A PERFUSION SYSTEM FOR MAINTAINING COCHLEAR TISSUE SLICES DURING MEASUREMENT OF POTENTIAL DISTRIBUTIONS IN RESPONSE TO ELECTRICAL STIMULATION ACROSS A BROAD RANGE OF FREQUENCIES

Christopher A. Dillon

A thesis submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Master of Science in the Department of Biomedical Engineering, School of Medicine.

Chapel Hill

2006

Approved by

Advisor: Professor Charles Finley

Reader: Professor Henry Hsiao

Reader: Professor Allison Fragale

ABSTRACT

CHRISTOPHER A DILLON: A Perfusion System for Maintaining Cochlear Tissue Slices during Measurement of Potential Distributions in Response to Electrical Stimulation across a Broad Range of Frequencies

(Under the direction of Dr. Charles Finley)

Cochlear implants are able to provide functional hearing to many deaf individuals through electrical stimulation of surviving nerves in the cochlea. While the general success of cochlear implants is well documented, there still remain many basic questions concerning an implant's interaction with the cochlear tissue. This research aims to explore the question of how electrical energy delivered by the implant as current pulses distributes through the tissue of the cochlea to generate extracellular potentials in the vicinity of surviving neurons. In particular, this work seeks to gain insight into how capacitive tissue properties effect the distribution of energy delivered by short-duration current pulses. This thesis describes a custom-build tissue chamber designed to maintain live cochlear tissue slices, as well as the hardware and software for implementing data collection and analysis protocols for characterizing potential distributions in the tissue slice during stimulation. Electrical stimulation is delivered via spatially-fixed ball electrodes, while a moveable metal microelectrode recorded data over a wide frequency range (30 Hz. to 100 kHz.) at 100 separate points evenly distributed across and within the tissue slice. To date the system has been tested with tissue substitutes such as bologna and potato slices to verify the function of the system. Continued work with cochlear slices will one day provide results that may improve the effectiveness of cochlear implants.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Charles Finley, for all his time, resources and insights without which completion of this project would not have been possible. It has been a rewarding and amazing experience to work with someone as smart and dedicated to his field as Dr. Finley. I would like to thank Dr. Henry Hsiao and Dr. Allison Fragale for serving on my committee and their valuable insights. I would additionally like to thank Dr. Robert Dennis for the use of his lab and his insight in the construction of the tissue chamber. I am deeply grateful for the support of my girlfriend and parents, without whom I would undoubtedly never have finished.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Chapter 1 INTRODUCTION

Hearing loss has become a common problem in today's society due not only to unchecked noise levels but also aging, heredity and disease. The National Institute of Deafness and Other Communication Disorders estimates that in the United States alone more than 28 million individuals are hearing impaired. Furthermore, in 2002, there were 59,000 individuals with cochlear implants in the world. In the United States at this time 13,000 adults in addition to 10,000 children had been implanted with cochlear implants (NIDCD 2002).

The cochlear stimulation technology has been evolving for the past 50 years and will continue to do so into the future. The first auditory nerve stimulation with an electrode was recorded in 1957, in this case the deaf patient was able to hear background noise. In 1966 the first patients were implanted with an array of electrodes placed within the cochlea. The goal of multi-electrode arrays was to stimulate distinct regions of the surviving nerve population within the cochlea corresponding to the regions that would be stimulated by particular sounds in normal hearing individuals, namely to drive the auditory system in a tonotopic manner. To this end much research has been conducted to determine the most advantageous method to target distinct nerve regions. Some of the methods that have provided significant gains towards this goal will be described in the background section of this thesis. However, there is one area for potential gain that has not been fully illuminated, that is the creation of a detailed map of the characteristics of the tissues within the cochlea.

The cochlear implants of today provide sound to profoundly deaf individuals with the use of multi-channel electrode arrays, which are wrapped around the cochlea within the scala tympani. To date, few papers have endeavored to uncover, in vitro, to what extent the tissues within the cochlea present resistive or capacitive characteristics when exposed to the electrical signals provided by these electrode arrays. The design of cochlear implants has allowed research to be conducted in which the electrodes within the electrode array are used as recording electrodes to get at the question of the tissues resistive characteristics. However, that research is not necessarily a direct representation of the signals that ultimately trigger a nerve action potential.

Further research has been conducted to understand the exact mechanisms that allow electrical stimulation within the cochlea to cause hearing within a profoundly deaf individual, the question of how the cochlear tissue transmits these electrical impulses has yet to be fully understood. The research, conducted in conjunction with this thesis project was done so to begin to illuminate these areas.

The remaining chapters of this thesis have the following organization:

Chapter 2. *Background*. A brief description is presented of the auditory system and hearing loss. It describes in greater detail the pertinent functionality of cochlear implants that will aid in the understanding of the remainder of this thesis.

