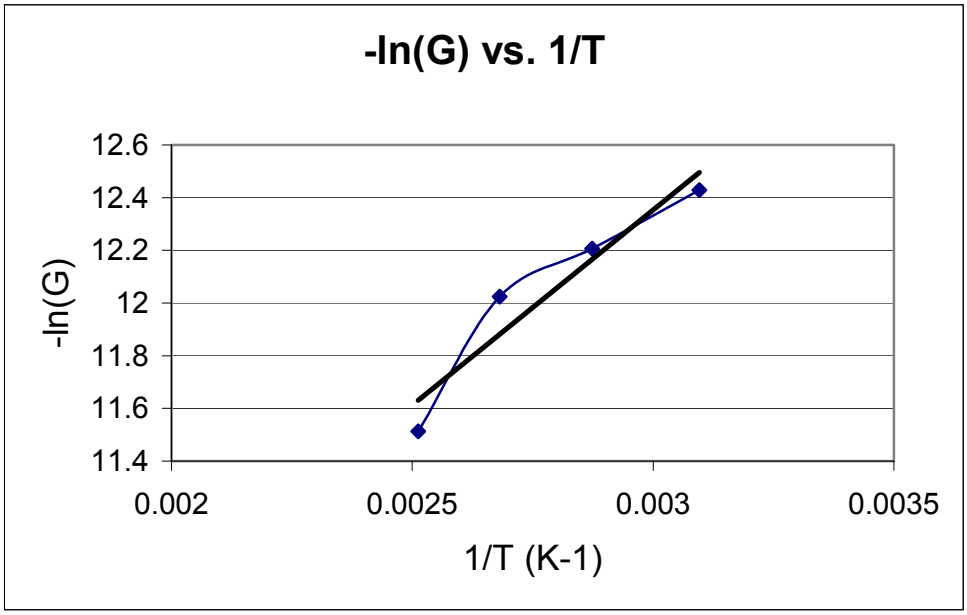


Figure 15. –Ln(G) vs. 1/T for an MWNT. This graph was used to calculate the energy barrier.

This value is quite small when compared with band gaps of semiconductors, because the MWNTs are not semiconductors. Their decrease in resistance with respect to temperature may not be due mostly to carrier concentration. On the contrary, the increasing temperature may be affecting the mobility of the charges, rather than their concentration. In that case, the above equations should be ignored.

Using the equation R = ρl/A, the theoretical resistance of these nanotubes can be calculated. The resistivity of the nanotubes is given as 55 μΩ cm. The nanotube was bridging a gap of 15 μm, so the length of the tube is approximated to 15 μm. The outer diameter of the nanotubes was estimated using a picture taken with a SEM. It was approximated to be 600 nm. The walls of the tubes were given as 12 nm, which yields an area of approximately 2.22e-10 cm^2. These values yield a theoretical resistance of 372 Ω.

The slopes on the graph of Figure 13 can also be used to calculate the resistance of the nanotubes. The slopes of the graph are equal to 1/R, where R is the resistance of the MWNT. At room temperature, the slope was approximately 5e-6 A/V, which gives a resistance of 200 kΩ. This value is three orders of magnitude larger than the theoretical resistance calculated in the previous paragraph. Therefore the measurements were

including a large contact resistance between the MWNT and the Ni-Cr electrode on which it rested.

Furthermore, data were taken on MWNTs that bridged gaps of varying sizes. At room temperature, a nanotube that was 10 μm long demonstrated a resistance of 33 kΩ. This finding is consistent with the fact that as the length decreases (from 15 μm to 10 μm) the resistance should also decrease. However, at room temperature, a nanotube that was 6 μm long demonstrated a resistance of 310 kΩ. Since this nanotube was shorter, it should have yielded a resistance that was smaller than the previous two nanotubes. The fact that it did not also supports the idea that the resistance measurements were including a large contact resistance.

However, the data presented here can still be used to assert the fact that resistance will decrease with an increase in temperature. The contact resistance between the nanotube and the electrode remained constant throughout the tests, as the nanotube did not move. Thus, the resistance values measured may not be the actual resistance of the tubes, but they will all differ from the actual value by the same constant. The trend of decreasing resistance is still valid.

The trend of decreasing resistance, however, does not continue with an increase in temperature. Between 125°C and 135°C, the I-V characteristic changes drastically. Between these two temperatures, the resistance of the MWNT increased from 100 kΩ to 2 MΩ. This trend of decreasing resistance and then increasing resistance was repeated in other trials. A possible explanation is that between 125°C and 135°C a critical temperature was reached that fundamentally changes the electrical properties of the tubes.

The resistance trend was then measured in the reverse direction to see if, after raising the temperature and lowering the resistance, lowering the temperature could result in an increase in resistance. Since in previous tests the data indicated a critical point between 125°C and 135°C, the temperature was raised only to 100°C. Figure 16 shows the data taken when the temperature was raised and then lowered. A (D) next to the temperature in the legend indicates that the temperature was raised and then lowered back down to the recorded temperature. Again, as expected, the current increased with an increase in temperature. This shows a decrease in resistance with an increase in temperature.

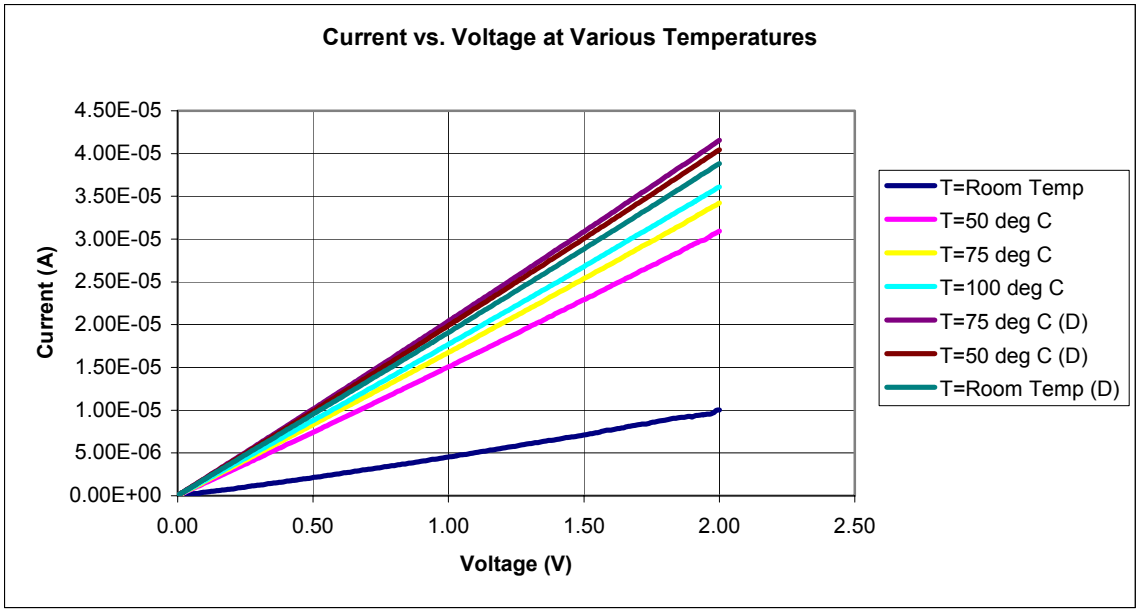


Figure 16. I-V characteristic for an MWNT at various temperatures.

When the heater was turned off and the temperature was lowered, however, the resistance did not increase again back to its original value. Instead, the resistance remained at the value it had reached at 100°C. The data indicate that the resistance may have even decreased a bit more after the heater was turned off.

There are various possible explanations. One explanation is that the temperature being measured was the temperature of the metal surface on which the wafer rested. The thermometer was resting on the same metal surface as the wafer, thus measuring the temperature of the surface. The temperature of the electrodes and the nanotube may have been different. The Ni-Cr electrodes may have taken longer to cool down than the metal surface. This explanation is supported by the data demonstrating that the resistance decreased a bit even after the heat was turned off. This shows that the electrodes and nanotube on the chip were slow to react to the change in external temperature. Thus, when the external temperature had cooled to 50°C, the temperature of the nanotube may still have been at 100°C. Also, data taken after the heat was turned off indicate a small increase in resistance. This suggests that the resistance is actually increasing, but it is unclear if the resistance value will return to the original room temperature value.

Another possible explanation is that heating the nanotube permanently changes the electrical properties of the tube. Hence, once the temperature is raised and the resistance is decreased, the resistance will never increase again. Resistance values were not coming down as quickly as they should have been once the heat was turned off. While the data did appear to be decreasing a bit, the changes in current could have been general fluctuations in measurement, errors caused by the tools that were used. The resistance value may not have been increasing, but rather fluctuating around a certain point. Unfortunately, not enough data were taken at the time to confirm this.

5. CONCLUSIONS

This report demonstrated the assembly parameters and wafer design that enabled electrical characterization of various types of nanostructures. It showed that large electrode pads on the Si wafer contributed large capacitive forces that made assembly difficult. Thus, when assembling nanowires or nanotubes, it is preferable to assemble on a chip without these large pads. However, for testing, these pads are necessary to probe from the outside world. One way to overcome these capacitive forces was demonstrated in this report. The solution was to assemble nanotubes or nanowires on wafers without these large pads and then do a final layer of processing in which they are added through photolithography and metal deposition. This method allows nanostructures to be assembled relatively easily and tested conveniently.

This report also demonstrated the I-V characteristic for Rh nanowires. This information can then be used to calibrate the nanowire if it is used as a sensor. It could, in effect, be used as a thermometer by measuring the resistance and finding the corresponding temperature. More important, however, were the tests done on the MWNTs, which showed that, in general, as temperature increases, the resistance of the nanotube will decrease. These tests also showed that heating the nanotubes permanently may change their electrical properties, as the trend could not be reversed.

6. RECOMMENDATIONS

Future work includes the characterization of GaAs nanowires. Because of the small size of the nanowires, this needs to be done at facilities that have greater resolution photolithography. Furthermore, mechanical tests can be done on all three structures. Resonant frequencies and quality factors could be found for these nanostructures and then be used to create other sensors. Ultimately, various types of nanowires and nanotubes could be placed in a large array. Since different types of nanowires react differently to different stimuli, this array would be capable of sensing multiple substances simultaneously. These types of sensors have applications in everything from medicine to defense.

7. ACKNOWLEDGMENTS

The authors would like to thank their advisor, Dr. Stephane Evoy, for guidance and direction throughout this project. They would also like to thank Mr. Vladimir Dominko, the manager of the microfabrication lab at the University of Pennsylvania, for his countless hours helping them fabricate wafers and design mask layers. The authors thank Mike Riegelman for his overall guidance and help during this project. They also thank Dr. Jan Van der Spiegel and Ms. Lois Clearfield for their organization of the program and general administrative help. Lastly, the authors want to thank the National Science Foundation for funding the SUNFEST Program and giving undergraduates the opportunity to have this research experience.

8. REFERENCES

1. MEMSnet. *What is MEMS Technology?* MEMS Exchange, 2003. Online. University of Pennsylvania Engineering Library. Internet. 30 July 2003.
2. X. Duan, J. Wang, and C.M. Lieber, Synthesis and optical properties of gallium arsenide nanowires, *Appl. Phys. Lett.*, 76 (2000) 1116-1118.
3. M.S. Gudiksen, L.J. Lauhon, J. Wang, D.C. Smith, and C.M. Lieber, Growth of nanowires superlattic structures for nanoscale photonics and electronics, *Nature*, 415 (2002) 617-620.
4. X. Wang, Z. Hu, Q. Wu, X. Chen, and Y. Chen, Synthesis of multi-walled carbon nanotubes by microwave plasma-enhanced chemical vapor deposition, *Thin Solid Films*, 390 (2001) 130-133.
5. Y. Cho, G. Choi, S. Hong, and D. Kim, Carbon nanotube synthesis using a magnetic fluid via thermal chemical vapor deposition, *J. Crystal Growth*, 243 (2002) 224-229.
6. J. Hu, T.W. Odom, and C.M. Lieber, Chemistry and physics in one dimension: Synthesis and properties of nanowires and nanotubes, *Acc. Chem. Res.,* 32 (1999) 435-445.
7. J. Westwater, D.P. Gosain, S. Tomiya, and S. Usui, Growth of silicon nanowires via gold/silane vapor-liquid-solid reaction, *J. Vac. Sci. Technol. B*, 15 (1997) 554-557.
8. A.M. Morales and C.M. Lieber, A laser ablation method for the synthesis of crystalline semiconductor nanowires, *Science*, 279 (1998) 208-211.
9. X. Duan and C.M. Lieber, General synthesis of compound semiconductor nanowires, *Adv. Mater.*, 12 (2000) 298-302.
10. P.M. Ajayan, Ph. Redlich, and M. Ruhle, Balance of graphite deposition and multishell carbon nanotube growth in the carbon arc discharge, *J. Mater. Res.*, 12 (1997) 244-252.
11. H..J. Lai, M.C.C. Lin, M.H. Yang, and A.K. Li, Synthesis of carbon nanotubes using polycyclic aromatic hydrocarbons as carbon sources in an arc discharge, *Mater. Sci. Eng. C*, 16 (2001) 23-26.
12. Z. Yuan, H. Huang, L. Liu, and S. Fan, Controlled growth of carbon nanotubes in diameter and shape using template-synthesis method, *Chem. Phys. Lett.*, 345 (2001) 39-43.
13. J. Kong, H.T. Soh, A. Cassell, C.F. Quate, and H.Dai, "Synthesis of Individual Single-Walled Carbon Nanotubes on Patterned Silicon Wafers" *Nature,* 395 (1998) 878.
14. A. Cassell, N. Franklin, E. Chan, J. Han, and H. Dai, "Directed Growth of Free-Standing Single-Walled Carbon Nanotubes" *J. Am. Chem. Soc.,* 121 (1999), 7975-7976.
15. A. Cassell, J. Raymakers, J. Kong, and H.ongjie Dai, "Large Scale CVD Synthesis of Single-Walled Carbon Nanotubes" *J. Phys. Chem.,* 103 (1999) 6484-6492.
16. H.T. Soh, A. Morpurgo, J. Kong, C. Marcus, C. Quate, and H. Dai, "Integrated Nanotube Circuits: Controlled Growth and Ohmic Contacts to Single-Walled Carbon Nanotubes" *Appl. Phys. Lett.,* 75 (1999), 627-629.
17. Mirkin, C. A., Letsinger, R. L., Mucic, R. C. & Storhoff, J. J. A DNA-based method for rationally assembling nanoparticles into macroscopic materials. *Nature* 382, 607 (1996).

18. R.C. Mucic, J.J Storhoff, C.A. Mirkin, and R.L. Letsinger, “DNA-Directed Synthesis of Binary Nanoparticle Network Materials,” *J. Am. Chem. Soc.,* 120 (1998) 12674-12675.
19. P. Smith; C. Nordquist; T. Jackson; T. Mayer; B. Martin; J. Mbindyo; T. Mallouk; “Electric-field assisted assembly and alignment of metallic nanowires” *Applied Physics Letters* 77 (2000), 1399-1401.
20. S. Evoy , B. Hailer, M. Duemling, B. R. Martin, T. E. Mallouk, I. Kratochvilova and Theresa S. Mayer, "Electrofluidic Assembly of Nanoelectromechanical Systems", in Materials Science of Microelectromechanical Systems (MEMS) Devices IV, MRS Symp. Proc. 687 (2002).

OPTIMIZED METHODS FOR EARLY CANCER DETECTION VIA OPTICAL IMAGING WITH A REDOX SCANNER

NSF Summer Undergraduate Fellowship in Sensor Technologies
Jennifer Geinzer, Electrical Engineering – University of Pittsburgh
Advisor: Dr. Britton Chance

ABSTRACT

Modern medicine requires more accurate resolution and newer detection methods to properly diagnose cancerous tumors. The purpose of the redox scanner is to utilize near-UV, visible light, and NIRS to image at low temperatures the biophysical signs of cancer, with the ultimate purpose of early detection and diagnosis for human patients. The scanner will perform these functions *in vitro* through 3-D imaging of metabolic redox biochemicals, hemodynamic signals, and injected artificial beacons of a tissue sample at 80 x 80 x 80 μm spatial resolution.

An outdated version of the redox scanner was operational but not ideal; in the scope of SUNFEST, the goal was to fabricate a prototype of an improved device based on the biomedical principles of the old machine, and then validate it in its initial stages by comparison of the actual test samples to the scanner data as modeled in Matlab™.

During the 10-week span of SUNFEST, the project progressed from early development and low functionality to near completion. Several vital system improvements were implemented, to include a new optoelectronics driver circuit to provide reliability, a DOS motor controller to fix critical Windows timing errors, and Matlab™ imaging software. Trial results were promising and signal the possible end of the system design phase. The next phase will involve testing on an animal model to find the efficacy of the sensors on actual biological systems.

Table of Contents

1. INTRODUCTION

In the field of medical imaging, precise diagnosis and treatment of cancerous tumors is a top priority. Modern-day medicine currently utilizes several imaging techniques to detect cancer, including magnetic resonance imaging, ultrasound, and x-rays. These techniques provide excellent anatomical status on the size and location of tumors, but poor functional information on the chemical processes within the target tissue.

Subsequently, many cancers go undetected, and a number of unnecessary biopsies are performed. In the field of breast cancer alone, it is estimated that 10%of patients with cancerous tumors (about 20,000 people) go undiagnosed per year, and approximately 50% of the biopsies done are later found to be benign [1]. These statistics indicate that current diagnostic methods are by themselves adequate for cancer identification. Optical tomography, near-infrared spectroscopy (NIRS) [2], and fluorescence spectroscopy [3] show promising results of illuminating the biochemical processes of tissue, without immobilization of the patient. In future years, these technologies may be coupled with extant techniques to detect, diagnose, and locate cancers with extreme precision. Until then, a medical standard is needed to list the biochemical indicators that definitively designate the presence and status of diseased tissue.

2. GOALS

The goal of the redox scanner is to provide 3-D high-resolution images of the biophysical signs associated with cancer. To accomplish this, visible light, NIR, and near-UV probes are used to examine six reduction/oxygenation (redox) biochemicals, hemodynamic signals, and injected artificial beacons in a given tissue sample. Through the imaging process, the efficacy and sensitivity of monitoring redox processes can be compared to that of metabolic signals and chemical beacons, to create a much-needed "gold standard" of the biochemical indicators for cancer.

This machine is intended for commercial utilization in the health care field. After the initial system design phase, it will be tested for cancer diagnosis in animal models, and eventually will serve as an early detection and biopsy tool for human patients.

3. BACKGROUND

The six metabolic, hemodynamic, and molecular target beacons observed by the redox scanner are chosen exclusively for their individual imaging characteristics, as well as their ability to cross-reference one another. The chemicals intrinsic to the human body include mitochondrial nicotinamide adenine diphosphate (NADH), mitochondrial flavoprotein (Fp), oxygenated hemoglobin (labeled Hb_O for this experiment), and total hemoglobin (Hb_T); the two extrinsic signals are the biochemical indicator dyes Cy5.5 and Indocyanine Green (ICG). These signals are imaged in one procedure at explicit temperatures that provide optimal emission and scanning conditions to ensure accuracy of cancer detection.

3.1 Chemicals Observed

Of the four intrinsic properties under study, the two redox chemicals are reduced NADH and Fp. Cells employ these chemicals in metabolic processes. In the biomedical field, it is common knowledge that cancer-induced hyper metabolism causes tissue hypoxia. This in turn decreases the amount of Fp (oxidized) and increases the presence of NADH (reduced) in tumor tissue.

Fluorescence emission spectroscopy can be employed to observe these properties,. In 1958, Chance and Baltscheffsky reported that NADH from the mitochondrial matrix auto fluoresces to a wavelength of 450 nm when tissue is excited with light of wavelength 366 nm [4]. Similarly, mitochondrial Fp emits light in the oxidized form at 520 nm with an excitation of 460 nm. In conventional methods of analysis, a redox ratio of NADH/Fp is calculated and then compared to a standard. Figure 1 illustrates the key locations of excitation, reflectance, and fluorescence emission spectra in the visible spectrum for the six chemicals under investigation.

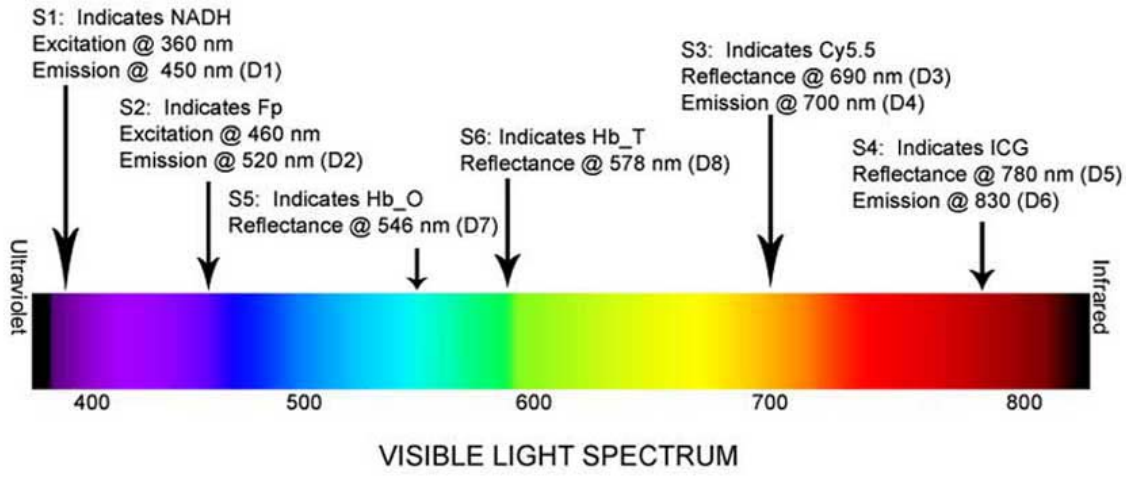


Figure 1. Source 1 (labeled S1) and S2 emit wavelengths in the near-UV and indicate redox chemicals, S5 and S6 create light in the visible spectrum and indicate molecular target beacons, and S3 and S4 provide NIR light and show hemoglobin.

Tissue health may also be evaluated by observing hemodynamics. Hb_O reflects light at 546 nm, and Hb_T reflects 578 nm light. Research in progress has indicated that malignant tumors show a low ratio of Hb_O/Hb_T when compared to healthy tissue, but benign tumors may show a normal to high ratio of the two [1]. Therefore, hemodynamics could be useful not only in detecting cancer but also in determining the difference between metastatic, malignant, and benign tumor types.

The two extrinsic probes used in this experiment are indicator dyes introduced to the tissue via localized or venous injection. ICG is a blood vessel indicator, and Cy5.5 is a cathepsin B indicator. Angiogenesis, the creation of new blood vessels, increases the bloodflow in a particular region, and thus increases total hemoglobin. The ICG beacon

can be cross-referenced with the previous two signals to understand hemodynamics more fully. The presence of the enzyme Cathepsin B has been shown to signify the presence of HT1080 type mouse cancers in recent experimentation [5]. The two indicator dyes both fluoresce and reflect light; Cy5.5 reflects light at 690 nm and fluoresces at 700 nm, while ICG reflects at 780 nm and fluoresces at 830 nm. Imaging of these beacons will indicate quantitatively how effective they are in molecular bonding and pinpointing tumors.

3.2 Temperature Specifications and Effects on the Scanner

Cryo-imaging has multiple advantages over scanning at room temperature. For this reason, the redox scanning procedure calls for the imaged tissue samples to be freeze-trapped and cooled with liquid nitrogen to 77 K. Molecular signals move very quickly toward the reduced state if the host tissue reaches temperatures over –20°C [6, p.15]. Freeze-trapping prohibits new oxygenation and metabolism of the test sample, enabling researchers to look at one biochemical "snapshot" in time. At liquid nitrogen temperatures, tissue is hard enough to be mounted and mechanically ground with standard steel milling tools. In addition, low temperatures have been proven to enhance tissue fluorescence [7], allowing greater signal transmission and thus provide greater signal-to-noise ratio for the fluorescing chemicals.

The extremely low temperatures significantly affect the operation of the opto-electronic components of the system and must be taken into consideration. Simple imaging methods where the source and detector pairs are very close to the scanned medium are impractical at these temperatures because of their severity and the sharp 221 K temperature differential between the immediate vicinity of the sample and the ambient temperature. If the sources and detectors are moved away to shield them from the temperatures, they will not provide the necessary 80 μm resolution due to diffusion of the light.

The solution to this problem is to use a light guide. The redox scanner utilizes one constructed from 14 fiber optic cables—six source fibers of 1000 μm each were connected around the circumference of a coupling, and eight detector fibers of 100 μm each are located in the middle of the coupling. The optimal design of the fiber tip coupling is shown in Figure 2.

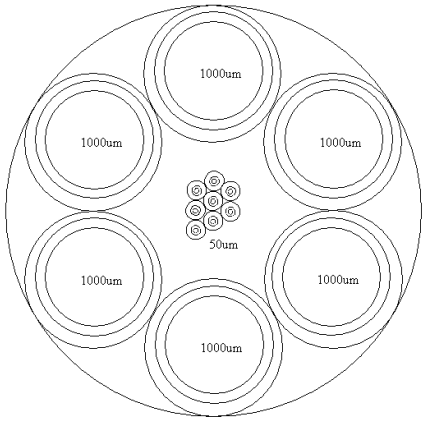


Figure 2. To maximize transmission of light through the fiber optics, the coupling is designed so the six source fibers surround the eight fibers.

The final component affected by the temperature conditions is the grinder. Before the cutting blades come in contact with the sample, both must be the same temperature. This condition is attained by immersing the blades in liquid nitrogen prior to the initial grind sequence.

4. SCANNING PROCEDURE

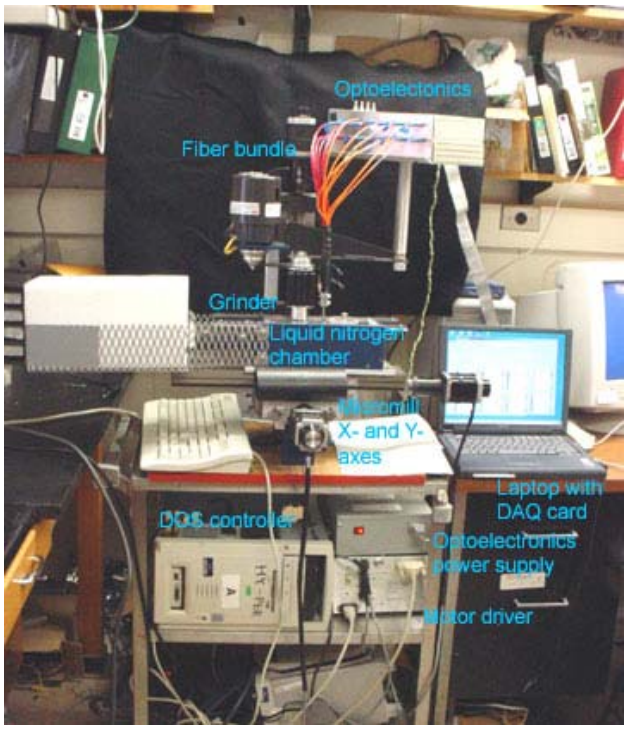


Figure 3. The main components of the redox scanner, to include the movement system, optoelectronics, and laptop, are labeled above.

In future testing, a tumor will be fostered on an animal model, and will subsequently be freeze- trapped, removed, and then mechanically secured with a chuck in a custom-built liquid nitrogen chamber. Figure 3 labels each component of the system. The liquid nitrogen will then be poured into the bottom of this chamber, freezing the chuck into place and keeping the tissue sample at low temperatures. The tumor is then prepared for scanning.

The imaging procedure is accomplished through use of the three main components of the redox scanner: the movement system, comprised of the MicroMill 2000™, driver, and a DOS controller; the opto-electronics; and a NI-DAQ™ PCMCIA-1200 card interfaced to laptop. The DOS controller is a recent addition to the system, and will be discussed in greater detail in Section5.5.

4.1 Mill and Motion

The MicroMill 2000™ is a standard hobbyists' milling tool that provides all of the movement required for the scanner. The mill shifts the liquid nitrogen chamber fitted to the x- and y-axes, and controls the vertical movement of the fiber optics and a 12-blade grinder on the z-axis.

The specific types of motors used for this platform include three 4-phase unipolar stepper motors enabled with half step mode. Each motor controlling one axis is run by four solenoids in the up, down, right, and left positions. Whenever the software provides current through any one or two coils simultaneously, the resulting magnetic response turns the gears. This allows eight steps of motion—the four cardinal phases, and the half steps in between. While the programming code provides a series of pulses through the driver to successive clockwise or counter-clockwise (cw/ccw) coils, the motor moves the nitrogen box in a positive or negative direction. As with any gear-based transport system, the mill cannot be run from a halt to a fast speed, so the velocity must be ramped with a constant acceleration. The motors can move the axes and grind with a step increment as

small as 3.175 μm. The axes also have knobs for manual control, while the motors are unpowered.

Affixed to the x- and y-axes are a Styrofoam™ filling chamber for adding liquid nitrogen as well as the Plexiglas® nitrogen chamber that houses the tissue sample. A chuck is fitted on the inside of this box to secure the sample. The optics bundle is held securely by the z-axis. The entire scanning sequence is run by the laptop via the "RedoxScanner" Microsoft Visual C++™ (hereafter referred to as VC++) software developed by the lab and a DOS computer.

The sequence begins by immersing the grinder in liquid nitrogen. The laptop prompts the user to run an initial calibration algorithm, and then the mill calls for the initial grind: It moves the sample underneath the grinder, smoothing out the top layer of the tissue.

The actual scanning motion begins after the sample is repositioned so that the optics bundle hovers over the top left corner of the sample. The sample is scanned from top to bottom (horizontally), and then the sample is moved left, reset to the top, and so on for a total of 128x128 pixels/cm^2 sample surface. The user can alter this scan size as needed.

Since the mill uses stepper motors, there is some play in the gears when shifting directions along one axis. To ensure that the starting position of each scan is on the same coordinates, after scanning down (horizontally) 128 pixels, the reset subroutine calls for fast movement of 133 pixels up and five back down to the y-axis start position.

After the first slice is scanned, the grinder removes 80 μm of tissue, and the scanning process restarts. The end result is eight tissue slices per compound, for a total of 64 images. These can then be analyzed to get a 3-D view of the biochemical, physiological, and metabolic processes of the tissue.

4.2 Optics and Data Acquisition

The optoelectronics and data acquisition funcion as follows. A National Instruments™ PCI 1200 data acquisition (DAQ) card is connected to a Windows 2000™ laptop. The "RedoxScanner" software supplies the light sequence in the form of voltage pulses, which are shuttled through the DAQ card to the driver circuit. The signals then run the light sources in the manner determined by the timing diagram. The flowchart and timing diagrams illustrate the steps in the process, see Figure 4 on the following page. The timing sequence for the sources labeled S1-S6 is {S1, S2, S3, S3, S4, S4, S5, S6}. Source 3 and S4 are given two pulses each, as they excite tissue such that it has both a reflection and fluorescence response. These two responses are measured by separate detectors. To generate near-UV light, SpectraLEDs are used for S1 and S2. The NIR S3 and S4 signals are provided by laser diodes, and S5 and S6 in the visible spectrum are provided by green and yellow LEDs.

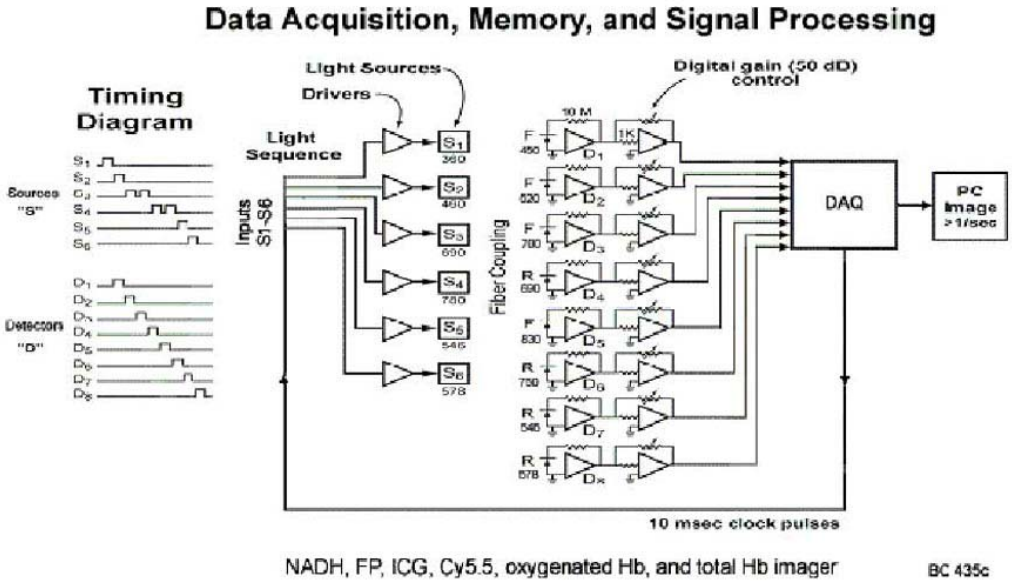


Figure 4. Light pulse timing is supplied by the DAQ card, as shown in the flowchart and the timing diagrams. Since the artificial molecular beacons Cy5.5 and ICG (light provided by S3 and S4) both reflect and fluoresce, they have two detectors each to measure these responses.

The light sources are held perpendicular to and flush against optical fibers. This light guide is secured by the mill, and is held 80 μm above the surface of the sample to maximize photon transmission and for the appropriate resolution. The light is then either reflected or fluoresced and transmitted back through the detector optics. Fluorescence signals are detected by photomultiplier tubes (PMTs), and the reflected light is measured with OPT101 photodiodes.

The DAQ card runs the sampling of the detector sequence simultaneously with the source sequence, in the manner {D1, D2, D3, D4, D5, D6, D7, D8}. These signals are amplified and then sent to the DAQ, where the data are converted from analog to digital, processed through an averaging filter, and output as a *.txt file. The resultant data can then be imaged by Matlab™. The current data acquisition calls for eight averaged samples taken per pixel, for a total of 16384 pixels and a spatial resolution of 80 μm. The entire scan sequence takes approximately 6.7 hours, which is 50 minutes per slice, or 183 ms per pixel.

5. TESTING AND EVALUTATION

This project is a work in progress, so all work accomplished, intermediate results, and debugging are presented here.

By the beginning of SUNFEST 2003, the macroscopic designs and specifications had already been determined. As an updated prototype of an older redox scanner, the

128x128 pixels/cm^2 (80 μm) resolution had been concluded to be optimal. The primary researchers in the lab had decided to add ICG, Cy5.5, Hb_O, and Hb_T to the list of scanned chemicals. The mill was already assembled, the liquid nitrogen box was built, VC++ code to run both the mill and the optics had been written, the fiber optics bundle had been coupled, and a first driver circuit had been constructed.

The following sections discuss the solutions and troubleshooting for various software and hardware problems observed in the system.

5.1 Software Calibration Bugs

During the first tests of the graphical user interface (GUI) for the mill, the original motion and calibration sections of the VC++ code prevented the end user from completing the calibration sequence. This problem stemmed from confusing conventions used to designate the axis motion, and the incorrect documentation of this movement. The "MMTest" code, designed to test the axes and familiarize the user with the positive and negative conventions, contradicted the "RedoxScanner" code concerning directional mapping.

The mechanical movement system on the mill was difficult to visualize, which exacerbated the situation. To scan, the fiber optics system that sampled the data was stationary while the box and the grinder moved, which was a somewhat unorthodox configuration. In addition, the knobs on all the axes did not turn clockwise or counter-clockwise consistently to move in any "positive" labeled direction.

After several false starts, it was determined that the easiest way to picture the system was to relate the horizontal x- and y-coordinates to standard graphing axes. It was important to note that the x and y direction conventions were taken with respect to the movement of the liquid nitrogen box underneath the optics. Facing the mill, the "positive" motion was right for the x-axis and up for y-axis. On the z-axis, vertical up and down moved the grinder and the optics in those directions. These conventions are illustrated by Figure 5 .

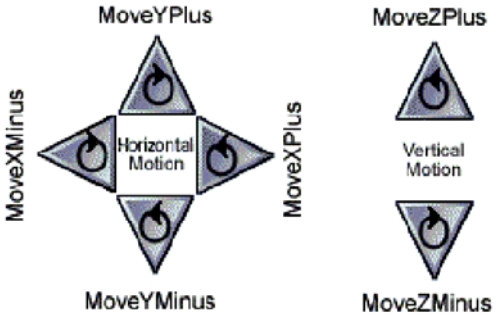


Figure 5. Directions are shown by the arrows and their corresponding VC++ functions. The circular arrows indicate the cw/ccw turning of the axis knob when moving in that direction.

Viewing the system from the perspective of the optics and sample provided an entirely opposite and wrong orientation. For instance, if the box moved left on the axes, the fiber tip over the sample moved right.

Several coding errors were discovered in the original programming based on direction assignments. In the initial calibration instructions, the final segment of code would not allow the user to move the box in the left direction and thus set the sample in the correct initial scanning position. This error was rectified, and the user was given more precise control during calibration. By altering the GUI, the motion code, and demonstration graphics for clarity purposes, the process was simplified. A glitch exhibited through faulty z-axis control on the grinder was located and corrected. In the original code, the x-axis movement was insufficient to smooth the entire sample. This could have lead to errors in the scan data, so the distance of the grinding sequence was lengthened.

5.2 New PCB Source and Detector Driver

The first circuit built to control the light sources and detectors never operated satisfactorily. The design for this printed circuit board (PCB) was pieced together from previous circuits in the lab that were built for the purpose of imaging, but incorporated a feature that was unnecessary for the purpose of the redox scanner—a sample and hold architecture inherited from a Time Resolved Spectroscopy system. The biological tissues that will eventually be imaged by the redox scanner system will be static, both physically because they will be *in vitro* and chemically as a result of the flash freeze process, so there is no time-dependency. This extraneous portion of the old circuit reputedly caused the most problems in the system.

A new version of the circuit was designed with Protel™software and sent to Advanced Circuits to be manufactured. Several items were updated in this version, including the deletion of the sample and hold construction. The circuit layout was also improved to provide for easier troubleshooting. Figures 6 and 7 illuminate the layout and final design of the circuit. After the new PCB was received, some additional work was required to make the circuit operational. The various components of the circuit were soldered to the board, and individual LED driver circuits were constructed and connected to the system. The old sample and hold architecture called for different source and detector timing, so the "RedoxScanner" software was updated to reflect the new sequences (shown in Section 4.2).

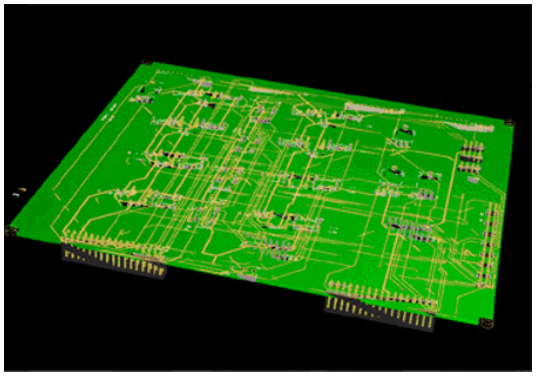


Figure 6. A model of the PCB was constructed via CAD software. The figure reveals how the circuit looked prior to adding electrical components.

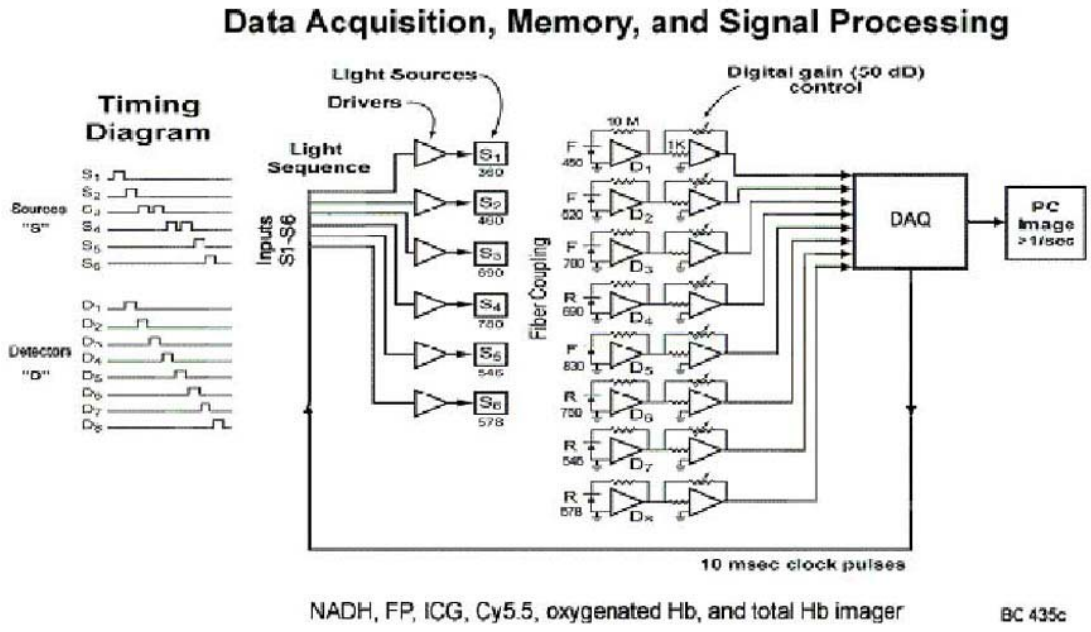


Figure 6. The new PCB schematic was designed with Protel™ prior to its manufacture. In the upper right are the DAQ card connections, below that are the high voltage power supplies for the PMTs, to the center are the source and detector busses; on the bottom center are various other power supplies, and the left shows the amplifiers for the signal outputs.

5.3 High-Signal Attenuation and Noise

5.3.1 Initial trial of redox scanner

After the new PCB was implemented and tested, the redox scanner was equipped to record its first image. To prepare for the trial run, the PMTs and photodiodes were exposed to a controlled amount of white overhead fluorescent lighting, and the voltage output was evaluated with a multimeter. This verified that white light, stronger than any signal that would be provided to the system, output nearly 5V, which is the maximum allowed by the system. Source 1 of wavelength of 360 nm fluoresced on bright white paper, and at 460 nm, S2 fluoresced with yellow highlighter. At the time, S1 and S2 were the only operational sources, so they were tested with a very simple standard—a smiley face drawn on bright white paper with black marker. This standard was affixed to the top of a plastic block held in place by the chuck. The hypothesis was that the output from D1 would be clearer than that from D2, since white is the optimum color for fluorescence of S1 and not S2. Results were achieved by using the default setting of 128x128 pixels/cm^2, with the number of slices set to 1 and the distance from fiber tip to sample under 1 mm (a method to calibrate this height to 80 μm has not yet been determined). The fiber tip was lowered closer to the sample in Trial 1. The testing outcomes are demonstrated in Figure 8.

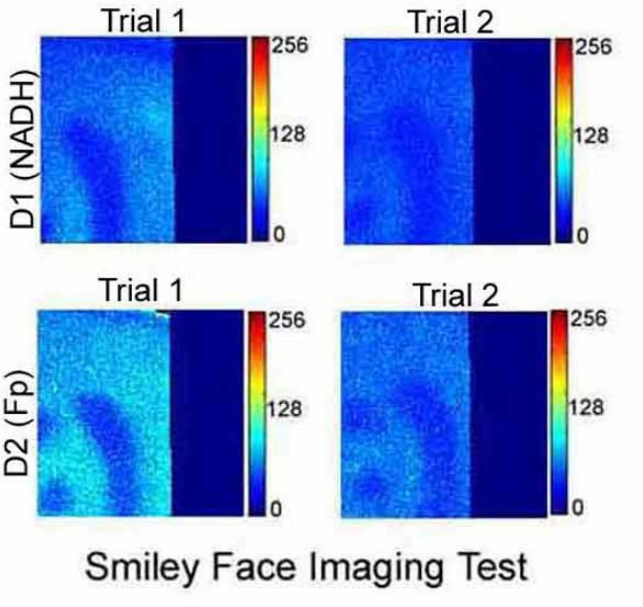


Figure 8. Red corresponds to the maximum output signal from the imager, and dark blue designates zero signal. Based on this scale, these two tests indicated very low signal-to-noise ratio. The rectangular areas signifying no signal on the right of the samples corresponded to times when the scan cycle was intentionally interrupted.

Comparing the images from the two different sources elucidated the capabilities of the circuit up to that point. The images clearly indicated that S2 experienced better signal transmission than S1, which contradicted the hypothesis, but was a reasonable result. This signified that not all source and detector fibers in the optics could be used interchangeably. If the distance between the fibers was too large, little or none of the signal was transmitted

These observations have established the methods for further testing. In future trials, pairs with more similar/better output characteristics should be chosen. The intensity of the images from two different light sources may not be directly comparable without a significant amount of calibration. The initial step in the trial should have been to calculate the offset (the noise measured when the detectors are held in complete darkness) via the VC++ software. This was not accomplished in these tests due to the difficulty of transporting the prototype to a dark room.

Performing this test in the future should eliminate small inconsistencies between the detectors, but still will not ensure that each signal will be amplified equally. To accomplish this, the scanner should be calibrated prior to use with a standard having a known value for each of the six chemicals. By scanning the standard prior to actual use, the intensity of the signals could be adjusted in the software to show an empirical, rather than a relative, voltage output.

The images indicated that other seemingly trivial distances also played significant roles. As the distance from source to sample decreased (starting from under 1mm), the signal transmission increased as long as the source-detector distance in the fiber was small. The intensity of the signals from the first trial was noticeably higher, thus reconfirming that 80 μm was a reasonable depth for good transmission. In addition, the separation between individual detectors meant that while the scanned areas overlap, the images from two chemicals would not show the same 128x128 pixels. Compared to the D1 image, the D2 image was shifted a great deal to the left. This finding lent further weight to the argument that output signals from different chemicals were not immediately comparable.

The core issue was that the signal-to-noise ratio was far too low. Ideally the lights should fluoresce very strongly with a white standard. Even the strongest signal from Fp did not show half of the maximum output voltage, which was unacceptable for accurate imaging.

5.3.2 Light guide signal transmission

Many factors contributed to the low signals. One issue involved the transmission of the light through the source optics, and the subsequent reflectance of the signal back into the detector fibers. The fiber tip coupling was designed to have the symmetry shown in Section 3.2 Figure 2. Note that in this design, there was one source fiber close in distance to at least one detector fiber. All source/detector pairs would have a similar separation distance.

Because of the difficulties of manufacturing such an optics cable, the machine shop was not able to build the coupling to the specifications. In the real fiber coupling, all the detectors were bundled together in the middle and toward one side, so that they nearly touched two of the source fibers. This greatly affected the transmission of the signals, as shown by the following evaluation of the optics.

To test which source and detector fibers should be combined to maximize output signals, a green LED (578 nm) signal and a specific photodiode were employed to evaluate each individual combination. The six source and eight detector fibers were labeled alphabetically for ease of identification, and a mirror was used to simulate absolute reflectance conditions of the sample. The tests were conducted with a fiber tip/sample separation of under 1 mm. With the "RedoxScanner" software running, the DAQ card supplied a pulse train throughout the system. The voltage output from the photodiode detector was measured with an oscilloscope set for data averaging of 16 samples. The results are shown in Figure 9 below.

		Source Fibers (Red)					
Detector Fibers (Orange)		**A**	**B**	**C**	**D**	**E**	**F**
	A	2.16	1.44	0.00	1.28	0.00	2.80
	B	0.00	0.00	0.00	6.80	0.00	5.84
	C	0.00	0.00	0.00	1.76	1.20	3.52
	D	2.80	1.72	0.00	1.68	0.00	6.80
	E	2.96	5.52	0.00	0.00	0.00	1.52
in mVpp	**F**	5.96	2.16	0.00	0.00	0.00	3.04
	G	0.00	2.24	1.20	0.00	1.20	1.44
	H	0.00	0.00	0.00	1.68	0.00	8.80

Figure 9. The highlighted output signals were the only ones completely distinguishable from the system noise. Source fibers F and A transmitted light such that multiple detectors could identify the signals, but Sources C and E did not have any detectors capable of sensing any signal. This signified that the fiber coupling geometry was not usable for the purposes of the redox scanner, since all fibers were necessary for experimentation.