Chapter 3. *Project overview and goals.* In this section, further development of the overriding hypothesis driving this project, and the planned approach that will be followed to verify the validity of the hypothesis is presented. The primary objective of this thesis is to establish and verify the basic measurement tools that will be employed in future studies. In particular, the principle aim is to develop a robust and repeatable method of recording the

distribution of electrical potentials from within 300µm-thick cochlear slices to determine the net effect of capacitive tissue characteristics on electrical current distribution within the cochlea.

Chapter 4. *Methods and Materials* This chapter includes individual descriptions of the equipment and processes that were developed in the achievement of the project's goals. The first section will describe the custom built chamber and the design considerations that dictated its final form. This will be followed by a discussion of the perfusion system and how it was designed to maintain the integrity of the living tissue. Aside from the hardware necessary for maintaining the tissue, the hardware and software needed to provide accurate electrode placement on a micron scale will be described in full. In the subsequent section the technique for optically imaging the tissue sample and specifying recording positions will be detailed. Additionally, the stimulating and recording electrode and amplification system design necessary for returning data that can justify the hypothesis will be outlined. This chapter also outlines the baseline recording tests that were conducted to demonstrate the accuracy of all the hardware and software systems described.

Chapter 5. *Results*. All data relevant to describing the characteristics of test tissues and tissue chamber will be presented, along with such data that were vital to the progression of the project.

Chapter 6. *Discussion*. This chapter will elaborate on the results from Chapter 5 – Results. The first section details the significance of the results. In the second section, limitations and problems that were overcome will be presented. Finally, the last section describes how these tools can be applied to further work examining the original hypothesis regarding electrical stimulation within the cochlea.

Chapter 7. *Conclusions*. The achievements of this work and the potential impact the results may have for patients in the future are presented. This section is followed by appendices containing source listings of the software code developed during this project.

Chapter 2 BACKGROUND

2.1 Auditory System and Hearing

In order to understand the driving forces behind the formulation of the hypothesis of this project, it is essential that one understand how a cochlear implant system functions to restore hearing based on the normal mechanisms of the auditory system. The following section will describe the functionality of the auditory system as well as the causes of the loss of this functionality.

2.1.1 How Hearing Works

Below is shown a diagram of the human ear, which is divided into the outer ear, middle ear and inner ear.

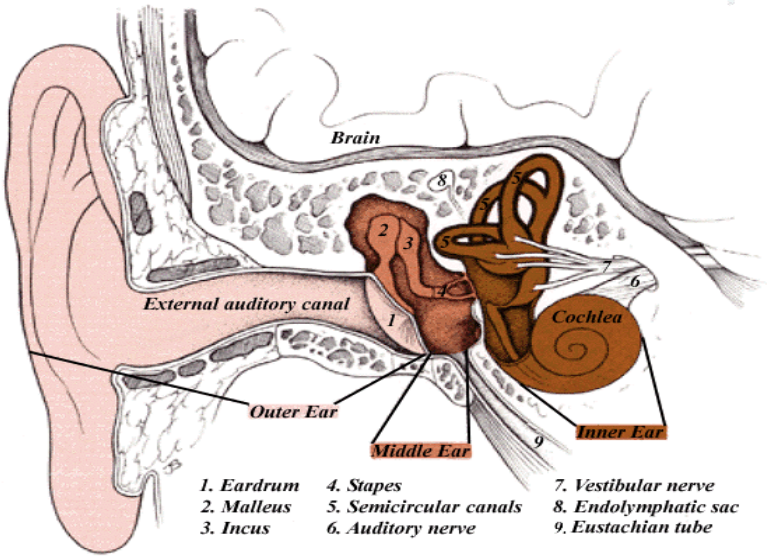


Figure 2-1 Diagram of the structure of the human ear
(obtained from The Ohio Rehabilitation Service Commission website – http://www.rsc.ohio.gov/images/Fun4kids/learn/ear.gif)

The outer ear, commonly referred by most people as the “ear”, is only the tip of the iceberg as far as the hearing process is concerned. The outer ear has multiple functions, which include sound localization, amplification and protection for the middle and inner ear. The middle ear is separated from the outer ear by the tympanic membrane (or ear drum). Connected to the tympanic membrane and bridging the gap between the air-filled outer ear and fluid-filled inner ear are three tiny bones known as the ossicles. These bones are the malleus, incus and stapes and are suspended loosely within the middle ear. Their function is to transform the sound energy incident on the tympanic membrane to mechanical energy incident on the oval window of the inner ear. Of all the parts of the auditory system, the inner ear is the most complex. It is composed of two parts, the vestibule and the cochlea. The vestibule assists in maintaining balance and posture and will not be not be dealt with further in this paper. This discussion will focus on the cochlea which further mediates the hearing function of the ear system. Once a sound wave has been converted in the middle ear, the inner ear completes the conversion process and outputs neural stimulation that is sent to the brain. This neural stimulation is conveyed to the brain via the auditory nerve where it is further processed to be perceived as sound.