It was important to note while the signals were very small, this was directly related to fiber tip/sample separation. An output voltage difference of up to two orders of magnitude larger was detected by adjusting this separation, so the results of this test are relative, not empirical. The relationships between the source and detector transmission remained constant at all separations however, so this test was a valid indication of coupling quality.

The yellow highlighted values specified the voltage pulses that were definitively distinguishable from the system noise. Two of the source fibers, C and D, do not have any detectors that transmitted the signal. This was a major problem, and indicated that with the current fibers two of the signals cannot be transmitted. On the other hand, fibers A and F conveyed signals well through multiple fibers. Making an informed estimate of the geometry of the system, it was concluded that source fibers A and F were the closest to the detector bundle, source fibers B and D were touching A and F to either side, and source fibers C and E were the farthest from the detectors.

Recoupling the fibers would be an expensive and time-consuming process, so the faulty optics configuration was used for the remainder of the SUNFEST program. The source and detector pairs were chosen through results from the chart to enhance transmission as best as possible with the extant geometry.

In addition to the combined source/fiber coupling problem, the coupling on the other end between individual fibers and the detectors was flawed. For the optimal configuration for signal transmission, the fibers were screwed into a coupling in the optoelectronics box. The fiber tips would then be secured flush against and perpendicular to the detectors. In practice, an acceptable signal could be transmitted only when the fibers were adjusted in the couplings by hand during the scanning process. Inspection of the couplings revealed that they were too long, causing a gap between the fibers and detectors. Cutting down the threads on the coupling removed this distance and greatly enhanced the signal.

5.3.3 OPT101 photodiode and LED efficiency

The efficiency of the OPT101 detectors was evaluated with the goal of enhancing the signal through other means. In the smiley face test, it was noted that the output signals from the green LED (578 nm) were extremely small. The output voltage of a yellow LED (546 nm) was assessed to find the signal was only slightly larger than that from the green LED.

The OPT101 specifications were reread to reveal that they were not optimal for detecting signals from the green and yellow LEDs. An ideal 850 nm infrared light source was expected to have an output of 0.60V/μW. The output decreased linearly with decreasing wavelength, such that the yellow LEDs could be expected to have a signal of 0.375 V/μW and the green LEDs 0.30V/μW, half of the optimal output voltage. After briefly evaluating the option of choosing different detectors that focused on the yellow and green wavelengths, the idea was discarded. The new PCB design was created

specifically to use the OPT101s, and the alternate pinout arrangements and voltage requirements of different detectors would have made it very complicated to integrate them into the chip.

It was ascertained that the best way to increase the output signal was to maximize the power output of the LEDs. Since the signals would be pulsed and not continuously powered, a new driver circuit using a BJT and various resistors was designed to run the LEDs at the high end of their power specifications. This circuit increased the LED supply current to 45mV (where $I_{peak\_max} = 50mV$) and the voltage to 3.0 (where $V_{green\_max} = 4.0$, $V_{yellow\_max} = 3.0$).

Several other measures were taken to increase the signal-to-noise ratio. To augment the voltage signal after the detectors, the feedback resistor on the amplifier was replaced with one of much higher value, 22MΩ. This linearly increased the amplification, but only increased noise by the square root of 22M, giving a signal-to-noise ratio of 5:1. During the testing process the photodiode output signals were discovered to have a high-frequency noise. To eliminate this, RC low-pass filters using 2.2μF capacitors and 1.1MΩ resistors were added to the circuit after the output voltage amplification.

The end result of all the troubleshooting was that the output signal was much improved, but still did not have the clarity and magnitude desired for accurate scanning.

5.4 Matlab Image Processing

The only imaging software that existed for the scanner was the status monitor on the "RedoxScanner" GUI. The "Redox" imaging program was constructed with Matlab v.6.1™ to reliably image and save in picture form the output from VC++. A screenshot detailing the "Redox" GUI appears in Figure 10.

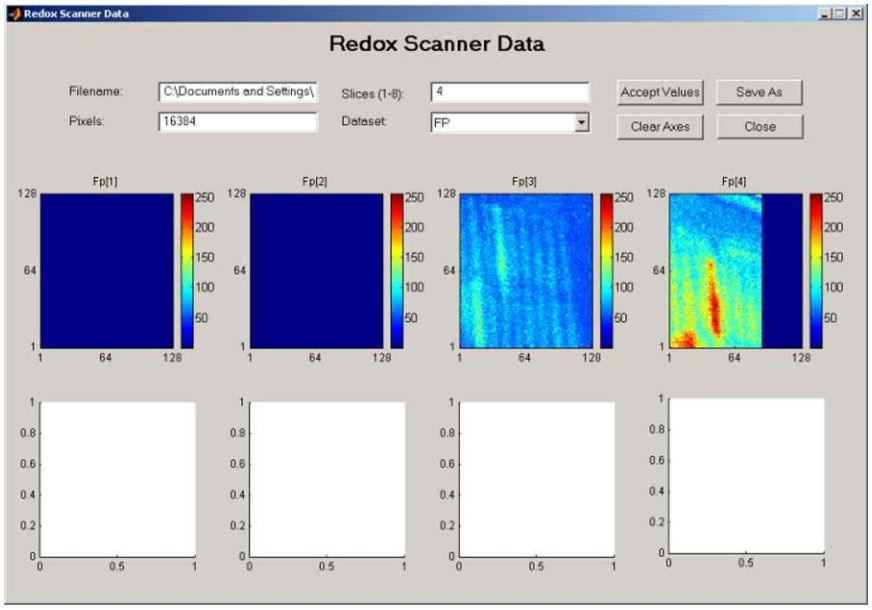


Figure 10. The Matlab™ "Redox" imaging software was designed to be as user-friendly as possible, and to show as many or as few slices of a sample that a user specifies. In addition, it was intended to resemble the VC++ GUI for simplicity.

The layout of the Matlab™ GUI was designed to look similar to the VC++ GUI; however, there were several notable differences. As with the VC++ software, the end user was presented with pictures of up to eight slices at one time, but all these slices were of the same chemical, rather than being specific slices of all chemicals. Since the amount of chemical present in tissue changes with depth, this layout was determined to give clearer information about tissue health. Through the GUI the user could open the *.dat output file for any scan sequence and set the number of sample pixels, the number of slices, and the chemical under observation.

The data were then imaged in the following manner. The number of pixels entered by the user was interpreted as a square sample size. The *.dat file information consisted of voltage values between –1000 and 4096, which corresponded to a signal output of –1 to +5V multiplied by 1000 by the DAQ card and coded in binary. Each pixel voltage value was imported into a matrix and rescaled to have a color value between 0 and 256. These data were printed out to the screen with a colorbar to show the color to intensity scale.

The user could then choose to view another chemical from the same scanning sequence, or select data from another scan sequence and reset the number of pixels and slices accordingly. The images are saved separately to a *.tiff file in a user-specified path.

5.5 Windows Timing Delay Causes Motion Lag

The original hardware configuration was a compromise between two system configurations. The Micromill 2000 was designed only to run under non-Windows DOS. The DAQ card was supported only under Windows™ platforms. Non-Windows DOS cannot run programs simultaneously with Windows on one computer, so the VC++ code had been created to bridge the two and run the mill under Windows-based DOS. Thus in this hardware arrangement, a Windows 2000™ laptop ran the optoelectronics through the DAQ card and simultaneously supplied pulses to the Micromill driver through the serial port. This configuration caused the most critical problem in the system, and was the most difficult to debug.

5.5.1 Initial hardware and software debugging

During the initial calibration of the Micromill prior to scanning, the motion along the x-axis was inconsistently choppy. In the original configuration, all the Micromill motion was controlled directly by the “RedoxScanner” software. Believing the lag to be a software issue, the calibration code was examined. The function FastMoveX() was responsible for the specific movement in question, but the other places where it was used did not seem to replicate the choppy motion. The DAQ card specifications verified that it was being utilized to the bottommost level of its capabilities, eliminating the possibility of a buffer bottleneck.

The next course of action was to investigate a hardware issue. When the machine was first implemented, the mill experienced similar lagging problems. These were

resolved by running the mill at a steady rate for approximately 36 hours to prime the machine. The mill had not been operated for several months prior to the new problem cropping up, so repriming the machine seemed a feasible solution. The machine was lubricated and simple VC++ test code, "MMBurnIn," was then written to ramp the mill to a high speed and then decelerate back and forth across the x-axis 500 times. Observation of the final sequences verified that the x-axis movement had not improved.

The proprietary "MPS2000" DOS Micromill software was then evaluated to determine whether the problem stemmed from the VC++ or a hardware malfunction. Simple G-code programs were coded to provide x-axis movement in the DOS software. At that time DOS could be run only from a Windows shell. When asked to move slowly the program behaved correctly, but when supplied a very fast ramp the motion was again choppy and the gears ground. To confirm that the problem was necessarily with the x-direction, the signal outputs to the x- and y-axis motors were switched from the power supply, and consequently the y-axis behaved erratically. This indicated that the problem was not in the VC++ code or in the Micromill 2000 hardware.

A series of tests was then conducted on the control signals from various hardware sources. The Micromill driver box that drove the motors was dismantled and probed with a multimeter, which did not provide any useful information. The graycode outputs from the DAQ card were tested with an oscilloscope. The DAQ was assessed using the "MMTest" software, causing the graycode to step by mouse click event, and the signals were found to be very clean 5V ramp functions. In hindsight, the frequency of the "RedoxScanner" graycode should then have been evaluated, which would have immediately explained the problem. Instead, this line of evaluation was suspended.

5.5.2 Grid imaging test

An experiment was conceived to find both the accuracy of the x- and y-movement and the affects of the lag on the sampling. The analysis methods used to capture the previously mentioned smiley faces were replicated here, with the exception of a different standard. Again, S1 and S2 were the test sources. A 1 cm^2 grid was drawn on bright white paper with a black fine-tipped pen. The grid consisted of vertical lines drawn every 2 mm, and horizontal lines drawn at the top, 5 mm down, and at the bottom of the grid. Two of the resulting rectangles were colored in with fluorescent yellow highlighter to test S2 fluorescence. The results of three different scans are shown in Figure 11.

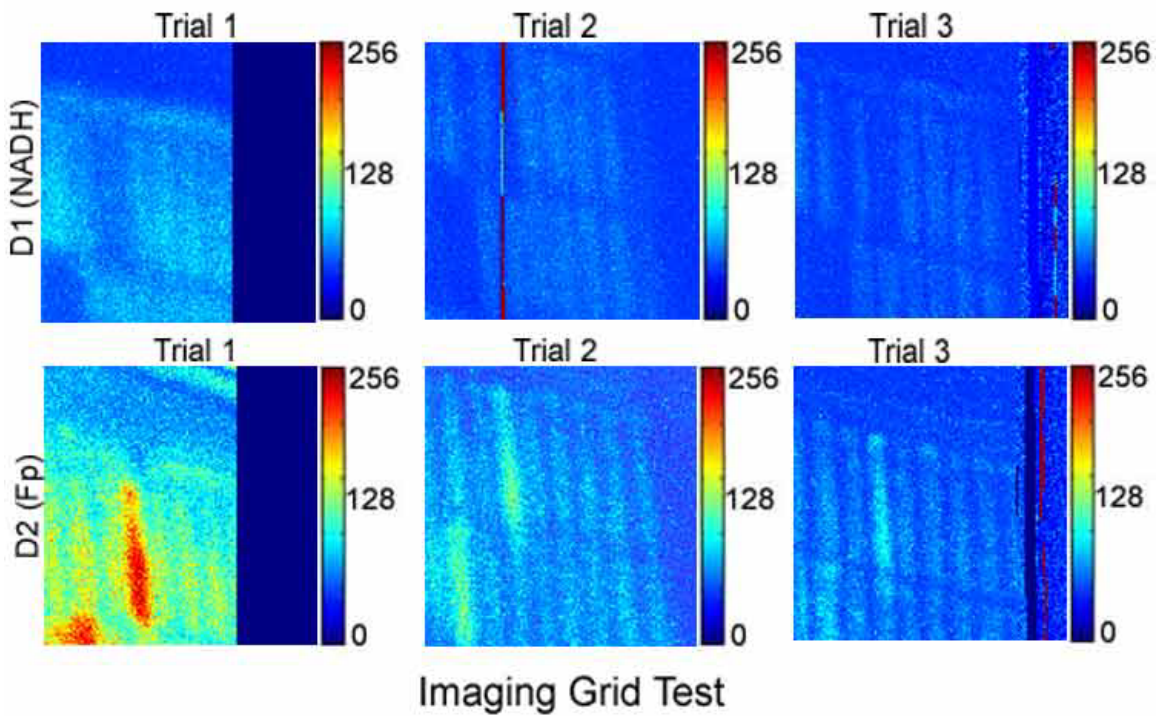


Figure 11. When imaged in the "Redox" software, the data denoted that the scanner had interpreted the grid as slanted lines. This proved that data had been lost somewhere in the scanning process, and and gave credibility to the theory that the delay was a Microsoft Windows problem.

These test results echoed those of the previous test in several ways. Source 2 revealed much better signal transmission than S1, as replicated in the smiley face test, and the dependency of signal to fiber height was firmly established. But the smiley face test utterly failed to convey one major fact.

In Figure 11 the images show straight vertical lines, signifying that the standard was placed straight on the sample bed. However, all the pictures reveal the center line to be slanted diagonally up and to the right, rather than at 90° to all the intersecting vertical lines as it was drawn. This demonstrates that not only are data lost in the process, but that it must occur during the y-axis movement as well, because no data are acquired during x-axis movement.

From all the evidence, troubleshooting, and process of elimination, it was ascertained that Windows™ timing itself caused the lag. The clock pulses provided by the OS did not have consistent timing frequency (which we could discovered earlier by testing the graycode). A consistent frequency is required only for acceleration and deceleration, which explains why the fast movement was affected while normal motion was not. It was now apparent that the Micromill could not be run directly from the laptop, so the system necessitated re-engineering.

5.5.3 Implementation of DOS controller

As recommended by the Micromill manufacturer, an alternative design was established that employed a DOS computer to function as a controller for the motors, isolating Windows™ from Micromill timing. DOS, unlike Windows™, does not multi-task, and thus could time pulses very accurately. The laptop was connected to the DOS box via serial cable, and the DOS computer was linked to the Micromill motor driver through its parallel port.

The VC++ software was updated, and new software was developed to interface the laptop/DOS and DOS/driver systems. In the new VC++ serial port protocol, the laptop passed five bytes to the DOS box every time a new movement was requested. The first byte could be one of two binary numbers. A value of 64 designated the most significant bit (MSB) of a new codeword, and 128 specified a reset. When the "RedoxScanner" program was closed, it output a 0 byte to the system. The DOS box read the 0 and output a 128 to clear previous values. The second byte signified the direction of the movement. The directions {X+, X–; Y+, Y–; Z+, Z–} were coded by {1, 8; 2, 16; 4, 32}. The third, fourth, and fifth byte determined the distance traveled in the direction requested. For each byte, the first six least significant bits were employed for distance, for a maximum of 262,143 steps per codeword. The two MSBs were purposely left blank, so they could not be misconstrued as 64 or 128. The DOS box sent a signal confirmation back to the laptop while the codeword was processing. It then provided the Micromill motor driver with all the timing and signal pulses required for motion along the axes.

This system was newly implemented, but initial testing confirmed that it completely solved the fast motion lag. The laptop could now devote more processing power to data acquisition and seemed to read the grid standard accurately, although no data are currently available to demonstrate this.

6. RECOMMENDATIONS

There are still several major topics to be addressed. The picture quality must be increased, and the optics coupling must be redesigned. Recoupling may increase the output voltage significantly and solve the signal attenuation problem, thus increasing image clarity. During testing, an interrelationship was noted between the source-to-detector distance, the fiber to sample distance, and there may have been a dependence on wavelength as well. This should be calculated ideally as well as empirically to determine the exact geometry of a new fiber coupling to optimize light transmission.

Better image quality can be attained through additional means. To increase the signal clarity, the VC++ software can be set to average more samples per pixel. The DAQ card uses only a small percentage of its processing capability, so the averaging could be increased from 8 samples per pixel to 50 or 500 if there is no resulting lag on the laptop. Also, using smaller-diameter detector fibers in the redesigned fiber coupling will provide increased resolution when the software is adjusted for a smaller step size. However, a

smaller fiber will be even less likely to transmit light, so the transmission characteristics should be calculated carefully to decide if this is feasible before implementation.

Several minor glitches in the VC++ software must be fixed. Currently, the software divides the number of slices by the assumed sample depth (1 cm) to calculate the depth of the sample grind. For instance, in a scan with eight slices, the depth of each grind will be 1/8 cm. The grind depth should be set at a constant value of 80 μm. Also, when the user enters a scan size other than 16384 pixels, the computer does not interpret this as a square image. Instead, it scans down 128 pixels for as many rows or partial rows as needed for the new scan size. This is not necessarily a problem, but is inconsistent with the Matlab™ imaging software, and one should be altered to match the other. In addition, the code was designed with the idea that the voltage output would vary from 0 to +5 V. The DAQ card ideally is set to unipolar, which scales these data in binary to a value between 0 and 4096. In practice the outputs from the DAQ are only multiplied by 1000, and can range from –1000 to approximately 4200. This indicates that the data acquisition is actually set to bipolar, which should be easily fixed.

A method must be developed to calibrate the light sources as well as the source-to-sample separation. If the signals are properly calibrated, a composite redox or hemoglobin ratio image can be created in Matlab™ by first comparing the images, and then creating an algorithm to shift the pixels before averaging.

7. CONCLUSIONS

The redox scanner still requires further work, but the all of the current problems seem solvable, whereas at the beginning of SUNFEST the list of known bugs was overwhelming. The implementation of the DOS computer solved the most critical and demoralizing problem in the system, so it is estimated that future work will be much more easily accomplished. Current test data are very encouraging and may herald the end of the construction phase of the redox scanner. The next phase will involve testing on an animal model to find the efficacy of the sensors on actual biological systems.

8. ACKNOWLEDGMENTS

I would like to thank Dr. Jan Van der Spiegel for his excellent support and chairing of the SUNFEST program, and Ms. Lois Clearfield for her wonderful administration and patience. I extend my sincere appreciation to everyone at the lab for their encouragement and help, especially Lanlan Zhou for introducing me to the redox scanner, Emily Hwang for performing much of the initial work, and Samantha Gaw for her constant assistance, supervision, and dedication to the project. I want to congratulate Dr. Britton Chance, as he is the inspiration for the entire lab and will outlive us all. Finally, I would like to thank the National Science Foundation for supplying the REU grant; without their support this SUNFEST project would not be possible.

9. REFERENCES

1. B. Tromberg, Development of diffuse optical spectroscopy for quantitative characterization of thick tissues, BCMCXC Symposium, Philadelphia, PA, USA, July 2003.

2. B. Chance, E. Anday, S. Nioka, S. Zhou, L. Hong, K. Worden, C. Li, T. Murray, Y. Ovetsky, D. Pidikiti, and R. Thomas, A novel method for fast imaging of brain function, non-invasively, with light, *Op. Ex.*, 2 (1998) 411-423.

3. N. Ramanujam, J. Chen, K. Gossage, R. Richards-Kortum, and B. Chance, Fast and non-invasive fluorescence imaging of biological tissues in vivo using a flying-spot scanner, *IEEE T-BME*, 48 (2001) 1034-1041.

4. B. Chance and H. Baltscheffsky, Binding of intramitochondrial reduced pyridine nucleotide. *J. Biol. Chem*., 233 (1958) 736-739.

5. R. Weissleder, C. Tung, U. Mahmood, Bogdanov, In vitro imaging of tumors with protease-activated near-infrared fluorescent probes*, Nature Biotech*., 17 (1999) 375-378.

6. S. Nioka, *NIH grant proposal for an updated redox scanner*, 2001.

7. B. Chance, H. Schleyer, and B. Schoener, Low temperature excitation and emission spectra of NADH in mitrochondria and tissues, *Biophy. Soc. Abstr.,* TE-8(1963).

Archeoviz: Improving the Camera Calibration Process

NSF Summer Undergraduate Fellowship in Sensor Technologies
Jonathan Goulet (Computer Science and Engineering) – University of Pennsylvania
Advisor: Dr. Kostas Daniilidis

ABSTRACT

The reconstruction of archaeological scenes as three-dimensional graphical models can provide a very powerful tool for the archaeologists studying them. The purpose of the Archeoviz project is to develop a program that archaeologists can use in the field to create these three-dimensional models themselves, accurately and easily, using two-dimensional photos of the scenes. This paper describes my development of a program to make the process of camera calibration, necessary before any model building, as fast and easy as possible for the user. This calibration process requires locating and recording image coordinates of numerous small markers in the scenes. Originally, this locating had been quite time-consuming and had to be done separately from any calibration calculations. I have developed an independent C++ program that requires users to locate only six of these markers themselves. The program will then automatically locate the remaining markers for the user and calculate the calibration data. This program includes a user-friendly zooming interface and a file browsing system for retrieving and loading the images and for saving the calibration data obtained. In addition, an extension for the program will allow users to estimate the location of the markers in the second image of the stereoscopic pair from the calibration data for the first. Also, in order to obtain more accurate three-dimensional models, the reconstruction will incorporate stereo pairs from multiple views of a scene. Through another extension, the calibration data from the first of these pairs can be used to predict the location of all markers in all subsequent stereo pairs. Through these various improvements, my program greatly speeds up this calibration process and allows archaeologists to quickly move on to constructing these valuable models.

Table of Contents

1. INTRODUCTION

The study of archaeological sites can give us both a link to the world of the past and a better understanding of that of our present. The reconstruction of these sites as three-dimensional graphical models can bring them directly to anyone with access to a computer.

The goal of the Archeoviz project is to provide full three-dimensional reconstructions of the pre-Columbian site of Tiwanaku, located in Bolivia, for the Museum of the University of Pennsylvania. The project also works to develop a program for archaeologists to use in the field that will allow them to create these models themselves, quickly and accurately, from two-dimensional photographs of the scenes they are studying. This program is intended to be as user-friendly and as easy to use as possible so that any archaeologist working in the field can, without any prior experience, create these models. These models would then provide them with the ability to study the sites and to collect valuable data from the scenes that they would not have been able to gain from the original two-dimensional photographs. In addition, the archaeologists would be able to create these models in just a few minutes, as opposed to the days or even weeks it would take them to create the accurate topographical maps that they now use for similar data taking purposes.

The first step in this reconstruction is the process known as camera calibration. This report describes my development of an independent program that will make this calibration as easy and as fast as possible for the archaeologists so that they can quickly proceed to creating these valuable models. The methods used to improve the calibration process are discussed in the main body of the report and a detailed description of the software implementation is provided in Appendix A.

2. MATHEMATICAL COMPONENTS

The two major mathematical components of the reconstruction process are camera calibration and stereoscopic reconstruction. My portion of the project focused mostly on the former, but I will give a short explanation of the latter as well.

2.1 Camera Calibration

In order to construct a three-dimensional model using only two-dimensional photographs, it is necessary to find a transformation from the two-dimensional image coordinates of points and features in the picture to some three-dimensional coordinate system in which these features can be represented. This process, known as camera calibration, is done in two steps, transforming from the two-dimensional image coordinates to the three-dimensional coordinates relative to the frame of the camera and from those to the three-dimensional coordinates relative to some fixed world system. The parameters describing the first transformation are known as the intrinsic parameters of the camera and those describing the second are called the extrinsic parameters. Figure 1 shows the three coordinate systems of interest in our calibration problem.

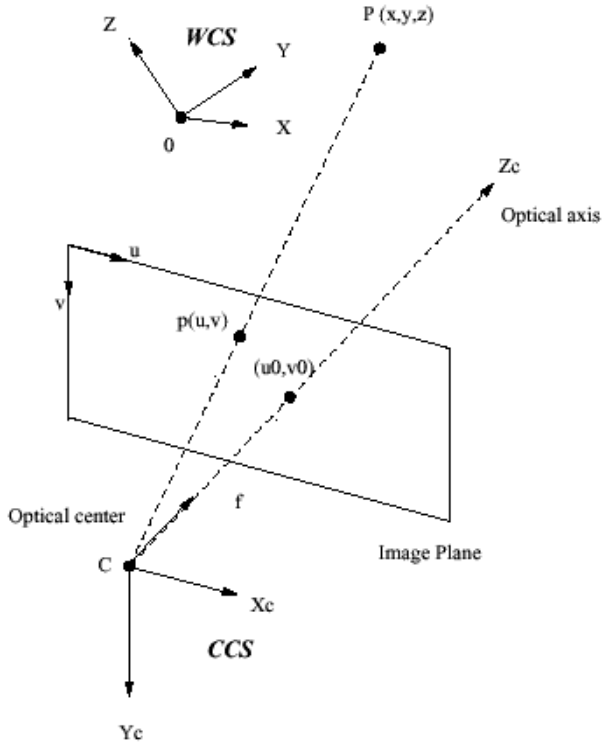


Figure 1: The world, camera, and image coordinate systems.
Figure taken from www.csd.uch.gr/~cgarcia/JANUS/pinhole.gif [1]

Four intrinsic parameters must be recovered in the calibration of the cameras. The first two describe the distance from the origin of the camera coordinate system, the optical center of the camera, to the image plane. These two parameters, denoted by s_u and s_v, also take into account the scale factors, pixels per unit length, in both the u and v directions in the image plane. The second pair of parameters describes the point in the image plane where the optical axis, the z-axis in the camera's coordinate system, intersects the image plane. These are denoted by u_0 and v_0 and are measured relative to the origin of the image coordinate system, in the upper left corner of the image plane. If these parameters are placed in a matrix, the transformation from image coordinates to camera coordinates can be described with the following equation, where the vector of homogeneous image coordinates is denoted by **(u, v, w)** and the vector of homogeneous camera coordinates by **(x_c, y_c, z_c, w_c)** and where ~ denotes equality up to a scale factor:

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} \sim \begin{pmatrix} su & 0 & u0 & 0 \\ 0 & sv & v0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} Xc \\ Yc \\ Zc \\ Wc \end{pmatrix}.$$

This equation gives the location in the image plane onto which a point with coordinates

(x_c, y_c, z_c, w_c) relative to the camera's coordinate system will be projected.

The extrinsic parameters of the camera are a rotation and a translation that relate the three-dimensional coordinate system centered at the camera to some fixed three-dimensional world coordinate system. Let the 3x3 matrix of rotation be denoted by **R** and the 3x1 matrix of translation by **t**. The transformation from camera to world coordinates can then be described by the following equation, where the camera coordinates of a point are denoted by **(x_c, y_c, z_c)** and the world coordinates of the point by **(x_w, y_w, z_w)**:

$$\begin{pmatrix} Xc \\ Yc \\ Zc \end{pmatrix} = \mathbf{R} \begin{pmatrix} Xw \\ Yw \\ Zw \end{pmatrix} + \mathbf{t}.$$

This transformation describes the representation, in the camera coordinate system, of the point with coordinates **(x_w, y_w, z_w)** in the world coordinate system. Equivalently, using homogeneous coordinates to represent a point in the two coordinate systems, the equation becomes:

$$\begin{pmatrix} Xc \\ Yc \\ Zc \\ Wc \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} Xw \\ Yw \\ Zw \\ Ww \end{pmatrix}.$$

By combining the above two transformations, we can obtain a transformation from the homogeneous world coordinates of a point to the homogeneous image coordinates of the projection of that point onto the image:

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} \sim \begin{pmatrix} su & 0 & u0 \\ 0 & sv & v0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} Xw \\ Yw \\ Zw \\ Ww \end{pmatrix} \rightarrow \begin{pmatrix} u \\ v \\ w \end{pmatrix} \sim \boldsymbol{M} \begin{pmatrix} Xw \\ Yw \\ Zw \\ Ww \end{pmatrix}.$$

In this equation, ***M*** is a 3x4 matrix that will be referred to as the projection matrix. My program's method for camera calibration recovers this matrix and extracts from it all of the intrinsic and extrinsic parameters. This matrix will be necessary for the stereoscopic reconstruction step of the process.

2.2 Stereoscopic Reconstruction

The method used in the Archeoviz project to create the three-dimensional models is known as stereoscopic reconstruction. This process uses multiple, slightly different, two-dimensional images of the same scene, for which camera calibration has already

been completed, and exploits these differences to create a three-dimensional model of the scene. This is the method that the human visual system employs in using the two slightly different pictures on the retinas to create a three-dimensional view of the world. The basic manner in which this process operates uses the projection matrix obtained in the camera calibration step. With this matrix, a ray in the three-dimensional world coordinate system can be associated with each point in the image plane. By intersecting the rays obtained from various calibrated images of the same scene, we can obtain the three-dimensional location of image points and features, and a three-dimensional model can be constructed.

3. THE ORIGINAL ARCHEOVIZ IMPLEMENTATION

In the Archeoviz project, the calibration of the cameras is accomplished through the use of markers that are placed in each of the scenes to be reconstructed at locations with known coordinates relative to the fixed world system. The world coordinates of these locations are measured using the Theodolite laser measurement system. This system uses a laser to accurately judge the distance to any point and, using that data, gives the *x, y,* and *z* coordinates of the point, corresponding to the distance east and north and the elevation from a set origin, respectively. The input to the calibration process is a text file containing these *x, y,* and *z* coordinates, along with the image to be calibrated. Calibrating requires that the markers in the image be located and their two-dimensional image coordinates recorded. Then, from the correspondences between the coordinates, we solve a system of linear equations, using linear least-squares techniques, to recover the projection matrix.

A program to accomplish the reconstruction of the scenes was created in the summer of 2002 as part of the Archeoviz project. However, the calibration process in this original implementation was quite time-consuming and difficult. In this original implementation, the user had to locate each of the 20 or 25 markers in the picture one by one, using a program such as Adobe Photoshop, and input their image coordinates into a text file manually, in a similar format to the text file containing the world coordinates of the points. The user had to make sure that these points were entered in a very specific order, corresponding to the order in which the markers were represented in the text file containing the world coordinates. This order was determined by the numbering assigned to the markers in hand-drawn diagrams that the archaeologists had created of the scenes they wished to be reconstructed. Since these markers are often quite small and are sometimes quite difficult to find, recording the image coordinates could take quite a long time. Once the user had found all of the markers in a particular scene, the text files containing the coordinates had to be taken to a program that would compute the projection matrix. Finally, the user would take a text file containing this projection matrix, along with the other text files created, to the reconstruction program, which would create the three-dimensional models. My major goal during my time on the Archeoviz project was to create a program to make this calibration much faster and easier for the user.

4. METHODS OF IMPROVING THE CALIBRATION PROCESS

4.1 The Six Points Algorithm

In the original implementation of the program, the most time-consuming part of the calibration procedure was locating the markers. Thus, my major focus was on reducing the number of markers that users had to find themselves, and on enabling the program to automatically find as many of these markers as possible. The projection matrix that needs to be recovered has 12 entries, and each correspondence between world and image coordinates provides two equations on these 12 elements. Consequently, six is the minimum number of correspondences that need to be known, and thus the minimum number of markers that need to be located, in order to recover a projection matrix for the scene. Therefore, my first major step in improving the calibration process was to implement an algorithm that requires users to find just six of these markers, then calculates an estimate of the projection matrix and, using that estimate, predicts the image coordinates of the remaining markers from their known world coordinates. This method reduces the number of markers that users must discover themselves from 20-25 in each picture to six.

My first step was to prototype this algorithm in Matlab, and to test how well the estimates of the markers worked with different combinations of the six markers first selected. In the majority of cases, the estimates for the remaining points were quite accurate if the six markers selected were relatively sparsely spread throughout the picture. However, certain degenerate point configurations caused the method of prediction to become quite inaccurate. This occurred mostly when the six points selected lay in a formation that was close to being planar in the three-dimensional world coordinate system. If these degenerate configurations were avoided, however, my approach would predict the majority of the remaining markers within about 30 pixels in both the u and v directions in the image coordinate system. Thus, I chose this value as the radius of the circle that I would implement my program to draw around the predicted locations of the markers to help the user find them. This value of 30 pixels is quite small in relation to the size of the entire image, and so these predictions are quite close, relatively, to the actual location of the markers.

As the above results illustrate, this new method provides an accurate way of predicting the location of the remaining markers in the scene from the known location of any six markers that do not lie in a planar formation. This will allow the program to automatically discover the location of all but the first six markers for users. Consequently, this method provides a great improvement over the original calibration implementation by significantly shortening the time needed to complete the calibration through reducing the number of markers that users must locate themselves.

4.2 Predicting the Right Projection Matrix from the Left

As mentioned earlier, the method used in the reconstruction program to produce the three-dimensional models is stereoscopic reconstruction. Thus, for each scene to be

reconstructed, it is necessary to complete calibration and to recover the projection matrix for each of two images, called the “left” and “right” images, corresponding to the relative locations of the cameras at the time the photos are taken. In my next improvement to the calibration process, I have implemented a new method that will use the projection matrix for the left image, obtained using the method described above, along with the information relating the relative location of the cameras, to estimate the projection matrix for the right image and thus predict the location of all markers in this second image. This will further speed up the calibration process, as it allows users to calibrate both images at once, as well as eliminating their need to find any markers in the right image by themselves. Figure 2 shows a sample pair of stereo images from one of the archaeological sites.

Figure 2: A stereo pair of images, the left above and the right below, illustrating the slight differences between the two views.

In this estimation, I have taken advantage of the information provided by the archaeologists who had taken the photos of the scenes in Tiwanaku regarding the setup of the cameras for the scenes. For all of the scenes to be reconstructed from Tiwanaku, the left and right cameras were set up on a common tripod, on opposite ends of a horizontal bar that sat atop the tripod. They were separated by a short distance, generally about one half meter, and they were moved to rotate only slightly to get a better view of the scene. This setup is shown in Figure 3, where the orientations of the camera coordinate systems relative to the physical cameras are apparent. In addition, this diagram illustrates that the only movements for the cameras correspond to a rotation about the y-axis of the camera coordinate system. From this diagram it is also apparent that the transformation between left and right camera systems can be modeled as a rigid transformation, with a translation along the x-axis of the camera system and a rotation about the y-axis of the same system.

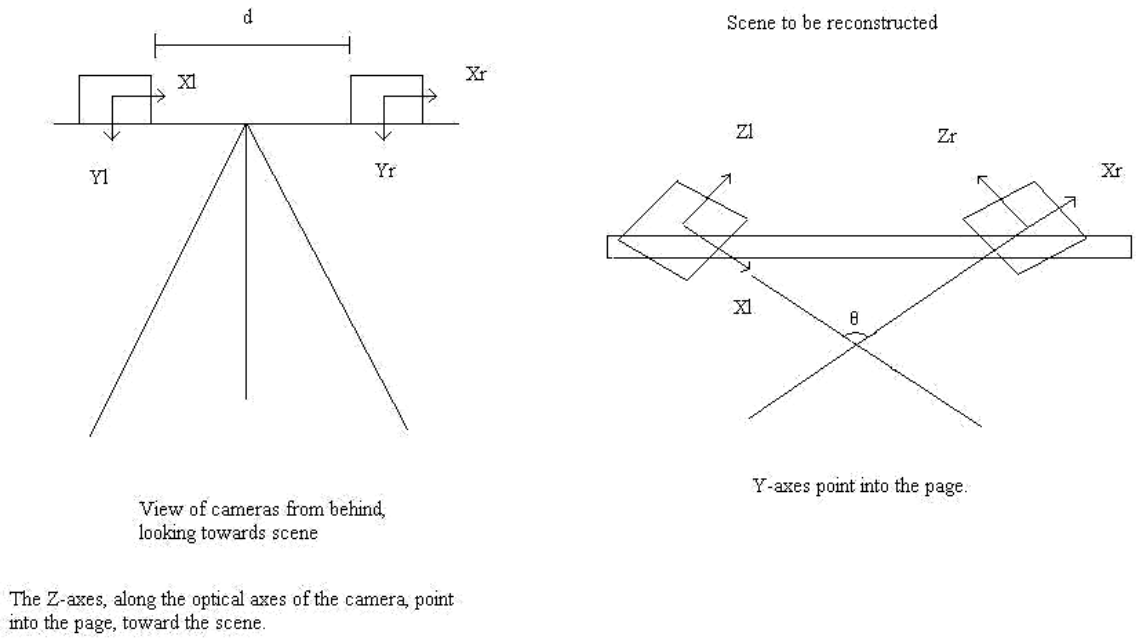


Figure 3: Geometric representation of camera coordinate systems, showing the relationship between left and right cameras.

Thus, two parameters will fully describe this transformation: the translation distance, d, and the angle of rotation about the y-axis of the camera system, θ. Consequently, this transformation can be modeled, mathematically, by the following equation, in which the coordinate vector of a point relative to the left camera frame is represented by $\mathbf{X_l}$ and that of a point relative to the right frame by $\mathbf{X_r}$**:**

$$\mathbf{X_l} = {}^{l}\mathbf{R_r} * \mathbf{X_r} + {}^{l}\mathbf{t_r}$$

where:

$${}^{l}\mathbf{R_r} = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \text{ and } {}^{l}\mathbf{t_r} = \begin{pmatrix} d \\ 0 \\ 0 \end{pmatrix}.$$

By adjusting these parameters appropriately, we can find the correct transformation from left camera system to right camera system. Combining this with the transformation from left camera system to world system, which is acquired by extracting the extrinsic parameters from the projection matrix of the left image, we can obtain an estimate of the transformation from world system to right system. If $\mathbf{X_w}$ represents the coordinate vector of a point in world coordinates, this transformation relating world and left camera systems can be modeled, as before:

$$\mathbf{X_l} = {}^{l}\mathbf{R_w} * \mathbf{X_w} + {}^{l}\mathbf{t_w}.$$

Using this and the prior relation, we can obtain the following transformation relating the world system and right camera system:

$$\mathbf{X_r} = {}^{l}\mathbf{R_r}^{-1} * {}^{l}\mathbf{R_w} * \mathbf{X_w} + {}^{l}\mathbf{R_r}^{-1} * ({}^{l}\mathbf{t_w} - {}^{l}\mathbf{t_r}).$$

Thus, using this formula, we can acquire an estimate of the extrinsic parameters for the right image. In addition, we assume that the intrinsic parameters of both cameras are the same. This assumption is based on data taken from many of the images from the Tiwanaku site, as well as the knowledge provided by the archaeologist that the zoom, and thus the focal length, of the cameras are kept relatively constant throughout the process of photographing. Therefore, using the estimated right extrinsic parameters, and setting the right intrinsic parameters to be the same as those obtained from the left projection matrix, we can estimate the right projection matrix and use it to predict the location of all of the markers in the right image. This will provide another method of speeding up the calibration, as users will not be required to locate any of the markers in the right image themselves.

4.3 Predictions Using the Location of the Tripod

In order to get more accurate three-dimensional models of the scenes, and so that these models can be viewed from many different viewpoints, the archaeologists have taken stereo pairs of images from many points of view for a given scene. Therefore, to capture a full 360º view of a scene, multiple stereo pairs of images must be calibrated. In my next improvement to the calibration process, I have implemented a method of predicting the projection matrices for all stereo pairs of images of a given scene using the calibration information obtained from just the first stereo pair. This is achieved through the inclusion of a few more pieces of data to be taken by the archaeologists in the field. In addition to recording the location of the markers in the fixed world coordinate system, the archaeologists in the field will have to measure the world coordinates of three fixed points on the tripod for each camera location that they use for the scene. Using these tripod measurements for each point of view, we can construct a transformation between the tripod positions. This transformation, along with the projection matrices obtained through the above method for the first stereo pair, can then be used to obtain an estimate for the projection matrices for all subsequent stereo pairs of images of the scene.

Using the knowledge of the world location of at least three points on the same rigid body, in this case the tripod housing the cameras, we can construct a rotation and a translation that relate the two positions of the rigid body. If $\mathbf{X_i}^{1}$ represents the coordinate vector of one of the measured points on the tripod at the first location, and $\mathbf{X_i}^{2}$ the coordinate vector of the same point at the second location, both measured relative to the world coordinate system, and $\mathbf{X_{avg}}^{1}$ and $\mathbf{X_{avg}}^{2}$ represent the average values of all points at each location, then the **R** and **t** that satisfy the following equations will give the desired transformation.

$$(X12 - Xavg2, \quad \ldots \quad , Xn2 - Xavg2) = \mathbf{R} * (X11 - Xavg1, \quad \ldots \quad , Xn1 - Xavg1)$$

$$\mathbf{t} = -\mathbf{R} * \mathbf{X_{avg}}^{1} + \mathbf{X_{avg}}^{2}$$

The above rotation and translation can be obtained using Singular Value Decomposition, among other methods. In addition, if it is assumed that the cameras are attached rigidly to the tripod and stay in the same position relative to the tripod when it is moved from location to location, this rotation and translation will also describe the movement of the camera body and thus of the camera coordinate system attached to it. If the calibration has been completed and the projection matrices are known for the first stereo pair of images, these matrices, along with the above transformation, can be used to get an estimate of the projection matrices for any pair of stereo images for which the location of these fixed points on the tripod is known. Using the **R** and **t** obtained above, this estimation can be accomplished mathematically by the following method, where $\boldsymbol{M_1}$ is a 3x3 matrix containing the first three columns of the known projection matrix, $\boldsymbol{m_1}$ is a 3x1 matrix containing the last column of this known projection matrix, $\boldsymbol{M_2}$ is a 3x3 matrix containing the first three columns of the projection matrix we wish to estimate, and $\boldsymbol{m_2}$ is a 3x1 matrix containing the last column of the estimated projection matrix:

$$\boldsymbol{M_2} = \mathbf{R} * \boldsymbol{M_1},$$

$$\boldsymbol{m_2} = - \boldsymbol{M_2} * (\mathbf{R} * (-\boldsymbol{M_1}^{-1} * \boldsymbol{m_1}) + \mathbf{t}).$$

These formulae will thus allow us to estimate the left projection matrix for any subsequent camera location from the left projection matrix obtained for the first camera location, and similarly for the right projection matrices.

As before, I first prototyped this method in Matlab to see how well it would predict the location of the points. I was able to do this using data from a local test of the photographing procedure, done by other researchers working on the Archeoviz project, which included recording the location of these tripod points, using the same Theodolite method used to record the world location of the markers. The resulting estimation of the projection matrix for the second scene was quite good, with almost all of the markers located within the 30-pixel circle. This was true using only the minimum number of points on the tripod, three. Also, we found that locating the three points on the tripod adds no more than a minute or two to the archaeologists' fieldwork. Thus, this method will allow the process of calibration to be sped up greatly at the cost of a very small increase in the work that the archaeologists in the field must do.

This final method of prediction greatly improves the speed of the calibration process by requiring that users independently locate only six markers in one image of a scene in order to accomplish the calibration for all images in all stereo pairs of that scene. In addition, if the world coordinate system is kept fixed for images of other scenes to be reconstructed, and if the world coordinates of the same, fixed tripod points are again recorded, this method can also be used to estimate the location of the markers in any image of any scene, as long as the location of the markers used in those scenes are recorded relative to this same, fixed world coordinate system. Thus, all of the images for all scenes that need to be reconstructed can be calibrated using only the projection

matrices obtained for the first stereo pair, along with the text files containing the world location of the fixed tripod points.

5. RESULTS AND CONCLUSIONS

Through the development of this calibration implementation, the process of calibrating the cameras has been greatly improved in both the time needed to accomplish it and in the work required by user. This new implementation requires that users find only six markers in one image of a scene by themselves, predicting the location of all of the remaining markers in all remaining images of the scene and of all markers in any images of any scenes in which the world coordinate system is kept fixed, using the three methods of prediction described above. In the original implementation, users needed to find all of 20-25 markers in each of the 6-16 images of a scene with no help at all.

All three of these prediction methods are contained within the same program, and users can utilize any of the three depending upon the situation. Thus, the calibration for all of the images for a given scene can be completed in one use of a single program. This is another improvement over the original method, in which users had to alternate between using Adobe Photoshop or some similar program to locate the markers and record their image coordinates, and Matlab to calculate the projection matrix. Thus, with the new implementation, the time needed to achieve the calibration and the amount of work that users must do independently have been reduced to a very small fraction of those in the original implementation. This allows users to quickly move on to the construction of the three-dimensional models, and thus will more quickly allow them to use this powerful tool for studying the archaeological sites.

6. ACKNOWLEDGMENTS

I would like to thank all of the participants in the Archeoviz project, particularly my advisor, Dr. Kostas Daniilids, for allowing me to be a part of his research. I would also like to thank Dianna Xu and Sandy Patterson for all of the help that they have given me throughout the summer in my work on the project. In addition, I would like to thank Dr. Jan Van der Spiegel for allowing me to be a part of the SUNFEST program and for organizing the various events and workshops that have made it a unique summer experience. Finally, I would like to thank Microsoft Corporation and the National Science Foundation for providing the funding that has made my summer experience possible.

7. RESOURCES

1. JANUS: A hand-eye humanoid robot: Pinhole camera model, Garcia C., Available from http://www.csd.uch.gr/~cgarcia/JANUS/pinhole.gif, accessed 16 July, 2003.

2. D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, Prentice Hall, Upper Saddle River, New Jersey, 2003.

3. CV online: Compendium of computer vision, Editor Robert B. Fisher, Available from: http://www.dai.ed.ac.uk/CVonline/, accessed 16 July, 2003.

4. Calculation of the rotation matrix, Young-Hoo Kwon, Available from http://kwon3d.com/theory/jkinem/rotmat.html, accessed 22 July, 2003.

5. E. Angel, *Interactive Computer Graphics: A Top-Down Approach with OpenGL*, Addison-Wesley, Reading, Massachusetts, 1997.

APPENDIX A: THE SOFTWARE IMPLEMENTATION

I implemented the final program using Microsoft Visual C++, and used the OpenGL graphics library to achieve the display of the images to be reconstructed. I utilized the GLUT library to handle both the user input, from the keyboard and mouse, and the interface with the operating system to create the windows in which the images are displayed. I also used the GLUI library to create the buttons, text fields, and other controls utilized by the users in the control panel windows.

A screenshot of the first implementation of the program, with only the six-point algorithm used to calibrate for just a single image, is presented in Figure 4 below. The top window on the left is the console window, in which all output from the program will be printed for the user. The main window, labeled "Six Points," is the window in which the image will be displayed, and the control panel on the right contains all of the buttons and controls that the user will need in order to complete the calibration process.

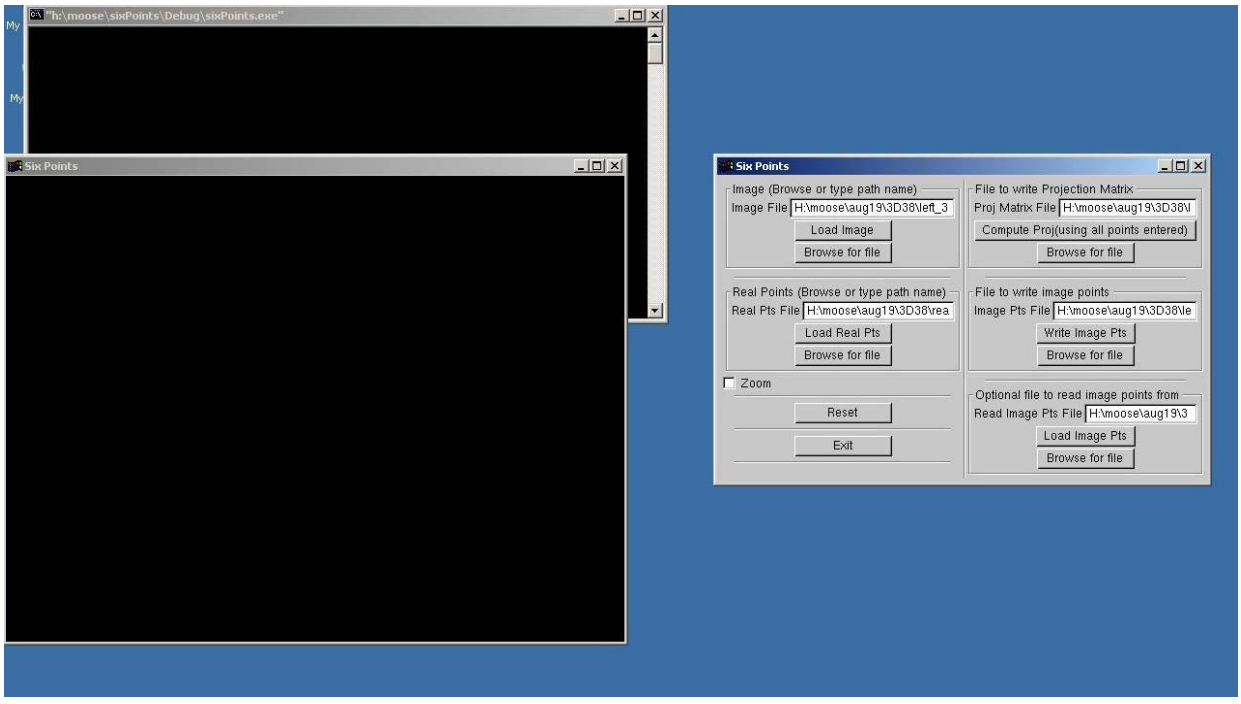


Figure 4: Screenshot of the six points program.

The input to the program is the image, in .tga format, and the text file containing the three-dimensional world coordinates of all of the markers. The output will be two text files, one containing the projection matrix for the scene and one containing the image coordinates for all of the markers in the scene, both of which will be correctly formatted for use in the reconstruction program. The user proceeds by first loading both the image and world coordinate files, and by specifying the two output text files. These files are specified in the text fields in the control panel by either using the browse buttons below the fields or typing in the file names manually. I have implemented the Windows style browsing windows to make the input of files easier, as the text fields are sometimes unstable and it is often tedious to type in the entire pathname of the file. (This problem was also present in the original Archeoviz program for the reconstruction.)

After loading all of the necessary files, the user begins the search for the markers. For the first six markers, the user types in the marker index using the keyboard and then clicks, with the mouse, on its location in the image. The program records the image coordinates of the point selected and places it in the correct location in the output file based on the input index. After these first six points are input, the program predicts the location of the remaining markers, as described above, and draws blue circles around these predicted locations. Figure 5 shows a screenshot of this step in the program, using a sample image from one of the scenes from Tiwanaku, with yellow circles displayed around the six markers entered.

Figure 5: A screenshot of the prediction step in the program. The yellow circles indicate the location of the first six markers, and the blue circles the predicted location of the remaining markers.

Next, the display will zoom to each of these predicted locations, one by one. When zoomed on a circle, the user will simply need to click on the appropriate location on the screen and the program will record the image coordinates, automatically identifying the correct index for the marker. If the prediction of a particular marker is not accurate, the user can simply type the "s" key to skip it and enter it later. After all of the markers have been entered or skipped, the display returns to the original view. If any more markers need to be entered, the user can enter them in the same manner as the first six markers. Alternatively, at any point after the first six markers are entered, the user can click on the "Compute Proj." button on the control panel to get a better estimate of the projection matrix using all of the point correspondences entered, and thus get a better estimate for the location of the points not yet entered. If the marker is within the new prediction circle, the user then can simply click on it, without worrying about entering the index of the marker, which will be displayed above the circle. After all of the markers in the scene have been clicked and their image coordinates entered, the program will calculate a projection matrix using the correspondences from all markers and will output both this matrix and the image coordinates to the appropriate text files. The user can then take these text files immediately to the reconstruction program and begin building the three-dimensional models.

I have implemented the method of estimating the right projection matrix from the left as an extension to the original six points program described above. I have created a second window and control panel for the right image, identical to the window and control panel used for the left. In addition, I have moved all of the variables and controls that relate to both images, such as the real points file and the controls that specify d and θ, to the main control panel at the top of the screen. Figure 6 shows a screenshot of the extended version of the program.

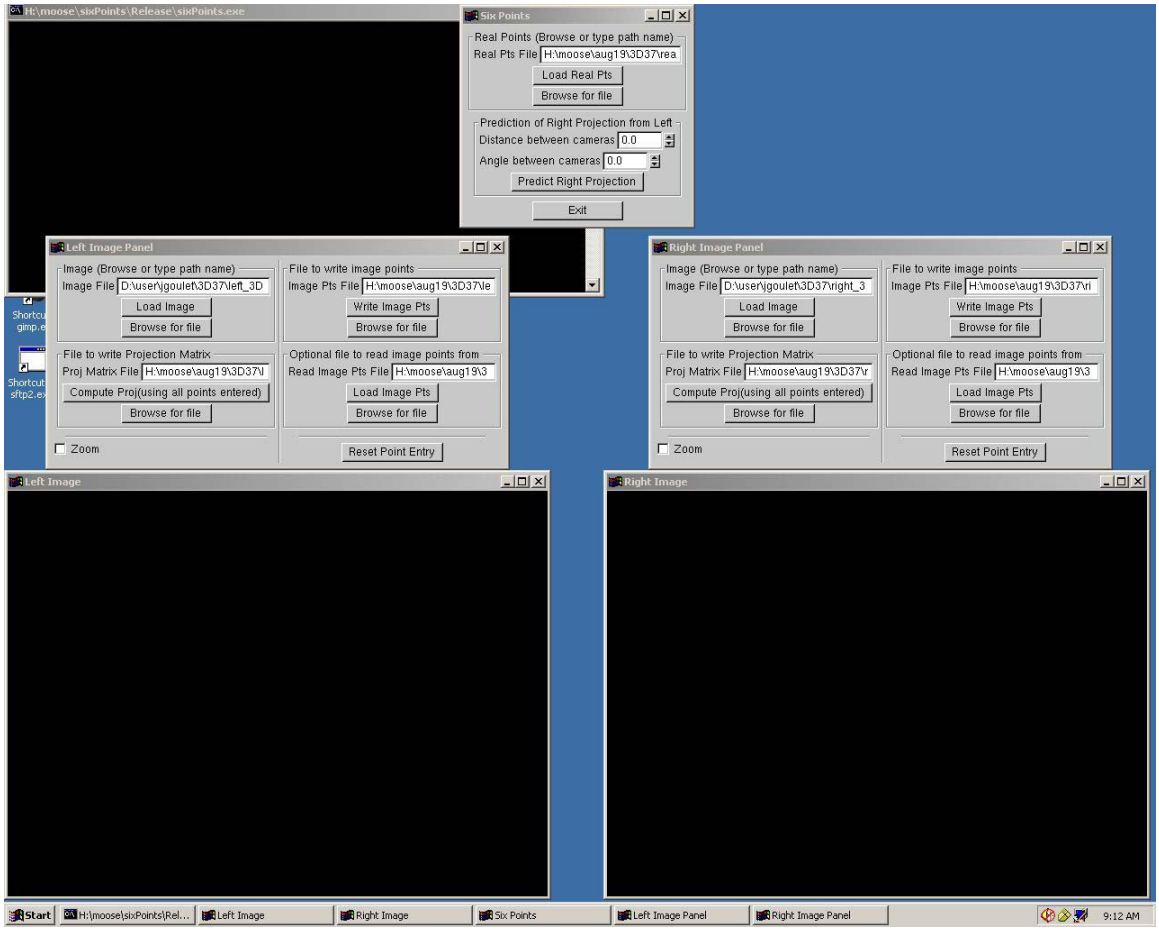


Figure 6: Screenshot of the extended version of the calibration program.

In the extended program, the user accomplishes the calibration of the left image exactly as in the original implementation of the program. Then, the user selects the appropriate values for d and θ, using the "spinners" in the main control panel, and clicks on the "Predict Right Projection Matrix" button. The program then extracts and stores the intrinsic and extrinsic parameters from the left projection matrix, using various formulae for calculating these parameters from the rows of the projection matrix. It calculates the transformation from left to right camera coordinate systems using the selected parameters and combines this transformation with the extrinsic parameters from the left projection to get an estimate for the extrinsic parameters of the right image. It uses these parameters, setting the right intrinsic parameters to be the same as those of the left image, to compute the projection matrix for the right image and then uses this to predict the location of all of the markers in the right image. The user then enters the image coordinates for the markers in the right image in exactly the same way as done for the predicted points in the left. After all markers have been selected in the right image, the image points and projection matrix text files are output exactly as they had been for the left image, in the same manner as in the original implementation of the program.

I have incorporated the final method of prediction, using the tripod data, as another extension to my program through adding four more text fields to the main control panel. These fields will be used to specify four additional text files. The first pair will be the text files containing the projection matrices for the left and right images in the first stereo pair, obtained using the program in the original manner. The second pair will contain the world coordinates of the points on the tripod, one at the location of the first stereo pair and the second at the subsequent location for which the user is currently calibrating. Figure 7 shows a screenshot of the new main control panel.

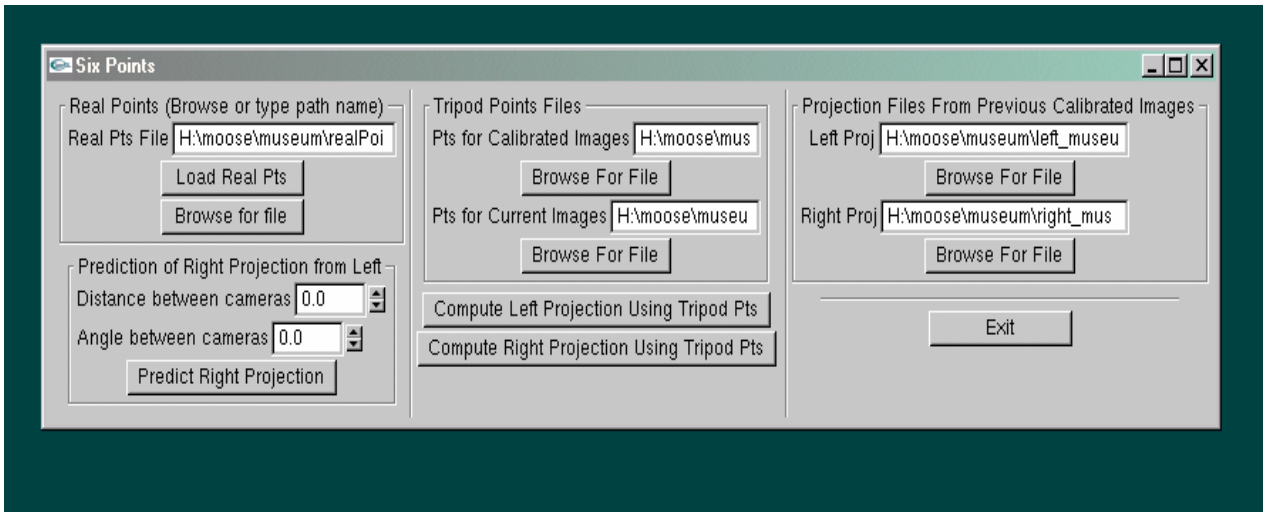


Figure 7: The new control panel for the final extension to the program.

The user specifies these four files, using the same browsing windows as before, and then clicks a button to calculate the projection matrices for the current images, using the above method for rigid transformations. The program will then predict the location of all points in both images in the current stereo pair, without requiring the user to locate any of the markers. The user can then utilize these predictions to input the image coordinates of the markers exactly as done for all of the predicted locations in the previous versions of the program. The output will again be the same projection and image point files for both the left and right images in the stereo pair, the first of which can be used in future scenes as the input projection matrices used in the tripod method.

This implementation includes various features that make it easy to use and user-friendly for the archaeologists in the field. The friendly graphical interface and easy-to-use zooming system allow the user to quickly learn how to use the software. In addition, the browsing windows provide a quick and easy method for loading all necessary files and for saving all data obtained. Also, the console provides the user with valuable information, including which markers are entered as they are clicked, how many markers remain to be entered, and tips on using some of the features of the program, such as the zooming buttons. In addition, the program automatically keeps track of the indices of the markers and displays them for the user above the circles around the markers. Finally, it automatically outputs all necessary files, properly formatted, to be used immediately in

the reconstruction. All of these features help to improve the performance of the program and add to the effectiveness of the underlying algorithms in speeding up the process of calibration and allowing the user to quickly proceed to creating these valuable graphical models.

THE CONSTRUCTION OF A MICRO-COULTER COUNTER DEVICE

NSF Summer Undergraduate Fellowship in Sensor Technologies
Mpitulo Kala-Lufulwabo (Electrical Engineering) – University of Pittsburgh
Advisor: Dr. Haim Bau

ABSTRACT

This report explains the general applications of the Coulter counter, discusses its benefits, and reviews literature in which researchers have used the device as a research tool and in biosensor applications. Coulter counters are electrofluidic devices commonly used to measure the size and the number of particles in a sample. The paper also reports the design and fabrication of a Coulter counter device constructed using Polydimethylsiloxane (PDMSX) and also one of a carbon nanotube based Coulter counter. These devices were desired to sense particle size through a change in resistance displayed through a current spike. Microfabrication was required to construct the device and prepare it for electrical and optical sensing. Potential concerns such as fabrication optimizing and mishaps, troubles with trapping, and as pore filling issues were evaluated and tested. Not all issues were resolved, and a functional Coulter counter device was not completed. However, with detailed microfabrication steps, revised trapping procedures, and improved troubleshooting, the groundwork for a functional device was laid out for implementation in the near future.

Table of Contents

1. INTRODUCTION

Science and engineering are being driven by the emerging field of nanotechnology. Many researchers believe that nanoscience may serve as a building block in the development of a sustainable society. One potential nanoscale device is the Coulter counter.

Coulter counters are electrofluidic devices commonly used for measurement of microscopic particles [1] such as in blood sample analysis. The Coulter counter helped revolutionize the complete blood count **(**CBC**)**, the standard initial blood test for a majority of medical practices. Blood quality can be analyzed rapidly, objectively, as well as quantitatively [2].

The Coulter counter can also be used as a chemical sensor or biosensor. Counters have been readily found capable in the detection of target analytes binding to functionalized nanoparticles. As the analytes bind to the nanoparticles, the visible mass of the particles seems to change and can be readily detected by observing the change in the electric current's magnitude passing through the nanotube or nanopore. Because of the favorable scaling laws in microfluidics, specifically surface-to-volume ratios, it is desirable to scale down the Coulter counter for use as a chemical or biosensor. Minimizing the counter can vastly improve these applications, as well as making procedures more chemically safe, cheaper, and more efficient than ever before [1].

This paper provides an overview of the basic Coulter principle as well as covering some specific device designs and experiments done in an attempt to implement a carbon nanotube based Coulter counter. Coulter counters' use as a research tool and in biosensor applications is also briefly reviewed.

2. BACKGROUND

2.1 Coulter Principle

The Coulter principle is illustrated in Figures 1 and 2.

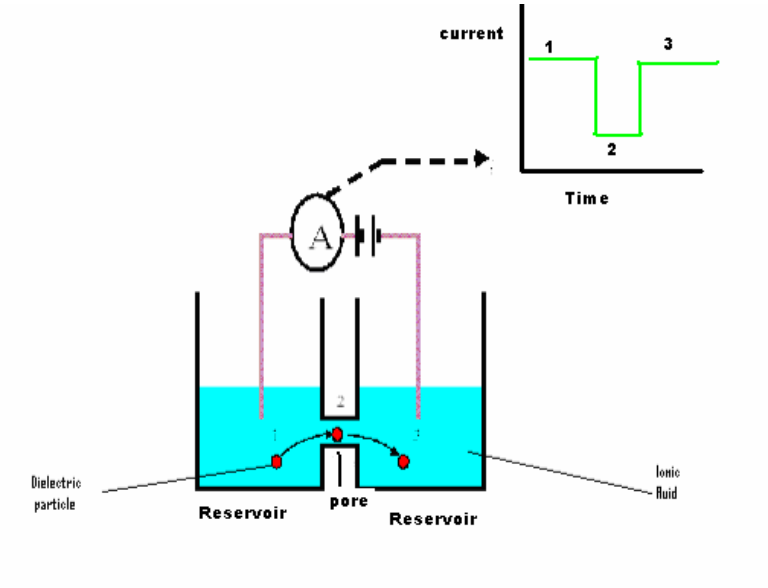


Figure 1. Basic Coulter principle – resistive pulse technique.

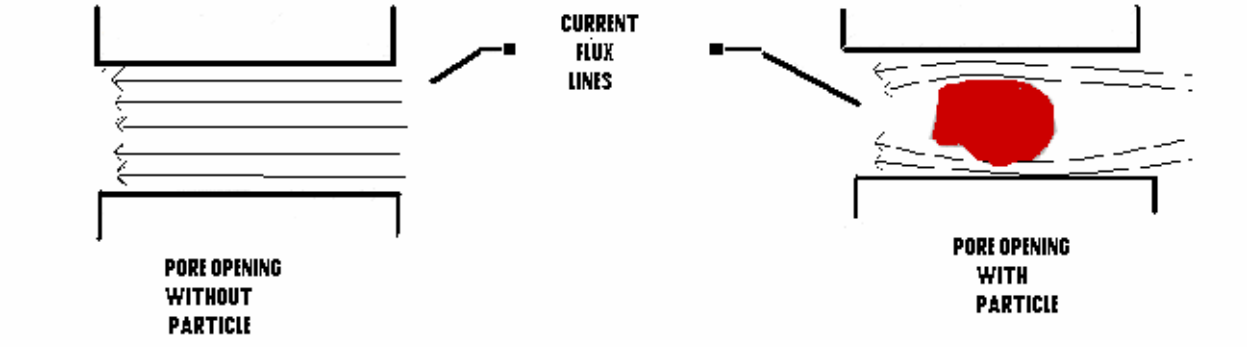


Figure 2. Particles in the pore bending the electrical flux lines.

As Figure #1 shows, a dielectric particle immersed in ionic fluid [2] is transferred through a pore by electrophoresis. As the particle enters the pore the ionic current fluctuates (See Figure #2). If the dominant resistance in the system is the pore, this can be modeled as two resistors in parallel; otherwise the resistance with the particle in the pore will be greater. The resistance in Figure 2a will be significantly less than the resistance in Figure 2b. The change in

resistance will appear as a current or resistance spike. The magnitude can be used to determine the volume of the particle [3].

The relationship between Δ resistance and volume can be expressed mathematically using equation 1:

$$\Delta R = \frac{-2\rho_f \pi^2}{A} \int_0^b \frac{(b^2 - l^2)}{\left[1 - \frac{\pi(b^2 - l^2)}{A}\right]} dl = -\frac{4\rho_f}{\pi D}\left[\frac{\sin^{-1}(d/D)}{\sqrt{1-(d/D)^2}} - \frac{d}{D}\right]$$

Where ρ_f = electrical resistance of the fluid, D = diameter of pore, and d = diameter of particles.

2.2 Resistance change and sensor performance

With other conditions such as noise, a device is assumed able to detect only approximately five percent of change in resistance. Figure #3 displays the resistance change in the pore with the addition of about 20 nm in diameter to the particle. For this test, it is assumed that the ratio of the pore diameter and particle diameter will be constant, 2 to 1.

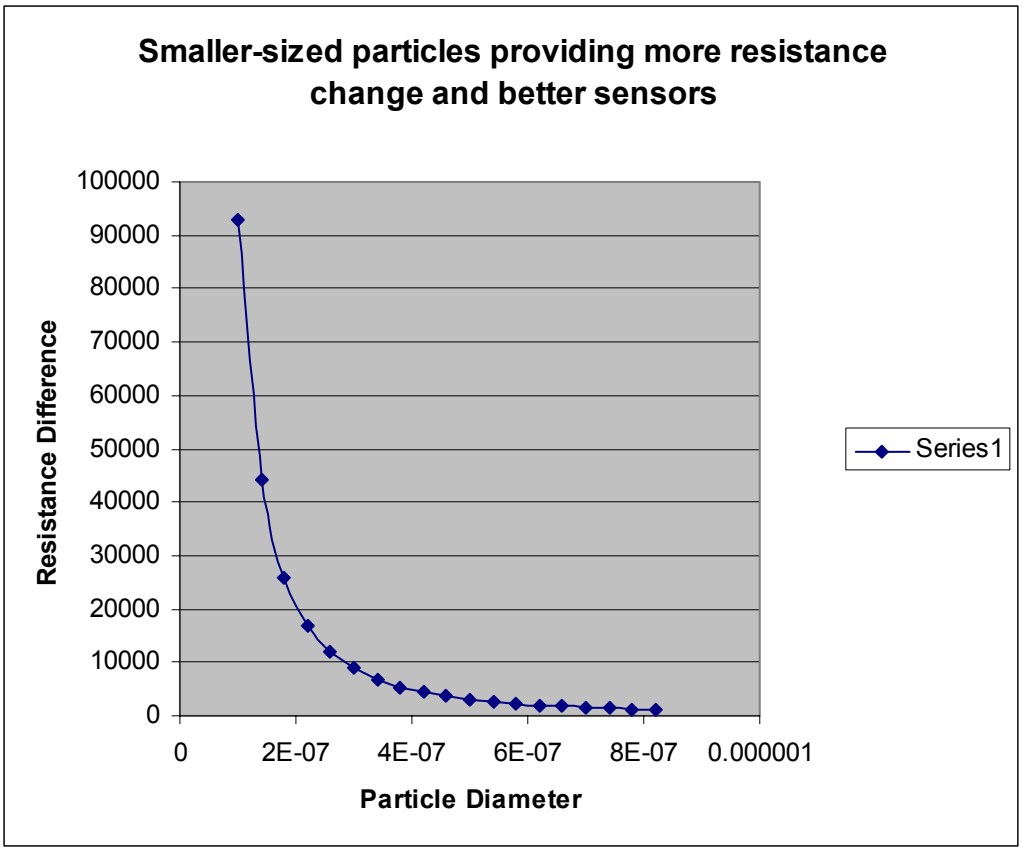


Figure 3: Shows how smaller-sized particles provide a greater resistance difference

The size of particles counted must be comparable to the size of the pore used in order for detection to occur [4]. Figure #4 shows how the surface to volume ratio exponentially increases as the particle size decreases. To obtain the surface area to volume ratio, the equations for surface area and volume were used:

$$\text{Surface Area} = 4 \times \pi \times Radius\text{^}2$$

$$\text{Volume} = \frac{4}{3} \times \pi \times radius\text{^}3$$

$$S/V \text{ Ratio} = \frac{4 \times \pi \times Radius\text{^}2}{\frac{4}{3} \times \pi \times Radius\text{^}3} = \frac{3}{Radius}$$

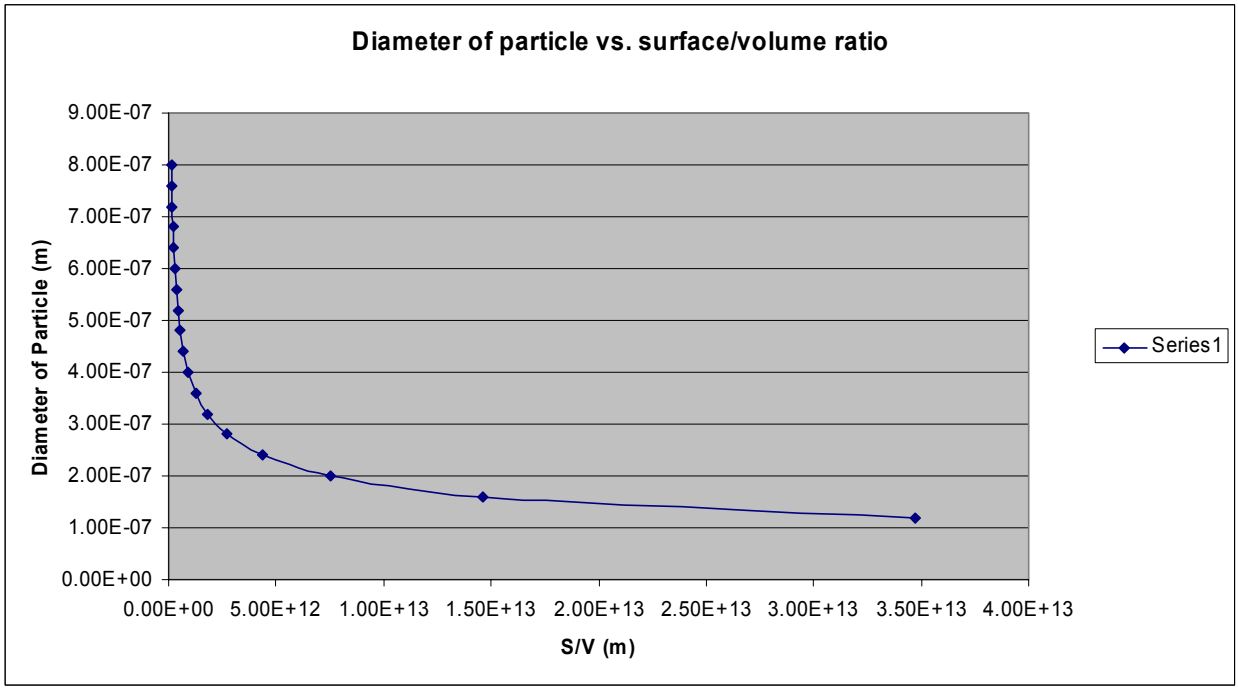


Figure 4: Smaller particles have greater surface to volume ratios

As the radius approaches infinity the surface-to-volume ratio approaches zero, while as the radius approaches zero, the surface-to-volume ratio approaches infinity.

As Figure #4 shows, the particles with the smaller diameter have greater slopes in their resistance, while the larger-sized particles have a change in resistance that tends to flatten out, thus showing that the smaller particles are the only ones whose resistance will change by more than five percent. The data from these graphs clearly show that the resistance is greatly affected by the volume of the particle that enters the pore. Figures # 3 and 4 show that the smaller particles have a greater surface-to-volume ratio, and thus a greater change in resistance. The charts are combined in figure #5 to display the connection more clearly. This supports the notion that particle volume plays an essential role in resistivity.

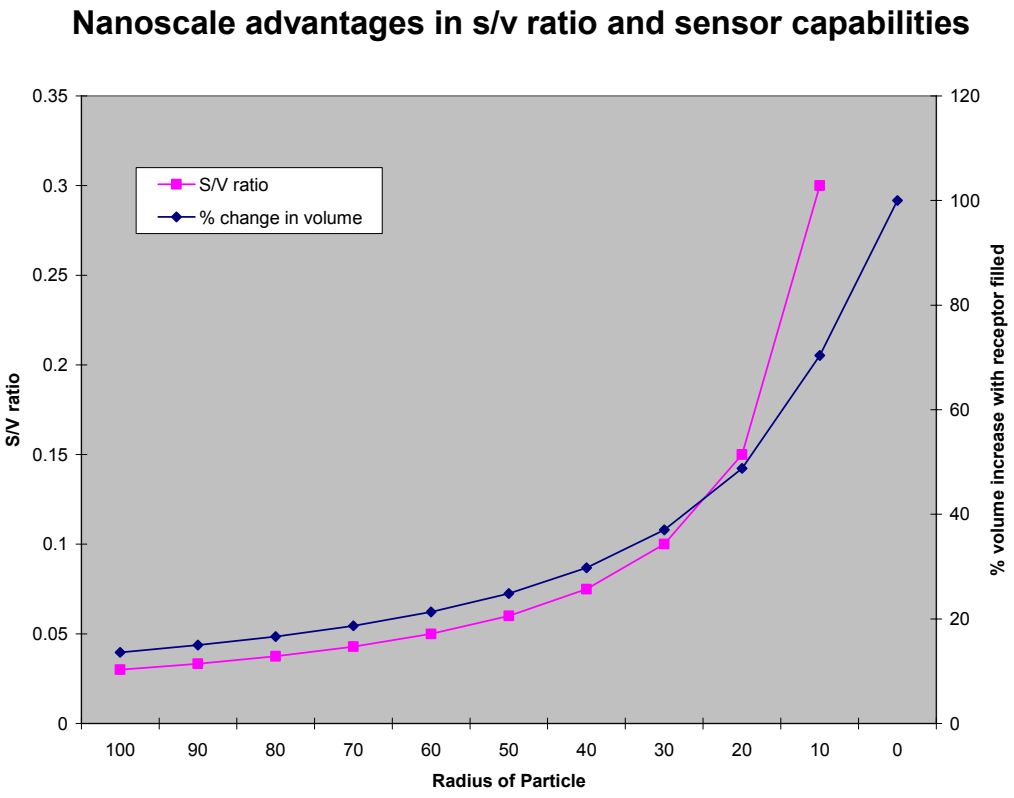


Figure 5: Particle volume playing a key role in s/v ratio and resistivity

3. Literature Review

The Coulter counter has been used in biosensor applications with particle sizes as small as 50 nm in diameter. These counters have been used for everything from blood analysis, to counting biological cells and colloidal particles, to recording debris number in cell tissue [5]; to even evaluating mass transport rates through nanoporous materials [6]. The smaller the tube's (pore) diameter the greater the device's sensitivity. Researchers predict that further reduction in pore size will allow them to build Coulter counters that can detect large polymeric molecules such as antibodies and viruses. Along with its other benefits, including simple construction, rapid production with good reproducibility, and miniaturization, counters will create many new

opportunities because many proteins, polysaccharides, and DNA do not have the electrochemical properties or are too small to be detected by available analytic mechanisms.

Various groups used Coulter counters as research tools and attempted to implement Coulter counters as biosensors. The counter produced among the most effective results for their tests. A few of the more significant studies are described below.

3.1 Characterizing Yeast Cells

In a 1997 study, Teeraoatar Srinorakutara [7] at the Thailand Institute of Scientific Research used the Coulter counter to find the cell wall thickness and cell diameter of yeast cells. This experiment involved comparing the yeast cell size distributions using the counter to the size obtained using image analysis. The results showed that although the mean cell diameter could be effectively measured using either image analysis or Coulter counting. Coulter counters were more efficient, because of the simplicity of the device, the mass transport and number capabilities, and its ability to render fast, accurate results. Any direct effects of vacuolation on the cell diameter, such as changing the resistance of the cell, were thought to be small. This theory proved consistent with the notion that volume is proportional to the pulse height and response of the instrument, as there is extremely little effect of particle resistivity on the diameter of the cell.

3.2 Efficiency of Aerosol Samplers

P. Gorner, R. Wrobel, and J.-F. Fabries of the National Institute of Research and Security in France [8] used a Coulter counter to measure the efficiency of aerosol samplers. Aerosol samplers measure the concentration of health-related aerosol fractions in work areas. They must satisfy the international requirements of sampling performance, including the penetration efficiencies for inhalable dust laid down by such hygiene organizations as the International Standards Organization in Geneva. The efficiency is normally measured and determined by comparing the ratio of concentration of sampled and referenced aerosols, as a function of particle aerodynamic diameter which is usually measured by time of flight method. However, for their case of high speed winds, normally a few meters per second, high sampler flow rates, or coarser particles, the researchers used the Coulter counter method. First, the test aerosol was generated, and then neutralized and weighed. The particles were recovered in an electrolytic solution with some surface active agent added. The electrolytic particles were forced through a calibrated pore using platinum electrodes on each side. The penetration efficiency data for the sample were also assessed compared using the time of flight method. The experiment determined that the Coulter counter produced comparable results but was slightly more effective than the time of flight method in determining the efficiency of aerosol samplers [6].

3.3 Implementation as a Biosensor

O.A. Saleh and L.L Sohn of Princeton University [2] implemented the Coulter counter as a biosensor. They fabricated a microchip counter on a substrate made of quartz, and used it to try to detect individual nanoscale colloidal particles that had a sensitivity in proportion to the size of each individual particle. The device was able to sense colloids as small as 87 nm in diameter,

and to also discriminate among colloids whose diameters differed by less than 10%. Consistent with the Coulter principle, the researchers performed a test in which four electrodes were patterned across the reservoirs, followed by small samples of platinum in an electron beam evaporator for conductive purposes. The tests were done using solutions of colloids of various diameters and varied applied voltage. It was found that the downward current pulses were inversely proportional as the applied voltage, as was expected. When the experimental data were compared to the predicted data, the measured and calculated values were found to be comparable. By further reducing pore sizes, the researchers concluded, their device would be applicable for measuring macromolecules, including DNA and proteins [2].

4. EXPERIMENTAL METHODS

This section reports on efforts to fabricate and implement a Coulter counter device. Discussed are two methods used: PDMS Coulter counter devices and a microfabricated device using both glass or silicon wafer chips and carbon nanotubes.

4.1 PDMS Device Design

The first device was basic in design. It contained a 2 micron inner diameter tube, which served as the pore. The outer coating of the tube was melted away by a Bunsen burner, enabling the pore to be transparent. Two small tops were to be taken, preferably saucer-style plastic tops. The tops, equal in size, served as the reservoirs in which the particles were submerged. A drill containing a bit designated for plastic materials was used. One tiny hole penetrating each top was drilled. After the tube was connected to the tops, the device was sealed with a silicone coated material called polydimethylsiloxane (Dow Corning Corp.), PDMS, made by mixing a 10 to 1 ratio of the PDMS material with its curing agent. The PDMS was placed in a vacuum in order to remove the air bubbles and was then cured. It generally takes 16-24 hours for the PDMS to cure in air, but with heat applied it cures quicker. After about two hours the still liquid PDMS was poured onto the device and allowed to cure on it. When the PDMS had finally cured, the micron tube was removed, thus leaving a small channel imbedded in the PDMS layer. Next the caps were removed as well, leaving two reservoirs overtop a thin PDMS coat. What remained were two reservoirs connected to a channel, similar to the one displayed in figure #6.

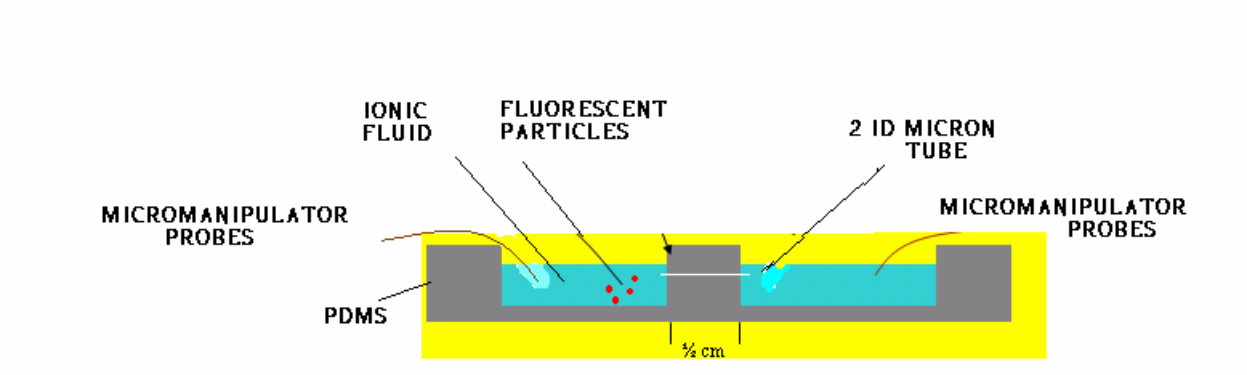


Figure 6. Model of Coulter counter device, similar to the one designed

4.1.1 Testing

For the PDMS device, fluorescent particles were placed into one of the reservoirs, which contained an electrolytic solution, so that the particles could be viewed optically in the pore. Attached at opposite ends of the pore were electrodes attached to micromanipulator probes. The probes were connected to an AC field power source; thus, when the power was turned on, an electric field within the pore was to be created. The voltage between these two fluidic reservoirs was to drive electroosmotic flow, and the fluorescent particles were to be transported by electrophoresis. The voltage needed to drive the particles through was then applied. An arbitrary potential of 10 volts was selected. The fluorescent particles were thus supposed to be viewed traveling through and out of the pore. If the particles were too small to be observed by the naked eye, a microscope would then be used and the device would be constructed under the microscopic lens and viewed at a necessary magnification.

4.1.2 Results

The PDMS device encountered a significant number of problems and therefore was disregarded over time. The first device was faulty in that the PDMS did not cure properly. There were many bubbles within the device, especially near the opening of the pore. The PDMS was also not solid in some areas. Several attempts were made to properly cure the PDMS as well as vacuum out any bubbles that arose during the curing stages. These attempts finally provided successful results. However, after the device was made, problems developed when filling the pore. As the particles were poured into the reservoir containing water and a hydrolytic solution, a 10 volt potential was applied. Neither the particles nor any liquid traveled through the pore, nor did any liquid. Tinkering with the voltage appeared to have no real effects on the device. Something within the pore was not allowing for transport, perhaps an air bubble. After weeks testing out possible solutions, the use of a PDMS device was halted and a silicon/glass device was introduced.

4.2 Silicon/Glass Wafer Device

Other Coulter counter devices used in the experiment employed glass or silicon wafer chips in the microfabrication of a functional device. The fabrication of these devices meshed customary microfabrication techniques with the dielectrophoretic trapping of single carbon multi-walled nanotubes.

4.2.1 Device Design

The first step in designing the device used a p-type silicon wafer. The wafer was cleaned with acetone and isoproponal. A small 1 micron layer of silicon dioxide (SiO_2) was applied to the top of the silicon. The SiO_2 layer serves as a thermal insulator for the chip, preventing heat or electrons from reflecting off the surface. With the use of an electron-beam machine, gold was then evaporated on the chip. The wafer is placed in a sealed, vacuumed chamber where a beam of electrons strikes a mound of gold and then evaporates gold onto the wafer. A layer of nickel

chromium was also applied on top to serve as an electrical insulator. The chip is then ready for the photolithography process.

Photolithography is a process of transferring geometric shapes on a mask to the surface of a substrate [9], in this case glass. The first step was to chemically clean the chip and remove any remnants of organic, ionic, and metal impurities from the chip as well as any other unwanted residue. Once again cleaning was done by applying acetone, water, and isoproponal. Positive photoresist was then added to the surface of the chip. High-speed spin coating was performed to apply the photoresist. The chips were spun at 5000 rpms for approximately 25 seconds. The chip is then soft baked on a heater for approximately 4 minutes to remove almost all of the solvents from the photoresist coating. The chip is then ready for exposing.

The positive resist was exposed with ultraviolet light wherever underlying matter was removed. The chip was then aligned with a mask, a square glass plate with geometric-shaped designs on the surface. The mask was put in contact with the gold plate and exposed to UV light for 21 seconds, in this case. The photoresist has properties that allow the contents of the mask to be transferred onto the gold plate. After exposure the plate was developed for approximately 45 seconds in photoresist developer. The chip was immersed in water and dried. It was then hard baked in order to harden the photoresist and improve adhesion. Next a wet-etching technique was used to remove the nickel chromium layer from the chip. What remained was a chip with gold-patterned electrodes. Finally the chip was dipped into a positive resist stripper for a minute and then into water.

After the gold electrodes were patterned onto the chip, multi-walled carbon nanotubes were prepared. These commercially available carbon nanotubes were treated with heat and suspended in a mixture of Isoproponal and H_2O. The tubes were then placed in an ultrasonic machine for approximately a half hour, in order to break apart the tube bundles. The tubes were then once again suspended in the mixture and put in the ultrasonic machine for another half hour. The tubes were then trapped onto the chips using dielectrophoresis. The dielectrophoretic process involved connecting an oscilloscope, frequency generator, resistor, and voltmeter to a probe station. Although many trials and experiments were done using various voltages and frequencies, for one trial a peak-to-peak voltage of 15 VAC was applied as well as a 2 MHz frequency for approximately one minute. Examples of other trials included 3 VAC, 2MHz for 1 minute, and 18 VAC, 2MHz for 5 minutes. Micromanipulator probes were attached to each of the isolated electrodes. The gap spacing differed with the varying experiments, anywhere from 4 microns to 15 microns in length. After the power source was turned on the CNT solution was poured in between the two remote trapping electrodes. The tubes were then trapped in between those electrodes. With the 4 micron gap spacing an efficient single tube was produced.

Next, SU-8 reservoirs were placed around the tube ends with a mask that was superimposed with the previous mask. SU-8, a negative toned, thick, chemically amplified photoresist, had to be fabricated onto the chip. The substrate was cleaned and dried, and then a coat of SU-8 was spun onto the chip at 5000 rpms for 25 seconds. The chip was then soft baked at 65 ºC on a hot plate for one minute and then 95 ºC for five minutes in order to evaporate the solvent. After heating, the chip was exposed, with the new mask superimposed on the old mask, for approximately 42 seconds. Alignment marks on the masks made it easy to line up the two masks appropriately. After exposing, the chip was placed on the hot plate for one minute at 65 º and five minutes at 95 ºC once again for post baking. The chip was then submerged in MicroChem SU-8 Developer (five parts water, one part developer) for one minute. After

developing the chip should be rinsed with water and air-dried. An SU-8 reservoir sidewall reservoir was patterned around the carbon nanotube endings and the gold electrodes. The device was then ready for testing.

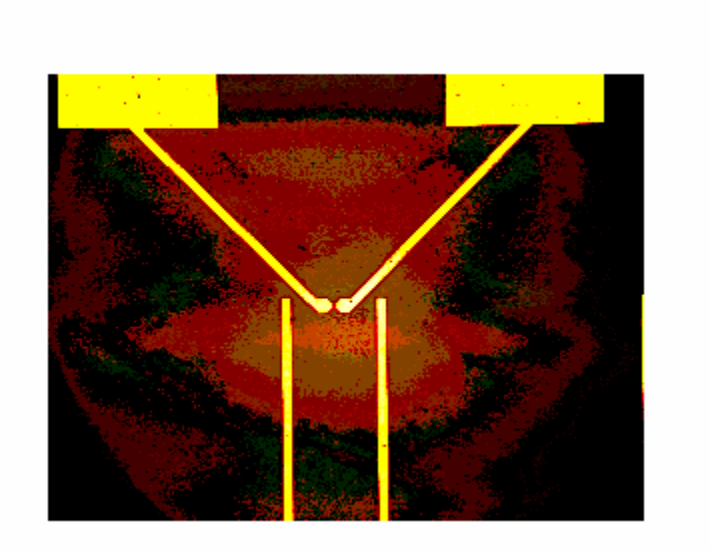


Figure 7. Gold electrodes in SiO_2 (MEAM laboratory University of Penn)

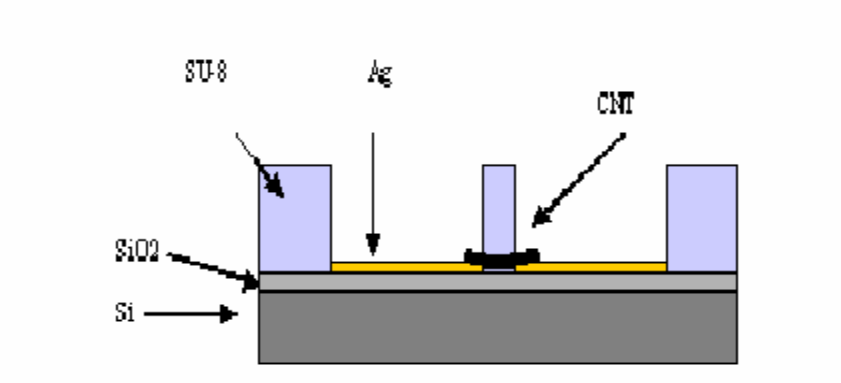


Figure 8. fabricated Coulter Counter device

4.2.2 Testing

The silicon wafers were to be tested electrically by measuring a change in resistance when the particles entered the pore. When the particle passed through the pore a current spike was intended to formulate. The carbon nanotubes that were trapped onto the chip were to then have spherical particles 49 nm in diameter driven through them. The particles are immersed in conductive fluid and then poured onto the substrate. They particles are suspended in the SU-8 reservoirs and once a potential was turned on they were to migrate through the pore. Attached to the device was a computer running a Matlab program that was to measure the current throughout the procedure.

4.2.3 Silicon/Glass Wafer

Trying to create the Coulter counter device using a silicon chip did not produce desired results. There was a problem with adhesion of the SU-8 onto the substrate as well as scum and other undesired particles forming on the chip. Methods were explored to remove such residue without damaging the chip altogether, and one seemed to work. The silicon wafer was dipped in a solution of hydrofluoric acid, which etched off a layer of oxide and therefore lifted off the scum with the oxide. This process was very effective and had very little effect on the chip. The next problem was the nanotubes trapping in between the electrodes. Clustered tubes and tubules not connecting to both electrodes continuously occurred, especially with an increased gap space and a higher potential. It is possible that the problem originated all the way from microfabrication. When wet etching was performed to remove the unwanted nickel chromium layer, the layer would be removed but some of the gold electrodes would be undercut and be etched away also, resulting in an increased and undesirable spacing of the electrodes.

The desired single tube was finally obtained with the specific 4 micron gap spacing of this chip. Once the tube was trapped, the test of driving particles and fluid through the pore commenced. The problem of tube filling also occurred with this device. The particles seemed to not be traveling through the pore. When trying to detect a current spike, the Matlab program did not perform enough readings per second. Also, when the current-voltage characteristics were measured there was relatively no change before and after the power was turned on. This in essence showed that the Coulter counter was not functioning. Flourescent particles were then poured onto the substrate and observed in a microscope. The particles looked to be trapped at the opening of the tube. Reducing the surface tension and contact angle of the pore and the particle by applying SDS detergent produced relatively no change in results.

5. DISCUSSIONS AND CONCLUSIONS

Although the Coulter counter devices did not work to the desired specification, some strides were made that suggest a functional device will be created in the near future. Extensive trial and error methods in fabrication as well as in testing have pinpointed prospective problems as well as changed and perfected specific techniques. For future tests, it will be necessary to measure the resistance of the pore before and after filling in order to determine if the Coulter counter is working properly. The numerous tests performed will make it easier to attack the operation from various angles in the future. Fabrication of the devices has been made simpler because of the different methods used in construction. Applications such as making the bake time 4 minutes instead of 2 minutes, spinning a specific layer of SU-8 on the chip, and curing the PDMS at specific temperatures, have all contributed to simplifying future experiments. When fluid and particles are properly filling the pore of the device, experiments characterizing and identifying particles will be possible.

6. ACKNOWLEDGMENTS

I would like to thank SUNFEST for selecting me to participate in such a beneficial program. I would like to thank my mentor, Michael Riegelman for all the help and guidance he has provided during the 10-week program. I would also like to thank my advisor, Dr. Haim Bau,

for all of his help. I thank NSF for funding my research, Vladimir Dominko, the rest of the faculty and staff as well as my family for all of their support during this period. I feel blessed to have been able to partake in such a worthwhile and satisfying experience. SUNFEST has undoubtedly fueled my desire to seek advancement in engineering.

7. REFERENCES

[1] Vijayaragavan, N; Shen, H; Shan, S; Wu, C. Micro Non Silicon Coulter Counter. Retrieved from ENEE 605 Term Project, University of Maryland [Online] 2002, pages 1-7. www.enee.umd.edu/class/enee605/grp/fg4_report.pdf. Accessed June 23, 2003

[2] Runker Room: Profile: Work: Dr. Dave's Museum. http://www.geocities.com/Tokyo/Island/6653/museum.htm#bckgd. Accessed July 20, 2003

[3] Koch, M; Evans, A G R; Brunnschweiler, A. Design and fabrication of a micromachined Coulter counter. *J. Micromech. Microeng.* [Online] 1999, *9*, 159-161. www.enee.umd.edu/class/enee605/grp/g4_report.pdf. Accessed June 23, 2003.

[4] Kobayashi, Y; Martin, C. R. Toward a molecular Coulter counter type device. *Journal of Electroanalytical Chemistry*. 1997, *431*, 29-33.