2.1.2 Structure of the Cochlea

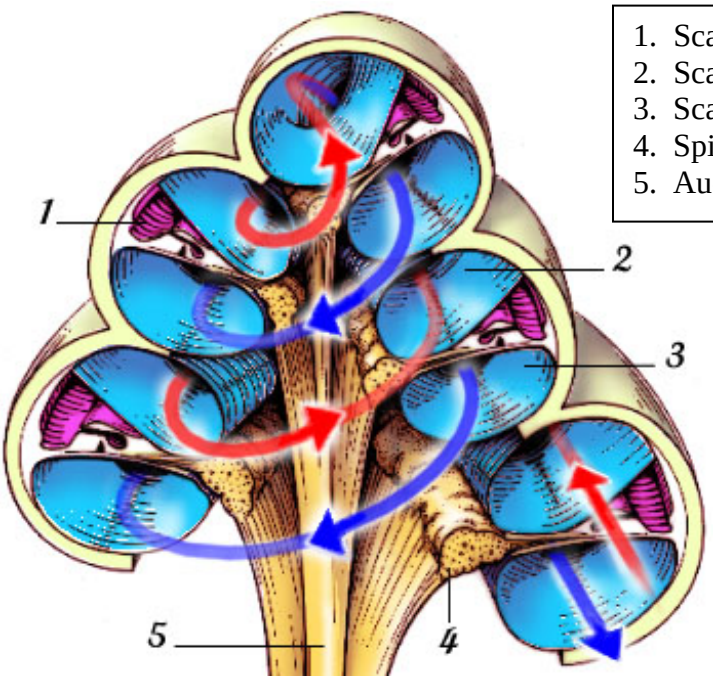


Figure 2-2 Mid modiolar section of a cochlea
(obtained from Promenade Around the Cochlea website – www.iurc.montp.inserm.fr/cric/audition/english/cochlea/cochlea.htm)

As can be seen in Figure 2.2 the cochlea is a spiraled structure with three fluid-filled channels: the scala vestibuli, scala media and scala tympani. The scala vestibuli and scala tympani are in fact one single canal that meets near the apex of the cochlea. A simpler way to envision the geometry of the cochlea is depicted in Figure 2.3 in which the cochlea has been uncoiled.

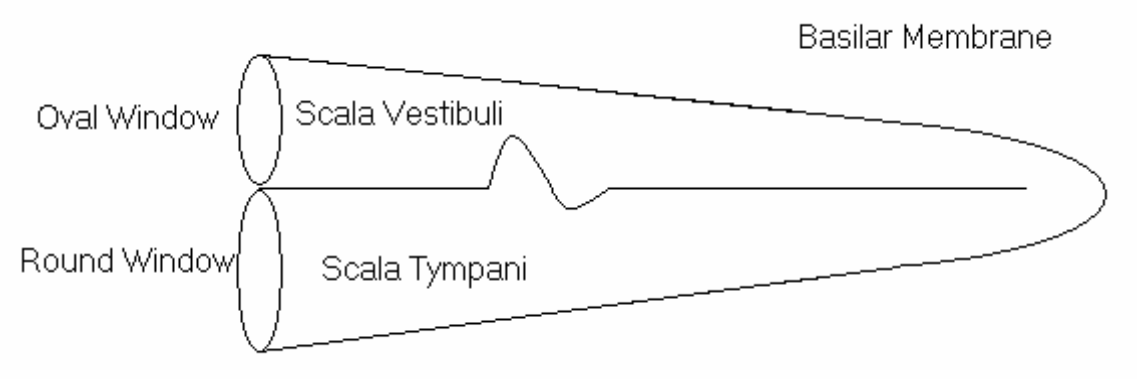


Figure 2-3 A cross sectional view of the cochlea

When the stapes, whose base is connected to the oval window, moves in response to a sound wave incident on the ear drum, a pressure wave is produced within the cochlear fluids. This wave produces a wave-like pattern on the basilar membrane, which is the membrane separating the scala media from the scala tympani. Atop the basilar membrane is the organ of Corti, which is the site of the mechanical transduction to nerve activation, which subsequently results in hearing percepts after further processing by the central nervous system.