[5] Elfick, A; Green, S; Unsworth, A. Laser Diffraction as a Tool for the Acurate Prediction of Debris Load in Periprosthetic Tissue. *47th Annual Meeting, Orthopaedic Research Society*. [Online] 2001. www.jbjs.org/ORS_2001/pdfs/1032.pdf. Accessed June 23, 2003.

[6] Sun, L; Crooks, R. M. Single Carbon Nanotube Membranes: A Well-Defined Model for Studying Mass Transport through Nanoporous Materials. *J. Am. Chem. Soc.* [Online] 2000, *122*, 12340-12345. http://www.orgchem.iisc.ernet.in/jacs/jtoc4900.html. Accessed June 24, 2003.

[7] Srinorakutara, T. Determination of Yeast Cell Wall Thickness and Cell Diameter Using New Methods. *Journal of Fermentation and Bioengineering.* 1998, *86*, 253-260

[8] Gorner, P; Wrobel, R; Fabries, J.-F. Experimental Method to Determine the Efficiency of Aerosol Samplers Using the Coulter Counter. *J. Aerosol Sci*. 2000, *31*, S268-S269

[9] Photolithography. www.ece.gatech.edu/research/labs/ vc/theory/photolith.html. Accessed July 20, 2003.

Binaural Sound Localization

NSF Summer Undergraduate Fellowship in Sensor Technologies
Emery Ku (Electrical Engineering) – Swarthmore College
Advisor: Dr. Dan Lee

ABSTRACT

Acoustic localization is an important process used by humans and many animals. Bringing this sense to an artificial system has many possible applications. The system explored here is the commercially available Sony Aibo ERS-210 robot dog. The primary goal is to use this robot dog to track white noise sources in the forward hemisphere. Physically modifying the original equipment allowed for better tracking of such sounds in the vertical direction; prior to this modification, there was very little variation in the spectrum of the recorded sound as a function of elevation. Templates of spectra at different elevations combined with the time delay between ears allowed for varied accuracies. The lowest standard deviation of errors occurred at a position directly in front of the robot dog, while the greatest errors were at the periphery of its sight. The range of standard deviation of errors for phi and delta in the forward hemisphere are as follows: lowest standard deviation of error in Delta occurred at 15 degrees azimuth and 0.3 radians elevation or (15, 0.3): 0.1042; highest standard deviation of error in Delta at (-90, 0): 25.804; there was never any error in Phi at these locations: (-45, -0.6), (45, -0.3), (5 & 15 & 45, 0), (-45 & -15 & 15 & 45, 0.3), (0 & 15, 0.6).

Table of Contents

1. INTRODUCTION

The sense of hearing is an integral part of the human experience. Audition allows us to communicate with others, navigate the world around us, and perceive and avoid dangers. The spatial component of hearing is crucial for these uses whether to center in on the voice of a speaker, or to steer through traffic. Several main characteristics of a sound act as cues for localization. Given the right circumstances, (eg. frequency spectrum, position) humans can detect a shift of approximately 1˚. [1] Translating the human auditory process to an artificial system poses an interesting challenge in addition to offering numerous significant applications.

This experiment reported in this paper evaluated the suitability of a robot dog for use as an acoustic localizer to a white noise sound source. The Aibo communicated with a PC via a wireless Ethernet connection and all calculations were performed in Matlab.

2. BACKGROUND

2.1 Human Localization of Pure Tones – Two Cues for Localization

The localization of steady pure tones is very unnatural for humans, as sinusoidal sounds do not occur in nature. However, it is instructive to examine this process since the same methods apply to more complex sources. **Figure 1** shows the three-dimensional space surrounding a subject's head, mapped onto a sphere. Thus, direction of any sound source relative to the listener can be specified by its azimuth and elevation.

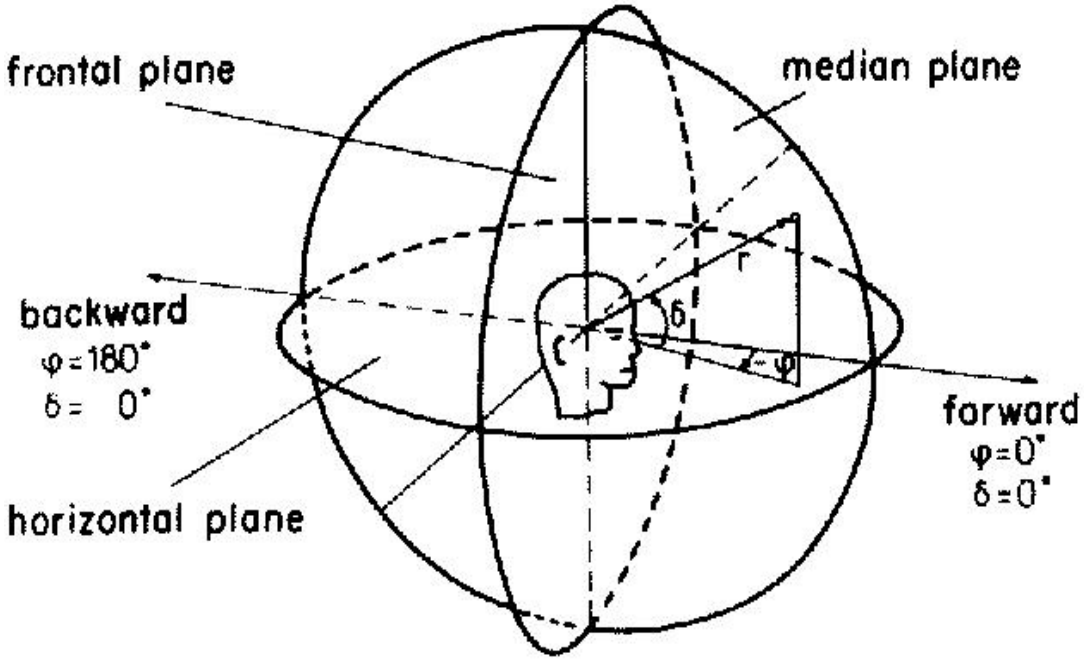


Figure 1: The spherical coordinate system about a human subject. [1]

The speed of sound is finite; a listener's two ears are spatially separated. Thus, any sound originating from a location that is not on the median plane will reach the two ears at different times. This acoustic cue is known as the interaural time difference (ITD). Additionally, if the sound comes from the left or the right, the head will diffract the sound at low frequencies and act as an attenuator at higher frequencies. This results in a second cue, the interaural intensity difference (IID). [1, 2, 3] This cue is nearly ineffectual below frequencies of approximately 500 Hz, though the difference may be as large as 20 dB at high frequencies. [1]

At lower frequencies, greater emphasis is placed on the ITD. [2, 3] In this instance, the time difference is manifested as a phase difference and is readily quantifiable. However, for a constant tone, changes in the ITD at frequencies above 1500 Hz are nearly undetectable; because the period of the signal is so short, phase shifts are essentially irrelevant.

The greatest precision of aural localization of pure tones occurs when the sound originates from straight ahead (zero degrees azimuth and elevation). [3] This is illustrated in **Figure 2**.

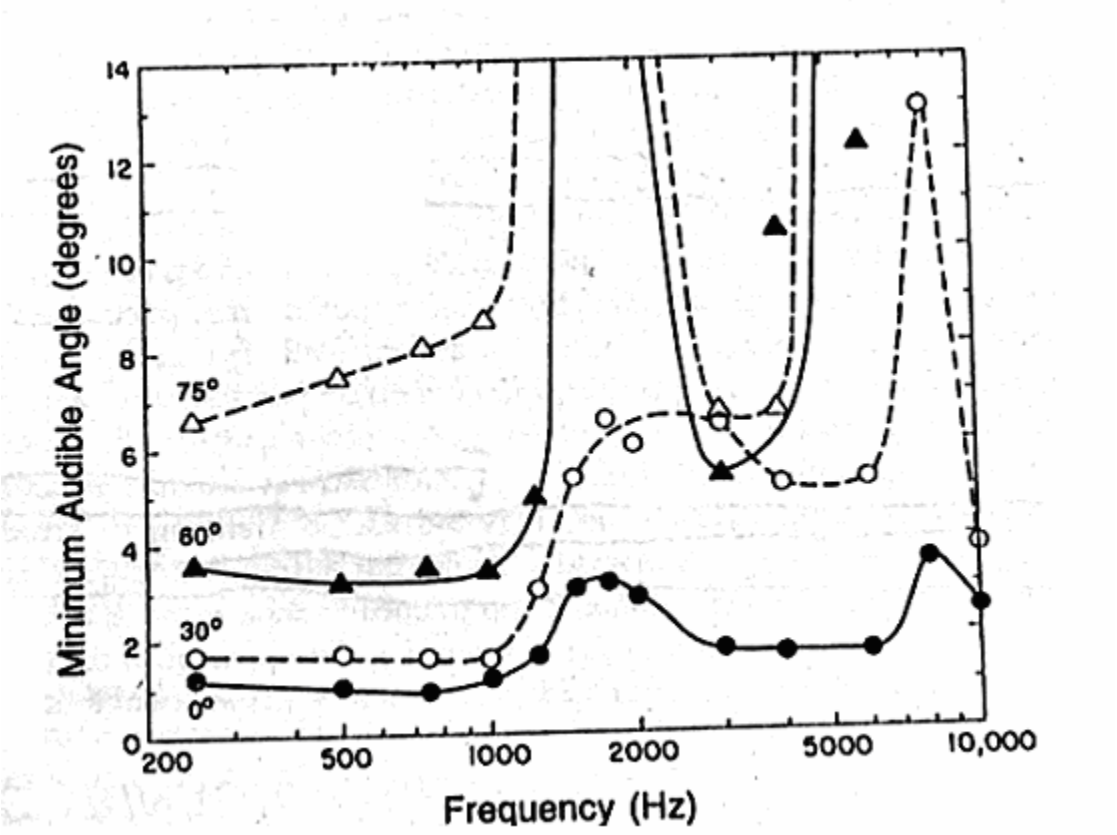


Figure 2: Localization precision decreases as the source moves away from the point directly ahead. [1]

2.2 Human Localization of Complex Sound Sources – A Third Cue

Complex sound sources are abundant in nature. They differ from steady pure tones in that they have onsets and offsets, as well as an amalgamation of many different frequencies. The additional characteristic of a sound helps in classifying its source direction is the shape of its frequency spectrum; by definition this applies only to complex sounds. This cue is known as the spectral cue, or the head-related transfer functions (HRTF). [1, 2]

Spectral cues allow for mapping of sounds in the vertical plane. The spectra of various sounds change with the elevation of the source as a result of various physical parameters. These may include the size and shape of the torso and shoulders, and most critically, of the outer ear or pinna. The changes in the magnitude of certain frequencies of the source spectrum are due to the reflections' cancellations and reinforcements given the shape of the human appendages. The frequency range that contains the most pertinent data (changes due to changes in source positioning) is approximately 5-18 kHz. [3]

3. SONY AIBO ROBOT DOGS

The goal of this project was to bring aural localization to an artificial system, in this case the commercially available Sony Aibo ERS-210A. The Aibo robot dog is equipped with two sets of internal microphones that can be set to "unidirectional" or "omnidirectional" mode in software. [4] For the purposes of this project, the microphones were left on omnidirectional mode. In addition, an important feature is the CMOS image sensor installed at the front of the head. This camera was important for standardizing placement of the speaker, as the head can be set to turn in precise directions. **Figure 3** shows a typical Aibo robot dog.

Figure 3: A Sony Aibo ERS-210 robot dog.

It was necessary to modify the original microphone configuration of the robot dog in order to obtain more usable data. **Figure 4** illustrates this alteration. It is further explained in section 6.

Figure 4: The modified Sony Aibo robot dog. The microphones have been placed inside the "ears," which now point downward.

4. THEORY AND METHODS

4.1 Cues

The two cues utilized for localization with the Sony Aibo are the ITD and the Spectral Cue. The ITD was chosen over the IID because of its greater accuracy. The sound source chosen contains information at all frequencies (white noise), which allows more flexibility regarding choosing cues.

The ITD alone provides enough information to map out an infinite cone about the axis formed by the two microphones. The surface of this cone contains the set of all possible source locations given a particular ITD. This is commonly known as the "Cone of Confusion," and is illustrated in **Figure 5**.

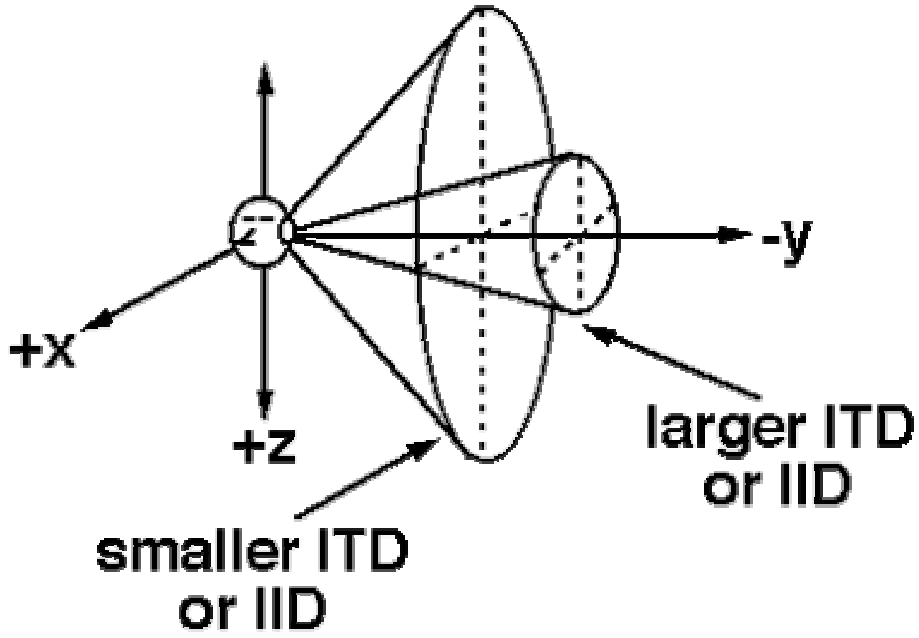


Figure 5: The Cone of Confusion.

The formula for the angle of incidence α (from the cone to the axis) is calculated given geometric constraints:

$$\sin(\alpha) = \frac{ct}{d} = \frac{cn}{fd}, \qquad (1)$$

where c is the speed of sound, t is the time delay in seconds, n is the time delay in samples, f is the sampling frequency, and d is the distance between the microphones. [2]

If the angle of elevation is also known, the set of possible source locations is reduced to two rays that lie on the cone. This theoretical ambiguity of two directions is explained by the horizontal symmetry of the head; one ray points onto the forward hemisphere while the other is a reflection onto the rear hemisphere. [1, 2] The solution pointing backward is considered extraneous in this experiment, as source locations are restricted to the forward hemisphere.

4.2 Calculations

4.2.1 Time Delay

The process to calculate the time difference between the two acquired samples occurs in the frequency domain. Although this may initially seem counterintuitive, the results are much more accurate than those obtained by matching the two time-domain samples. Often there is some degree of distortion from one channel to the other due to the head, thereby rendering direct comparison very difficult. In addition, work in the time domain is limited by the sampling frequency. In the case of the Aibo robot dog, the maximum

audio sampling frequency is 16,000 Hz. [4] Given that the distance between the two sets of microphones is approximately 5cm, the number of samples corresponding to different time delays at different points on the sphere therefore has a range of about 10 points. This range corresponds to a maximum accuracy of 18˚ along the horizontal plane which is inaccurate at best.

A more refined approach uses a cross-correlation algorithm based in the frequency domain. A time delay translates to the following linear phase shift in the frequency domain [5]:

$$m(t - dt) \Rightarrow M(f) \bullet e^{-j2\pi f * dt} \tag{2}$$

Therefore, maximizing the cross-correlation function between two complex functions X_l and X_r by varying the value of the time shift **dt**, it is possible to obtain a much more precise angle of incidence [6]:

$$dt = \max\left[X_l * conj(X_r) \bullet e^{-j2\pi fdt}\right] \tag{3}$$

Thus, the correct value of **dt** is obtained when the above argument in equation 3 is maximized. This applied in conjunction with equation 1 provides the angle of incidence for the Cone of Confusion.

4.2.2 Elevation

The calculation of the elevation relies solely on information obtained from spectral cues. The HRTF contains spectral information and is a function of source position (azimuth, elevation, distance). A transfer function is given by the following:

$$H_{\theta,\phi}(f) = X_{\theta,\phi}(f) \Big/ S(f), \tag{4}$$

where X(f) is the recorded spectrum at a given location and S(f) is the source spectrum. The recorded and source spectra are calculated by taking the Discrete Fourier Transform of the respective time-domain signals. **Figure 6** illustrates an averaged HRTF at various elevations and zero degrees azimuth.

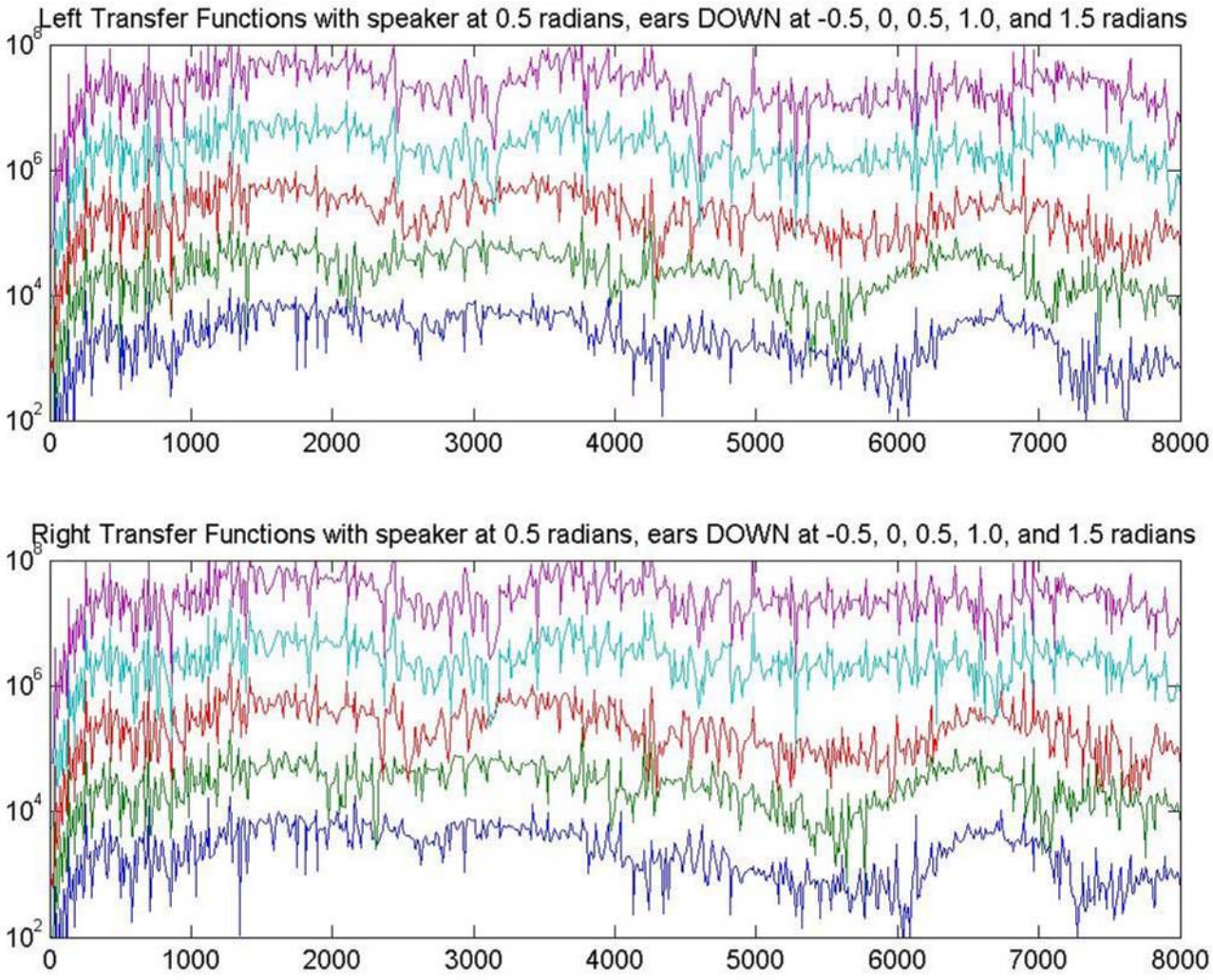


Figure 6: HRTF (frequency in Hz vs. magnitude) at five different elevations, zero degrees azimuth. Note that because of the Nyquist Rate, the frequency range of acquired data is limited at 8 kHz. [8]

The numerous sharp spikes and dips are problematic for a simple template matching algorithm, though matching a new transfer function with known templates that represent different locations is the simplest method available. The most relevant features of the transfer functions are the wide troughs and peaks. To isolate these data, the Discrete Fourier Transform is applied to the absolute value of the log of the transfer functions. Speech analysts refer to this strategy as Cepstral Data Analysis. [7]

$$\text{Cepstral Data} = \text{fft}(\log(\text{abs}(\text{TF}))) \qquad (7)$$

In order to ensure accuracy, numerous samples of several different white noise sources are taken at chosen elevations and azimuths. Each sample is then processed into complex Cepstral Data. **Figure 7** shows the Cepstral Coefficients for one particular white noise source at a source direction straight ahead (zero azimuth and elevation).

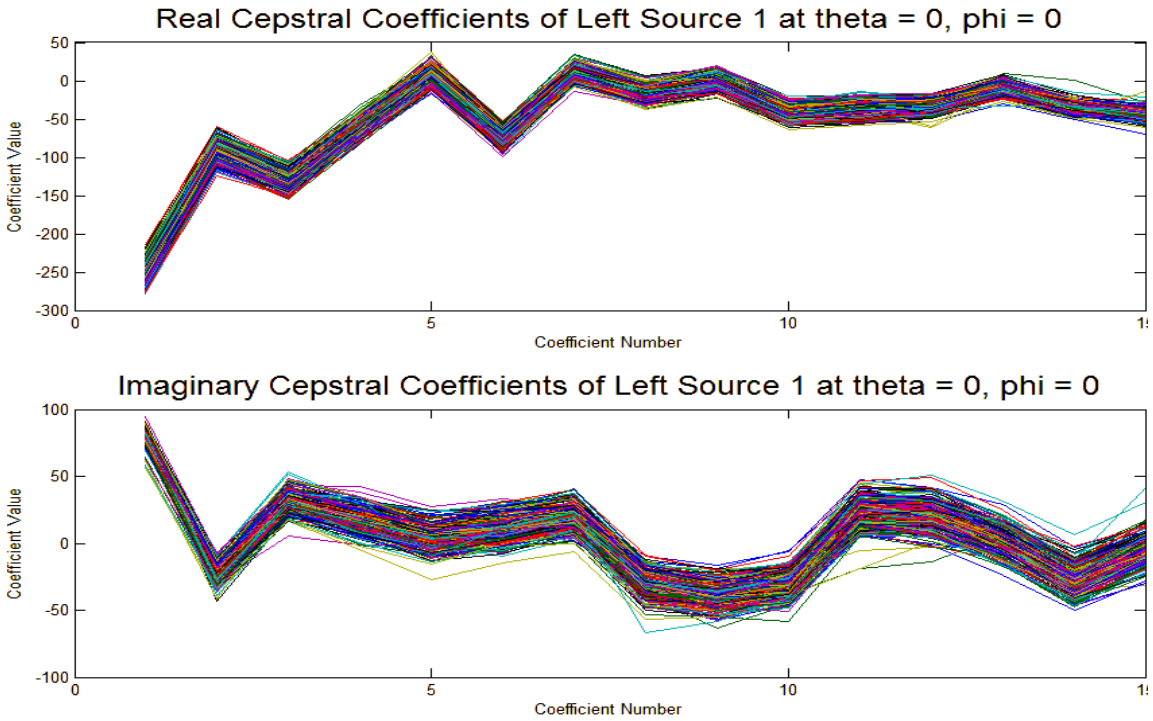


Figure 7: First 15 Cepstral Coefficients at a given source location (250 samples).

These values are then averaged together to form one template for this position:

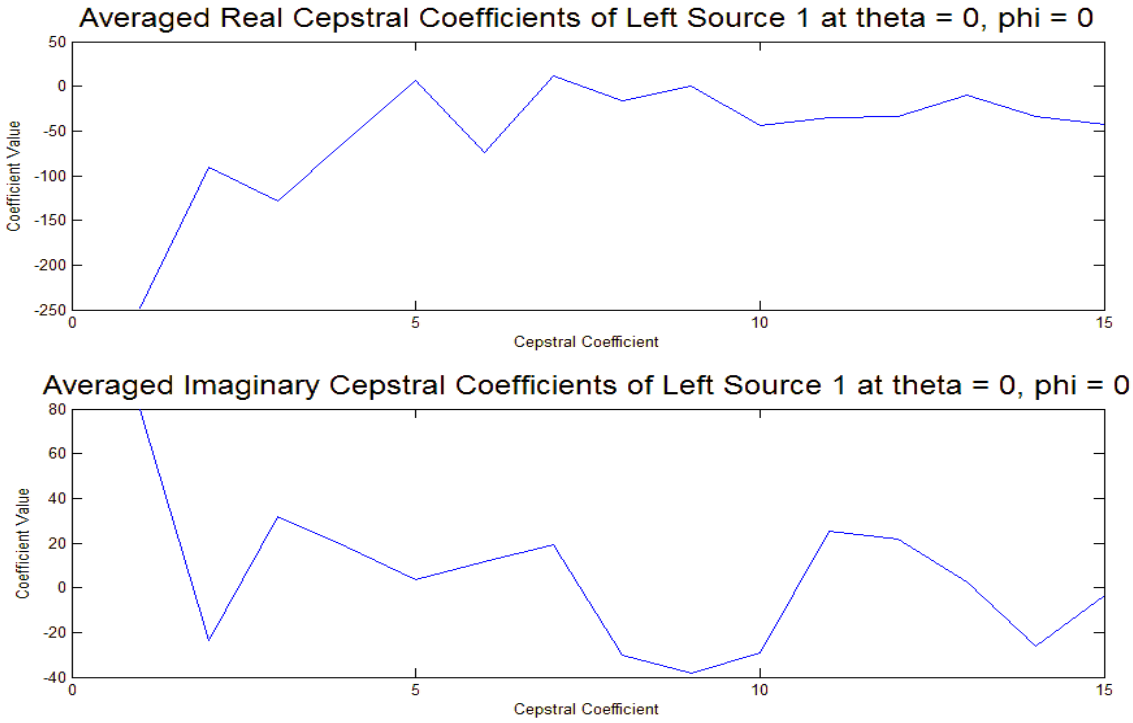


Figure 8: Mean of Cepstral Coefficients at a given source location.

It is not by accident that only the first 15 coefficients are displayed. In fact, the very first coefficient has been omitted. The 0^{th} coefficient signifies the D.C. offset of the transfer function and thus can only introduce error. The higher coefficients are ignored as they represent the amount of successively higher frequency components present in the transfer function shape. These 30 numbers (15 real, 15 imaginary) are sufficient to represent the overall shape of the transfer function.

For each position in space (azimuth, elevation), there are four sources and thus eight templates total (four for the left, four for the right). Three azimuths were chosen: –45, 0 and +45 degrees. Sixteen elevations were sampled at 0 degrees, and 13 for both ±45˚. In the final verification of position program, **verify5.m** (see Appendix A), the same process outlined above is repeated for a single acquired sample. The first 15 complex Cepstral Coefficients (not counting the D.C. offset) are then compared to the first 15 Cepstral Coefficients on all of the templates. The minimum of the sum of errors between the new sample and the templates is then found. The following figure shows a typical graph of errors at different elevations for a single acquired sample:

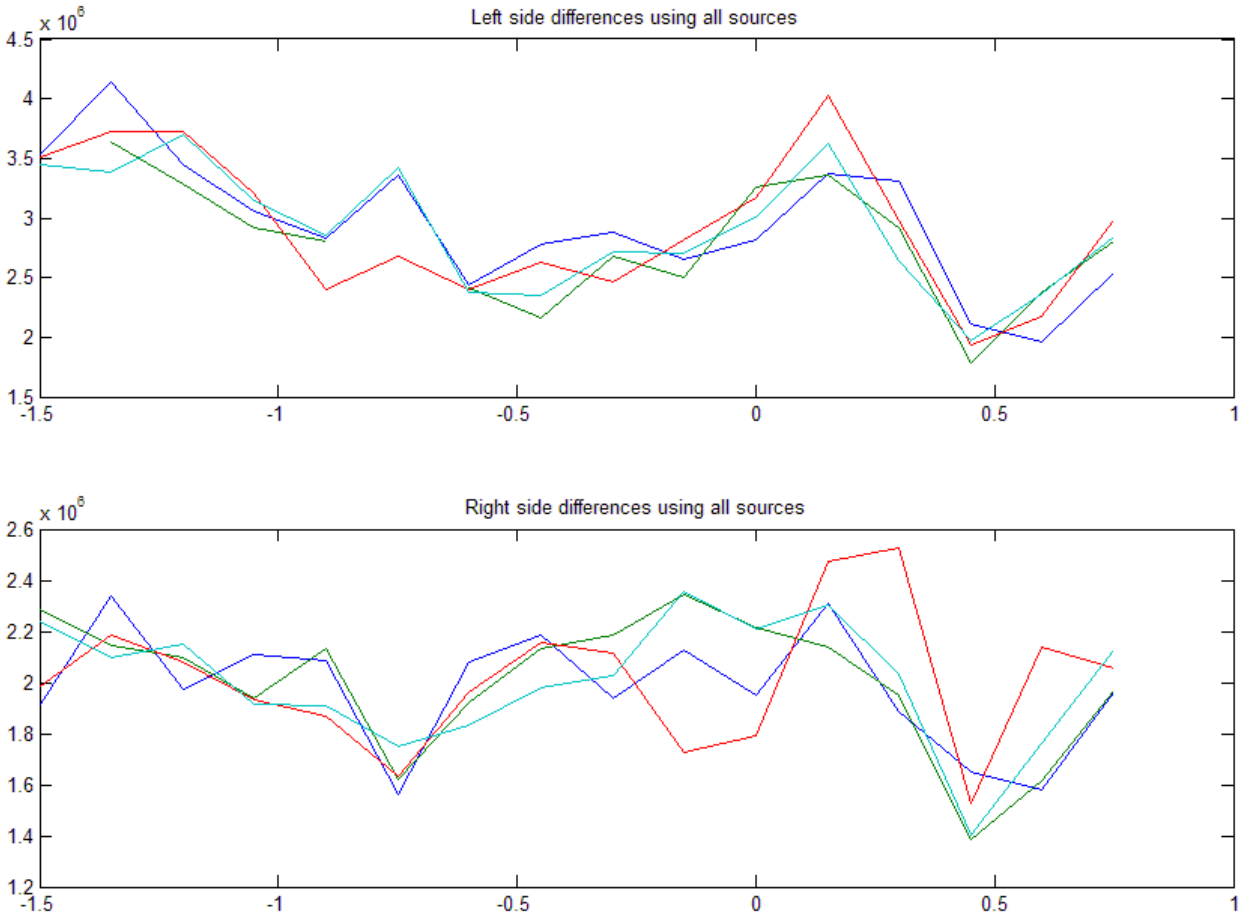


Figure 9: Sum of errors between acquired Cepstral Data and Cepstral Templates at various elevations. Note that the absolute minimum occurred in source 3 (dark green line) on from the right differences at 0.45 radians elevations.

5. RESULTS

Figure 10 shows the standard deviation of errors in the expected vs. measured delta (azimuth) and phi (elevation) at multiple positions along the horizontal plane, zero radians elevation. As expected, the angles nearest the center (zero degrees azimuth) represent the area with least error.

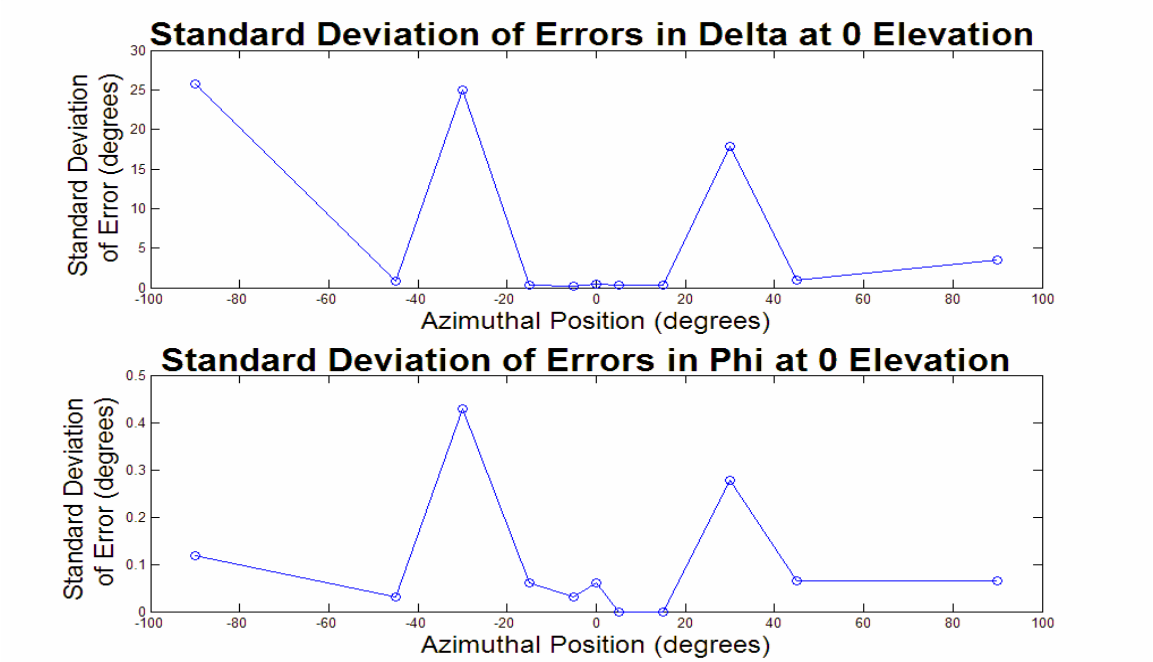


Figure 10: A measure of the error of azimuth (delta) and elevation (phi) at various points at 0 radians elevation.

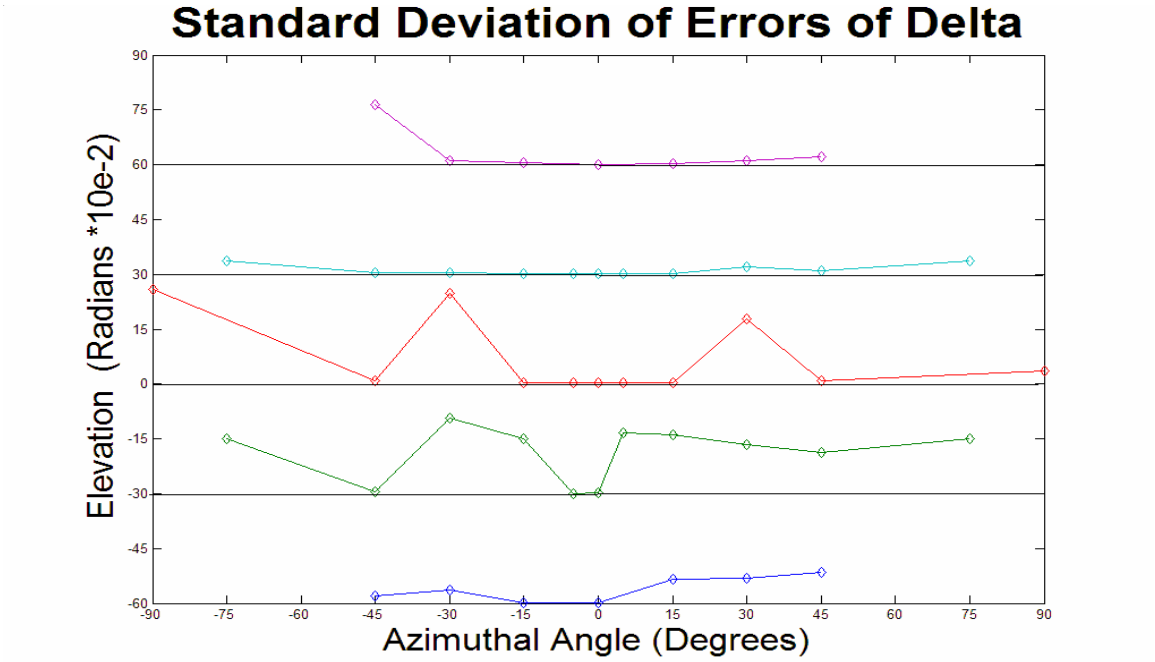


Figure 11: Standard deviation of errors of delta (azimuth) along various positions and five separate elevations ranging from –0.6 radians to 0.6 radians.

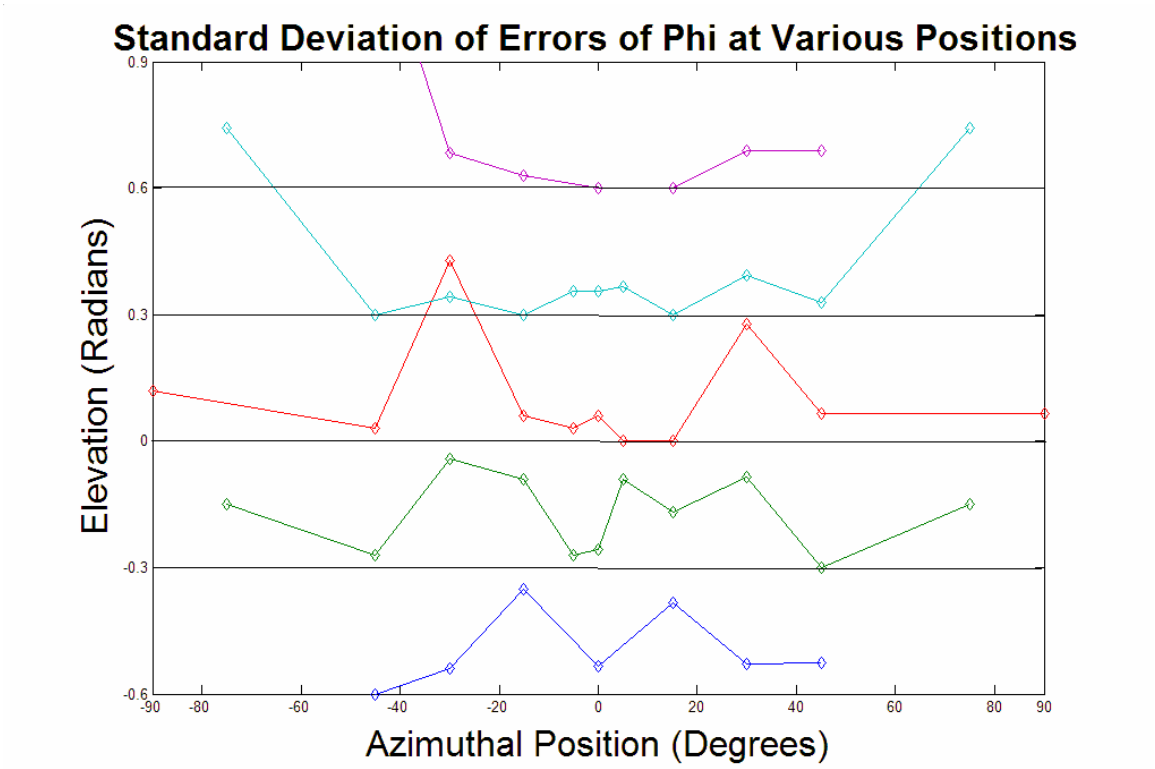


Figure 12: Standard deviation of errors of phi (elevation) at various positions in space along five chosen elevations ranging from –0.6 to 0.6 radians.

Precise values of standard deviation of errors can be found in Appendix B.

6. DISCUSSION AND CONCLUSIONS

A number of non-idealities and limiting factors contribute to the imperfections in the system. One major pitfall is the low sample rate, capped at 16 kHz. Though this seems reasonable, the Nyquist Rate specifies that the spectrum can only accurately range up to half the sampling rate, or 8 kHz. [8] This is sufficient to provide enough variation in the transfer function to distinguish various elevations, but as stated earlier, most spectral information pertaining to physical parameters occurs within the range of 5-18 kHz.

When the HRTF of the unmodified robot dog was first analyzed, it was found that there was insufficient variation in the transfer functions at different elevations to allow for positive matching against a template. Thus the microphones were placed under the ears to act as an artificial pinna. Given more time, different microphone positions could have been tried, and perhaps an outer ear could have been built around them to create more variations in the HRTF that depend on elevation. Another interesting idea might be to have one ear pointing up and the other pointing down. This asymmetry is used by certain animals in nature- [2] and might increase the confidence level of the result of a given elevation.

The greatest weakness of the final algorithm is that, in order to correct for position on the sphere, the accuracy of the angle of elevation is as crucial as the time delay. The greater the error in either, the more skewed the calculated position. This is most troublesome in the case of the elevation. As is apparent from **Figure 12**, the error of phi is much greater in the lower elevations. This is probably due to the fact that the robot dog's ears are pointed down, so there is no diffraction of sound around the ear when the sound emanates from below. The best solution may be to build a custom outer ear with folds and sharp corners in order to ensure marked variations in the HRTF at various elevations.

Another significant issue is that there are only three sets of Cepstral templates at -45, 0 and +45 degrees azimuth. This is the main reason the errors for phi are so high at ±30 degrees. Adding more templates would solve this issue.

Although a number of improvements can be made to this program, the robot dog is able to accurately localize a white noise sound source at most points in the front hemisphere. Any additions to the project would be refinements to improve precision.

7. ACKNOWLEDGMENTS

I would like to thank Professor Dan Lee of the University of Pennsylvania for his encouragement, support, and supervision. In addition, Yuan-Qing Li of the University of Pennsylvania provided invaluable ideas and comments. I would also like to thank the National Science Foundation for their support through an NSF-REU grant that made this program possible.

8. REFERENCES

1. B. Moore, *An Introduction to the Psychology of Hearing,* Vol 1., Academic Press, New York. 4th Ed. 1997, p. 210-230.

2. E. Milios, and G. Reid, Active Stereo Sound Localization. *Technical Report CS-1999-09,* Department of Computer Science, York University, p. 1-26.

3. E. Ben-Reuven, and Y. Singer, Discriminative Binaural Sound Localization. Hebrew University, Jerusalem, Israel. 1999.

4. Sony Aibo website. [electronic] Available: http://www.us.aibo.com/

5. M. Selik, and R. Baraniuk, Properties of the Fourier Transform. The Connexions Project, Rice University, Houston, TX. 2003.

6. Page: 139
Y. Lin, Paper on Fourier Cross-Correlation Methods, Unpublished manuscript, Dept. of Electrical and Systems Engineering, University of Pennsylvania, 2003.

7. P. Zakarauskas, and M. S. Cynader, A computational Theory of Spectral Cue Localization. *Journal of the Acoustical Society of America,* vol. 94 (1993) 1323-1330.

8. B. A. Carlson, *Communication Systems*, McGraw-Hill Companies, New York, 4th Ed. 2001. p. 37.

9. J. Hung, Sound Localization in Reverberant Environment Based on the Model of the Precedence Effect, *IEEE Transactions on Instrumentation and Measurement*. vol. 46, no 4 (1997). p. 842-846.

9. APPENDICES

Appendix A: Matlab Code

```
%  verify5.m
%  dog finds elevation of speaker

close all;

%  load templates for cepstral data at -45, 0 and 45 degrees azimuth
load ceptemplate.mat;
load ceptemplateN45.mat;
load ceptemplateP45.mat;
SetDefaultRobot(2);
Effector(15);

%  choose which white noise source to try to find.
source = input('Chooses source; enter 1, 2, 3, or me: ','s');
if source == '1'
   ss = load('C:\MATLAB6p5\work\ss.dat');
   ss1 = load('C:\MATLAB6p5\work\ss.dat');
end
if source == '2'
   ss = load('C:\MATLAB6p5\work\s1.dat');
   ss1 = load('C:\MATLAB6p5\work\s1.dat');
end
if source == '3'
   ss = load('C:\MATLAB6p5\work\s2.dat');
   ss1 = load('C:\MATLAB6p5\work\s2.dat');
end
if source == 'me'
   ss = load('C:\MATLAB6p5\work\ssme.dat');
   ss1 = load('C:\MATLAB6p5\work\ssme.dat');
end

ss1 = [ss1' zeros(1,1536)]';
ss1fft = fft(ss1);
pause(1);

%  shows you a picture of what the dog sees before proceeding
incorrect = 'y';
while incorrect == 'y'
   Effector([0 0 0 0]');   %prep dog position
   pause(1);
   h=figure(1);
```

```
   rb_ima=YUVRead;
   image(rb_ima);            %gives you a different picture for every new head position
   hold on;
   plot(88,66,'r.')          %red dot at center of dog's view
   axis off;
   incorrect = input('readjust? (y/n): ','s');
   pause(1);
end
close all;

% set infinite loop to keep finding speaker until interrupted by user
while 1
close all
Effector([0 0 0 0]');        %set dog's head to standard position
Effector([0 0]')             %make sure ears are down
clear check;
earstr = 'down';
pause(2);

[y, userdata] = MicRead;
Effector(15);
micPort = 5002;
xlave = 2;
xrave = 2;
% set while loop to catch white noise sound
rdelta = 2000;
ldelta = 2000;
dt = 1;
% keep taking data points until the onset of the sound source is within the
% 2048 point-long recorded window.  Also make sure the time difference does
% not exceed physical limitations to ensure that the data is reliable.

while rdelta > 1535 | ldelta > 1535 | abs(ldelta - rdelta) > 10 | abs(dt) > 2.5e-4
   close all;
   xlave1 = 1000;
   xlave2 = 0;
   xrave1 = 0;
   xrave2 = 0;
   xlave = 1;
   xrave = 1;
   while xlave < 200 | xrave < 200 | xlave1 > 500 | xlave2 > 600 | xrave1 > 500 | xrave2 >
600  %set loop to make sure sound is captured correctly
      soundsc(ss,16000);         %play white noise, fs = 16000
      pause(.0625);              %delay to give computer time to play noise
      x = double(micread);       %dog acquires data from both microphones
      subplot(2,2,1), plot(x(:,1));
```

```
        axis([0 2048 -4000 4000]);
        subplot(2,2,2), plot(x(:,2));
        axis([0 2048 -4000 4000]);
        xlave = mean(abs(x(:,1)));  %these are the thresholds to make sure the sound is
within the window
        xrave = mean(abs(x(:,2)));
        xlave1 = mean(abs(x(1:150,1)));
        xlave2 = mean(abs(x(1900:2048,1)));
        xrave1 = mean(abs(x(1:150,2)));
        xrave2 = mean(abs(x(1900:2048,2)));
    end
    yl = x(:,1);
    yr = x(:,2);
    xlfft = fft(yl);  %take fourier transform of acquired time-data
    xrfft = fft(yr);
    Clf = xlfft.*conj(ss1fft);  %multiply by the complex conjugate of the source
    Crf = xrfft.*conj(ss1fft);
    clt = ifft(Clf);            %take the inverse fourier transform of the above product
    crt = ifft(Crf);
    lhighest = max(clt);        %determine number of shifted data points
    rhighest = max(crt);
    ldelta = find(clt==lhighest);
    rdelta = find(crt==rhighest);
    %use the program RefinedCC.m to get a much more accurate time delay (steps through
small time differences in the freq domain)
    [c,peakValue,dt]=RefinedCC(x(:,1),x(:,2),16000,(-10:10)/16000);
    dt
end

%take the data we want (first 512 data points of recorded sound- removes echo) and shift
back to 0
for s = 0:511
    shiftxl(s+1,1) = yl(ldelta + s,1);
    shiftxr(s+1,1) = yr(rdelta + s,1);
end
shiftedxl = [shiftxl' zeros(1,1536)]';
shiftedxr = [shiftxr' zeros(1,1536)]';
close all;

%acquired data processing here
    xlfft = fft(shiftedxl);
    xrfft = fft(shiftedxr);
    xltf = xlfft(1:1024,1)./ss1fft(1:1024,1);   %get transfer function from just acquired
sample
    xrtf = xrfft(1:1024,1)./ss1fft(1:1024,1);
    clear shiftedxl shiftedxr
```

```
   cepltf = fft(log(abs(xltf)));              %calculate cepstral data
   ceprtf = fft(log(abs(xrtf)));
   dt

%converting dt to an angle
   theta = -asin(dt*344/.085);
   thetadeg = theta*180/pi
   hyp = sqrt(1000^2+1000^2);

%choose which template to compare
if thetadeg < -20 %choose right template
   thetashift = theta + pi/2;
   dcepdifferencesN45;  % finds sum of square of differences of first derivative of all
cepstral data
   xval = [-1.2 : 0.15 : .60];

   subplot(2,1,1), plot(xval,[ldiffs(1,:); ldiffs(2,:)+50; ldiffs(3,:)+100; ldiffs(4,:)+150]);
   title('Left side differences using all sources; N45-Template');
   subplot(2,1,2), plot(xval,[rdiffs(1,:); rdiffs(2,:)+50; rdiffs(3,:)+100; rdiffs(4,:)+150]);
   title('Right side differences using all sources');

   [rl,cl] = find(ldiffs==min(min(ldiffs)));
   [rr,cr] = find(rdiffs==min(min(rdiffs)));
   figure;

   %choose left or right side with lowest error value
   if ldiffs(rl,cl) < rdiffs(rr,cr)
      elevation = xval(cl)
            %calculating delta, actual horizontal turning angle required
            delta = asin(sqrt((hyp*sin(thetashift))^2 - (hyp*sin(elevation))^2) / sqrt(hyp^2 -
(hyp*sin(elevation))^2));
      xd = cos(delta)*hyp;
      yd = sin(delta)*hyp;
      zd = tan(elevation)*hyp;
      %effector([qangle 0 0 0]');
      PointHead([xd yd zd]);
      pause(1)
      k=figure(2);
      rb_ima=YUVRead;
      image(rb_ima);            %gives you a different picture for every new head position
      hold on;
      plot(88,66,'r.')          %red dot at center of dog's view
      axis off;
   else
      elevation = xval(cr)
         %calculating delta, actual horizontal turning angle required
```

```
        delta = asin(sqrt((hyp*sin(thetashift))^2 - (hyp*sin(elevation))^2) / sqrt(hyp^2 -
(hyp*sin(elevation))^2));
      xd = cos(delta)*hyp;
      yd = sin(delta)*hyp;
      zd = tan(elevation)*hyp;
      %effector([qangle 0 0 0]');
      PointHead([xd yd zd]);
      pause(1)
      k=figure(2);
      rb_ima=YUVRead;
      image(rb_ima);            %gives you a different picture for every new head position
      hold on;
      plot(88,66,'r.')          %red dot at center of dog's view
      axis off;
   end
end

if thetadeg > 20 %choose left template
   thetashift = abs(theta - pi/2);
   dcepdifferencesP45;  % finds sum of square of differences of first derivative of all
cepstral data
   xval = [-1.2 : 0.15 : .60];

   subplot(2,1,1), plot(xval,[ldiffs(1,:); ldiffs(2,:)+50; ldiffs(3,:)+100; ldiffs(4,:)+150]);
   title('Left side differences using all sources; P45-Template');
   subplot(2,1,2), plot(xval,[rdiffs(1,:); rdiffs(2,:)+50; rdiffs(3,:)+100; rdiffs(4,:)+150]);
   title('Right side differences using all sources');

   [rl,cl] = find(ldiffs==min(min(ldiffs)));
   [rr,cr] = find(rdiffs==min(min(rdiffs)));
   figure;
   %choose left or right side with lowest error value
   if ldiffs(rl,cl) < rdiffs(rr,cr)
      elevation = xval(cl)
         %calculating delta, actual horizontal turning angle required
         delta = asin(sqrt((hyp*sin(thetashift))^2 - (hyp*sin(elevation))^2) / sqrt(hyp^2 -
(hyp*sin(elevation))^2));
      xd = -cos(delta)*hyp;
      yd = sin(delta)*hyp;
      zd = tan(elevation)*hyp;
      %effector([qangle 0 0 0]');
      PointHead([xd yd zd]);
      pause(1)
      k=figure(2);
      rb_ima=YUVRead;
      image(rb_ima);            %gives you a different picture for every new head position
```

```
      hold on;
      plot(88,66,'r.')          %red dot at center of dog's view
      axis off;
   else
      elevation = xval(cr)
         %calculating delta, actual horizontal turning angle required
         delta = asin(sqrt((hyp*sin(thetashift))^2 - (hyp*sin(elevation))^2) / sqrt(hyp^2 -
(hyp*sin(elevation))^2));
      xd = -cos(delta)*hyp;
      yd = sin(delta)*hyp;
      zd = tan(elevation)*hyp;
      %effector([qangle 0 0 0]');
      PointHead([xd yd zd]);
      pause(1)
      k=figure(2);
      rb_ima=YUVRead;
      image(rb_ima);            %gives you a different picture for every new head position
      hold on;
      plot(88,66,'r.')          %red dot at center of dog's view
      axis off;
   end
end

if abs(thetadeg) < 20 %choose center template
   if thetadeg < 0
      thetashift = theta + pi/2;
   else
      thetashift = abs(theta - pi/2);
   end

%  sums the derivatives of differences
dcepdifferences;  % finds sum of square of differences of first derivative of all cepstral
data

figure
xval = [-1.5:0.15:0.75];
subplot(2,1,1), plot(xval,[ldiffs(1,:); ldiffs(2,:)+50; ldiffs(3,:)+100; ldiffs(4,:)+150]);
title('Left side differences using all sources');
subplot(2,1,2), plot(xval,[rdiffs(1,:); rdiffs(2,:)+50; rdiffs(3,:)+100; rdiffs(4,:)+150]);
title('Right side differences using all sources');

[rl,cl] = find(ldiffs==min(min(ldiffs)));
[rr,cr] = find(rdiffs==min(min(rdiffs)));
figure;
%choose left or right side with lowest error value
if ldiffs(rl,cl) < rdiffs(rr,cr)
```

```
   elevation = xval(cl)
   %effector([qangle 0 0 0]');
         %calculating delta, actual horizontal turning angle required
         delta = asin(sqrt((hyp*sin(thetashift))^2 - (hyp*sin(elevation))^2) / sqrt(hyp^2 -
(hyp*sin(elevation))^2));
   if thetadeg > 0
      xd = -cos(delta)*hyp;
   else
      xd = cos(delta)*hyp;
   end
   yd = sin(delta)*hyp;
   zd = tan(elevation)*hyp;
   Pointhead([xd yd zd])
   pause(1)
   k=figure(2);
   rb_ima=YUVRead;
   image(rb_ima);            %gives you a different picture for every new head position
   hold on;
   plot(88,66,'r.')          %red dot at center of dog's view
   axis off;
else
   elevation = xval(cr)
   %effector([qangle 0 0 0]');
         %calculating delta, actual horizontal turning angle required
         delta = asin(sqrt((hyp*sin(thetashift))^2 - (hyp*sin(elevation))^2) / sqrt(hyp^2 -
(hyp*sin(elevation))^2));
   if thetadeg > 0
      xd = -cos(delta)*hyp;
   else
      xd = cos(delta)*hyp;
   end
   yd = sin(delta)*hyp;
   zd = tan(elevation)*hyp;
   Pointhead([xd yd zd])
   pause(1)
   k=figure(2);
   rb_ima=YUVRead;
   image(rb_ima);            %gives you a different picture for every new head position
   hold on;
   plot(88,66,'r.')          %red dot at center of dog's view
   axis off;
end
end
pause(2.5)
end
```

Appendix B: Standard Deviation of Errors of Delta (azimuth) and Phi (elevation)
(azimuthal angles are the first row)

δ

Standard Deviation of Errors of Delta at -0.6 radians elevation

-45	-30	-15	0	15	30	45
1.9404	3.5187	0.19015	0.20193	6.4555	6.8373	8.5096

Standard Deviation of Errors of Delta at -0.3 radians elevation

-75	-45	-30	-15	-5	0	5	15	30	45	75
15	0.46391	20.698	14.897	0.08409	0.16593	16.616	16.166	13.329	11.272	15

Standard Deviation of Errors of Delta at 0 radians elevation

-90	-45	-30	-15	-5	0	5	15	30	45	90
25.804	0.806	24.928	0.30239	0.24028	0.39559	0.322	0.38098	17.904	0.91274	3.5024

Standard Deviation of Errors of Delta at 0.3 radians elevation

-75	-45	-30	-15	-5	0	5	15	30	45	75
3.5842	0.40616	0.34686	0.12882	0.11271	0.17792	0.28313	0.1042	2.1068	0.96298	3.5842

Standard Deviation of Errors of Delta at 0.6 radians elevation

-45	-30	-15	0	15	30	45
16.406	1.1356	0.46876	0.10966	0.39	1.0335	2.1171

φ

Standard Deviation of Errors of Phi at -0.6 radians elevation

-45	-30	-15	0	15	30	45
0	0.06	0.24875	0.065955	0.21599	0.071414	0.073485

Standard Deviation of Errors of Phi at -0.3 radians elevation

-75	-45	-30	-15	-5	0	5	15	30	45	75
0.15	0.03	0.25807	0.21	0.03	0.043301	0.21	0.13191	0.21442	0	0.15

Standard Deviation of Errors of Phi at 0 radians elevation

-90	-45	-30	-15	-5	0	5	15	30	45	90
0.11874	0.03	0.42927	0.06	0.03	0.061237	0	0	0.27699	0	0.065383

Standard Deviation of Errors of Phi at 0.3 radians elevation

-75	-45	-30	-15	-5	0	5	15	30	45	75
0.44396	0	0.041533	0	0.056125	0.056125	0.065383	0	0.094074	0	0.44396

Standard Deviation of Errors of Phi at 0.6 radians elevation

-45	-30	-15	0	15	30	45
0.63273	0.083066	0.03	0	0	0.09	0.09

BUILDING A PROTOTYPE OF A 3-D DISTRIBUTED MOBILE SENSOR NETWORK

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

ABSTRACT

Biological organisms have the ability to gather information over a variety of senses from multiple viewpoints for accurate sensory perception. Artificial systems typically use a static array of sensors that lack mobility, or employ motion in a robotic platform in two dimensions.

The goal of this project is to design and build a prototype of an adaptive, distributed sensor network utilizing small, modular sensors and actuating components that will accurately position sensors in three-dimensional space. A working prototype was built, using a simple motorized spool design and a Motorola HC11 microcontroller. The system monitors the current position in three dimensions using encoders. Based on the change of lengths of the network cables, the motors are set to spin at speeds that will achieve motion in the desired direction. Control of the system was implemented in C to position an adaptive network in three dimensions. A user interface using serial communication and infrared remote control was designed.

Table of Contents

1. INTRODUCTION

In nature, biological organisms use multiple viewpoints to accurately perceive their 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]. Visually, multiple cameras are used to create binocular viewing systems to allow stereo depth perception.

However, a number of constraints limit the ability to gather information from multiple viewpoints. If an object is not placed exactly in the designated field that is being sensed, the information gathered can be spurious and inaccurate. Mobile sensor arrays have limitations of being constrained to two dimensions, and are usually wheeled. A wheeled sensor network, such as a robot, can easily disturb the environment it is trying to monitor. for example, a wheeled robot will disturb an animal, and the information gathered would not necessarily be accurate.

To replace static sensor arrays and wheeled robots, we propose to build an adaptive distributed sensor network that can be accurately positioned in three dimensions to gather information from multiple viewpoints. The system will consist of small, modular sensors and actuating components that will make the sensor network mobile. The sensor network can contain visual, audio, and olfactory sensors. The goal of the project is to build a prototype of an adaptive mobile network to assess the feasibility of such a system.

2. BACKGROUND

2.1 Biological Motivation

Being able to move provides a variety of viewpoints that can be used to perceive and make decisions about an area. The dependence of an observer's perception on his or her viewpoint can be illustrated with the Ames room illusion. Figure 1 shows the actual construction of the Ames room. If an observer looks through the viewing hole, limited to a single viewpoint, a pair of identical twins standing on opposite sides of the room,will be perceived as 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 appeared different in size merely because they are at different distances from the observer [2]. This simple illustration demonstrates the role that an observer's location plays in sensory perception and highlights the importance of gathering information from multiple viewpoints for accurate perception.

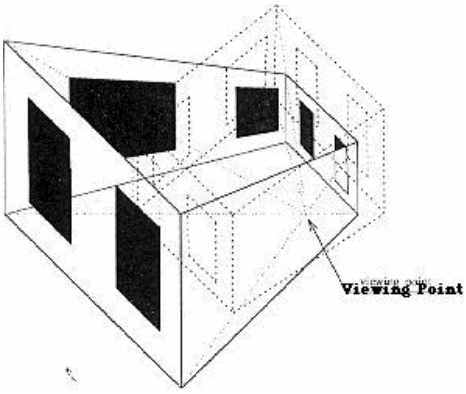


Figure 1: Actual shape and structure of an Ames room.

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

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 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 [3]. 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 Existing Sensor Network Projects

Many different sensor network research projects are currently being conducted. The Palms Fixed/Mobile Experiment at the University of California–Berkeley is trying to deploy a sensor network via unmanned aerial vehicles. This sensor network will be able to spread quickly across an area and deliver information from numerous viewpoints [4].

The Smart Dust project, also 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 information to a base station for processing [5].

2.3 Description of Proposed Prototype

The inspiration for the three-dimensional positioning of the system came from theatrical performances such as *Peter Pan* that incorporate "flying" actors and props that are suspended in the air by multiple cables.

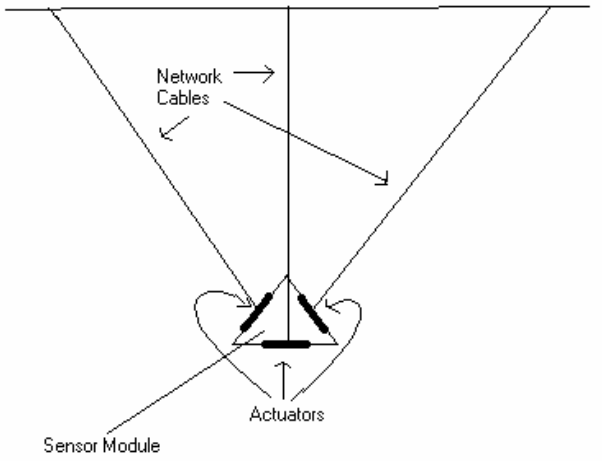


Figure 3: Sketch of Proposed Prototype.

The sensor module will have a number of thin cables connecting it to the ceiling that will provide mechanical support. Small actuators will be controlled by software to adjust the lengths of connected cables by winding or unwinding them. Changing the lengths of the supporting cables will enable the three-dimensional position of the sensor node to be quickly and freely adjusted. A diagram of the proposed system is shown in figure 3.

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

3. HARDWARE DESIGN

The goal for the summer was to develop an adaptive network that is able to move freely in three dimensions. By the end of the project, a system had been built consisting of three motors attached to a single platform that also acts as the sensor node. The three motors allow mobility in three dimensions constrained to a triangular area. Using the algorithm the system implements, it is straightforward to implement a fourth motor that will allow mobility across an entire room.

3.1 Hardware Description

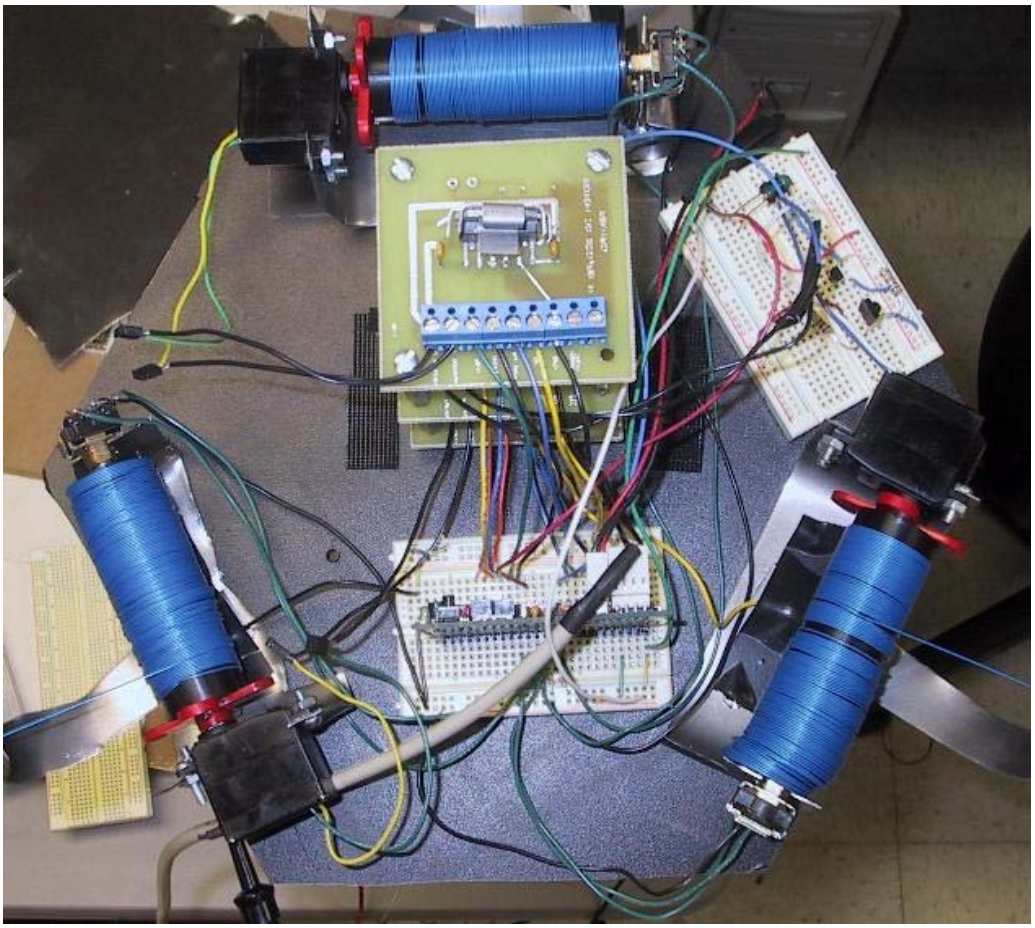


Figure 4: Fully assembled sensor node.

The spool system for controlling the lengths of the cables consists of an HS-225MG Hitec Mini Precision Servo that is modified to act as a geared DC motor, and custom-made plastic, aluminum and steel parts that were designed and machined. Connected to the ends of each spool is a CTS-288 two-bit rotary encoder that enables the location of the node to be accurately detected. To allow for the power source to be located off the sensor system, the network cables are 28-guage wire, which will trickle charge an onboard battery that powers the motors.

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 has 24K of external RAM and 32K of EEPROM. To interface a remote control, an infrared detector circuit was designed and connected to the microcontroller board. A lead-acid battery is used to provide power to the motors and is recharged through the 28-gauge wire, which is connected to a laboratory DC voltage source. On board is a battery pack containing 4 AA batteries that is used to power the

microcontroller and infrared detector circuitry. The completed prototype is shown in figure 4.

3.2 HC11 Setup and Infrared Receiver Circuit

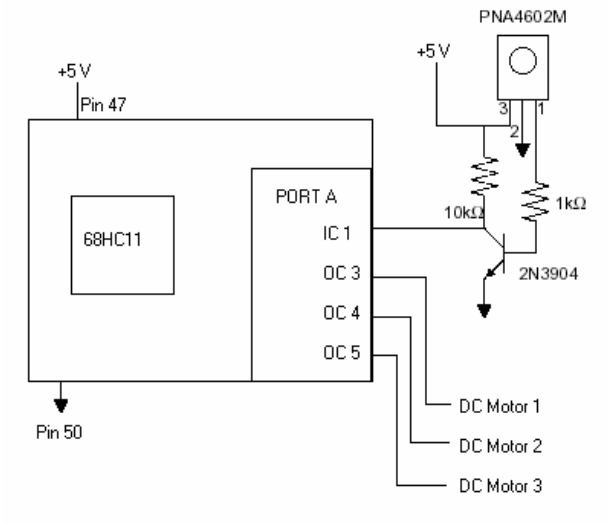


Figure 5: HC11 pin connections and infrared receiver circuit.

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 DC motors. The pins are connected to a National Semiconductor LMD18201 H-Bridge chip, which allows for an input signal of 5 volts and an output of 12 to 55 volts, as well as ease in changing direction and braking a DC motor. The H-bridges outputted 12 volts.

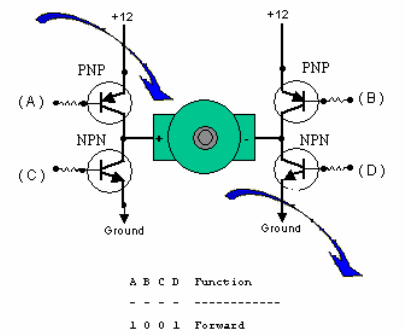
An infrared detector circuit, seen in figure 5 along with the pin connections for the 68HC11, consisting of a Panasonic PNA4602M IR detector chip and a 2N3904 transistor for signal amplification, was connected to the input capture pin of the HC11 Board to allow for detection and processing of infrared signals [6].

3.3 Modification of the HS-225MG Hitec Mini Precision Servo

A servo, 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. It has a limited range of motion, usually being unable to spin more than 180 degrees.

Inside the servomotor are gears connected to the DC motor to slow down the output spin and also allow for increased torque. The HS-225MG has a rated torque of 66.7 oz-in at 6 volts. High torque is necessary in this application to hold the sensor nodes firmly in place. The servo is also small, 1.25 x 1.25 x .75 inches, and lightweight. This is necessary as well because to allow easier mobility the servos will need to carry their own weight plus the weight of the sensors on board.. To allow use of the compact size and high torque, the servo was modified to act like a DC motor, which will enable continuous rotation and speed control. Two holes were punched into the back of the servomotor. The onboard electronics were disconnected from the DC motor, and two wire leads were attached, thus enabling the DC motor to be controlled from outside the servo.

To control the DC motor in the servo, a pulse-width modulated (PWM) signal is required. To allow greater ease in controlling direction and voltage, an H-Bridge circuit is attached from the output compare pin of the HC11 chip to the DC motor. The H-bridge circuit allows for control of current through a DC motor. Turning on and off certain transistors controls the current to spin the motor in one direction or another. As is seen in figure 6 and 7, turning on and off certain transistors allows the motor to spin in opposite directions with only a single user controlled input. Putting the H-bridge in brake position turns off the current flowing through the motor and prevents the motor from spinning by maintaining the same voltage across the two nodes of a DC motor. As the motor tries to turn, it acts like a generator, producing its own voltage, but the H-bridge maintains the same voltage across the two nodes of the motor, turning the motor back to its original position and keeping it there [7].



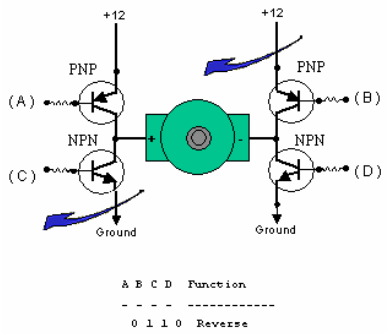


Figure 6: H-bridge flow diagram forward. Figure 7: H-bridge flow diagram backward.

The PWM signal is sent into the H-bridge at 5 volts (the output signal of the HC11) and changed to the equivalent signal at 12 volts (user-decided voltage of 12 to 55 volts). The PWM signal is high for a set amount of time, and then goes low. This repeats itself every 7.3 milliseconds. By setting the amount of time high, the user can control how fast the motor spins. The amount of time set high, which is called the percent duty cycle, is a certain percentage of the total time.A PWM signal is shown in figure 8. The motor spins faster with a high percent duty cycle, as it gets a high voltage across it for a longer time.

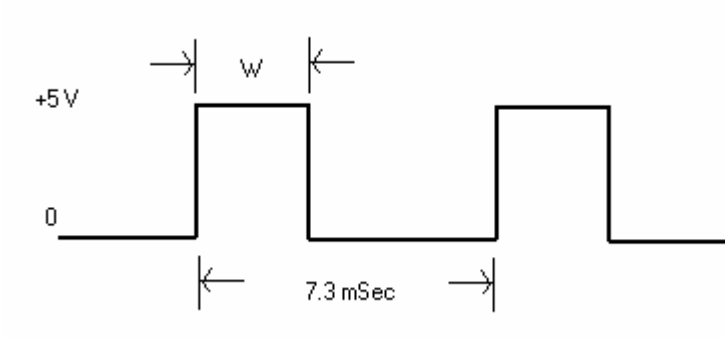


Figure 8: A Pulse-width modulated (PWM) signal.

The servo itself allows for rotation of only 180 degrees. To allow for greater rotation, the mechanical stopper must be removed using pliers. On the bottom of the output shaft, another stopper is also removed, as shown in figure 9. The rest of the circuitry is then cut from the DC motor, and two leads are soldered on, as shown in figure 10. Two holes are drilled into the back of the servo case to allow access to the leads [8].

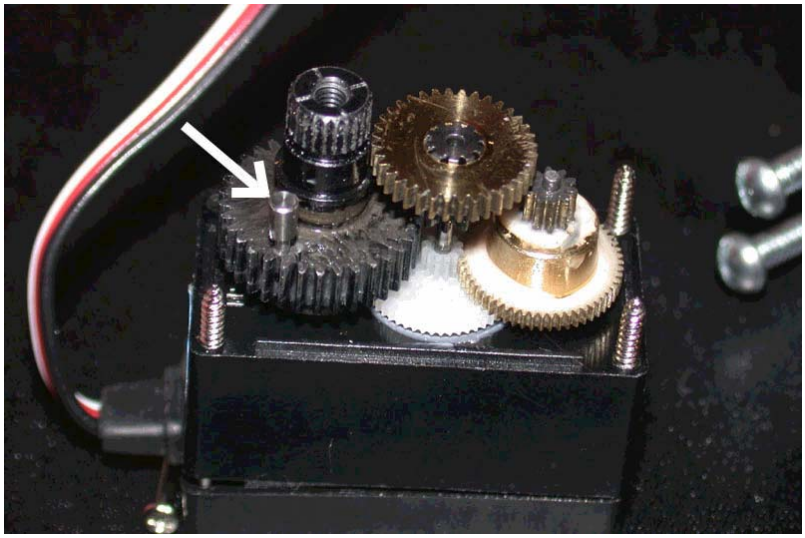


Figure 9: Modified servomotor gears.

Figure 10: DC motor leads.

3.4 Design of Mechanical Spool System

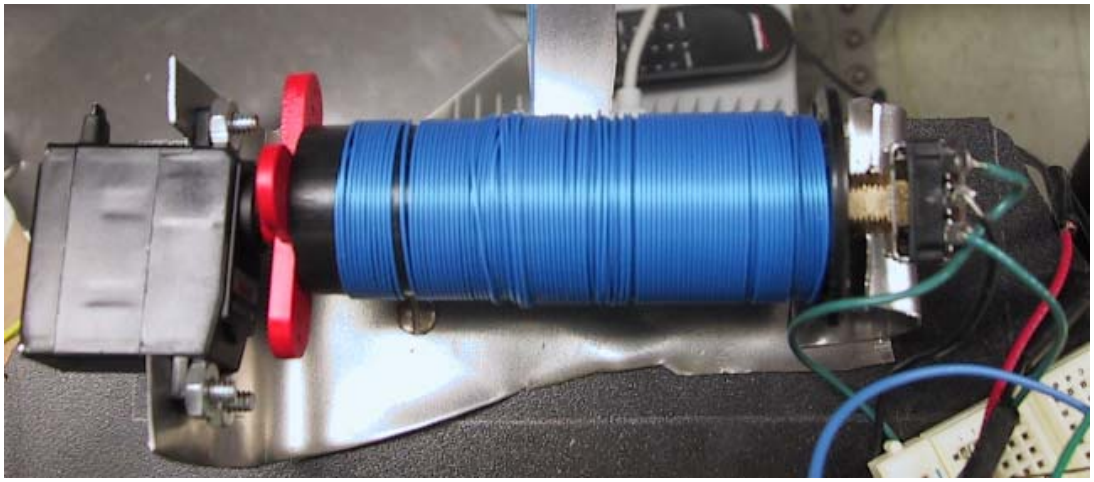


Figure 11: Mechanical spool.

The actual spool is a simple piece of cylindrical plastic, three inches long and one inch in diameter, that is screwed into the servo motor shaft. On the other end, a CTS-288 encoder is held firmly in place with epoxy, so it will spin with the motor. The encoder has two channels and a common pin. A 5-volt source is connected to the common pin. The two channels have a signal that goes high and low four times per revolution of the motor. The "A" channel leads the "B" channel so direction can be discerned. Since one channel leads the other, there are a total of 16 positions per revolution [9]. The timing diagram for the two channels are shown in figure 12. The outputs of the two channels are grounded and directed into the HC11. The encoder shaft is made of solid aluminum. A screw is drilled into the end going into the spool to enable a wire to be attached. The wire is then connected to the 28-gauge wire connected to the power source that charges the battery. The other end of the encoder is held by a piece of steel that also holds the motor for support. The steel, since it is connected to the other end of the encoder, has a connection to the voltage source and has a wire running to the battery, which trickle charges it. The entire spool with motor, encoder, wire, and steel holder are shown in figure 11.

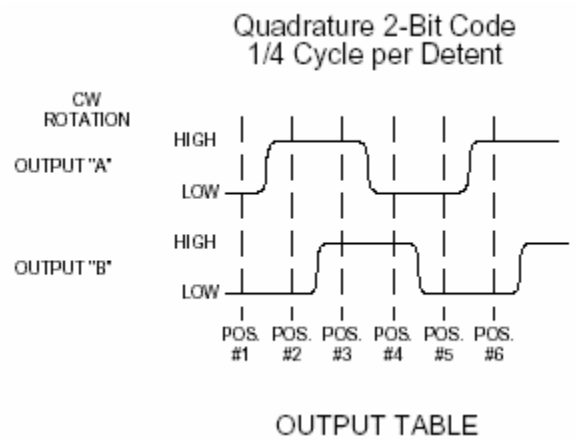


Figure 12: Encoder channel diagram.

3.5 Design of Sensor Node

The first prototype of the system, shown in figure 13, had three motors mounted on the ceiling and had a weight as the sensor node. This system allowed ease of testing because of the lightweight and simpler design.

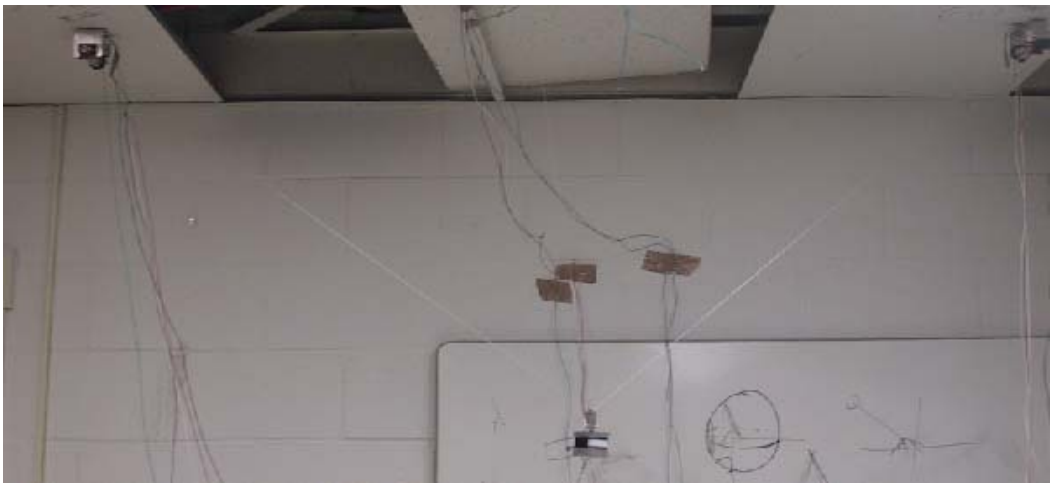


Figure 13: Original prototype.

The next system (see Figure 4) took the three motors onto a single platform that acts as the sensor node. This allows the system to adapt easily, requiring attachment to three hooks, instead of needing to mount motors on the ceiling. The node itself is a hexagon-shaped sheet of plastic board, with each side measuring six inches. The motor spools and encoder are placed on three edges to form an equilateral triangle. In the center is the 68HC11 and the three H-bridges, each H-bridge controlling a single motor. On another edge is the remote control circuitry. Each spool has 28-guage wire wound around it, which is used to deliver power to trickle charge the lead-acid battery on board. The wire is then pulled and connected to hooks in the ceiling. The individual wires are called

Side A, Side B, and Side C, respectively, throughout the remainder of this report. The battery pack for the 68HC11 and remote control circuitry is located next to the H-bridges, and the lead-acid battery is not placed on the board, because it is too heavy, but would ideally be placed on the other side of the H-bridge.

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. The Imagecraft ICC11 compiler was also used to load the program onto the 68HC11. The code consists of three major sections: code to decode user input, either via serial interface or remote control interface; code to find the position of the node; and code to control the DC motors.

4.2 User Input Decoding

There are two forms of user input: serial interface and remote control. The serial interface is used to communicate with the 68HC11 over a serial cable that allows the user to enter commands on the computer. The remote control communicates directly to the 68HC11 using infrared signals.

When the program first boots, the serial interface asks for the dimensions of the system. A base assumption is that the three points on the ceiling to which the wire connects forms an isosceles triangle, which is the most stable system to hold three sides. Each of the string lengths of the initial system must be entered, and the equal sides of the isosceles triangle following the non-equal side of the isosceles triangle are entered. Using this information, the system calculates the x, y, and z coordinates of the system. The x, y, and z coordinates are then outputted to the screen, and the user can then use either the remote control or the serial interface to move the system. With the serial interface, the user can then input the new x, y, and z coordinates he or she would like the system to go to. After the system has moved, the system calculates the new x, y and z coordinates. The coordinates are discussed further in section 3t.

For easy control of the system, remote control functionality was added. A Hauppauge remote control was used as an input device. Whenever a key is pressed, the remote control transmits two infrared pulse trains, each containing 12 pulses. The infrared 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 two time counter values is then calculated to get the width of each received pulse. Pulse widths larger than 3000 clock ticks are assigned a value of 1, while shorter pulse widths are assigned a value of 0. Pulse width assignments are shown in figure 14.

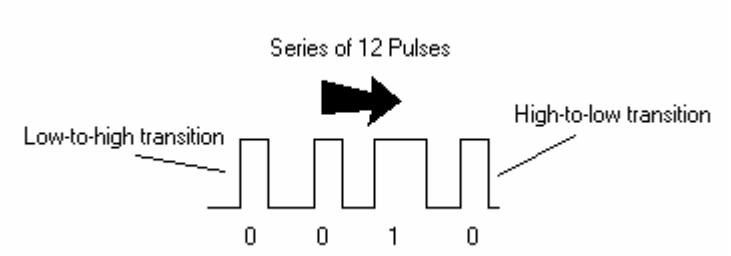


Figure 14: Decoding of pulse sequences.

Since each key of 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 x, y, and z coordinates of the system are established, the user enters on the remote control the change desired in each direction. The channel plus and minus buttons tell the system to respectively add or subtract in a certain orthogonal direction. To change the position of the node, the user enters plus or minus, followed by two digits for change in x, followed by the same for y and z. An LED turns on for half a second to let the user know that the system has received a command. After z has been entered, the node is moved to the desired position, and then the new x, y, and z coordinates are calculated.

4.3 Finding the Position of the Node

As noted in section 2, when the program is first started, it asks for the dimensions of the system and uses them to calculate the x, y, and z coordinates. The x, y, z plane used has the base of Side A on the ceiling as the origin, with positive x going toward Side B, positive y going toward Side C, and positive z going toward the floor. Since the dimensions of the isosceles triangle in the x, y plane are known, the x, y, and z coordinates can be delineated. Using each length of the wire as a radius of a sphere centered on each point of the isosceles triangle, the system can calculate the x, y, and z coordinates of the node by finding the intersection of the three spheres. After the system runs and moves the node to a new position, the new x, y, and z coordinates are calculated. The system uses the number of encoder ticks seen at each of the motors to find the new lengths of the wires. From these lengths, the x, y, and z coordinates are calculated.

4.4 Code to Control the DC Motors

When the user inputs the new location of the node, either through the serial interface or remote control, the program determines the speeds at which each motor needs to move to reach the new point. The program does this by calculating the new length of each wire and subtracting the old length from the new length. If the number is negative, then the direction is set for the motor to wind the string inwards, and vise versa. The program also takes the difference between the two lengths and determines how many encoder ticks will be necessary for an individual motor to have spun the correct amount. The program monitors the number of encoder ticks, and when it sees the correct amount

and the node has reached the correct point, the brake is turned on for the motors. After the difference in the lengths of string is determined, the ratio of the other two side differences to the maximum difference is found. The ratios are used to set the motors to spin at a speed corresponding to the ratios to achieve smooth motion to the user-defined destination.

To achieve the correct speed for the motors, two arrays are created: one for unwinding and one for winding. Two arrays are necessary because there is more torque required to wind the wire than to unwind it. Each number position of the array is the number of time units between encoder ticks. Stored in that position is the correct PWM size to give the respective time unit. With the use of the ratios, the correct PWM size can be quickly assigned to each motor. Testing was done on the motors to find the correct PWM size for each time. Precisely timed PWM signals are easily generated with the output compare function of the 68HC11 microcontroller. The programmable timer on the 68HC11 is used to produce high and low voltages on the output compare pins to generate appropriate pulse sequences to control the DC motors.

5. DISCUSSION AND CONCLUSIONS

The completed system contains three motors that enable the system to be positioned anywhere in three dimensions within the boundaries formed by the isosceles triangle to which the wire is connected. The system is adaptable to any room as long as the connections to which the wires hook are in the shape of an isosceles triangle. When different sensors are connected to the node then multiple viewpoints will be accessible, allowing for a more accurate perception of the environment.

Certain difficulties were encountered. One issue was weight. The motors are not designed to handle the amount of weight that the entire node is. Therefore the node has variable performance and does not achieve desired results on every trial. A related issue is the power supply. The lead-acid battery is too heavy to be placed on board the node to power the system, and the system cannot be powered directly off the 28-gauge wire because the resistance of the long, thin wire does not allow enough current to pass.

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. The system currently has trouble supporting its weight, therefore making reliable movement to a desired location difficult. When a large amount of tension is placed onto a motor, the motor begins to unwind. Using better motors can solve this problem. The motors used are servomotors with 66.7 oz-in torque. Using DC motors that can produce much more torque will allow the system to move more reliably.

The system currently uses three motors to move the node in three dimensions. This allows movement within a constrained triangular area. Since the movement of the motors is not reliable, a fourth motor that would have allowed a greater range of

movement was deemed unpractical. The use of more reliable motors, as discussed earlier, will allow a fourth motor to be interfaced reliably.

The system also does not have a great deal of feedback while in movement. The 68HC11 is not powerful enough to be able to watch for correct speeds and move the motors at the same time. A more powerful microcontroller will allow for the addition of feedback control and give more reliable movement.

Once a reliable working prototype of a single node is created, multiple nodes can be hooked together to create a network of sensors that can simultaneously provide information about a network. 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 cable lengths.

If a reliable 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, and 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. Special thanks to Dr. Daniel Lee, who has been a great mentor, advisor, and source of encouragement throughout this project. In addition, I'd like to thank Sid Deliwala for his invaluable assistance in the Electrical Engineering lab, Lois Clearfield for taking care of the many administrative details, and James Tripp of the Math Department for his help in solving various geometrical problems. Last but not least, I would 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. S. Del Prete, Ames room, Available at http://psylux.psych.tu-dresden.de/i1/kaw/diverses%20Material/www.illusionworks.com/html/ames_room.html, 6/10/03.

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

4. 29 Palms Fixed/Mobile Experiment Tracking vehicles with UAV-delivered sensor network. University of California at Berkeley and MLB Co, http://www.eecs.berkeley.edu/~pister/29Palms0103, 6/10/03.

5. Smart Dust project website. University of California at Berkeley, http://robotics.eecs.berkeley.edu/~pister/SmartDust, 6/10/03.

6. D.D. Lee, IR asynchronous communicator, https://courseweb.library.upenn.edu/courses/1/EE400-2003A/content/_287694_1/lab05.pdf, 6/26/03.