The organ of Corti rests on top of the basilar membrane and is the home of several thousand sensory nerve cells called “hair cells”. Each hair cell has several cilia protruding from the top of the cell. The organ of Corti contains four rows of hair cells. The inner most row of cells are referred to as “inner hair cells” and are responsible for 90-95% of the information carried by the auditory nerve to the brain (Bess 1995). The cilia of the inner hair cells extend from the top of the hair cells and form loose attachments to the overlying tectoral membrane, while the base of the hair cell is connected to nerve fibers of the auditory nerve via synaptic connections. The remaining three rows of hair cells are referred to as the “outer hair cells”. The cilia of the outer hair cells are embedded into the tectorial membrane. Their connection to the tectorial membrane is used to mechanically adjust and tune the basilar membrane to enhance its response to auditory stimulation. In combination with systematic changes in the width and flexibility of the basilar membrane along its length, the tuning of the basilar membrane causes the cochlea to be tonotopically responsive to the vibration frequencies present in incoming sound. This means the sounds waves of different frequencies cause wave spikes at different locations within the cochlea. The tonotopic organization of the cochlea results in neurons originating near the base of the cochlea being

maximally responsive to high frequencies, where as neurons originating near apex or far end of the cochlea being maximally responsive to low frequencies. For example, low frequency sound would cause activation of hair cells on the right hand side of Figure 2.3 and high frequency sounds wound activate hair cells and neurons on the left end near the oval window. Between these extremes there is a monotonic progression of frequency sensitivity, hence the idea that the cochlea is 'tonotopically" organized and provides the first, and very important, level of processing in the auditory system. In a normally fuctioning system this tonotopic organization is retained at all levels of auditory processing in the brain all the way to the cortex.

2.1.3 Causes of hearing loss

The properly performing auditory system functions as a transducer converting acoustic energy to neural impulses, but when one part of the system fails that transformation is not completed. The mechanism of hearing loss that is of relevance to this research is the result of the loss of hair cells within the cochlea, referred to as "sensorineural deafness". It has been well documented that the degeneration of hair cells and partial loss of auditory nerves are at the center of sensorineural deafness. This will become an important fact in the following section when discussing cochlear implants. The loss of hair cells can be the result of diseases (e.g., meningitis, Meniere's disease), congenital disorders, brief exposure to transient extremely loud sounds, sustained exposure to moderately loud sounds (e.g. workplace noise, concerts, lawn mowers, etc), the normal process of aging, and/or certain drug treatments (Loizou, 1998). When hair cells are damaged or lost, the auditory neurons to which they connect may also atrophy and die, but to a lesser extent. Hearing loss occurs once a substantial number of auditory neurons and hair cells of the cochlea are damaged or

lost in a particular region of the cochlea. As the cochlea is a tonotopic structure, hearing loss can affect selected frequencies (e.g. high frequencies as in the case of age related hearing loss) and can often be treated effectively by application of hearing aids if sufficient numbers of hair cells remain. However, in the case of severely hearing impaired and/or profoundly deaf individuals very few hair cells remain and virtual no acoustic energy is transformed and passed to the brain, even when substantial amplification of the sound is provided by a hearing aid. Cochlear implants were developed to assist these populations of patients, who would otherwise have no alternative therapies for their deafness.

2.2 Cochlear Implants

Cochlear implants are neuroprosthetic devices designed to restore hearing in patients with severe to profound sensorineural hearing loss. As pointed out in the previous section the auditory neurons in patients with sensorineural deafness do survive to varying extents and are still functionally able to conduct action potentials to the central nervous system. It is the hair cells that have been predominately lost or damaged. The key design principle of the cochlear implant utilizes this fact and is focused on directly stimulating the surviving auditory neurons, thus bypassing the functional need for the damaged or lost hair cells.

2.2.1 Internal and External Components

Cochlear implants can be functionally divided into two parts: the surgically implanted internal component and the wearable external component.

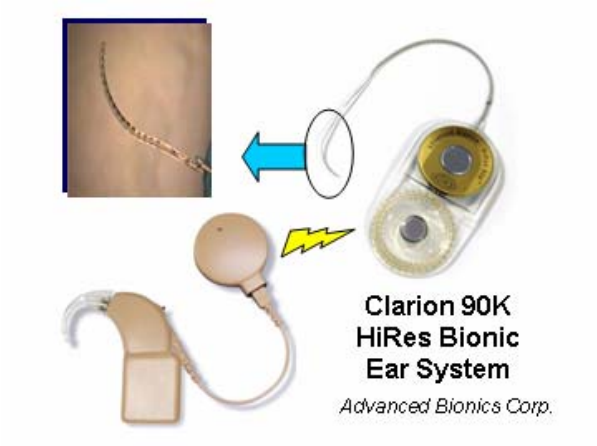


Figure 2-4 Present day cochlear implant system in clinical use

The external component has two distinct parts. First, the head piece provides the connection between the internal and external components. The head piece locates itself on the skin directly above the case of the internal component magnetically. The magnetic alignment of the head piece and implanted case provides the alignment for telemetric coils, which pass power as well as the stimulating signal to the implanted components. The second external part is the speech processor. Sound that is picked up by a microphone is passed to the signal processor where the sound information is encoded into strings of biphasic pulses. This process will be described further in section 2.2.2. The speech process also houses the rechargeable battery which powers the entire system. The internal component of a cochlear implant consists of an implanted stimulator sealed within a case and an electrode array. The stimulator receives the signal encoded by the speech processor and outputs the required signal to the proper electrodes within the electrode array. The stimulator is surgically placed just beneath the skin behind the ear and is secured to the skull. Depending on the particular device selected, the electrode array may contain up to 22 separate electrodes spaced evenly along the length of the array. The electrode array is carefully inserted into the scala tympani