7. J. Brown, DPRG: Brief H-Bridge Theory of Operation, http://www.dprg.org/tutorials/1998-04a/, 7/10/03.

8. Hacking Servos, http://www.geocities.com/kokop76/tips/tips.html, 7/3/03.

9. CTS 288 Series 16mm Rotary Encoder, http://rocky.digikey.com/WebLib/CTS/Web%20Data/288%20Series.pdf, 7/2/03.

APPENDIX A

```
/*
SUNFEST project
3-D Sensor Network
Greg Kuperman
August 1, 2003
*/

#include <hc11.h>

#pragma interrupt_handler OC3ISR
#pragma interrupt_handler OC4ISR
#pragma interrupt_handler OC5ISR
#pragma interrupt_handler SCIISR
#pragma interrupt_handler RTIISR
#pragma interrupt_handler IC1ISR

#define NUMBER 15000L /*Total Length of ticks on HC11 counter of duty
                                                    cycle length for motor output*/

void OC3ISR();
void OC4ISR();
void OC5ISR();
void SCIISR();
void IC1ISR();
void RTIISR();
void TOIISR();

/*uparray and downarray are what duty cycle will give that
 position of the array time between encoder ticks. anything
 below 5 are unaccesable positions. so to get 25 units of time
 between encoder clicks for bringing object up, choose duty cycle
 at uparray[25], values for this array are assigned from testing
 (write program to assign duty cycle to a motor and direction, and
 have RTI count and every encoder tick, output number of RTI's for
 that specific duty cycle*/
int uparray[105] = {04,04,04,04,04,04,04,04,04,04,04,04,04,04,04,04,04,04,04,
           95,95,95,90,85,80,78,77,75,70,68,67,65,63,62,60,59,57,56,
           55,55,54,54,53,53,52,52,51,50,48,47,45,45,44,44,43,43,42,42,
           41,40,40,40,40,40,39,39,39,39,39,39,38,38,38,38,38,38,38,37,37,
           37,37,37,37,37,37,36,36,36,36,36,36,36,36,36,35,35,35,35,35,35,
           35,35,35,34,34,34,34,34};
int downarray[105]={04,04,04,04,04,04,04,04,04,04,04,04,04,04,95,90,85,80,
                    70,65,63,60,58,55,52,50,45,44,42,41,40,39,38,37,36,35,
                    34,33,32,31,31,30,29,28,28,27,27,26,26,25,25,24,24,23,23,
                    22,22,22,21,21,21,20,20,20,19,19,19,18,18,18,18,
                    17,17,17,17,16,16,16,16,16,15,15,15,15,15,15,15,15,14,
                    14,14,14,14,14,14,14,14,14,14,14,14,14,14,14,14};

int distance;

int newx,newy,newz;
int x,y,z;
long deltaa,deltab,deltac;
```

```
int sidea,sideb,sidec,sided,sidee;
int x3,y3,range;
int a,b,c,d,da,db,dc,dx;

int acount,bcount,ccount,pulsetrack1,pulsetrack2,pulsetrack3;

int remotex,remotey,remotez,remotecount,xdir,ydir,zdir;
int sciflag;

int flagc,flagd,flage,ecount1,ecount2,ecount3;
int testarray1[10], testarray2[10],testarray3[10];
int pulsecount1,pulsecount2,pulsecount3;

int flagcA,flagcB,flagdA,flagdB,flageA,flageB,flagc1,flagd1,flage1;
int currentstateC,currentstateD,currentstateE,laststateC,laststateD,laststateE;

int i,j,k;
int ledflag,ledcount;

int ii;
unsigned int period[48];
int diff[47];
int count;

int changeflag,countflag;

int direction1=1, direction2=1,direction3=1;

unsigned int t3;

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

unsigned int t4;

unsigned int t_high4; /*pulse width for OC5*/
unsigned int t_low4;

unsigned int t5;

unsigned int t_high5; /*pulse width for OC5*/
unsigned int t_low5;

void main(){

*(unsigned char *)0xD9 = 0x7E;
*(void (**)())0xDA = OC3ISR;
*(unsigned char *)0xD3 = 0x7E;
*(void (**)())0xD4 = OC5ISR;
*(unsigned char *)0xD6 = 0x7E;
*(void (**)())0xD7 = OC4ISR;
*(unsigned char *)0xC4 = 0x7E;
*(void (**)())0xC5 = SCIISR;
*(unsigned char *)0xEB = 0x7E;
*(void (**)())0xEC = RTIISR;
*(unsigned char *)0xE8 = 0x7E;
```

```
*(void (**)())0xE9 = IC1ISR;


/*Set initial duty cycle*/
t3 = 5;
t4 = 5;
t5 = 5;

t_high3 = NUMBER*t3 / 100;
t_low3 = NUMBER - t_high3;

t_high4 = NUMBER*t4 / 100;
t_low4 = NUMBER - t_high4;

t_high5 = NUMBER*t5 / 100;
t_low5 = NUMBER - t_high5;

/*SCI settings*/

BAUD = 0x30; /*9600 BAUD*/
SCCR1 = 0x00;
SCCR2 = 0x0C; /* Enable SCI transmitter and reciever */
SCCR2 |= 0x20; //Reciever interrupt enable

/* OC5 settings */
PACTL &= 0xFB; //Set OC5 in PACTL
TOC5 = 0x7FFF;
TCTL1 |= 0x03; //set output pin high for OC5
TMSK1 |= 0x08; //Enable OC5 interrupt locally
TFLG1 &= 0x08; //Clear OC5 Flag

/* IR input settings */

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

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

/* OC4 settings */
TOC4 = 0x7FFF;
TCTL1 |= 0x0C; //set output pin high for OC4
TMSK1 |= 0x10; //Enable OC3 interrupt locally
TFLG1 &= 0x10; //Clear OC3 Flag

/*Real time interrupt settings*/
//Timerover flow settings
TMSK2 |= 0x40;
TFLG2 |= 0x40;
PACTL |= 0x01;

DDRC=0; //all PORTC is input
```

```
PORTB |= 0x2A; //turn brake on

/*clear testarrays*/
for(i=0,j=0;i<10;i++,j++){
        testarray1[i]=0;
        testarray2[j]=0;
        testarray3[j]=0;
        }
i=0;
j=0;
k=0;

asm("cli");

while(1){

        }
}

/*absolute value function*/
int absolute(int value){
        if(value<0) value*=-1;
        return value;
        }


/*Square root without use of floating point*/
unsigned int sqrtnewton (unsigned int num) {
        unsigned temp, div = num;
        unsigned rslt = num;
        if (num <= 0)
                return 0;
        while(1){
                        temp = num / div + div;
                        div = temp >> 1;
                        div += temp & 1;
                if (rslt > div)
                  rslt = div;
                else return rslt;
        }
}

/*Calculate x y z coordinates of object using intersection of three spheres*/
void xyz(){

        x = (sideb*sideb - sidee*sidee - sidea*sidea) / (-2*sidee);
        y = (sidec*sidec + 2*x3*x - x3*x3 - y3*y3 - sidea*sidea) / (-2*y3);
        z = sqrtnewton(sidea*sidea - x*x - y*y);

        range=(y3-y)*x3 / y3;/*width of accessable x for triangular area*/

        printf("\nx=%d y=%d z=%d",x,y,z);
        }

/*maximum value function*/
long max(long first, long second){
```

```
        if (first > second) return first;
        else return second;
        }


/*change lengths of strings to new lengths based off of encoder readings*/
void change(){

         int TICKS=1460; /*ticks * 100 per revolution on encoder,
                           adjusted by -8% for accuracy*/
         int DIAMETER=1; /*Diameter of spool in inches*/

         deltaa = (100 * (long)pulsecount1*22)/TICKS/7; //change in sidea
         pulsecount1=0;

         deltab = (100 * (long)pulsecount2*22)/TICKS/7; //change in sideb
         pulsecount2=0;

         deltac = (100 * (long)pulsecount3*22)/TICKS/7;       //change in sidec
         pulsecount3=0;

         /*change side lengths*/
         sidea += (int)deltaa;
         sideb += (int)deltab;
         sidec += (int)deltac;

        putchar('\n');
        if (deltaa<0){
                printf(" SideA IN ");
                  }
        else if (deltaa>0){
                 printf(" SideA OUT ");
                  }

        if (deltab<0){
                printf(" SideB IN ");
                  }
        else if (deltab>0){
                 printf(" SideB OUT ");
                  }

        if (deltac<0){
                printf(" SideC IN ");
                  }
        else if (deltac>0){
                 printf(" SideC OUT ");
                  }

        printf("\nCHANGE: deltaa=%d deltab=%d deltac=%d NEW: sidea=%d sideb=%d sidec=%d",
                          (int)deltaa,(int)deltab,(int)deltac,sidea,sideb,sidec);

         }


/* calculate and output duty cycle to motors to achieve desired motion*/
void dist(){
```

```
        int ratio1,ratio2,ratio3;
        int temp,temp1,temp2;

        printf("\nnewx=%d newy=%d newz=%d",newx,newy,newz); //desired x y z

        /*find change in length for each the sides*/
        /*start*/
        temp = newx*newx + newy*newy + newz*newz;
        temp1 = (sided-newx)*(sided-newx) + newy*newy + newz*newz;
        temp2 = (newx - sided/2)*(newx - sided/2) + (newy - (int)*
                (866L*(long)sided / 1000))*(newy - (int)((long)866L*sided / 1000))
                + newz*newz;

        a = sqrtnewton(temp);
        b = sqrtnewton(temp1);
        c = sqrtnewton(temp2);

        da = (sidea-a);
        db = (sideb-b);
        dc = (sidec-c);
        /*finish*/

        /*find number of encoder ticks for each motor for change in length*/
        pulsetrack1 = (long)da * 1460 * 7 / 22 / 100;
        if (pulsetrack1<0) pulsetrack1 *=-1;
        pulsetrack2 = (long)db * 1460 * 7 / 22 / 100;
        if (pulsetrack2<0) pulsetrack2 *=-1;
        pulsetrack3 = (long)dc * 1460 * 7 / 22 / 100;
        if (pulsetrack3<0) pulsetrack3 *=-1;

        /*set direction of string movement for each motor*/
        if (da<0) {direction1 = 2; PORTB &= 0xFB; putchar('A');   }
        else if(da>0) {direction1 = 1;PORTB |= 0x04; putchar('a'); }
        if (db<0) {direction2 = 2; PORTB &= 0xFE;putchar('B');  }
        else if (db>0) {direction2 = 1;PORTB |= 0x01; putchar('b');}
        if (dc<0) {direction3 = 2; PORTB &= 0xEF;putchar('C');  }
        else if (dc>0) {direction3 = 1;PORTB |= 0x10; putchar('c');}

        putchar('\n');

        printf("\nda=%d db=%d dc=%d direction1=%d direction2=%d direction3=%d",
                        da,db,dc,direction1,direction2,direction3);

        if (da<0) da *= -1; //absolute value
        if (db<0) db *= -1; //absolute value
        if (dc<0) dc *= -1; //absolute value

        ratio1=1001;
ratio2=1001;
ratio3=1001;

        /*setting the duty cycle*/
        /*algorithm: 1. find the maximum length change
         2. find the ratio of the other two side changes to the maximum change
        multiplied by 1000 to keep it an integer with 3 decimal precision
```

```
 3. use 20th position in each array as maximum speed (20 unit time between
encoder ticks)
4. assign other motor speeds by dividing 20 by ratio (scaled by 1000)
5. if motor speed too slow (ratio below 200) assign manually based on
specifications of the system
*/

if (max(max(da,db),dc) == da) {
        ratio2=(1000*db)/da;
        ratio3=(1000*dc)/da;
        printf("\ndA max, ratio2=%d ratio3=%d",ratio2,ratio3);


                if (direction1==1) t4 = uparray[20];
                else if (direction1==2) t4 = downarray[20];
                if (direction2==1) t5 = uparray[20000/ratio2];
    else if (direction2==2) t5 = downarray[20000/ratio2];
    if (direction3==1) t3 = uparray[20000/ratio3];
    else if (direction3==2) t3 = downarray[20000/ratio3];

                 if ((ratio2<200 && ratio2>100) && direction2==1) t5 = 33;
                 else if ((ratio2<200 && ratio2>100) && direction2==2) t5 = 15;
                 else if(ratio2<100 && direction2==1) t5=31;
                 else if(ratio2<100 && direction2==2) t5=13;
                 if ((ratio3<200 && ratio3>100) && direction3==1) t3 = 33;
                 else if ((ratio3<200 && ratio3>100) && direction3==2) t3 = 15;
                 else if(ratio3<100 && direction3==1) t3=31;
                 else if(ratio3<100 && direction3==2) t3=13;


}

else if (max(max(da,db),dc) == db){
        ratio1=(1000*da)/db;
        ratio3=(1000*dc)/db;
        printf("\ndB max, ratio1=%d ratio3=%d",ratio1,ratio3);


                if (direction1==1) t4 = uparray[20000/ratio1];
                else if (direction1==2) t4 = downarray[20000/ratio1];
                if (direction2==1) t5 = uparray[20];
            else if (direction2==2) t5 = downarray[20];
            if (direction3==1) t3 = uparray[20000/ratio3];
            else if (direction3==2) t3 = downarray[20000/ratio3];

                 if ((ratio1<200 && ratio1>100) && direction1==1) t4 = 33;
                 else if ((ratio1<200 && ratio1>100) && direction1==2) t4 = 15;
                 else if(ratio1<100 && direction1==1) t4=31;
                 else if(ratio1<100 && direction1==2) t4=13;
                 if ((ratio3<200 && ratio3>100) && direction3==1) t3 = 33;
                 else if ((ratio3<200 && ratio3>100) && direction3==2) t3 = 15;
                 else if(ratio3<100 && direction3==1) t3=31;
                 else if(ratio3<100 && direction3==2) t3=13;

}
```

```
        else if (max(max(da,db),dc) == dc){
                ratio1=(1000*da)/dc;
                ratio2=(1000*db)/dc;
                printf("\ndC max, ratio1=%d ratio2=%d",ratio1,ratio2);

                        if (direction1==1) t4 = uparray[20000/ratio1];
                        else if (direction1==2) t4 = downarray[20000/ratio1];
                        if (direction2==1) t5 = uparray[20000/ratio2];
                   else if (direction2==2) t5 = downarray[20000/ratio2];
                   if (direction3==1) t3 = uparray[20];
                   else if (direction3==2) t3 = downarray[20];

                         if ((ratio1<200 && ratio1>100) && direction1==1) t4 = 33;
                         else if ((ratio1<200 && ratio1>100) && direction1==2) t4 = 15;
                         else if(ratio1<100 && direction1==1) t4=31;
                         else if(ratio1<100 && direction1==2) t4=13;
                         if ((ratio2<200 && ratio2>100) && direction2==1) t5 = 33;
                         else if ((ratio2<200 && ratio2>100) && direction2==2) t5 = 15;
                         else if(ratio2<100 && direction2==1) t5=31;
                         else if(ratio2<100 && direction2==2) t5=13;

        }

        /*adjusted for this system specifically to correct for erroneous movement*/
        if (direction1==2){t4+=10;putchar('Q');
        else if (t4<30) {t4+=10;putchar('W');
        if (direction2==2) {t5+=10;putchar('E');
        else if (t5<30) {t5+=10;
        if (direction3==2) {t3+=10;
        else if (t3<30) {t3+=10;

        printf("\nt3=%d t4=%d t5=%d pulsetrack1=%d pulsetrack2=%d pulsetrack3=%d",
                t3,t4,t5,pulsetrack1,pulsetrack2,pulsetrack3);
}


/*called when user asks system to change position*/
void changetime(){

                PORTB |= 0x2A; //turn brake on

                putchar('\n');

                printf("\npulsecount1=%d pulsecount2=%d pulsecount3=%d",
                        pulsecount1,pulsecount2,pulsecount3);

                change();       //find new lengths of sides
                xyz();          //calculate new xyz
                printf("\nNew x=%d y=%d z=%d",x,y,z);

                putchar('\n');

                /*remote input*/
                if(sciflag==0){
                        newx=remotex+x;
                        printf("\nx has %d added, newx=%d",remotex,newx);
```

```
                    newy=remotey+y;
                    printf("\ny has %d added, newy=%d",remotey,newy);
                    newz=remotez+z;
                    printf("\nz has %d added, newz=%d",remotez,newz);
                    }

            /*serial input*/
            else if(sciflag==1){
                    printf("\nPlease enter new X: ");
               newx=scanchar();
               printf("\nPlease enter new y: ");
               newy=scanchar();
               printf("\nPlease enter new z: ");
               newz=scanchar();
            }

            dx = newx-x; //change in x direction
            if (dx<0) dx*=-1;

            if (y>y3) printf("\nRange Error: y too large");
            else if (x>(x3+range) || x<(range-x3))
                    printf("\nRange Error: x out of trianlge");

            dist(); /*assign duty cycles for each motor to get
                     system to move correctly*/

            acount=0;
            bcount=0;
            ccount=0;

            putchar('\n');

            printf("\ndx=%d",dx);

            putchar('\n');

            t_high3 = NUMBER*t3 / 100;
t_low3 = NUMBER - t_high3;

            t_high4 = NUMBER*t4 / 100;
t_low4 = NUMBER - t_high4;

            t_high5 = NUMBER*t5 / 100;
t_low5 = NUMBER - t_high5;

            PORTB &= 0xD5; //turn brake off

            /*if duty cycle below 5, leave brake on for that motor*/
            if(t3<5)PORTB |= 0x20;
            if(t4<5)PORTB |= 0x08;
            if(t5<5)PORTB |= 0x02;

            remotex=0;
        remotey=0;
            remotez=0;
            remotecount=0;
```

```
}

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

/*if system being accessed for first time, assign side lengths
 and dimensions of system*/
void first(){
			changeflag=1;
			printf("\nPlease enter length of SideA: ");
			sidea=scanchar();
			printf("\nPlease enter length of SideB: ");
			sideb=scanchar();
			printf("\nPlease enter length of SideC: ");
			sidec=scanchar();
			printf("\nPlease enter Equal Length of
				Isosceles Traingle: ");
			sided=scanchar();
			printf("\nPlease enter Non-Equal Length of
				Isosceles Traingle: ");
			sidee=scanchar();

			x3 = sidee/2;
			y3 = sqrtnewton(sided*sided - x3*x3);
			printf("\nx3=%d y3=%d ",x3,y3);

			xyz();
			}

void SCIISR(){
	unsigned char scsrval,scdrval;
	scsrval=SCSR;
	scdrval=SCDR;
			sciflag=1;
			if(changeflag==0)first();
	else changetime();
}

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

```
            TCTL1 |= 0x10;
            }
}

void OC4ISR(){

      TFLG1 &= 0x10;
      if (TCTL1 & 0x04){
            TOC4 += t_high4;
            TCTL1 &= 0xFB;
            }
      else {
            TOC4 += t_low4;
            TCTL1 |= 0x04;
            }
                  }

void OC5ISR(){

      TFLG1 &= 0x08;
      if (TCTL1 & 0x01){
            TOC5 += t_high5;
            TCTL1 &= 0xFE;
            }
      else {
            TOC5 += t_low5;
            TCTL1 |= 0x01;
            }
                  }

/*poll for encoder ticks*/
void RTIISR(){
          TFLG2 |= 0x40;

          /*to count time between encoder ticks, use the ecount's below,
           reset them everytime
            a tick occurs, but print out a value before you do so*/
          ecount1++;
          ecount2++;
          ecount3++;

          ledcount++;

          /*turn on LED for remote for 1/2 second*/
          if(ledflag==1){
                  PORTB |= 0x40;
                  if(ledcount > 60){
                          PORTB &= 0xBF;
                          ledflag=0;
                  }
          }


          /*Dual channel encoders, edge A leads edge B*/

          if ((PORTC & 0x08) && flagc<3) flagc++;
```

```
if ((PORTC & 0x08) && flagc==3){
  flagcA=1;
  flagc++;
  }
if (!(PORTC & 0x08) && flagc==4) {flagc=0;flagcA=0;}

if ((PORTC & 0x01) && flagc1<3) flagc1++;
if ((PORTC & 0x01) && flagc1==3){
  flagcB=1;
  flagc1++;
  }
if (!(PORTC & 0x01) && flagc1==4) {flagc1=0; flagcB=0;}


if(flagcA==0 && flagcB==0)      currentstateC=0;
else if(flagcA==1 && flagcB==0) currentstateC=1;
else if(flagcA==1 && flagcB==1) currentstateC=2;
else if(flagcA==0 && flagcB==1) currentstateC=3;

if(laststateC != currentstateC){
            if (((laststateC < currentstateC)&&!(laststateC==0 && currentstateC==3)) ||
             (laststateC==3 && currentstateC==0)){ pulsecount1++;acount++;putchar('C');}
            else if((laststateC > currentstateC) || (laststateC==0 && currentstateC==3)){
                        pulsecount1--;acount++;putchar('c');}
            laststateC=currentstateC;
}

if ((PORTC & 0x02) && flagd<3) flagd++;
if ((PORTC & 0x02) && flagd==3){
  flagd++;
  flagdA=1;
  }
if (!(PORTC & 0x02) && flagd==4) {flagd=0;flagdA=0;}

if ((PORTC & 0x10) && flagd1<3) flagd1++;
if ((PORTC & 0x10) && flagd1==3){
  flagd1++;
  flagdB=1;
  }
if (!(PORTC & 0x10) && flagd1==4) {flagd1=0;flagdB=0;}

if(flagdA==0 && flagdB==0)      {currentstateD=0; }
else if(flagdA==1 && flagdB==0) {currentstateD=1; }
else if(flagdA==1 && flagdB==1) {currentstateD=2; }
else if(flagdA==0 && flagdB==1) {currentstateD=3; }

if(laststateD != currentstateD){
            if (((laststateD < currentstateD)&&!(laststateD==0 && currentstateD==3)) ||
             (laststateD==3 && currentstateD==0)){ pulsecount2++;bcount++;putchar('D');}
            else if((laststateD > currentstateD) || (laststateD==0 && currentstateD==3)){
             pulsecount2--;bcount++;putchar('d');}
            laststateD=currentstateD;
}

if ((PORTC & 0x20) && flage<3) flage++;
if ((PORTC & 0x20) && flage==3){
```

```
            flage++;
            flageA=1;
            }
          if (!(PORTC & 0x20) && flage==4) {flage=0;flageA=0;}

          if ((PORTC & 0x04) && flage1<3) flage1++;
          if ((PORTC & 0x04) && flage1==3){
            flage1++;
            flageB=1;
            }
          if (!(PORTC & 0x04) && flage1==4) {flage1=0;flageB=0;}

          if(flageA==0 && flageB==0)     { currentstateE=0; }
          else if(flageA==1 && flageB==0){ currentstateE=1; }
          else if(flageA==1 && flageB==1){ currentstateE=2; }
          else if(flageA==0 && flageB==1){ currentstateE=3; }

          if(laststateE != currentstateE){
                    if (((laststateE < currentstateE)&&!(laststateE==0 && currentstateE==3)) ||
                     (laststateE==3 && currentstateE==0)){pulsecount3--;ccount++;putchar('e');}
                    else if((laststateE > currentstateE) || (laststateE==0 && currentstateE==3)){
                     pulsecount3++;ccount++;putchar('E');}
                    laststateE=currentstateE;
          }

          /*wait until one of the motors goes the distance necessary and the other two are
           within 3 encoder ticks, the stop the system*/
          if ((pulsetrack1 < acount && pulsetrack2 < bcount + 3 && pulsetrack3 < ccount + 3) ||
           (pulsetrack1 < acount + 3 && pulsetrack2 < bcount && pulsetrack3 < ccount + 3) ||
           (pulsetrack1 < acount + 3 && pulsetrack2 < bcount + 3 && pulsetrack3 < ccount)){
            PORTB |= 0x2A; //turn brake on
            acount=0;
            bcount=0;
            ccount=0;
            }
}

/*Remote control*/
/*Enter ch+ or ch- to increase or decrease coordinate respetively.
currently, two digit change for each coordinate.
example: to increase x by 10, not change y, and decrease z by 3, enter the following:
CH+ 1 0 CH+ 0 0 CH- 0 3.
LED turns on for 1/2 second to confirm each time something is entered
*/

void IC1ISR(){

                    if(countflag>100){
           ii=0;
           countflag=0;
           }

      TFLG1 &= 0x04;
      period[ii] = TIC1;
      ii++;
      if (ii==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");}


else if (diff[13]==1 && diff[14]==1 && diff[21]==1){
	printf("CH -");
                                                    if(remotecount==0)xdir=0;
                                                    else if(remotecount==3)ydir=0;
                                                    else if(remotecount==6)zdir=0;
                                                    remotecount++;
                                                    ledflag=1;
                                                    ledcount=0;
}
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");}


else if (diff[13]==1 && diff[14]==1){
	printf("CH +");
                                                    if(remotecount==0)xdir=1;
                                                    else if(remotecount==3)ydir=1;
                                                    else if(remotecount==6)zdir=1;
                                                    remotecount++;
                                                    ledflag=1;
	ledcount=0;
}


else if (diff[15]==1 && diff[16]==1 && diff[21]==1){
	printf("VOL -");

}


else if (diff[15]==1 && diff[16]==1){
	printf("VOL +");

}


else if (diff[17]==1 && diff[18]==1 && diff[21]==1){
	printf("9");
                                                    if(remotecount==1) remotex=9;
                                                    else if(remotecount==2) remotex=remotex*10+9;
                                                    else if(remotecount==4) remotey=9;
                                                    else if(remotecount==5) remotey=remotey*10+9;
                                                    else if(remotecount==7) remotez=9;
                                                    else if(remotecount==8) remotez=remotez*10+9;
                                                    remotecount++;
                                                    ledflag=1;
                                                    ledcount=0;
```

```
}


else if (diff[17]==1 && diff[18]==1){
      printf("8");
                                                if(remotecount==1) remotex=8;
                                                else if(remotecount==2) remotex=remotex*10+8;
                                                else if(remotecount==4) remotey=8;
                                                else if(remotecount==5) remotey=remotey*10+8;
                                                else if(remotecount==7) remotez=8;
                                                else if(remotecount==8) remotez=remotez*10+8;
                                                remotecount++;
                                                ledflag=1;
                                                ledcount=0;

}

else if (diff[14]==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");

}


else if (diff[20]==1 && diff[21]==1 && diff[19]==1){
      printf("5");
                                                if(remotecount==1) remotex=5;
                                                else if(remotecount==2) remotex=remotex*10+5;
                                                else if(remotecount==4) remotey=5;
                                                else if(remotecount==5) remotey=remotey*10+5;
                                                else if(remotecount==7) remotez=5;
                                                else if(remotecount==8) remotez=remotez*10+5;
                                                remotecount++;
                                                ledflag=1;
                                                ledcount=0;

}


else if (diff[20]==1 && diff[19]==1){
      printf("4");
                                                if(remotecount==1) remotex=4;
                                                else if(remotecount==2) remotex=remotex*10+4;
                                                else if(remotecount==4) remotey=4;
                                                else if(remotecount==5) remotey=remotey*10+4;
                                                else if(remotecount==7) remotez=4;
```

```
                                        else if(remotecount==8) remotez=remotez*10+4;
                                        remotecount++;
                                        ledflag=1;
                                        ledcount=0;

}


else if (diff[19]==1 && diff[22]==1){
        printf("6");
                                        if(remotecount==1) remotex=6;
                                        else if(remotecount==2) remotex=remotex*10+6;
                                        else if(remotecount==4) remotey=6;
                                        else if(remotecount==5) remotey=remotey*10+6;
                                        else if(remotecount==7) remotez=6;
                                        else if(remotecount==8) remotez=remotez*10+6;
                                        remotecount++;
                                        ledflag=1;
                                        ledcount=0;

}


else if(diff[19]==1){
        printf("7");
                                        if(remotecount==1) remotex=7;
                                        else if(remotecount==2) remotex=remotex*10+7;
                                        else if(remotecount==4) remotey=7;
                                        else if(remotecount==5) remotey=remotey*10+7;
                                        else if(remotecount==7) remotez=7;
                                        else if(remotecount==8) remotez=remotez*10+7;
                                        remotecount++;
        ledflag=1;
                                        ledcount=0;
}

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


else if (diff[22]==1 && diff[21]==1){
        printf("2");
                                        if(remotecount==1) remotex=2;
                                        else if(remotecount==2) remotex=remotex*10+2;
                                        else if(remotecount==4) remotey=2;
                                        else if(remotecount==5) remotey=remotey*10+2;
                                        else if(remotecount==7) remotez=2;
                                        else if(remotecount==8) remotez=remotez*10+2;
                                        remotecount++;
        ledflag=1;
                                        ledcount=0;
}


else if (diff[21]==1){
        printf("3");
                                        if(remotecount==1) remotex=3;
```

```
                                                else if(remotecount==2) remotex=remotex*10+3;
                                                else if(remotecount==4) remotey=3;
                                                else if(remotecount==5) remotey=remotey*10+3;
                                                else if(remotecount==7) remotez=3;
                                                else if(remotecount==8) remotez=remotez*10+3;
                                                remotecount++;
            ledflag=1;
                                                ledcount=0;
        }


        else if(diff[23]==1){
            printf("1");
                                                if(remotecount==1) remotex=1;
                                                else if(remotecount==2) remotex=remotex*10+1;
                                                else if(remotecount==4) remotey=1;
                                                else if(remotecount==5) remotey=remotey*10+1;
                                                else if(remotecount==7) remotez=1;
                                                else if(remotecount==8) remotez=remotez*10+1;
                                                remotecount++;
                                                ledflag=1;
                                                ledcount=0;
        }

        else {
                                        printf("0");
                                        if(remotecount==1) remotex=0;
                                        else if(remotecount==2) remotex=remotex*10+0;
                                        else if(remotecount==4) remotey=0;
                                        else if(remotecount==5) remotey=remotey*10+0;
                                        else if(remotecount==7) remotez=0;
                                        else if(remotecount==8) remotez=remotez*10+0;
                                        remotecount++;
                                        ledflag=1;
                                        ledcount=0;
                                        }

        printf("\n");

        ii=0;
        countflag = 0;

                                    if (remotecount==9){
                                      if(xdir==0)remotex*=-1;
                                      if(ydir==0)remotey*=-1;
                                      if(zdir==0)remotez*=-1;
          printf("\nRemote Entered: deltax=%d deltay=%d deltaz=%d",
                                remotex,remotey,remotez);
                                      sciflag=0;
                                      changetime();
                                      }
    }
}
```

MAGNETORESISTANCE OF ELECTROSPUN CARBON NANOFIBERS PYROLYZED AT LOW TEMPERATURES

NSF Summer Undergraduate Fellowship in Sensor Technologies
Linda Lamptey (Electrical Engineering) – University of Pennsylvania
Advisors: Dr. Jorges Avilles-Santiago and Yu Wang

ABSTRACT

Carbon fibers with diameters of approximately 100 nm can be produced by heating electrospun polyacrylonitrile (PAN) nanofibers in a reduced-pressure chamber. These nanofibers' outstanding properties, especially their high specific surface area, make them promising materials for scaffolds in tissue engineering and also for high-performance filtration and sensor applications. However before any of these potential uses can be explored, the nature of these nanofibers must first be better understood.

One important quantum effect phenomenon that occurs in carbon nanofibers is magnetoresistance (MR).This is a measure of how the electrical resistance of the carbon nanofiber changes in the presence of a transverse magnetic field. To make this measurement, the PAN nanofibers were placed in the right orientation on a patterned silicon wafer by electrospinning. The PAN nanofiber samples were then pyrolyzed in a vacuum chamber at a temperature of 1173K to produce carbon nanofiber samples. The four-point probe method was used to measure their conductivity. Initially, the temperature dependence of the nanofiber was observed, with no applied magnetic field, within a temperature range of 0K to 300K. The resistance decreased exponentially as the temperature was increased. Then measurements were taken at temperatures of 1.9K, 3K, 5K, and 10K within a magnetic field range of –9T to 9T each. These current and voltage measurements were then manipulated to calculate MR. In general, MR was negative, and its magnitude increased with an increase in magnetic field and a decrease in temperature. This result was attributed to the weak localization effect model.

Table of Contents

1. INTRODUCTION

Conventional fiber spinning methods usually use mechanical forces to drive fiber formation. These conventional methods are strong enough to produce fibers with micron-meter-range diameters. The significance of these carbon fibers lies in their chemical inertness, strength, and electrical and thermal conductivity. These materials can thus be conveniently used for batteries, plastic, and asphalt applications. However, when mechanical forces are combined with electrostatic forces, the diameter of the carbon fibers produced reduces significantly to the nanometer range (see Figure 1). This more effective spinning technique is known as electrospinning.

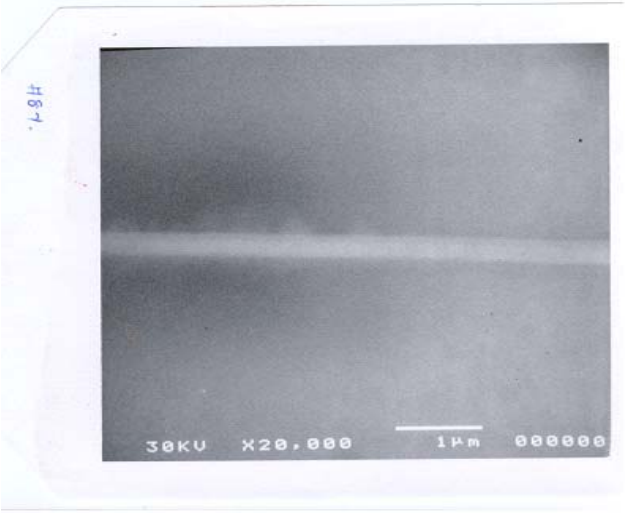


Figure 1: Image of carbon nanofiber obtained from scanning electron microscope.

The ultrafine nanofibers produced by electrospinning are more outstanding than ordinary carbon fibers because they have unusually large specific surface areas [1]. There has been much recent research on carbon nanofibers because of their promising potential uses in the field of nanotechnology. These nanofibers also have high porosity and very small pore size, making them likely candidates for use in highly selective filtration applications. They are also absorbent. This quality, combined with their chemical inertness and high conductivity properties, makes them prospective materials for sensors. Their electrical resistance and thus conductivity is highly dependent on the physical conditions of their surroundings. Consequently, the practical application of this project is to use these carbon nanofibers as sensors for physical measurements such as temperature, pressure, and levels of certain atmospheric gases. Nevertheless, the feasibility of this application depends on the scope and accuracy of knowledge of the nature of the carbon nanofiber. Unfortunately, the nanoscopic size of these fibers makes them hard to manipulate; thus, not enough research has been done in this field.

This project helps to further characterize carbon nanofibers by measuring magnetoresistance (MR) — the change in electrical resistance when an electric field is

applied perpendicularly to the carbon nanofiber. MR reflects the electron transport properties of the nanofiber. In accordance to classical electron theory [2], electrical resistance should increase when a perpendicular magnetic field is applied to the carbon nanofiber. Thus, MR should be positive. However, it has been observed that in carbon fibers resistance tends to increase as the magnetic field increases. When temperature is lowered, the degree of disorderliness increases, also increasing MR. The fibers thus exhibit negative MR properties that are attributable to their structure and quantum mechanic effects. The Bright Model [3] explained this negative MR in terms of an increase in the density of carrier electrons when a magnetic field is applied at low temperatures. Although the Bright Model represented a significant breakthrough in understanding the negative MR of carbon nanofibers, it could not efficiently explain all observed data. Another model [4] explained negative magnetoresistance in terms of weak localization effects. This phenomenon is an inherent interference effect based on the scattering of carriers at low temperatures.

This project will further investigate if the same phenomenon of negative magnetoresistance that exists in carbon fibers also exists in nanofibers. Previous research has examined nanofibers pyrolyzed at high temperatures of 2273K and above [5,6]. However, not much work has been done on nanofibers pyrolyzed at low temperatures. Thus this project aims to measure MR at a low temperature of 1173K to observe the trends at different low temperatures.

2. BACKGROUND

2.1 The Potential Sensor Application

The effects of certain physical parameters on electrical resistivities of carbon nanofibers demonstrate that they can be used in sensor applications. Resistivity (ρ), an intrinsic property of a material, is a measure of its longitudinal electrical resistance. Resistivity is related to electrical resistance (R) by the equation $R = \rho L / S$ (1), where L is the length of the fiber and S is the surface area. According to Ohms Law, for certain materials such as carbon nanofibers, voltage (V) across and the current (I) flowing through the material are related by V=IR (2). A combination of these two equations yields $\mathbf{I = V/R = VS/ \rho L}$. Hence in making a sensor, some of these properties of the nanofiber can be fixed and only one left varying. In the sensor circuit shown in Figure 2, the length, surface area, and voltage across the nanofiber remain fixed.

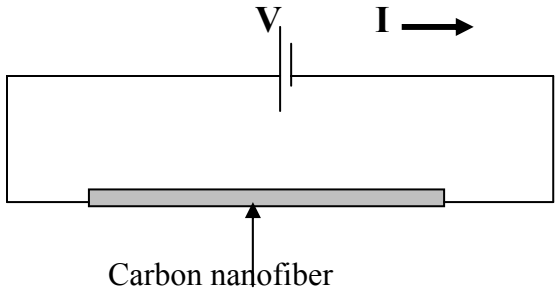


Figure 2 Circuit of potential carbon nanofiber sensor

Hence, the only property affecting the current flowing through the circuit is the resistivity. If a physical parameter that affects resistivity, such as temperature, changes, resistivity will change, thus affecting the current through the circuit. Thus if this parameter is being monitored, this change in current output can be interpreted by a program that will trigger a desirable response (see Figure 3).

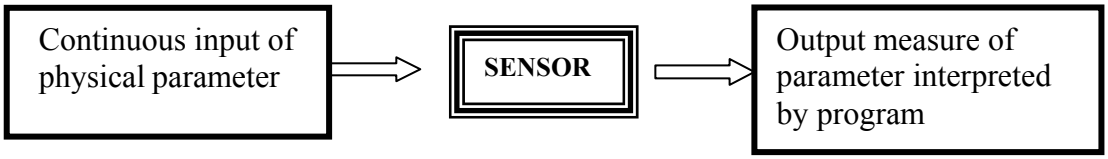


Figure 3: Diagrammatic representation of sensor application.

2.2 Substrate Preparation

The substrate is a silicon wafer that undergoes certain processes before it is ready to be used for the MR measurements.

2.2.1 Photolithography

Photolithography [7] is the process of transferring geometric shapes on a mask to the surface of a silicon wafer. The first step is to prepare a mask with the desired pattern on a glass plate. The wafer then has to be thoroughly cleaned chemically and dried, making it ready to be coated with a layer of silicon dioxide. The next step is to add another thin layer of an ultraviolet sensitive polymer known as a photoresist (see Figure 4). Photoresists can be either positive or negative. For our experiment a positive photoresist was used.

The mask is then aligned with the silicon wafer. The parts of the photoresist that need to be removed to form the pattern are exposed to ultraviolet light through the mask. This process modifies the chemical structure of the photoresist, making it soluble in the developer. The wafer is then developed, thus transferring the pattern on the mask onto the wafer. The solvents are then removed from the photoresist to make the coating photosensitive

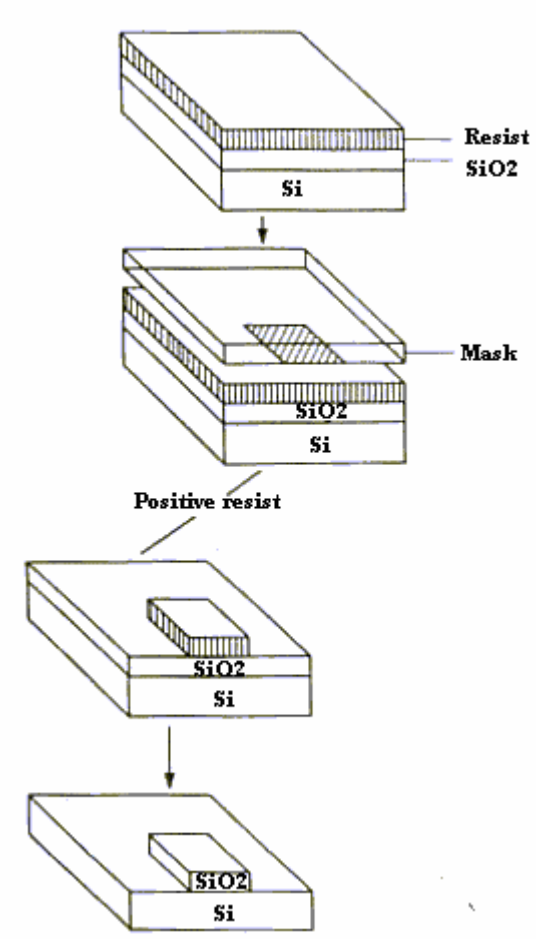


Figure 4: The photolithography process.

2.2.2 Liftoff etching

The next step is an etching process known as "liftoff": a stenciling technique used to pattern noble metals such as gold. To ensure better adhesion, a layer of nickel is deposited on the wafer before depositing the thin layer of gold. The pattern is then stenciled through the gaps in the resist and the unwanted metal is lifted off (see Figures 5 and 6).

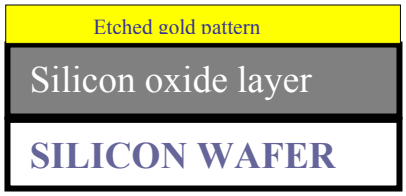


Figure 5 Cross-section of sample

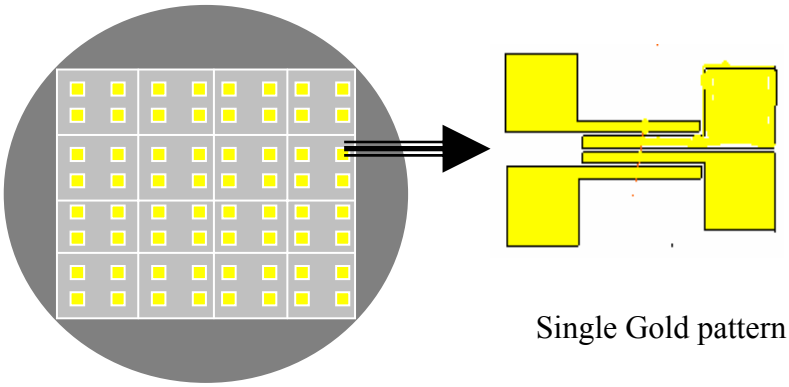


Figure 6 Top view of sample with several etched gold patterns

Single Gold pattern

2.3 Electrospinning

Conventional fiber spinning techniques, such as melt spinning, dry spinning, and wet spinning, use mechanical forces in fiber production. With these processes, a polymer solution is forced out of a spinnerette to produce non-woven fibers as the drawn solution solidifies. Electrospinning combines these mechanical forces with electrostatic forces. The process was discovered in the early 1930s but not significantly exploited because of limited comprehension, but recent years have seen increased research on electrospinning [1]

Electrospun fibers usually have diameters in the nanometer range, in the order of 100 nm. These ultrafine fibers are obtained by spinning polymer solutions in a high-voltage electric field. A simple setup involves a plastic syringe with a metal tip, a clamp, a high-voltage source, and a grounded static screen. These objects are connected as shown in Figure 7, with the clamp slightly inclining the syringe at an angle and the voltage source connected to the metal tip. The polymer is put in the syringe, and a drop of solution is held at the metal tip by the surface tension force of the particles. Simultaneously, the high-voltage source positively charges the particles of the polymer solution, and the repulsion between them increases. At a certain critical point, the electrostatic force due to repulsion overcomes the opposing surface tension force, thus forcing out a jet of the charged polymer solution. While in the electric field, the solvent from this jet solution evaporates, leaving nanofibers targeted at the grounded screen.

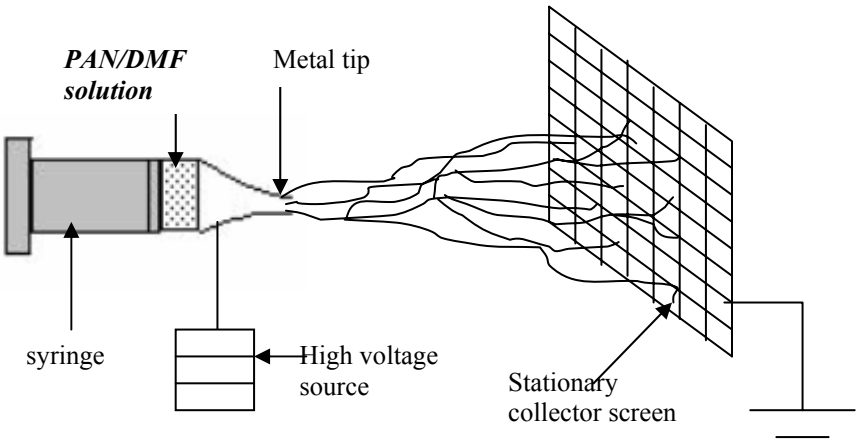


Figure 7: Electrospinning setup.

2.4 Four-Probe Method

The-four probe configuration can be used for electrical resistance measurements. The setup (see Figure 8) has four equally spaced collinear probes. The two outer probes are current-carrying probes, while the two inner ones are for voltage measurements. Since very little current flows through the voltage probes, the voltage measured is about the same as the voltage across the nanofiber. This method thus decreases the contact resistance and makes the voltage measurements more accurate.

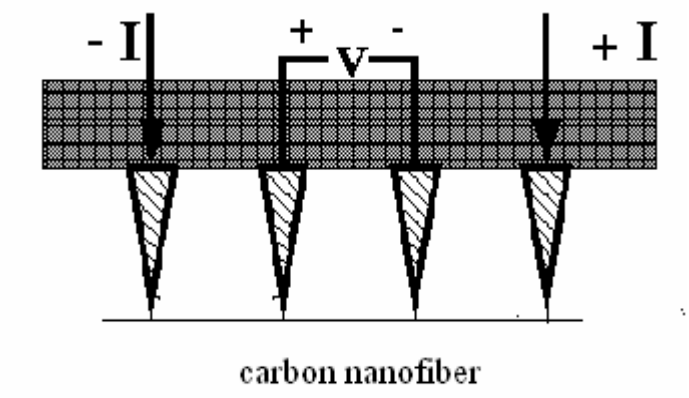


Figure 8: The four-probe method [8].

3. The Experimental Procedure and Results

The first step was to pattern the silicon wafer substrate by the process of photolithography and "liftoff" etching. The next arduous task was to obtain a carbon nanofiber placed in the right orientation across one gold pattern. To achieve this we prepared a polymer solution using 8 mg of polyacrylonitrile (PAN) and 10 ml of N, N-dimethyl formamide (DMF) solvent. This solution was then electrospun and the silicon wafer was momentarily placed in the electric field to collect some PAN nanofibers. It was important to ensure that a high density of PAN nanofibers was not deposited on the wafer, as this would decrease the probability of having a desirable sample. The wafer was then viewed under a microscope to observe whether a nanofiber had been placed in the right orientation. If not, the wafer was cleaned with DMF solution and dried to prepare it for another trial. This experiment was repeated until a single PAN nanofiber was stretched across the central part of one of the gold patterns, as shown in Figure 9. This nanofiber had to not touch any of the four contact pads nor be crossed by another nanofiber since that would affect the resistance measurements of the nanofiber.

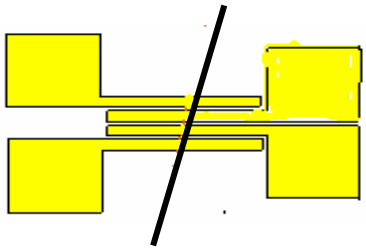


Figure 9: Pattern after successful electrospinning.

After successful electrolysis, the PAN nanofiber was chemically changed to a carbon nanofiber. This was done by heating it in a vacuum chamber at 1173K in a process known as pyrolysis: high-temperature decomposition under reduced pressure. For pyrolysis, a Brew Model 466-S with a mesh element was used. The sample was heated for 30 minutes in a vacuum of 106 Torr. The complications of pyrolysis were the high probability of either moving or breaking the nanofiber. After successful carbonization, the sample was then ready for MR measurements by the four probe method.

To enhance these measurements, silver epoxy was used to attach thin gold wires to each of the four contact pads. These gold wires were connected to the appropriate current and voltage probes. The purpose of the first set of measurements was to investigate how the resistance of the carbon nanofiber varied with temperature. Thus without applying any magnetic field, with a temperature range from 0K to 300K, the electrical resistance of the sample was calculated using the recorded current and voltage measurements. These measurements were made using the four-probe method with a Model 6000 Physical Properties Measurements System by Quantum Inc., equipped with a Keithley 237 high-voltage source measurement unit. As Figure 10 shows, as the temperature increases, electrical resistance also decreases.

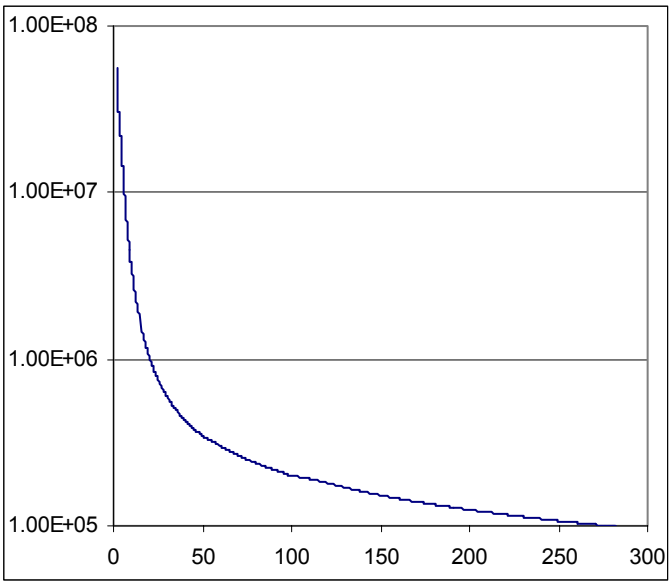


Figure 10: Relationship between electrical resistance and temperature.

Then, with the perpendicular magnetic field ranging from – 9T to 9T, voltage and current measurements were taken on the sample at temperatures of 1.9K, 3.0K, 5.0K, and 10K. MR and magnetoconductance (MC), which is the inverse of MR, were calculated using the formulae on the next page.

$$MR = \{R(B) - R(0)\} / R(0)$$

$$= \{ R(B) / R(0) \} - 1$$

$$MC = \{G(B) - G(0) \} / R(0)$$

$$= \{ G(B) / G(0) \} - 1$$

R(B) is the resistance of the carbon nanofiber in a perpendicular magnetic field of strength B. R(0) is the resistance of the carbon nanofiber when no magnetic field is applied.

G(B) is the conductance of the carbon nanofiber in a perpendicular magnetic field of strength B. G(0) is the conductance of the carbon nanofiber when no magnetic field is applied.

The figures below show the data collected for MR/MC against the magnetic field strength.

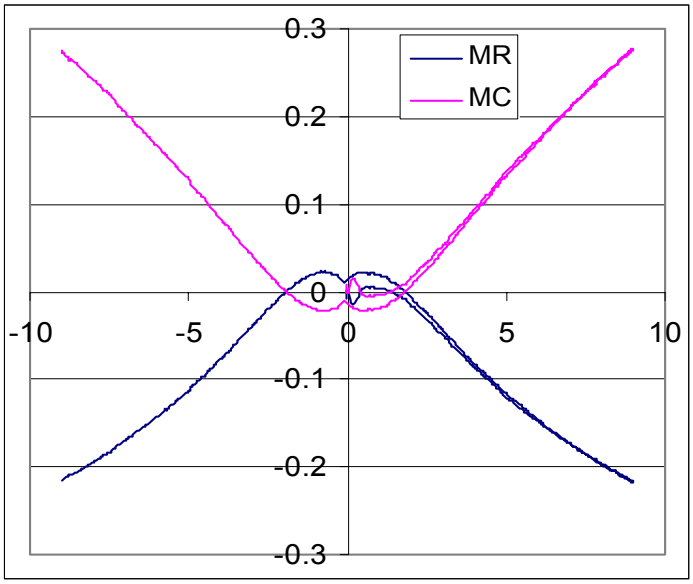


Figure 11: MR/MC measurements taken at 1.9K.

As can be seen from Figure 11, as the magnitude of the magnetic field increases, the magnitude of MR increases. However, a greater part of the curve is below the x-axis, showing that MR is negative, while a greater part of the MC curve is above the x-axis, showing that MC is positive. But an interesting result was observed between about –1.8 T to 1.8T, where on the contrary MR is positive and MC is negative. This could mean that there is a change in the scattering of the carriers in this region which affects the effect of the magnetic field. This speculation could be investigated further in future

experiments. In general, for the sample at 1.9K, the maximum MR obtained in the range was about -21.8 % when magnetic field was 9T.

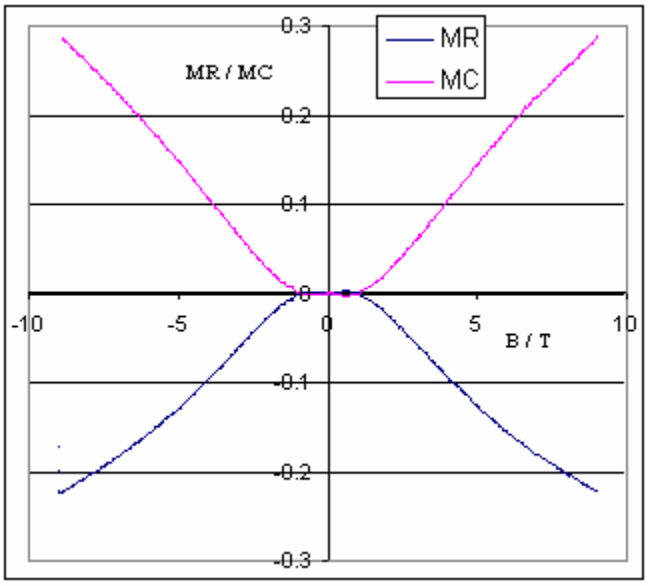


Figure 12: Measurements taken at 3K.

Just as in the previous graph, in Figure 12 MR was generally negative while MC was generally positive. Unlike the measurements taken at 1.9K, here the MR curve is fully negative, while the MC curve is totally positive. In general, for the sample at 3K, the maximum MR obtained in the range was almost -22.3%.

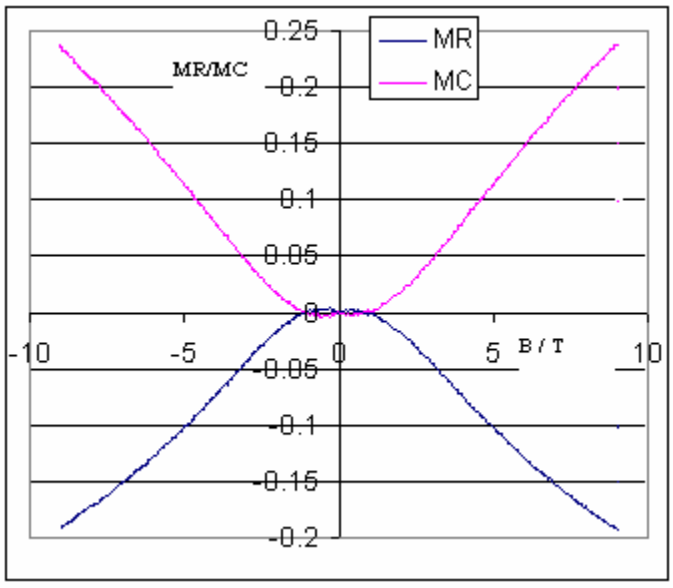


Figure 13: Measurements taken at 5K.

In Figure 13, once again MR is totally negative, while MC is totally positive. The maximum value is about -19.2%.

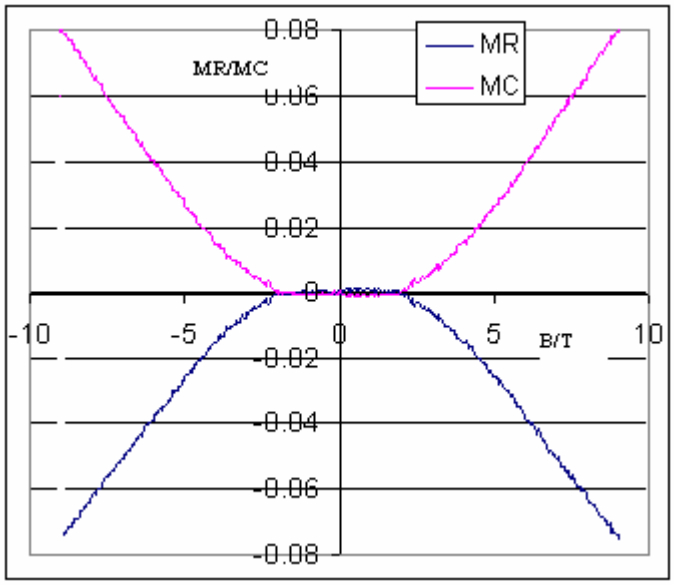


Figure 14: Measurements taken at 10K.

As expected, in Figure 14 MR is negative while MC is positive. The maximum measurement is about -7.4 %

A combination of all the results shows the relationship between MR and temperature more clearly. In general, as the temperature increases, MR decreases. It was expected that the measurements taken at 1.9K would have the highest results while those taken at 10K would have the lowest values. However, the measurements taken at 3K generally show the highest MR results.

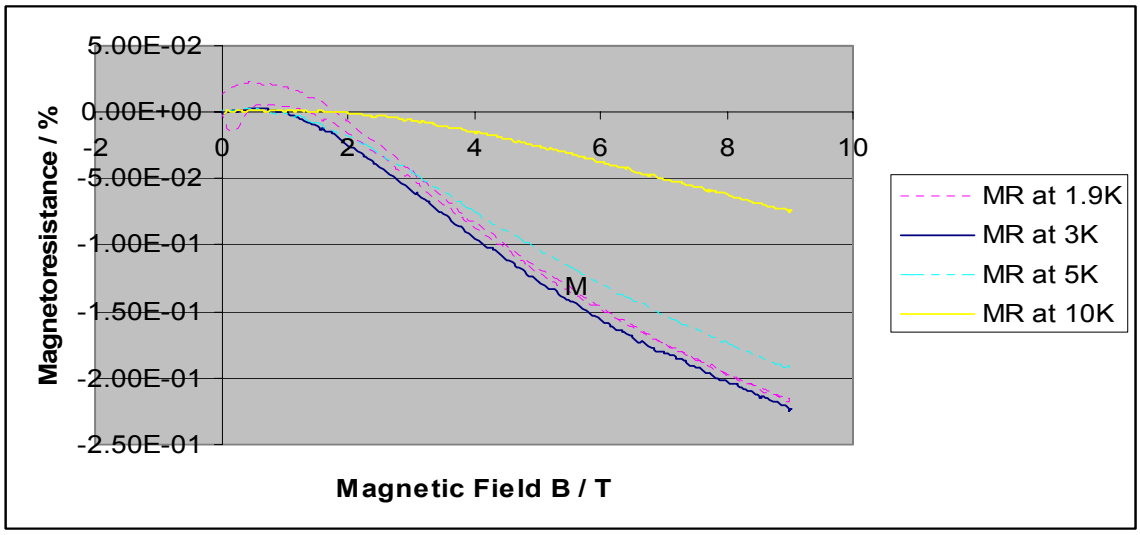


Figure 15: Graph showing combined MR measurements at 1.9K, 3K, 5K and 10K

4. DISCUSSION AND CONCLUSIONS

For most conductive materials, as temperature increases, there is an increase in the vibration amplitudes of the atoms thus increasing the frequency of collision of carriers. As a result, their electrical resistance decreases with temperature increase. However, semiconductors behave differently because of their atomic structure and the nature of their charge carriers. When temperature increases, semiconductors' electrical resistance decreases because the increased amplitudes of vibration may break some bonds between the valence electrons and the lattice atoms, hence releasing more charge carriers. The relationship between resistance and temperature is represented by

$R = R_0 e^{(1/T - 1/T_0)\Delta E/2K}$

R_0 is the resistance of the semiconductor at a reference temperature T_0 (K), usually 273K.

Thus since the shape of Figure 10 shows this relation, it confirms that carbon nanofibers have semiconductor tendencies. This observation is explained by the weak localization phenomenon discussed below.

Figures 11, 12, and 13 showed that, in general, MR is negative. The Bright Model [3] explained this in terms of free carriers. Bright argued that the density of free carriers is increased with magnetic field, causing negative MR. However, this model failed to explain certain observed phenomena. Other authors [4,5,6] turned to weak localization effects which occur in weakly disordered electronic systems like that of the carbon nanofiber because of its turbostratic structure. This weak localization is an inherent interference effect common to a wave-propagated system in a disordered medium. It occurs when the probability of elastic scattering is greater than inelastic scattering of carriers, causing the interference of the electron wave to move in the backward direction. Magnetic fields suppress the phase coherence of backscattered wave. Thus the change in electrical resistance when a magnetic field is applied tends to be negative.

It was also observed that as temperature was increased, MR decreased. The maximum for 10K was -7.4%; for 5K, -19.2%; for 3K, -22.3%; and for 1.9K, 21.8%. MR is more prominent at low temperatures because there is increase in the disorderliness of the system. Contrary to expected results, the values at 1.9K were lower than those at 3K. However, the values for these two measurements were very close. This is most likely because there is not a great change in temperature to affect the carriers immensely. The experiment can be repeated at these temperatures to verify the results and confirm that this observation was not due to experimental limitations. Since the nanofiber also displayed some positive MR at 1.9K, it will be interesting to investigate if the nanofiber undergoes some significant changes at that very low temperature.

In conclusion, the MR of carbon nanofibers pyrolyzed at 1173K is generally negative and increases while temperature decreases.

5. RECOMMENDATIONS

Resistivity, which is a measure of the longitudinal resistance, is usually a better measurement than resistance because it is an intrinsic property. However, resistivity is dependent on the surface area and length of the carbon nanofiber. Time constraints made it impossible to investigate the cross-section of the nanofiber. This should be done as a next step, and its surface area measured. The cross-section can be viewed with a scanning probe microscope operated in tapping mode. Since it is suspected that the cross-section is elliptical [9], the surface area can be calculated by measuring the vertical diameter and the full width at half maxima horizontal diameter. The length of the nanofiber can be measured using an optical microscope.

The nanofiber can also be characterized further using Raman scattering. This will show different peaks and thus the presence and levels of disorder in the nanofiber. These steps would allow us to better understand the MR results.

Finally the results can be compared to MR measurements of carbon nanofibers pyrolzed at other temperatures to observe any trends.

6. ACKNOWLEDGMENTS

I am indebted to Yu Wang for his immense help and constant guidance with this project. I will also like to thank Dr. Jorges Santiago-Aviles for remaining a fountain of knowledge and being ever willing to help me. I am also grateful to Vladimir Dominoko for his help in preparing the samples and to Dr. Jay Kikkawa for his help in taking measurements. Thanks also to Kennedy Gachiri for proof-reading this document. Finally, but most importantly, I will like to express my sincere gratitude to the National Science Foundation for their support through an NSF-REU grant and Microsoft Corporation for their financial support.

7. REFERENCES

1. G.C. Rutledge, M.Y. Shin, S.B. Warner, A. Buer, M. Grimler, and S.C. Ugbolue, A Fundamental Investigation of the Formation and Properties of Electrospun Fibers, <http://heavenly.mit.edu/~rutledge/PDFs/NTCannual00.pdf >. Ac Acessed July 15, 2003.

2. E. H. Putley, *The hall effect and related phenomena*, Butterworths Scientific Publications Ltd, London, 1960, p66-99.

3. A. A. Bright, Negative magnetoresistance of pregraphitic carbons, *Phys. Rev. B,* 20 (1979) 5142-5149.

4. Y. Koike and T. Fukase, Anomalous electric conduction in carbon fibers at low temperatures, *Solid State Comm.* 62 (1987) 499-502.

5. V. Bayot, L. Piraux, J.-P. Michenaud, and J.P. Issi, Weak localization in pregraphitic carbon fibers*, Phys. Rev. B*, 40 (1989) 3514-3523.

6. V. Bayot, L. Piraux, J.-P. Michenaud, and J.P. Issi, Two-dimensional weak localization in partially graphitic carbons*, Phys. Rev. B*, 41 (1990) 11770-11779.

7. Georgia Tech School of Electrical and Computer Engineering, Photolithography, <http://www.ece.gatech.edu/research/labs/vc/theory/photolith.html>. Accessed July 20, 2003.

8. University of California in Berkeley. Four point probe manual, 1994, <http://microlab.berkeley.edu/~ee143/ Four-Point_Probe>. Accessed June 30, 2003.

9. Y. Wang, J. J. Santiago-Aviles, R. Furlan, and I. Ramos, Pyrolysis temperature and time dependence of electrical conductivity evolution for electrostatically generated carbon nanofibers, *IEEE T-NT 2,* (2003) 39-43.

10. Y. Wang, J. J. Santiago-Aviles, Large negative magnetoresistance and two-dimensional weak localization in carbon nanofiber fabricated using electrospinning*, J. Appl. Phys. 94*, (2003) 1721-1727.

Remote Cognosensors: Developing an NIR Imaging Model to Map Brain Function

NSF Summer Undergraduate Fellowship in Sensor Technologies
Prasheel Lillaney, Department of Bioengineering, University of Pennsylvania
Advisor: Dr. Britton Chance

ABSTRACT

Near-infrared (NIR) imaging has provided new insight into optical imaging of human tissue. The purpose of this study is to develop an NIR imaging system that could potentially be used to study cognitive function by measuring the optical parameters (μ_a and $\mu_s`$) of the human prefrontal cortex. The desired system should be a remote sensing system that does not need direct contact with the subject. It should also be able to work over a distance of 50 cm to 2 m from the subject. This study presents a method of remote sensing via a time resolved spectroscopy approach. Two methods of analysis are used: a single photon counting (SPC) method; and the second being a series of gated integrating circuits (Box Car). System design and initial results are shown for both systems. Actual photon migration patterns, from which values of μ_a are calculated, were obtained via the SPC method over a distance of 60 cm. Also, experimental results demonstrating pulse shaping are shown from the Box Car system. Finally, noise level reduction in the SPC method is taken into consideration, along with an option for achieving correct Box Car gate timing.

Table of Contents

1. INTRODUCTION

Human brain function imaging plays a crucial part in determining how the brain works under different conditions. Using techniques such as magnetic resonance imaging and computed tomography scans, it is possible to see different areas of the brain "activate" as a test subject is asked to solve a puzzle or recognize a familiar face. Another technique uses near-infrared (NIR) imaging to try to determine the optical parameters μ_a and $\mu_s`$, which have been shown to correlate to blood volume and oxygenation in human tissue [1].

One of the current NIR methods uses a multi-source, multi-detector array placed in a plastic holder. The holder itself is held in place over the forehead with the use of Ace bandages and Velcro straps. As a result when the subject moves his forehead the placement of the source/detector network shifts slightly relative to the subject's forehead. These movements lead to outlier signals and questions about the reproducibility of the experiment, mainly how well the source/detector array can be replaced over hundreds of trials [2].

Two ideas have been proposed to avoid these problems with head movement during testing. The first minimizes the electronic devices that would be attached to the subject's forehead. Complementary metal-oxide semiconductor (CMOS) technology could be used to minimize the detectors to about the size of a quarter. The detectors could then be placed in any fashion around a central light source (see Figure 1). The whole configuration could be held in place with one bandage tied around the head, and any motion artifacts found in the previous system would be minimized. The detectors would then transmit the data to a remote source where they could be monitored [2].

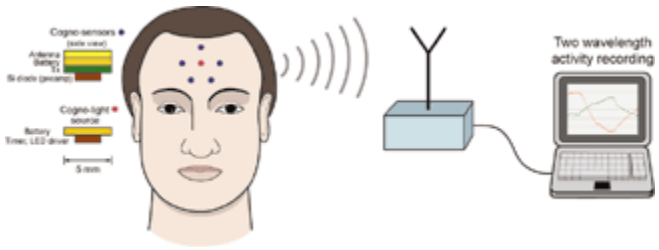


Figure 1 – The small light source is denoted in red and the detectors are shown in blue. The whole system transmits data to a remote source shown to the right [2].

In the second approach, the source and detector are placed anywhere from 50 cm to several meters from the subject. The source is focused through a lens onto the subject. A Fresnel lens then captures the reflected and diffuse photons and focuses them onto a photomultiplier tube (PMT). In this setup, the source would be placed at approximately a 10-degree angle with respect to the subject [2].

This paper will address the design and testing of the above non-contact system. The following sections will focus on the background pertaining to the analysis of the

PMT signal, the design and layout of the non-contact system, experimental results, conclusions, and recommendations for future work.

2. BACKGROUND

2.1 Time Resolved Spectroscopy (TRS) and Time Correlated Single Photon Counting

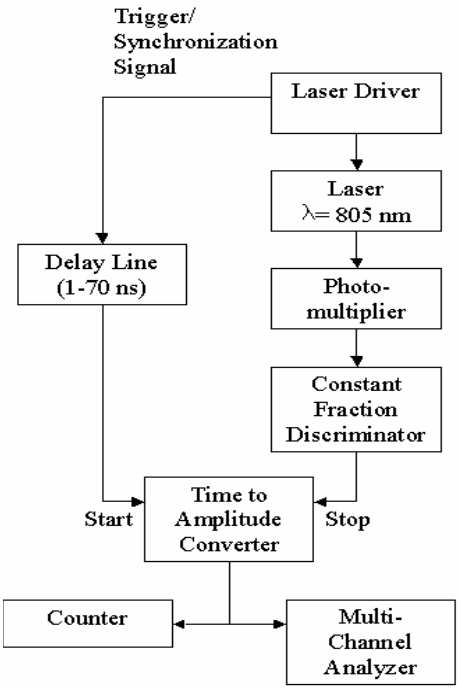


Figure 2 - This is a block diagram for the TRS system showing the progression of the signal through the different components. The CFD, TAC, and MCA can all be bought on an integrated board that can be inserted into a PC so that the photon migration pattern can be displayed in real time as the voltage from the TAC is put into memory by the MCA.

The initial approach used to analyze the PMT signal will be a Time Correlated Single Photon Counting method. The basic principle is that for every pulse from the laser many photons are released into the subject. The job of the PMT, Constant Fraction Discriminator (CFD), and Time to Amplitude Converter (TAC) system is to measure the amount of time each photon takes to arrive at the detector (PMT) with respect to the original excitation event. Once the optical signal arrives at the photo-cathode of the PMT, it triggers the release of electrons that are then cascaded down a series of dynodes to amplify the signal. The signal then arrives at the anode and is then fed into the CFD. The time that it takes the signal to travel from the photo-cathode to the anode is known as the transit time, and as this time becomes shorter the time resolution of the whole system increases [3].

The PMT signal that comes out of the anode still has considerable amplitude jitter and as a result must be sent to the CFD. The CFD creates a constant amplitude pulse for every PMT pulse that is over a certain minimum voltage. The CFD accomplishes this via a system of delays and inverters [see Appendix A]. The data from CFD is then fed into the stop input of the TAC, while the trigger signal from the laser driver is fed into the start input after it travels through a variable delay line. The TAC starts a voltage ramp when it receives the start signal and stops the voltage ramp when it receives the stop signal. Thus the TAC directly correlates time to voltage [see Appendix B]. The output of the TAC is fed into a counter, mainly to check whether the signal is reasonable, and into a multi-channel analyzer (MCA). The MCA converts the output voltage of the TAC into a histogram that displays photon counts versus time (ns), where the time corresponds to how long it took the PMT to detect a photon relative to the time of laser excitation. The photons that diffused farther into the subject would take a longer time to be detected, and as a result the histogram would show a photon migration pattern.

The slope of this photon migration is directly related to the absorption coefficient (μ_a) by the Patterson-Chance-Wilson equation ($\mu_a * c$ = slope, where c is the speed of light) [2].

2.2 Continuous Wave (CW) Cognosensor

Previous work has shown that NIR imaging offers insight into muscle activation and, more importantly, cognitive function. Several systems have used the continuous wave (CW) model, where a constant intensity of light is put into the tissue and the intensity of light leaving the surface of the tissue is recorded [4]. In experiments focusing on skeletal muscle activation resulting from increased load, it was shown that the 730 nm NIR signal decreased because of increased deoxy-hemoglobin (Hb), whereas the 850 nm NIR signal increased because of decreased oxy hemoglobin (HbO_2). Furthermore, once the exercise was stopped the signals began to approach their original values as a result of re-oxygenation (see Figure 3).

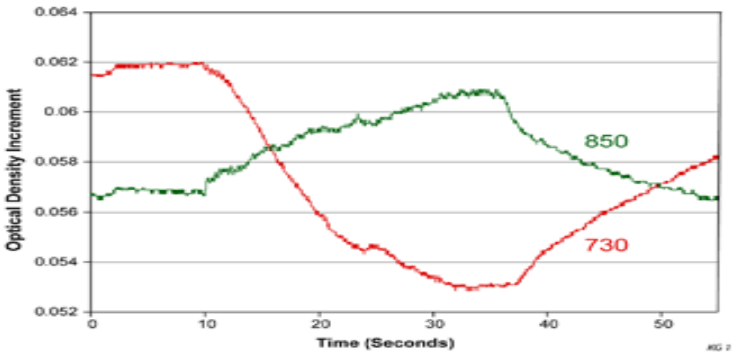


Figure 3 – Illustration of 730, 850 nm NIR signals of skeletal muscle during exercise [5].

Other NIR experiments done with similar instrumentation have shown interesting signals relating to subject behavior. Histograms can be created for numerous positions on the subject's forehead, and data can be collected on activation in those areas over a large number of trials. Interestingly, it has been shown that when a subject lies there is far more activation than when the subject is telling the truth (see Figure 4). It is important to remember, however, that since different people have different characteristic activation patterns results cannot be compared across subjects [2].

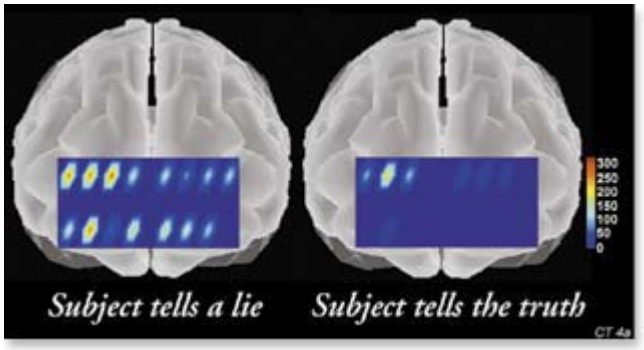


Figure 4 – Illustration of the fact that there is more activation in the prefrontal cortex when a subject lies when compared to the response when the subject tells the truth [2].

2.3 TRS Contact Sensor

The obvious question that arises about the non-contact system is not whether it is possible to get signals similar to those shown above, but whether the system is sensitive enough to detect changes in the optical parameters for a dynamic system such as human tissue. This difficult question can be partially answered by results from a previous TRS contact model (see Figure 5). In this setup the signal from the detector was fed into a single photon counting system (SPC) that had all the components integrated on to one board. The board had two inputs, one for the detector signal and the other for the trigger signal from the source, which in this case was a 780/830nm, 40μW laser driven at 5 MHz. Results from the experiment demonstrated that quantification of absorption changes were accurate to 10^{-3} cm^{-1} with sensitivity to 10^{-4} cm^{-1}. Moreover, these results were mainly limited by laser instability [6]. The task remains to replicate these results for the remote condition, but at least it is known that the TRS system can provide the above-mentioned degree of sensitivity and accuracy.

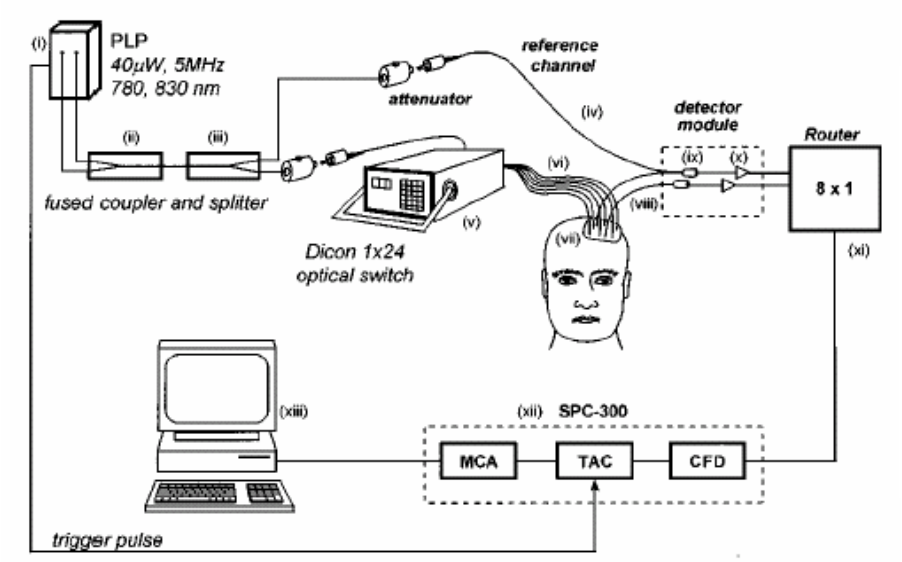


Figure 5 - The laser source is shown to the top left of the diagram. It is fed into the optical switch allowing for variation of source position on the subject's forehead. The detector signal is then fed into the SPC-300 board shown to the bottom right. The results from the multi-channel analyzer (MCA) on the board can be displayed o a PC [6].

2.4 Proposed Model

The model that will be used to carry out the experiment will have the subject at a distance of approximately 50 cm from the source and detector. The source will be put at an angle of roughly 10 degrees with respect to the detector. Focusing the source through a lens is optional and will be done if the signal on the detector side is too weak to recognize. The detector will be placed 10-15 cm behind a Fresnel lens, which will collect the diffuse photons returning from the subject and focus them onto the PMT fiber. The signal from the PMT will be fed into either a TRS system or a Box Car detector (see Figure 6) . (Additional details are provided in section 3.)

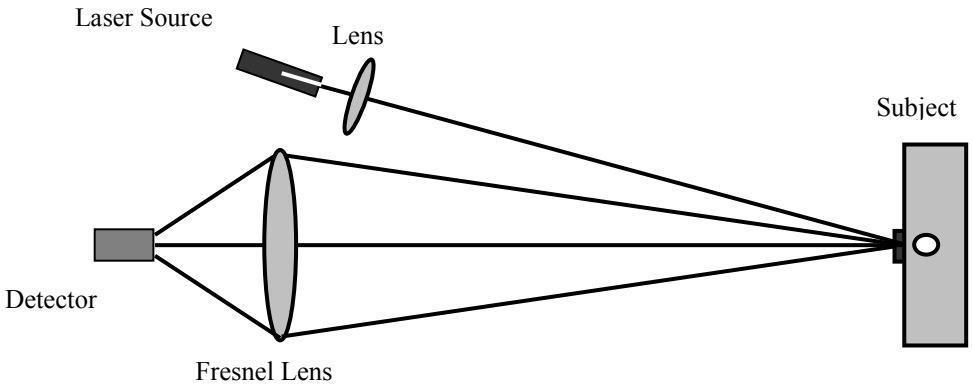


Figure 6 – Illustration of detector/source positioning with respect to the subject

3. SYSTEM DESIGN

3.1 Analysis by Conventional TRS

The TRS system design is fairly similar to that shown in the background section, with a few variations. The driver used is a Hewlett Packard 8082A Pulse Generator on a frequency range of 5 to 50 MHz. The amplitude and FWHM (Full-Width, Half-Maximum) of the pulses are below 1.0 V and 5.0 ns, respectively. It is optimal to minimize the FWHM of the driver so that the instrument response function in the result is also minimized (for further discussion, see section 4). The signal from the driver is split

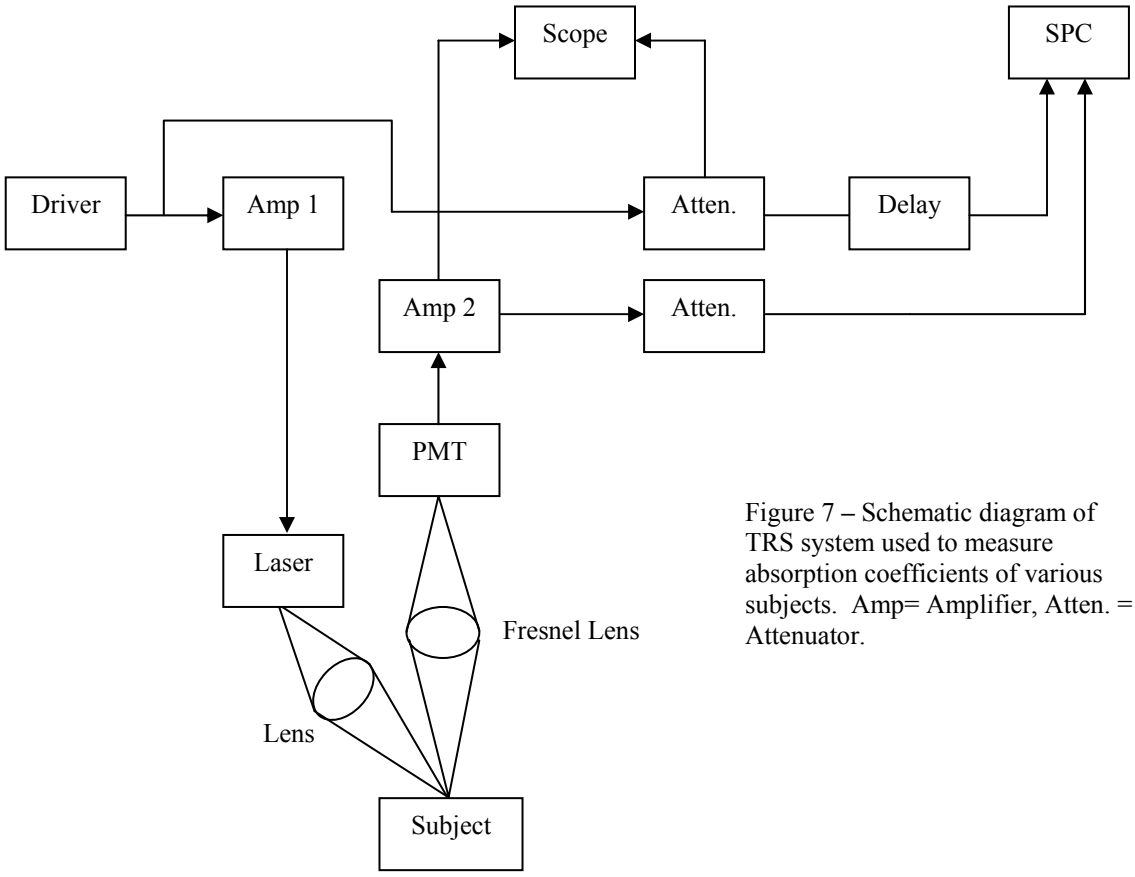


Figure 7 – Schematic diagram of TRS system used to measure absorption coefficients of various subjects. Amp= Amplifier, Atten. = Attenuator.

into an amplifier and an attenuator. The amplifier (Amp 1) used is a Mini-Circuits Monolithic Amplifier (Model HELA-10b) with 12 dB typical gain and 20 dBm (6.3 V p-p for 50-Ω impedance) max power rating at input. This amplifier feeds the driver signal into the laser (780 nm) for a typical optical power output of approximately 1.2 mW. The two attenuators used are identical and are variable from 1 to above 40 dB attenuation, with 1 W max input power and 50-Ω impedance at input. They are used to keep the signals being fed into the SPC 300 board below 80 mV p-p, so the attenuation used can range from 10 to 15 dB. The SPC board, purchased from Edinburgh Instruments Ltd, creates the photon migration pattern and displays it on a PC. The attenuator used for the driver signal is fed into a variable delay line (1-70 ns delay via coaxial cable with 20 ps accuracy) before it enters the SPC board, so that the position of the photon migration pattern can be shifted to allow complete visualization of the photon decay.