under surgical conditions. Typically the electrode array coils approximately one and a half turns into the cochlea. With this position of the array in the cochlea, stimulation of individual electrode contacts causes activation of selective regions of surviving auditory neurons, thus providing electrical access to the tonotopic structure of the cochlea and the remaining auditory system. Together the internal and external components of the cochlear implant are able to transduce acoustic energy into neural signals.

2.2.2 Bipolar vs. Monopolar Stimulation

The two most commonly used methods of stimulation utilized by cochlear implants are monopolar and bipolar. Both techniques employ the same principle of electrical currents. When a current is generated by a source electrode it must have a ground point with which to complete its current path. In monopolar stimulation the ground point is located on the stimulator case or a separate electrode spaced away from the electrode array. In bipolar stimulation, two electrodes in the electrode array are used to create the electrical field, with one electrode in the array acting as the source and the other acting as the ground.

Both monopolar and bipolar stimulation have their own distinct advantages and disadvantages in cochlear implant operation. The advantage of monopolar is that the electrical field that is produced is much larger thus making stimulation of the auditory neurons possible at lower current levels. The disadvantage is that since the field is so large it is virtually impossible to limit the number of neurons that are activated. Since all neurons across the cochlea are activated there is diminished ability to discrimination between high- and low-frequency sounds. By using bipolar stimulation the electrical fields are better contained and can be localized to neurons responsive to particular frequencies. However, by using a smaller field, a larger current must be passed between the source and ground

electrode to ensure the field spreads far enough to reach the neurons. Higher current results in electrolysis of the cochlear fluid and dissolution of the electrode contacts, which can in turn damage the surviving cochlear tissue. Additionally bipolar stimulation, with its smaller field, is still not able to create the fine frequency structure of a healthy cochlea. The results from this research project will provide further information what will help characterized the electrical distribution of these stimulating strategies within the cochlea.

2.2.3 Compressed Analog vs. Continuous Interleaved Sampling

Not only is it important to understand output from the electrodes, but also to understand the signals that are sent to the electrodes. This will provide greater insight to understand the importance of this project. Compressed Analog (CA) was the initial encoding strategy for multi-electrode implants. Under the CA stimulating strategy sound taken in by a microphone is compressed and broken into evenly spaced groups of frequency with band pass filters. Each group is then sent to its respective electrode within the electrode array. The stimulation level output of each electrode would be equivalent to the amount of acoustic energy that was parceled into its select frequency grouping. The problem with this strategy was that each electrode would be outputting its signal at the same time as every other electrode. When electrical current comes in contact with each other they interact either by summing or subtracting from each other. This “channel interaction” results in the loss of frequency specific signals and uncontrollable spread of stimulation.

In order to correct for the channel interaction that was encountered in CA, researchers developed continuous interleaved sampling (CIS), which continues to be the strategy of choice today. In the CIS strategy, the first step is to divide the raw sound signal into frequency groups. The energy level of the frequency groupings is estimated with a full-wave

rectification and low-pass filter setup. A non-linear mapping component can then bring the energy levels to a desired level specific to each individual. Finally, and most importantly, a post processor sequencer is used to sample the energy level of each grouping separately and apply stimulation to only one electrode at a time, thus reducing the effect of channel interactions, at least interactions due to the simultaneous summation of two or more electrical fields. Because the bulk tissue and neural membranes have capacitive and refractive characteristics, respectively, there is another form of channel interaction referred to as "temporal" interactions. These interactions arise from the lingering effects of charge and refraction from stimulation on one electrode overlapping with subsequent stimulation on another electrode. By lengthening or shortening the rate of pulsatile stimulation one can control the amount of temporal channel interaction between electrodes, which can be directly related to a patient's ability to perform. Figure 2.4 shows a block diagram of a CIS system for a 6 channel processor.

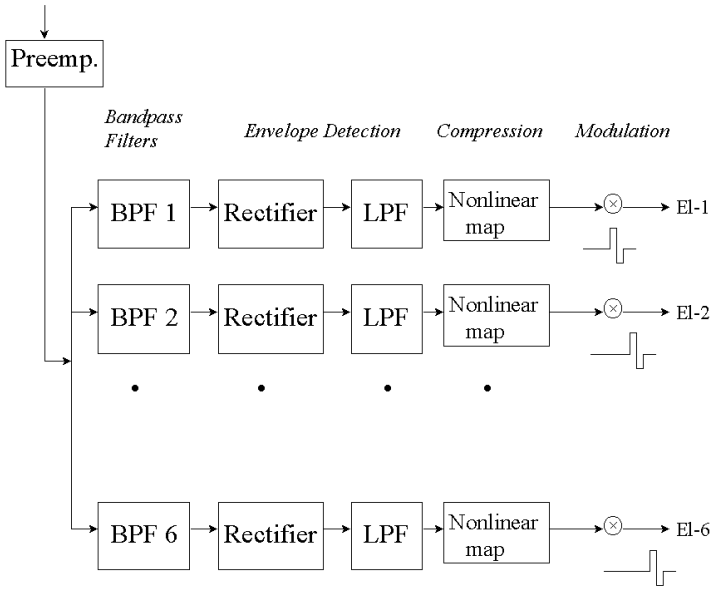


Figure 2-5 Block diagram of CIS stimulation strategy (Loizou 1998)

Chapter 3 PROJECT OVERVIEW AND GOAL

The primary objective of this work is to develop a procedure capable of measuring the effect of capacitive tissue characteristics on the potential distributions generated in the cochlea due to brief electrical stimulation with biphasic current pulses. This project pursues this goal by making potential distribution measures in living cochlear slices in vitro using a custom-designed tissue bath and facilities for automated collection of potential data during active electrical stimulation over a broad range of frequencies. In furthering the knowledge of how the capacitive characteristics of the cochlea tissue influence stimulation, it is hoped that new insights into stimulation mechanisms may be gained and cochlear implant designs can be improved.

Research to date provides evidence in support of hypothesizing that cochlear tissue is of a resistive nature. In a recent report by Spelmen et al, guinea pigs and monkeys were implanted with electrode arrays and impedances were measured with in the scala tympani and between the scala tympani and the internal auditory meatus of the modiolus. The measurements revealed that the impedances inside and outside the scala tympani are resistive for frequencies between 8 Hz and 12.5 Hz. This data leads to the conclusion that the tissue in the cochlear will behave in a resistive manner, showing a linear relationship to varied frequencies of stimulation. However, the structure of the cochlea is filled with nerve cells and nerve bundles, which extend outward from all parts of the scala tympani, unite and proceed to the auditory nerve. It is known that nerve cells with their extensive myelination

display capacitive characteristics. Thus, it is proposed that the tissue of the cochlear due to its large makeup of nerve cells may be capacitive. This project will utilize cochlear slices that are 200 to 300 microns thick from which impedance measurements will be made throughout the sample.

Spelmen's research finding were presented in 1982, since that time many advances have increased the speed at which cochlear implants operate. At the time of Spelmen's research cochlear implants were outputting biphasic pulses with duration between 200 to 500 micro seconds. Today's implants stimulate with biphasic pulses of 10 micro second durations. Figure 3-2 illustrates the power density of today's implants and depicts the frequency at which the Spelmen study was stopped. The manner in which the Spelmen study was conducted also neglected the presence of the various tissue types in the cochlea.

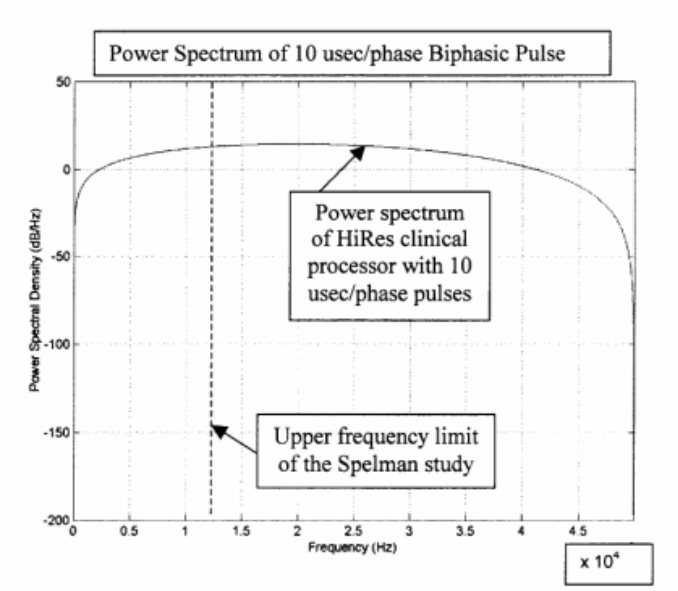


Figure 3-1 Power spectrum of 10 micro second biphasic pulse

Two important requirements drove the progression of this project towards its final goal. First, the data that described the characteristics of the cochlear slices would be extracted from a potential field distribution constructed at every desired frequency. Second

the tissue had to be maintained in a living condition in order to obtain useful information. Consequently, the tissue was maintained in a profusion chamber which would have its own characteristics that would be present in the potential field distributions. Due to this fact, the final potential distributions of the tissue needed to eliminate the baseline characteristics of the tissue chamber. With this knowledge, much of the work leading up to the final data collection was conducted in order to determine the baseline characteristics of the tissue chamber and to determine a repeatable method for doing so.