The PMT used in the system is a Hamamatsu R5600U and is operated in the 800 V range. The PMT signal is fed directly to a Hamamatsu C5994 amplifier (Amp 2), which is made especially for use with PMT signals (10 dBm max power input, 36 dB gain, 50 KHz to 1.5 GHz frequency range). The signal from this amplifier is split into two. One half goes to an attenuator and eventually to the SPC board. The other half is monitored on the oscilloscope in order to make sure that there is a live PMT signal during testing. It may seem odd that the PMT signal is being amplified and then immediately attenuated before entering the SPC board. The design is set up this way because the SPC board is very sensitive to PMT signals and can create photon migration patterns with a PMT signal that is less than 5 mV. Yet, it is difficult to monitor such a small signal on the oscilloscope. As a result the signal must be amplified first so that it can be monitored, but then attenuated at the next step so that the SPC board does not overflow. The cables used to connect all the components are coaxial BNC cables with 50-Ω impedance. Finally, a 6.5-inch diameter Fresnel lens is placed 10-15 cm in front of the PMT fiber in order to capture the optical signal from the subject and focus it onto the fiber. Another lens maybe used in order to focus the laser source if deemed necessary. (see Figure 7).

3.2 Analysis by Box Car Detectors

The TRS system mentioned above is limited by the pulse pile-up distortion. These errors create an upper limit of 0.1 for the ratio of detected photons to the number of excitation events from the laser. For higher values of this ratio mathematical corrections to the data must be made to remove the error [3]. In order to avoid this problem of having an upper limit placed on the number of photons that can be detected, the Box Car detector system can be used instead of the SPC 300 board.

The principle of the Box Car system relies on the concept of gated integrators (see Figure 8). The integrators being used are simple resistor capacitor networks (R_1, C_1, R_2, C_2, R_3) and are on a timing schedule, which is controlled by a series of delay lines, flip-flops (F.F.), and pin diodes. For example, the first gate is turned on when the impulse from the pulse generator arrives at the first flip-flop. At this point the flip-flop transmits

the pulse to the pin diode in front of the first integrator. Up to this point the pin diode was acting like a closed switch, but upon receiving the impulse from the pulse generator it conducts current and acts as a closed switch. As a result the signal from the PMT is allowed to travel across the pin diode and enter into the first integrator.

The integrator then reports a DC voltage to the first channel of the analog-to-digital converter, which is then stored to memory. Hence, the pulse generator was simply

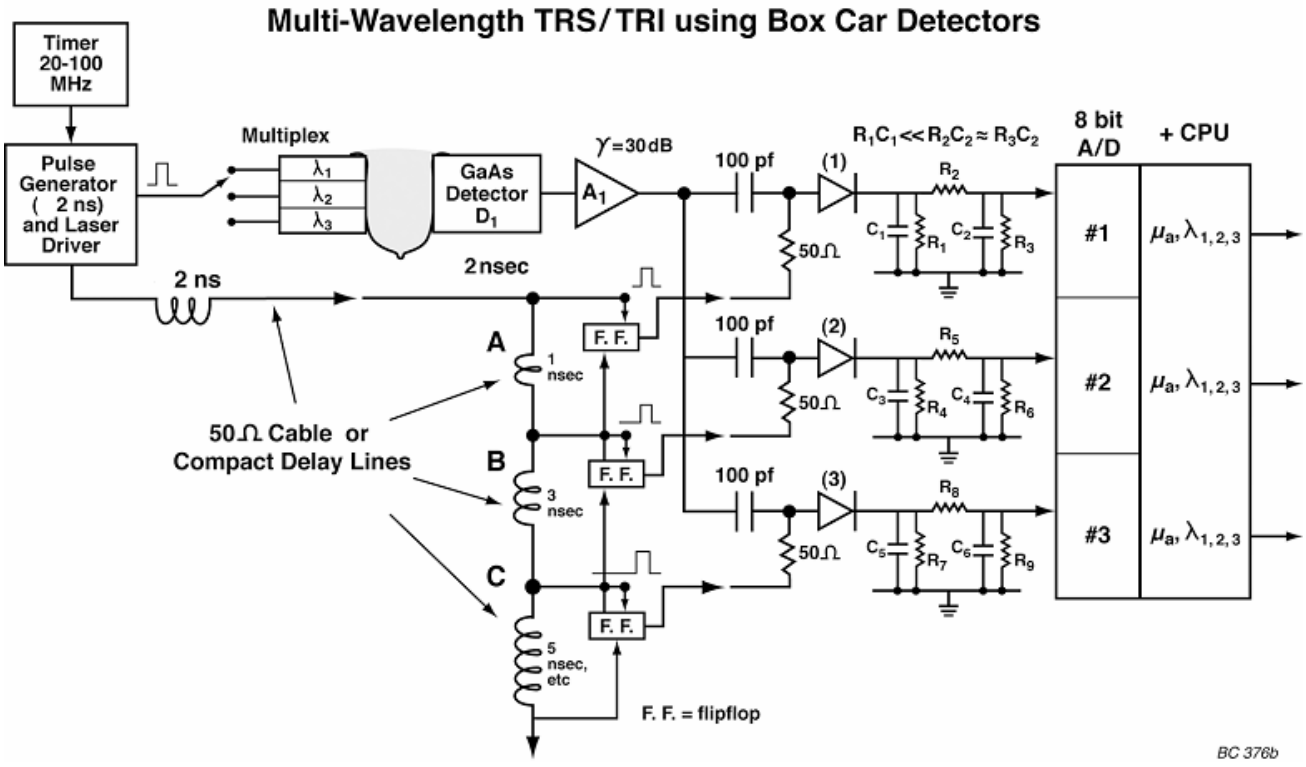


Figure 8 – Illustration of Box Car detector system. #1,2,3 stand for gate 1,2,3 and so forth. The element before each resistor, capacitor integrator is a pin diode. F.F stands for flip-flop and GaAs detector (D_1) shown is a Gallium-Arsenide PMT.

acting as a "platform" for the PMT signal to be integrated upon. In order to turn off the first integrator, the original impulse from the pulse generator is also fed though delay line A (shown here as 1 ns). After traveling through the delay line the impulse arrives at the second input of the flip-flop. Once the flip-flop receives the second delayed impulse from the pulse generator it no longer transmits any signal. As a result the pin diode again acts as a closed switch, and the first integrator is turned off. Yet, the same pulse that arrives after a 1 ns delay to the second input of the first flip-flop now arrives at the first input of the first flip-flop and turns on the second integrator according to the same principle. The second flip-flop receives a second input after a 3 ns delay (delay line B). This process can go on for as many gates as needed to reconstruct the logarithmic decay curve in order to measure the absorption coefficient of the subject. The delay lines get longer for the sequential gates because the PMT signal is also getting weaker, and in order to preserve equal signal to noise ratio across all the gates the integration time must increase for the later gates (see Figure 8).

The major drawback to the system is the difficulty of finding a flip-flop or other logic device with a response time that can handle the 1 and 3 ns timing needed to run the first few gates. The advantages to the system, on the other hand, are considerable. First, the problem of pulse pile-up can be avoided, since the system is not using a conventional SPC approach. Next, the system can be miniaturized and replicated for as many gates and channels as are needed. For example, it is feasible to have a 3-wavelength source, 8 detectors, and multi-position scanning system via Box Car analysis. It should be noted that Figure 8 shows the system in a contact breast imager model, but it can easily be interchanged with the non-contact NIR cognosensor system. [Note: All Box Car specifications credited to Chance Lab].

4. EXPERIMENTAL RESULTS

4.1 TRS Results

The following data were recorded with the SPC 300 board for various remote distances of approximately 60 cm from the source/detector system with a high voltage of approximately 900 volts applied to the PMT. Figure 9 shows a linear plot of photon counts versus time across three trials. The first two trials were performed with the subject (phantom 1) in a position to minimize the specular reflection response. Trial 3 was performed with a slight change of angle on the phantom to increase the reflective response.

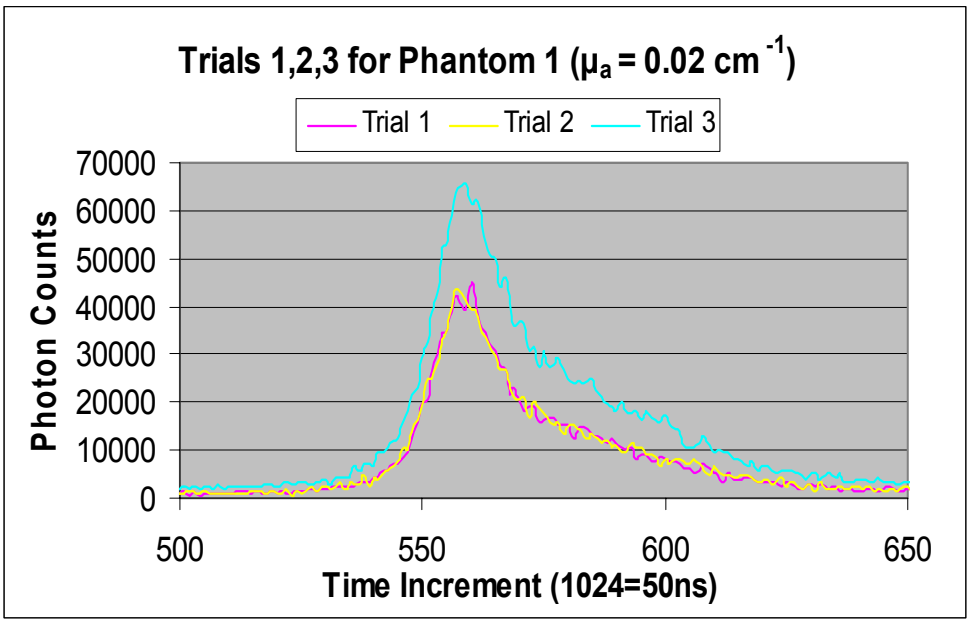


Figure 9 – Graph of photon counts versus time for Phantom 1 at a remote distance of 60 cm.

Figure 10 shows the same data as above except with the y-axis changed to a logarithmic scale. At time = 575 the reflection has ended, and the true photon migration pattern is seen.

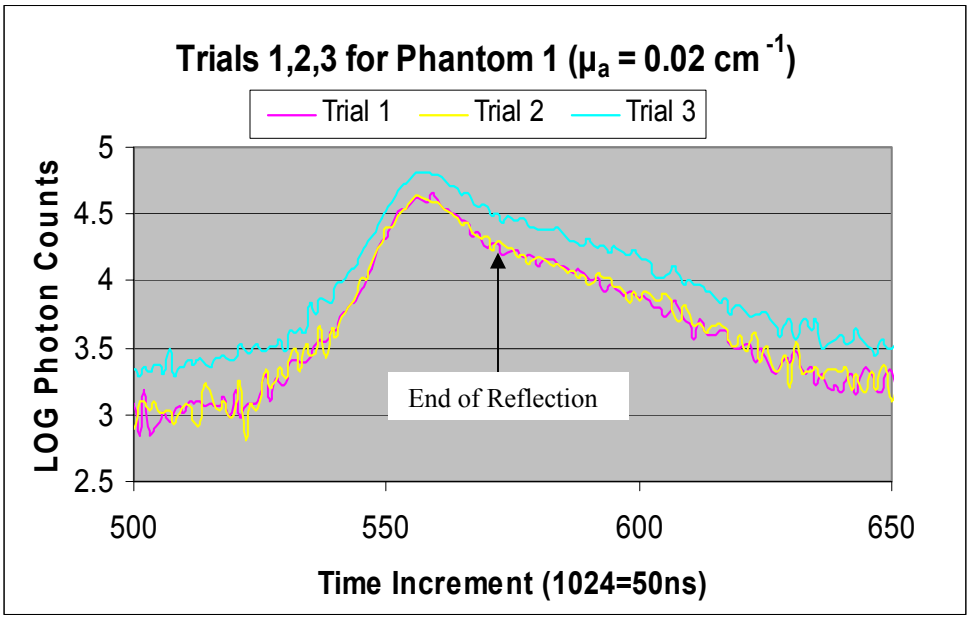


Figure 10 - Graph of log photon counts versus time for Phantom 1 at a remote distance of 60 cm.

Data points 575-625 were used to fit via regression and determine the slope of the logarithmic decay for the photon migration pattern (see Figure 11). Only trial 1 is shown in the figure for convenience (statistics on trial 1 and other two trials are in Appendix C). Converting the decay slopes across all three trials into absorption coefficients yields an average μ_a of 0.01387 cm $^{-1}$ ± 4.7% (95% confidence).

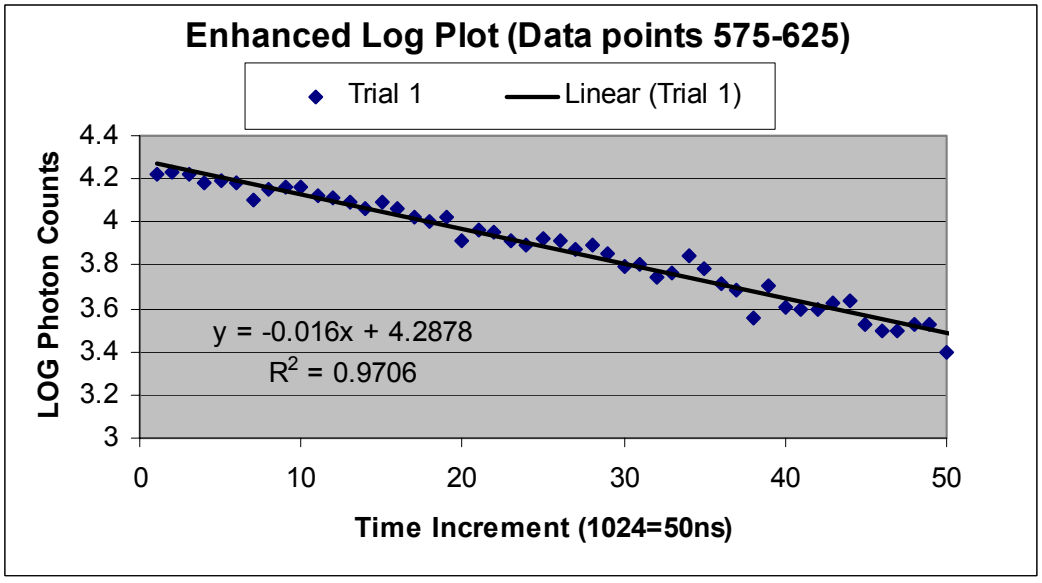


Figure 11 – Selected points for log plot of trial 1 used to perform regression analysis.

The above three trials displayed that the photon migration pattern and specular reflection from Phantom 1 flowed into each other, meaning that there was no time difference between the two responses. However, by changing the attenuation value on the PMT signal entering the SPC 300 board, increasing the PMT voltage to near 1100

volts, and changing to a much higher absorption phantom (μ_a= 0.37 cm $^{-1}$), it was possible to separate the two signals (see Figure 12). The signal was too noisy, however, to perform regression analysis (the standard error of slope was greater than 10%). This problem is discussed further in section 5.

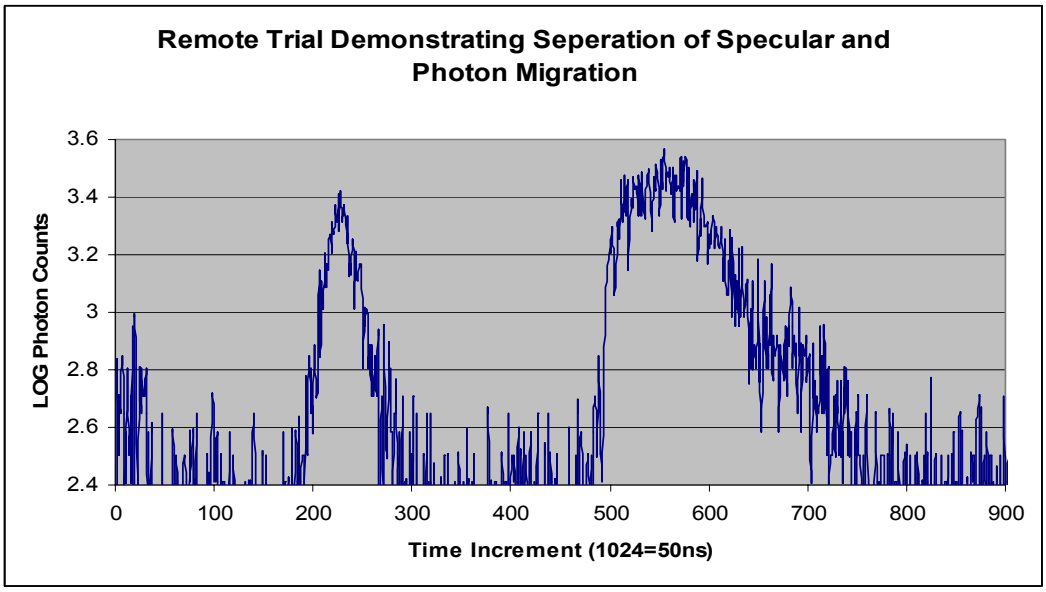


Figure 12 – Graph of separation of specular reflection and photon migration pattern when attenuation, PMT voltage, and phantom was changed.

4.2 Initial Box Car Results

Although no flip-flop or logic device was found that was fast enough to handle the timing controlling the gates in the Box Car system, it was still possible to design a 1-gate integrator without the pin diode, delay line, and flip-flop timing system. Figure 13 demonstrates the voltage dampening out of the output of the integrator (R_1= 50 – Ω, C_1= 100 pF, R_2= 50 – Ω, C_2= 3 nF, no R_3 used) when a 1 V p-p impulse was sent through at various frequencies. The voltage was decreased by a factor of 10 at 5.2 MHz.

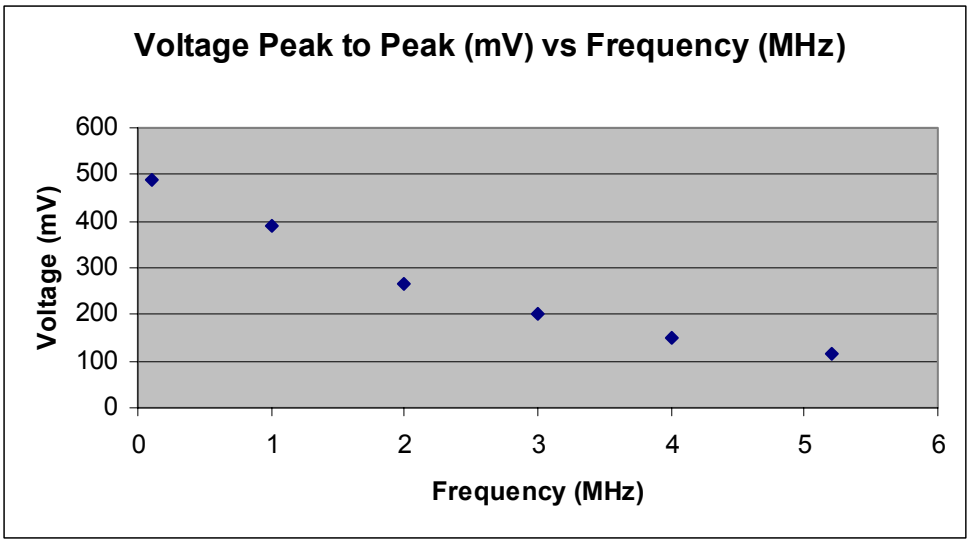


Figure 13 – Graph demonstrating voltage dampening out of 1 gated Box Car.

One of the main purposes of the Box Car system is to stretch the input pulses as they go through the integrator, making the result at the analog-to-digital converter more stable. Figure 14 demonstrates the above-mentioned pulse stretching. At a frequency of 5.2 MHz the input pulse had a FWHM that was 37% as long as the FWHM of the output pulse, thus showing a considerable amount of stretching.

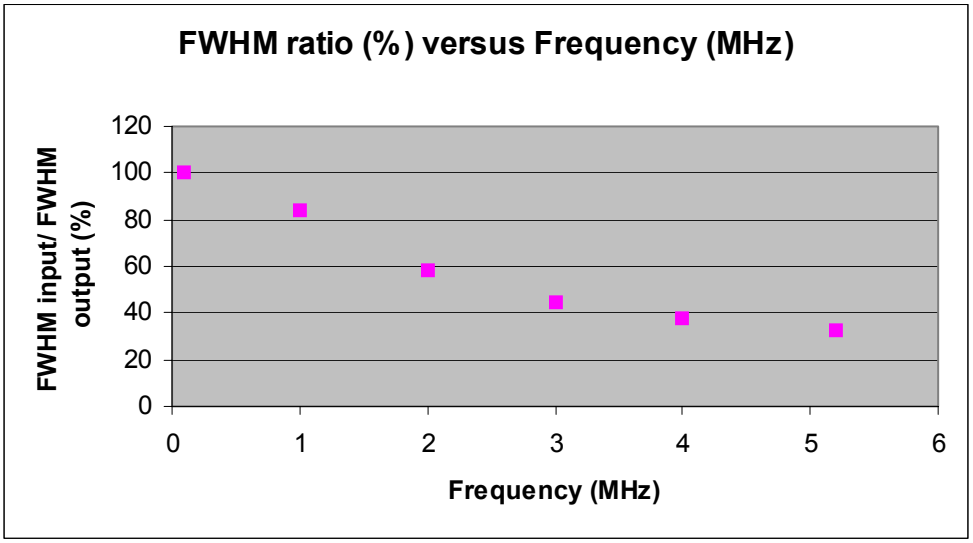


Figure 14 – Graph demonstrating pulse stretching out of 1 gated Box Car.

5. DISCUSSION AND CONCLUSIONS

The results from the TRS system for Phantom 1 (μ_a of 0.01387 cm $^{-1}$ ± 4.7%) show roughly 30% error from the actual value of 0.02 cm^{-1}. This error would suggest that perhaps the signal being analyzed for the decay slope is not the true photon migration pattern. However, it is important to remember that the signal measured by the SPC 300 board is a convolution of the true signal and another signal known as the instrument

response function. The instrument response function is a result of the pulse that is driving the source for the experiment (for this experiment, the Hewlett Packard 8082A Pulse Generator). Therefore, in order to measure an absolute value for μ_a the data must first be de-convolved, for which there are various methods [6]. Since it was shown in the results that it is perhaps possible to further separate the specular response and the actual photon migration pattern (see Figure 12), no steps were taken to de-convolve the data since it is not yet known which data set is optimal.

The results from the Box Car system demonstrated that pulses in the frequency range of 1-5 MHz could be handled with the 1-gated integrator in the appropriate fashion (see Figures 13 and 14). If frequencies of up to 50 MHz were applied, one would simply have to experiment with the values of C_1 and C_2 to achieve the appropriate pulse shaping for the analog-to-digital converter.

6. RECOMMENDATIONS

Although it was shown that data could be seen over remote distances with the TRS system, significant work is needed to optimize the system. First, further tests should be done to try to minimize the noise seen when the specular reflection and photon migration pattern are separated. If noise can be reduced, it will be possible to confirm that the second peak that appears is actually a photon migration pattern and not just an artifact from the SPC 300 board or from another component in the system. The first step should be to move the SPC 300 board to another PC and see if the same results can be obtained.

The Box Car system remains largely untouched. However, one option remains to achieve the nanosecond timing needed for the gates. It may be possible to program an emitter coupled logic (ECL) chip or purchase an ECL flip-flop that bhaves in the fashion needed to control the integrator gates.

7. ACKNOWLEDGMENTS

First, I would like to thank my project advisor, Dr. Britton Chance, for his encouragement, advice, and most importantly his patience. I would also like to thank Dr. Jan Van der Spiegel, the SUNFEST committee, the National Science Foundation, and the Microsoft Corporation for making it possible for undergraduates to pursue meaningful research projects. Lastly, I would like to give special thanks to Leonid Zubkov for developing the laser driver and Zhongyao Zhao and Xavier Intes for their technical expertise and advice with the TRS system.

8. REFERENCES

1- V. Ntziachristos, X. Ma, and B. Chance. Time-Correlated single photon counting imager for simultaneous magnetic resonance and near-infrared mammography. *Rev. of Sci. Instruments, Vol. 69, No. 12,* December 1998, pg 4221.

2- B. Chance, S. Nioka, and Y. Chen. Shining New Light on Brain Function. *OE Magazine,* July 2003, pg 16-19.

3- D.V. O'connor and D. Phillips. *Time~correlated Single Photon Counting.* Academic Press, London, 1984. Pg 103,164,165.

4- F.E.W. Schmidt. *Development of a Time-Resolved Optical Tomography System for Neonatal Brain Imaging.* Thesis submitted at University of London, November 1999. Pg 59,60, 102.

5- K. Gangavarapu, Q. Liu, H. Liu, J. Liu, S. Nioka, and B. Chance. *Future Applications of Nanotechnology in the Field of Functional Brain Imaging*. DOE Nanoscale Science Research Center Workshop, Feb. 26-28, 2003.

6- V. Ntziachristos, X. Ma, A. G. Yodh, and B. Chance. *Multichannel photon counting instrument for spatially resolved near infrared spectroscopy. Rev. of Sci. Instruments, Vol. 70, No. 1 ,* December 1999, pg 194, 200.

9. APPENDICES

Appendix A : CFD Principle

The CFD takes the input pulse from the PMT and splits it into two signals. The first signal is attenuated and the second is inverted and delayed. The pulses are then added together creating a bipolar signal that has a zero crossing level. The zero crossing level is then used by the CFD as the point to create the new impulse, which is independent of the original amplitude of the input pulse (see Figure A1). The CFD can also be set to reject any input pulses that have amplitudes falling outside a variable window. This setting allows for rejection of input pulses that are caused by double triggering or are part of the dark current of the PMT [4].

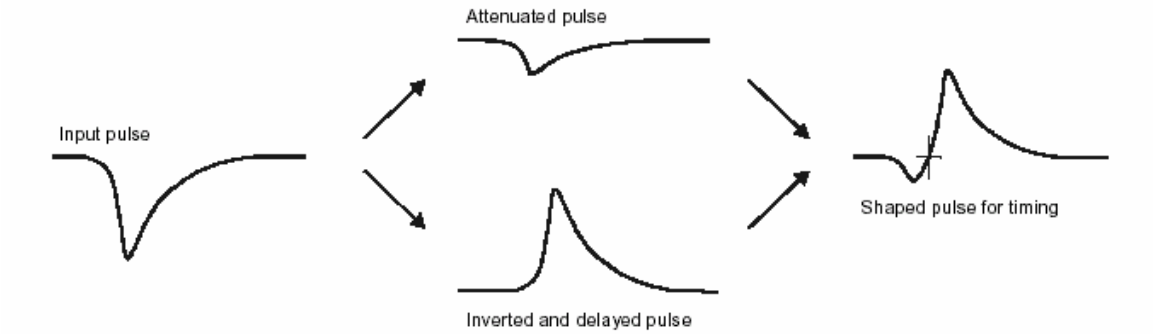


Figure A1 - Diagram of CFD timing principle [4].

Appendix B : TAC Principle

The TAC works with two inputs, a start input and a stop input. Once the TAC receives a start impulse it creates a voltage ramp (after a certain characterstic time delay inherent in the TAC being used). When the TAC receives the corresponding stop impulse it will stop the voltage ramp. Hence, the amplitude of the voltage ramp will be directly proportional to the time between the start and stop impules. Therefore the TAC takes time and directly correlates it to a voltage (see Figure B1).

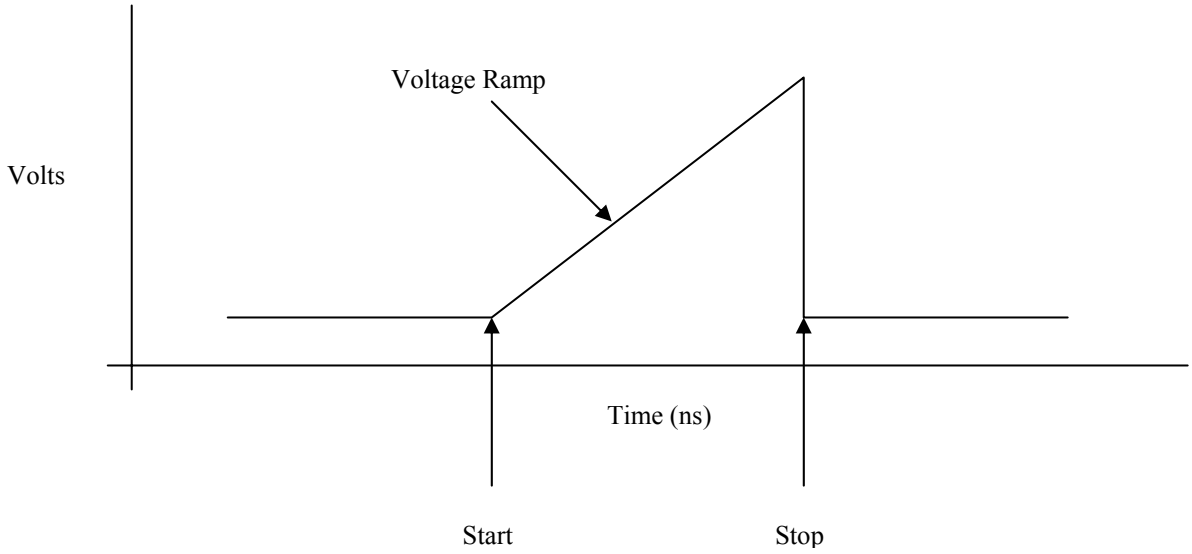


Figure B1 - Illustration of ideal TAC function when start and stop inputs are provided that are separated on the nanosecond time scale.

Appendix C : TRS Regression Analysis

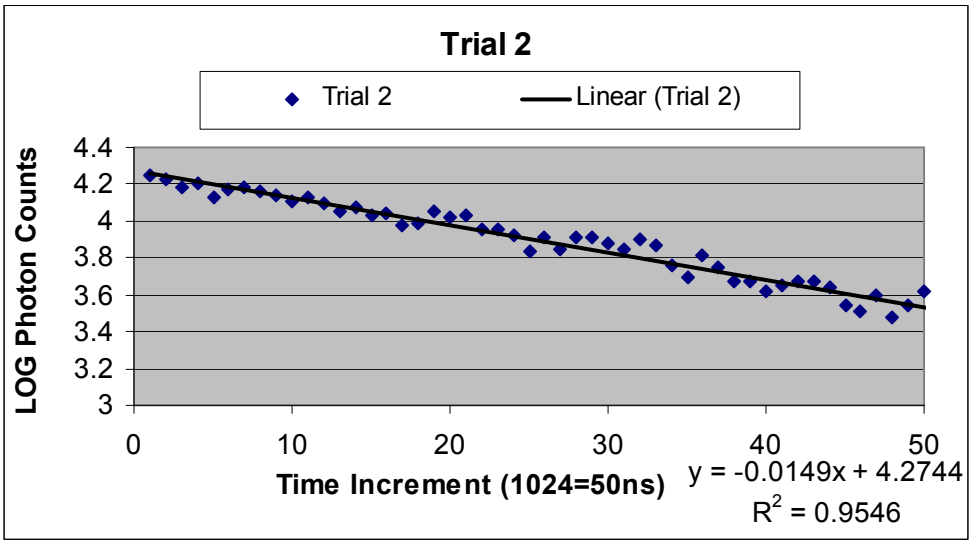


Figure C1 – Graph of points used to do regression analysis on Trial 2 for Phantom 1. Points 575-625 were used and the slope and R^2 value are listed to the bottom right.

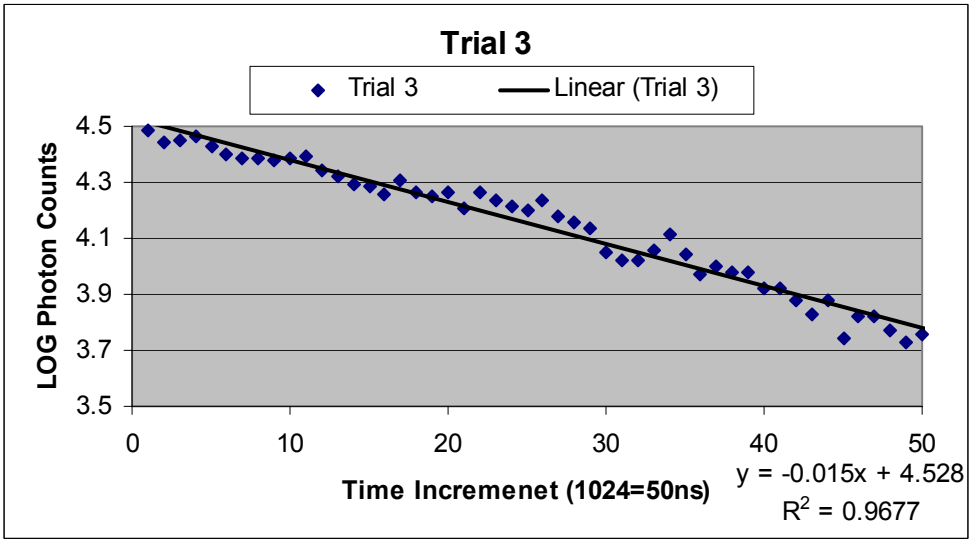


Figure C2 – Graph of points used to do regression analysis on Trial 3 for Phantom 1. Points 575-625 were used and the slope and R^2 value are listed to the bottom right.

Table C1 : Regression statistics for all three trials on Phantom 1.

Statistic	R^2 Value	Max Slope (95% CI)	Min Slope (95% CI)	Standard Error
Trial 1	**0.971**	**-0.01522**	**-0.01682**	**0.000398**
Trial 2	**0.955**	**-0.01392**	**-0.01578**	**0.000463**
Trial 3	**0.968**	**-0.0142**	**-0.01577**	**0.000391**

CARBON NANOTUBE-BASED BIODETECTION OF TRIIODOTHYRANINE

NSF Summer Undergraduate Fellowship in Sensor Technologies
Enrique Rojas
Department of Physics and Astronomy – University of Pennsylvania
Advisor: Charlie Johnson

ABSTRACT

We report biodetection of the thyroid hormone triiodothyranine (T3) using carbon nanotube field-effect transistors. T3 molecules bind non-covalently the semiconducting nanotube and donate electrons to the nanotube lattice, thereby altering its electronic characteristics via chemical doping. The effect is semi-reversible by rinsing the transistors with the T3 solvent without the biomolecule. The nanotube transistors are found to have no electrical response to the solvent itself. The results demonstrate that T3 can have an electrostatic effect on a one-dimensional system and may suggest an electrical interaction between T3 and DNA.

TABLE OF CONTENTS

1. INTRODUCTION

Since their discovery in 1996[1], it has been understood that the mechanical, thermal, and electronic properties of carbon nanotubes distinguish them as one of the most interesting materials on earth. Not only are nanotubes the strongest material ever discovered, they also are among the best conductors of heat and electricity. As condensed matter science and engineering has increasingly moved toward the "nano-" world, carbon nanotubes have become one of the most widely researched nanomaterials.

As laboratory methods become more refined, it is becoming easier to isolate and manipulate single-walled carbon nanotubes (SWNTs) as opposed to the bundles and multi-walled nanotubes (MWNTs), the form in which they were originally dis covered. This improvement is particularly favorable for studying and engineering electronic transport of nanotubes since transport measurements of bundles and MWNTs are difficult to check against theory and SWNTs represent the smallest possible scale for electronic devices. The versatility of SWNTs has been exploited in several different applications. SWNTs have been incorporated into prototypes of field effect transistors,[2] atomic force microscopy tips,[3] and chemical sensors.[4] Recently, SWNTs have been applied as key components in nanoscale biodetectors.[5]

Carbon nanotube-based biodetection of various enzymes and proteins has been reported in the literature. We choose to use the hormone triiodothyranine (T3) as a target molecule for our experiments – a brief background illuminates our motivation. T3 is a hormone produced in the thyroid gland that is essential to human development and metabolism. Deficiency during fetal development results in permanent mental impairment to the child. In adults, symptoms of deficiency are severe but can by treated through drug therapy. T3 takes its active form after entering the cells nucleus and is known to interact with DNA, but it is not known through what biological mechanism information is transferred between T3 and DNA. Furthermore, the structure of the T3 molecule raises a question – Why are the iodine atoms necessary? T3 is formed, in part, by two benzene rings decorated with three iodine atoms. The role of the iodine atoms is not understood, especially since there are so many iodine-deficient areas in the world whose population is unable to produce enough thyroid hormone. Why hasn't another substituent replaced the iodine atom evolutionarily?

One suggested answer is that iodine is essential to the *electrostatic* interaction between T3 and DNA. Unfortunately, the electronic properties of DNA are not well understood, and transport measurements are still in their infancy. Carbon nanotubes, though, provide a possible model for DNA as a 1-dimensional conductor, at least to see if the T3 molecule as a whole has some electronic properties.

The motivation for this project, then, is twofold. First, biodetection of T3 by SWNTs will increase the capability of nanoscale biosensors and confirm carbon nanotubes as useful biosensor elements. At the same time we will learn something about the role of thyroid hormone in the body, and whether or not it can act as an electrostatic gate or a dopant for DNA.

2. BACKGROUND INFORMATION

2.1 Carbon Nanotube Physics

Nanotubes are of a family of molecules including diamond, graphite, and buckminsterfullerene (C60). Structurally, a SWNT is equivalent to a sheet of graphene rolled up to form a cylinder. Experimentally, SWNTs used in our laboratory are 1-3 nm in diameter and as long as several hundred μm. A class of SWNTs can be uniquely defined by how its hexagonal graphene lattice orients itself with respect to the axis of the tubes. Specifically, a pair of indices, (n,m), known as the wrapping vector indicates the chirality of the nanotube in terms of the unit vectors of graphite (see Figure 1).[6]

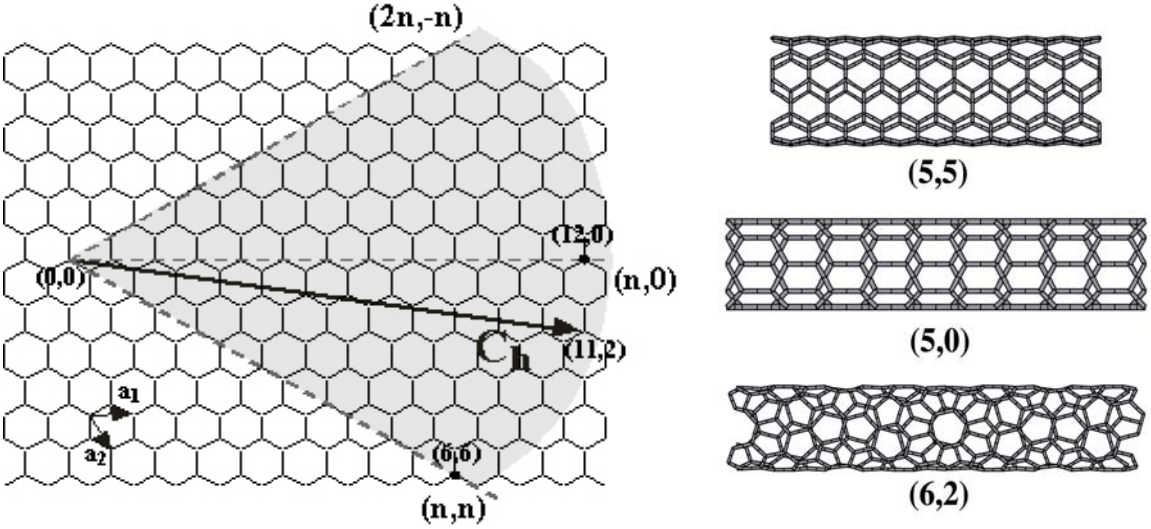


Figure 1 (Left) The wrapping vector (n,m) specifies the so-called chirality of the nanotube. These coordinates define not only the circumference and geometry of the nanotube, but also its electronic properties. (Right) Three examples of SWNTs.

One reason carbon nanotubes have warranted so much investigation is their electronic versatility. SWNTs can behave as metals or semiconductors, depending on their chirality. The band structure of graphene is interesting in that it touches periodically at six Fermi points making it a zero-gap semiconductor. This six-fold symmetry is a result of the hexagonal carbon lattice. The hexagon in reciprocal space that is formed by the Fermi points is known as the Brillouin zone. When the sheet is wrapped up to form a tube, waves become confined along the circumference of the cylinder, and a boundary condition is imposed. Only certain energy states, corresponding to waves that satisfy this condition, are allowed in that dimension. The effect in k-space is that the Brillouin zone becomes sliced along at discrete k_y values. If the allowed states intersect with the Fermi points, the conduction and valence bands are continuous, and electronically, the nanotube behaves as a metal. Conversely, if the allowed states miss the singularities, the nanotube acts as a semiconductor. Experimentally, the range of resistances of semiconducting SWNTs is ~0.3-100 MΩ, while that for metallic SWNTs is about 10-500kΩ. In practice, resistance depends not only on the chirality of the nanotube, but also on the nature of the electrical contacts to the tube and the environment of the nanotube.

2.2 Triiodothyranine

Thyroxine is a hormone produced in the thyroid gland. Its most obvious structural feature is the iodine atoms that bond specifically to the benzene rings. Immediately after it is synthesized in the thyroid, four iodine atoms line the molecule and are subsequently removed by a series of deiodinations. After deiodination, the iodine atoms proceed to the urine unless the thyroid recycles them. That is, they play no other physiological role. The most active form of thyroxine is T3 (see Figure 3), which is known to alter gene expression via a receptor molecule in an interaction with DNA. It is not understood how this interaction is carried out but the presence of the iodine atoms suggests that perhaps the T3 molecule induces charge transfer to the DNA, thereby altering its conductivity.

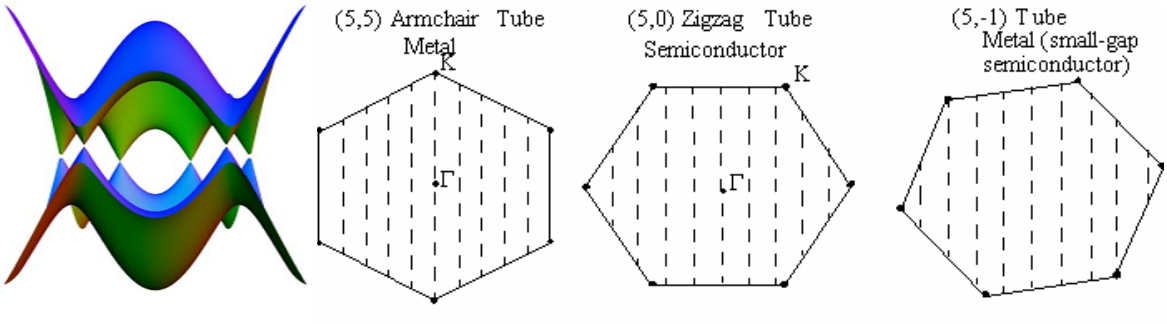


Figure 2 (Left) A diagram of the 3D conduction and valence membranes of graphene. Those points at which the two membranes touch are the Fermi points. Together, the Fermi points form the vertices of the Brillouin zone. (Right) Only certain states, represented by the dotted lines, are allowed when the boundary condition for the cylindrical nanotube is applied. If the lines pass through the Fermi, point, the tube is a metal. Otherwise the tube is a semiconductor.

An *in situ* experiment that can properly model the environment in which DNA is biologically functional has yet to be designed. A more robust 1D system for which the electronic properties are better understood is the SWNT. Additionally, SWNTs are a particularly relevant model for DNA the aromatic ring structure of the carbon nanotube resembles the pi-stack geometry of DNA.

3. THE CARBON NANOTUBE BIOSENSOR

The geometry of the carbon nanotube biosensor looks exactly like that of the carbon nanotube field effect transistor (FET). Nanotubes are grown via chemical vapor deposition (CVD) similar to the procedure described by Hafner et al.[7] Twenty drops of a 100 mg/L catalyst solution containing particles of iron nitrate, $Fe(NO_3)_3 \cdot 9(H2O)$, in isoproponal are spun onto a p-doped silicon wafer covered with 400 nm of silica. The wafers are heated to 900° C in an Ar atmosphere and growth is then carried out for 15 minutes flowing 3500 sccm CH_4 and 450 sccm H_2. The wafers are then cooled flowing Ar.

Electrical contacts of the nanotubes are made using a novel electron beam lithography technique and thermal evaporation. Several hundred pairs of source and drain leads, 10 nm Cr followed by 30 nm Au, with a gap of ~1 μm between them, are patterned at random on the substrate wafer. We are able to control the nanotube density on the wafer so that we consistently get ~10-12 circuits with one semiconducting SWNT

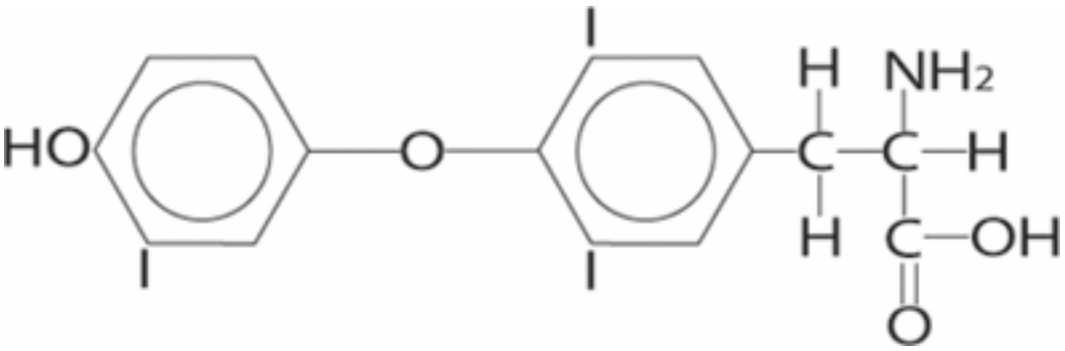


Figure 3 The structure of the T3 molecule.

for every ~150 pairs of leads we write, while 30+ other pairs will have metallic SWNTs or multiple SWNTs completing a circuit, and the remaining pairs will be empty leads. The p-doped silicon serves as a backgate separated by the insulating silica.

The semiconducting SWNTs are then externally contacted and characterized. The response of the conductance of a semiconducting SWNT to changes in the backgate voltage is the most important characteristic of the FET to our biodetection scheme. A simplified classical explanation for this response is as follows. For a p-type semiconducting SWNT, one where holes carry the current, applying a negative voltage to the gate attracts more holes to the nanotube, making it easier to carry current, and thereby increasing the conductance. On the other hand, positive gate voltages will deplete positive charges, leaving none to carry current, and the current will be suppressed. The gate, then, acts like a rheostat for the current where for a given bias voltage we can turn the current through the SWNT on and off. This is a transistor, called a field effect transistor because the electric field between the backgate and the nanotube is responsible for the changes in conductance of the tube. An n-type material exhibits the opposite gate response, since the current is carried with electrons. All of the devices we measured were p-type semiconducting SWNTs, with the exception of one that was ambipolar, with n-carriers conducting slightly better than p-carriers at saturation.

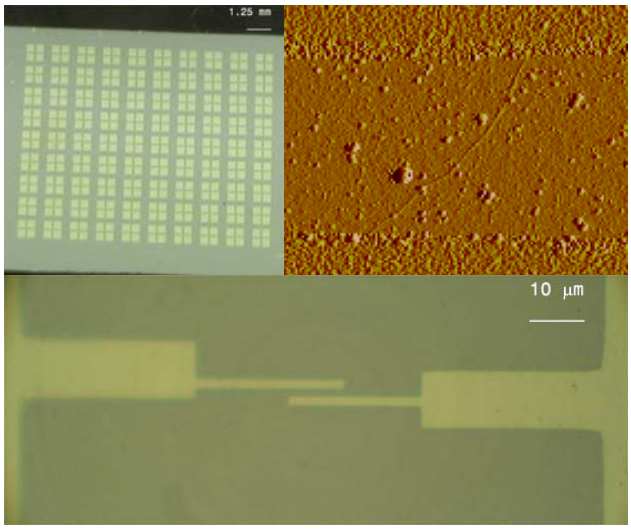


Figure 4 (Top Left) 180 FET devices patterned using ebeam lithography on ~1cm^2 chip. (Bottom) Close-up. (Top Right) AFM image of SWNT connecting leads. The spacing between the leads is 1 µm.

T3 is applied locally to the FETs in solution through a microcapillary tube (World Precision Instruments, Inc.). Drops of ~10 nL-0.1 µL of T3 solution can be held over the device for an indefinite amount of time by applying a back pressure in the capillary equal to the vapor pressure of the drop.

Changes in the gate response of a device during and after application of T3 shed the most light on how the T3 molecule interacts with the nanotube. The hysteresis in the gate curves can be explained by charge injection from the nanotube to the substrate (For an example of this hysteresis, see Figure 6).[8] At large negative gate voltages, positive

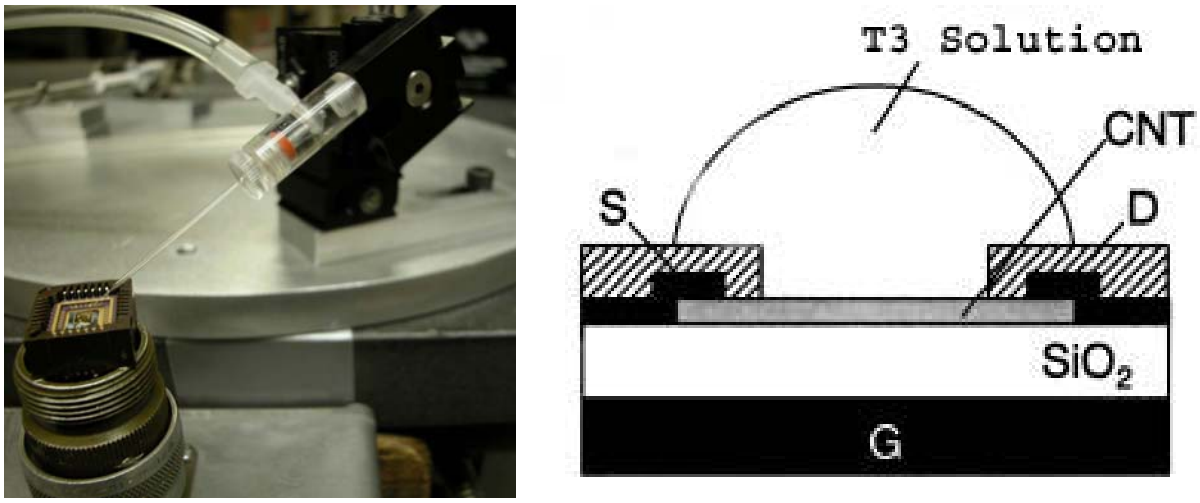


Figure 5 (Left) Microcapillary apparatus designed to apply drops as small as 10 nL to SWNT biosensor devices. The capillary delivers solution to the FET, which is connected to the electronics through the socket. (Right) Schematic of nanotube biosensor.

charges become trapped in the substrate, and so when the voltage is ramped back up, the effective gate voltage that the nanotube sees is more positive than the applied V_{Gate}. For large positive values of V_{Gate}, the opposite effect occurs. A simpler form of data is measures the current as a function of time while applying, removing, and rinsing the drop.

4. SOLVENT CONTROL EXPERIMENTS

Three solvents are known to dissolve T3 well: NaOH in water, NaOH and Ethanol in water, and Dimethyl Formamide (DMF). The first is favorable in terms of T3 solubility, but for biosensing, doping should occur regardless of the choice of solvent. Also, it is important to rule out the possibility of electronic response due to the solvent by itself, or at least to understand the response in order to subtract it from the signal due to the solubilized T3. To that end, several control experiments were conducted.

Drops of solution were applied over the biosensors while the current through the nanotube was recorded. 10 V Gate sweeps were taken before the experiment, while the drop was applied, and after it had evaporated. Typical bias voltages for conductance and gate measurements ranged from about 0.1 to 0.5 V.

Deionized water (ρ~18 MΩ-cm), by itself, increases the current through the nanotube by <.001%, and the effect is reversible. That is, the conductance due to the water is negligible. Up to a molarity of 1 mM, NaOH in water performs like DI water – it does not alter significantly either the conductance of the nanotube or its gate characteristics (See Figure 6). At higher concentrations a change in conductance is observed, but it is predictable and repeatable. Presumably, there is a threshold concentration of NaOH for which the ions begin to contribute to the conductance. NaOH

and Ethanol in water does not perform quite as well; even at .1 mM we observe an electronic response conductance (~30%), but it is predictable as well. Unfortunately, at higher concentrations ≥ 1mM, NaOH/Ethanol irreversibly quenches the gate characteristics for some nanotubes. DMF, a strong organic solvent, performs the worst of the three. In addition to altering the conductance of the nanotube and suppressing its gate characteristics, it also damages our equipment. The apparatus is currently being modified to accommodate DMF. All three solvents are useful to solubilize T3 for biosensing, with NaOH in water performing the best.

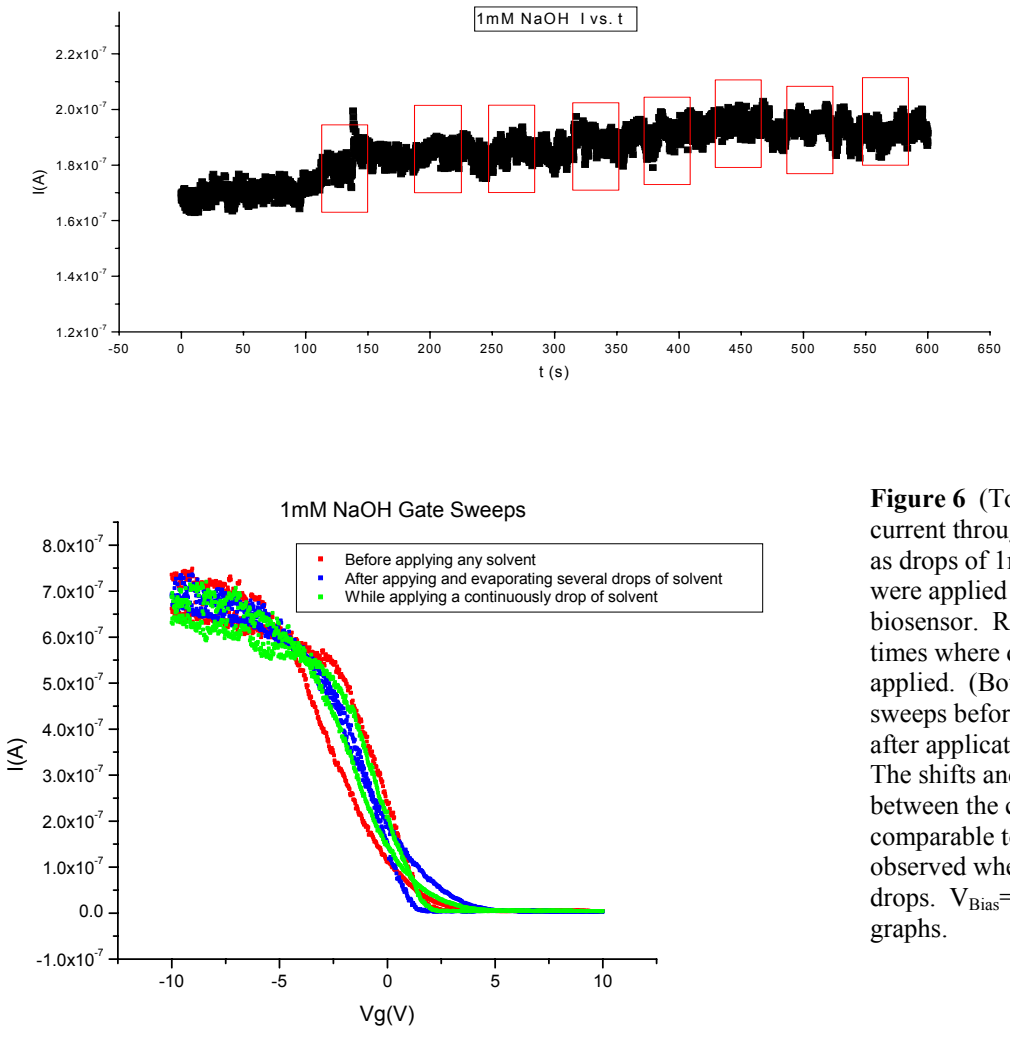


Figure 6 (Top) Plot of the current through the device as drops of 1mM NaOH were applied to the biosensor. Red boxes are times where drops were applied. (Bottom) Gate sweeps before during and after application of drops. The shifts and variations between the curves are comparable to those observed when adding no drops. V_{Bias}=1V for both graphs.

5. RESULTS WITH T3

Our procedure for applying T3 is similar to that for applying the control solution. First, several 2V gate sweeps are taken to characterize the device pre-experiment. Then the current is recorded as a drop of solution is applied to the device. Several more 2V

gate sweeps are taken for comparison to the pre-experiment sweeps. If no gate response is observed, the gate voltage is ramped progressively to higher and higher absolute values, as far as ±30V. Then the current is recorded as the drop is removed. Again the range of V_{Gate} is swept in search of a gate response. If none is found, a rinsing is performed by adding and removing several drops of 1mM NaOH without T3 to try to reverse the effect of T3 by diffusing it off of the nanotube.

For all the experiments using T3 so far, we have used the 1mM NaOH solvent. We can vary the concentration of T3 up to a limit of 1mM for this concentration of solvent. For the lowest concentration of T3, 10 μM, no change in conductance is observed when drops are applied other than that associated with the conductance of the water. Additionally, after the drops are applied the gate hysteresis curves do not change more than what is expected due to charge fluctuation in the substrate. For a 100 μM T3 solution, again there is no significant difference in conductance between times when the drop was on and those when it is off, but the gate curves before, during, and after drop are not as stable as they would be without any solution over the time period of the experiment. This may be because T3 molecule promotes additional charge fluctuation around the substrate. Still at this concentration we see no clear detection of T3 attributable to an electrostatic interaction of T3 with the nanotube.

For a concentration of 1 mM T3, though, when the drop is put on, after ~30 seconds, the current is suppressed to that of the transistor's off-state, where it is stable for as long as the drop is applied (See Figure 7). These measurements were taken for four p-type transistors. For three, subsequent gate sweeps out to –30 V show no recovery of on-state current with the drop still applied and after it evaporates. After rinsing, though, gate response is recovered for two of the samples, but the onset of the current turn-on is shifted to ~ -5V. For the fourth device, no rinsing was needed to find the shifted on-state onset, which occurred at ~ -3 V. This suggests that for the three samples for which the on-state disappeared, the on-state was actually shifted to huge negative values past -30 V and rinsing shifted the onset back toward 0 V. For another ambipolar sample, the drop also suppressed the current *to the lowest value of the gate sweep*, not to on-state for positive voltages. Measurements at gate voltages past 2 V have not yet been taken. This sample contradicts the shift theory and suggests that the T3 actually suppresses the gate response completely and when it is rinsed off, the response reappears.

6. DISCUSSION AND CONCLUSIONS

Further measurements are warranted. Specifically, it is necessary to ramp the gate voltage lower than –30 V after the T3 is applied to search for the onsets to the FET on-states. If the onsets can always be found, it would be interesting to see whether the amount of shift depends on the concentration of T3, or if there is simply a threshold molarity for which the biodetection is activated. Because iodine, atomic number 53, has an open space for an electron in its valence band, we originally predicted a shift in the gate hysteresis toward the positive. That is, when the iodine binds to the nanotube, it

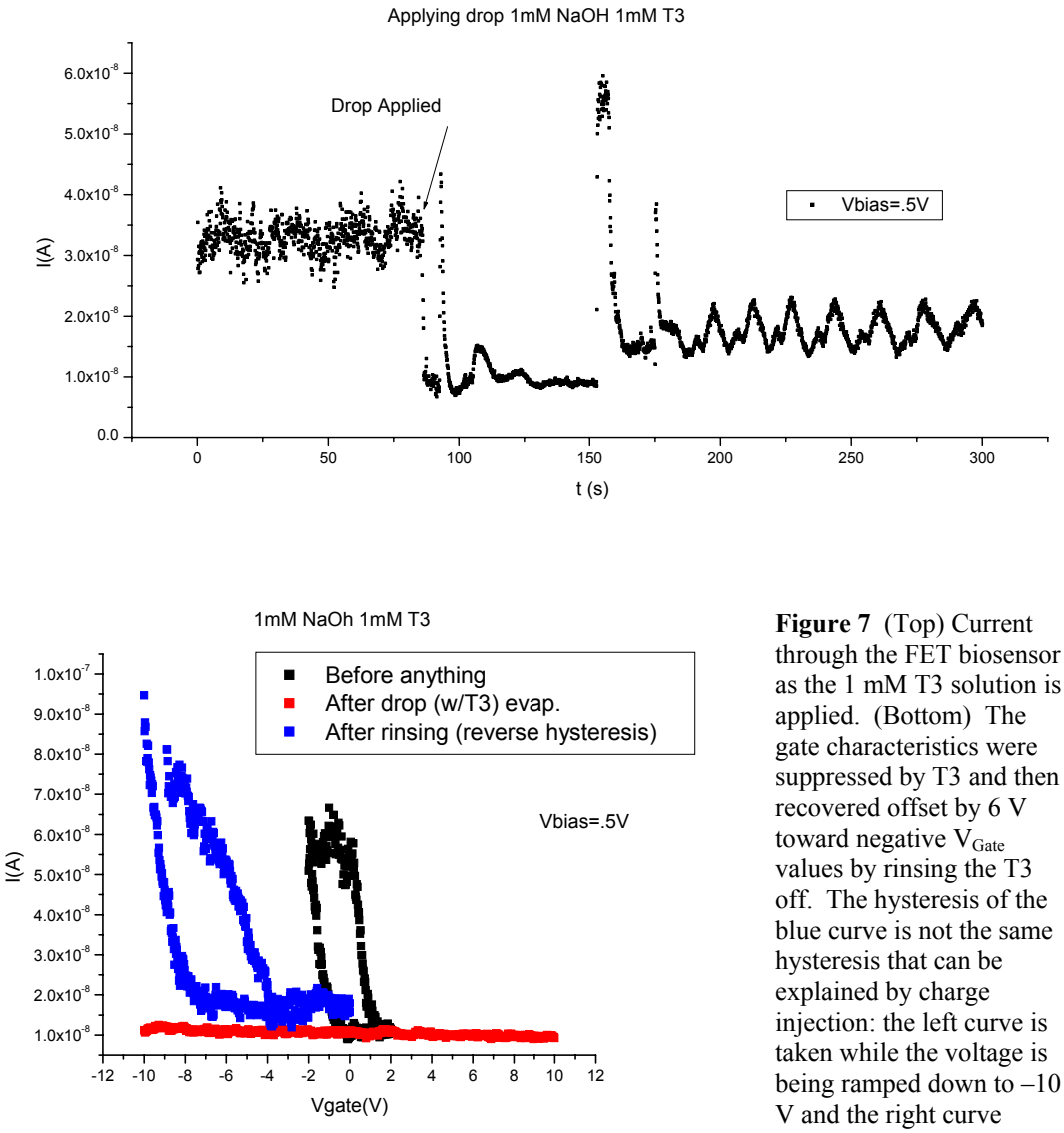


Figure 7 (Top) Current through the FET biosensor as the 1 mM T3 solution is applied. (Bottom) The gate characteristics were suppressed by T3 and then recovered offset by 6 V toward negative V_{Gate} values by rinsing the T3 off. The hysteresis of the blue curve is not the same hysteresis that can be explained by charge injection: the left curve is taken while the voltage is being ramped down to –10 V and the right curve taken on the way back up.

would accept an electron into its valence shell from the nanotube, making it easier to turn the transistor on. After observing a shift in the opposite direction, we have to modify this theory. Since the T3 molecule is electrically neutral, we have no reason to believe *a priori* that electron donation from the nanotube would be the mechanism for doping as opposed to electron donation from the T3, except for the presence of the iodine atom. It is still possible that the iodine atom is essential to the chemistry of the molecule in order for it to donate n-carriers. Detection experiments could be run on a deiodinated species of thyroxine to test this. We could also try to calibrate the shift with respect to T2 and T4 as well as T3 to see how important the iodine atoms are. On the other hand, perhaps the iodine atoms contribute critically to the electric dipole of T3, and binding to the SWNT with the dipole pointing radially outward away from the tube lowers the valence and

conductance band with respect to the Fermi energy of the metal, thereby increasing the height of the tunneling barrier for holes to get on and off the nanotube.

Experiments using the other two solvents will also contribute to the overall picture. Crude detection experiments with T3 on bulk material were done several years ago, and it appeared that the effect of T3 changed given a different solvent.[9] If we can take data that are consistent with this, they would support the charge transfer mechanism for biodetection and would demonstrate the electrical versatility of T3: It can donate holes or electrons depending on its chemical environment.

In conclusion, this experiment shows that it is possible to detect T3 using carbon nanotube sensors. More interestingly, it has been demonstrated that T3 can be electrically active and can interact electrostatically with a 1-dimensional system. This may have physiological implications. Until we can construct a biologically relevant model in which to measure the electrical properties of DNA, carbon nanotubes are the closest we can come.

7. ACKNOWLEDGEMENTS

I would like to thank my co-workers on this project, Mike Stern, Scott Paulson, Mary Dratman, and Charlie Johnson. I would also like to thank Jonas Goldsmith and Cristian Staii for experimental help. Funding for this project was provided by the National Science Foundation and by the Nanotechnology Institute. I would also like to thank Jan Van der Spiegel, Lois Clearfield, and the SUNFEST summer research program.

8. REFERENCES

[1] S. Iijima, Helical microtubules of graphitic carbon, *Nature*, 358 (1991) 56-58.
[2] R. Martel, et al., Single- and multi-wall carbon nanotube field-effect transistors, *Appl. Phys. Lett*., 73 (1998) 2447-2449.
[3] J. Hafner, et al., Direct Growth of Single-Walled Carbon Nanotube Scanning Probe Microscopy Tips, *J. Am. Chem. Soc.*, 121 (1999) 9750-9751.
[4] J. Kong, et al., Functionalized carbon nanotubes for molecular hydrogen sensors, *Adv. Mat.*, 13 (2001) 1384-1386.
[5] A. Star, et al., Electronic Detection of specific protein binding using nanotube FET devices, *Nano Letters*, 3 (2003) 459-463.
[6] R. Saito, et al., Electronic structure of chiral graphene tubules, *Appl. Phys. Lett.*, 60 (1992) 2204-2206.
[7] J. Hafner, et al., High-yield fabrication of individual single-walled nanotube probe tips for atomic force microscopy, *J. Phys. Chem. B*, 105 (2001) 743-746.
[8] M. Radosavljevic, et al., Nonvolatile Molecular Memory Elements Based on Ambipolar Nanotube Field Effect Transistors, *Nano Lett.*, 2 (2002) 761-764.
[9] M. Radosavljevic, Improving carbon nanotube nanodevices: ambipolar field effect transistors and high current interconnects, Doctoral Thesis, University of Pennsylvania 2001.