The diagram below (Figure 3.1) illustrates the setup for the experiment. All components of the diagram will be described in detail in the materials and methods section of this paper.

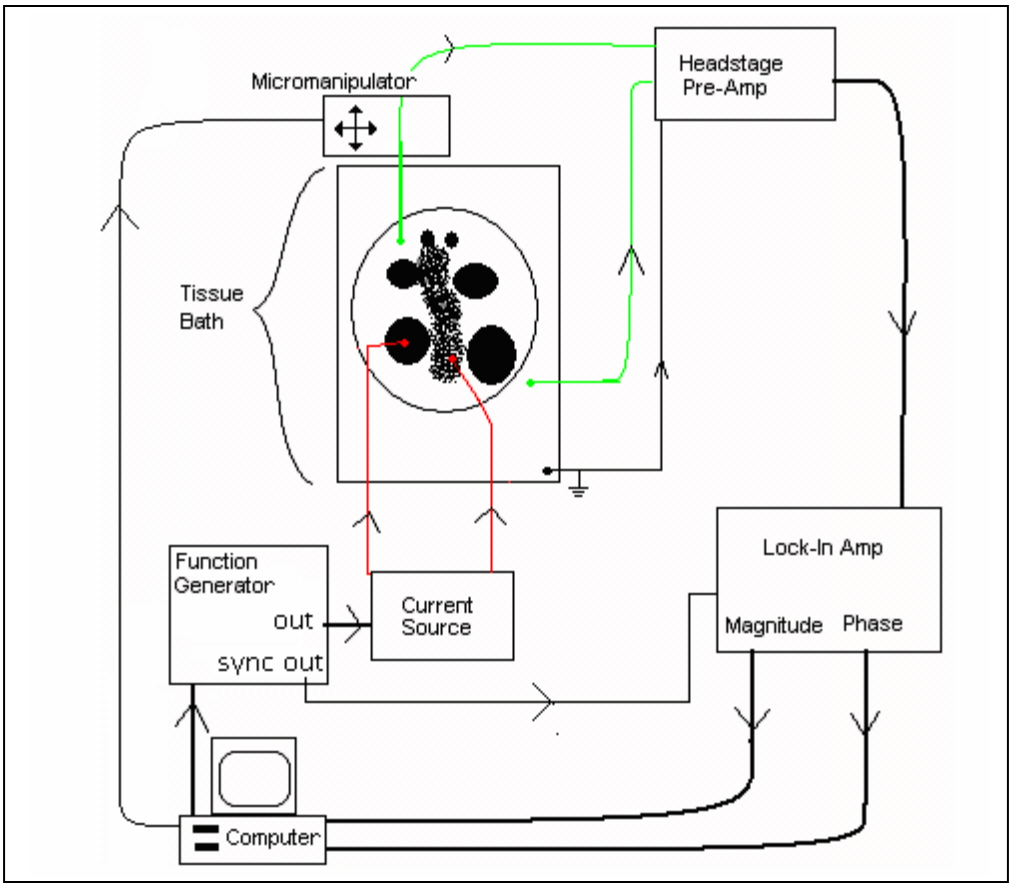


Figure 3-2 Overview of experimental setup

Prior to testing in the tissue bath, a much simpler setup was constructed and tested. The simpler setup allowed for a quick and easy way to test the hardware and parts of the software, while also illuminating some of the problems that would be encountered with the more complex setup. The simpler setup was tested with a variety of test materials which would also be tested in the tissue chamber and allow for comparison of the two setups.

The process of computing potential field distributions is a useful concept that was tested prior to being used with this experiment. In fact, the software and hardware were used to create the potential distribution field plots produced by an actual cochlear implant prior to this project. The creation of these field plots for cochlear implants not only demonstrated the potential to create such plots but also the nature of the results that can be inferred by such plots. The creation of potential field plots made it possible to determine the response of all parts of the tissue slice to a wide range of frequency stimulation. Since the cochlea is composed of many different types of tissues (ex. bone, nerve, and hollow space) it will be possible to determine how each effects the propagation of stimulation. By comparing the potential field plots at various frequencies, the capacitive nature of each tissue type will be illuminated. The knowledge extracted from the plots will be used to fulfill the ultimate goal of this project; that of characterizing the capacitive characteristics of cochlear tissue slices. This in turn will potentially one day improve the effectiveness of cochlear implants.

Chapter 4 METHODS AND MATERIALS

The following section describes the materials and methods used to develop the tissue recording chamber and the stimulation and recording systems.

4.1 Bulk Tissue Measurements

4.1.1 Simple Syringe Setup

Prior to final construction of the tissue chamber described above a simpler setup was created which allowed for preliminary tests to be conducted. A metal box was filled with two holding stands that would fix the syringe in place and prevent electrical conduction to the box. The box provided highly consistent results as well as an easy way to change the test material. The test materials were contained within a modified 8cc syringe. Round disk electrodes (AgCl) (A-M Systems 550025) were fixed to the ends of syringe plungers and electrically connected to BNC connects on the front of the box. The electrodes presented the stimulation created by the current source, while the lock-in amplifier made recordings from the BNC connections. Samples were loaded into the syringe from either end, and holes were created in the side of the syringe to allow excess air to escape until the electrodes were in contact with the sample material.

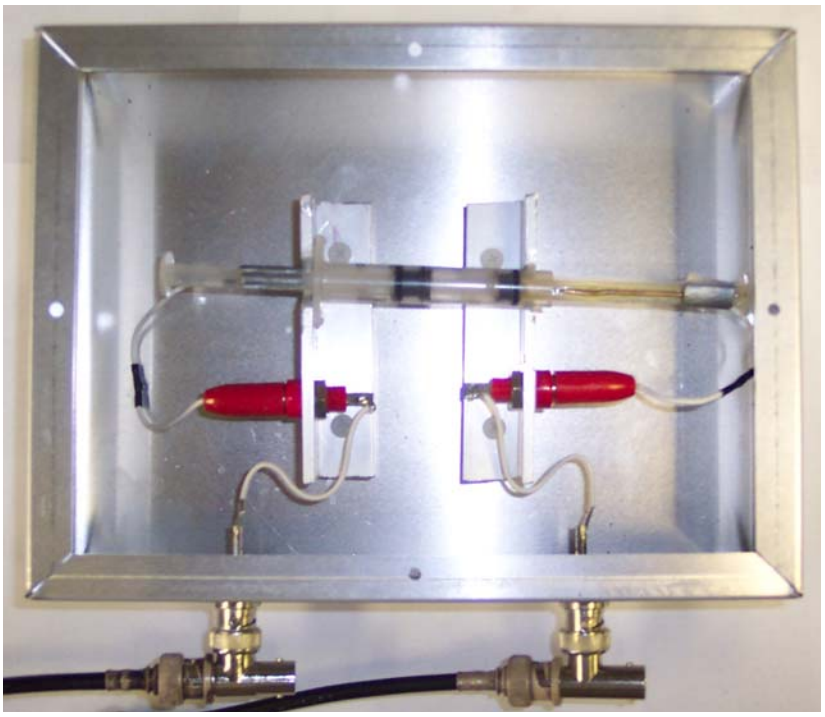


Figure 4-1 Picture of simple syringe setup for bulk tissue impedance measures

4.1.2 Simple Syringe Recording Protocol

The experimental protocol used to obtain these data is nearly identical to that described in the Chapter 3. Custom Visual Basic software code controlled the frequency of the function generator. The function generator drove the current source at the desired frequency and stimulation amplitude. The current source connected directly to the two way BNC connectors on the front of the metal box seen in Figure 4.5. The second side of the BNC connectors connected to either the A or B input of the lock-in amplifier. In this manner current was simultaneously passed through the syringe and its contents while the voltage across the syringe was being measured by the lock-in amplifier. The lock-in amplifier was operated in its differential setting, A-B, thus the lock-in was measuring the drop of voltage across the syringe's contents. The results that were obtained from this experiment describe the voltage drop across the whole system including any electrode and test material interface

characteristics. Separation of these factors was not attempted as the ultimate data set would be a relative reading, thus any constant characteristics such as electrode/sample interface, would be ignored. The lock-in amplifier was set to output the magnitude and phase which was read back into the custom Visual Basic software code though the DAQ card described in section 4.3.2.

4.2 Tissue Slice Measurements

In order to make the desired recording from a cochlear tissue slice a large number of custom designed systems had to be employed. These systems included a custom designed tissue bath for maintaining the tissue's viability during the recording period, a custom build micromanipulation stage as well as individually made Platinum-Iridium (PT-IR) ball electrodes. Due to the fact that this project was the first of its nature to be conducted in Dr. Finley's lab, many additional parameters were resolved that will not be described in detail in this paper.

4.2.1 The Tissue Bath

The final design for the tissue bath that used in this project was initially based on the design provided by Dr. Kerry Zbicz in his paper "Transient Voltage and Calcium-Dependent Outward Currents in Hippocampal CA3 Pyramidal Neurons." The chamber was designed around the ability to support the tissue slice on a piece of nylon mesh, which would allow perfusion above and below the sample. A system of o-rings was decided on for two reasons. First, the o-rings could stretch and hold the mesh in place. Secondly, the o-rings would prevent water from escaping from the chamber. The goal of the chamber was to provide continuous perfusion to the tissue while being open enough to collect from all parts of the sample. Figure 4.1 illustrates the chamber in its complete assembly.

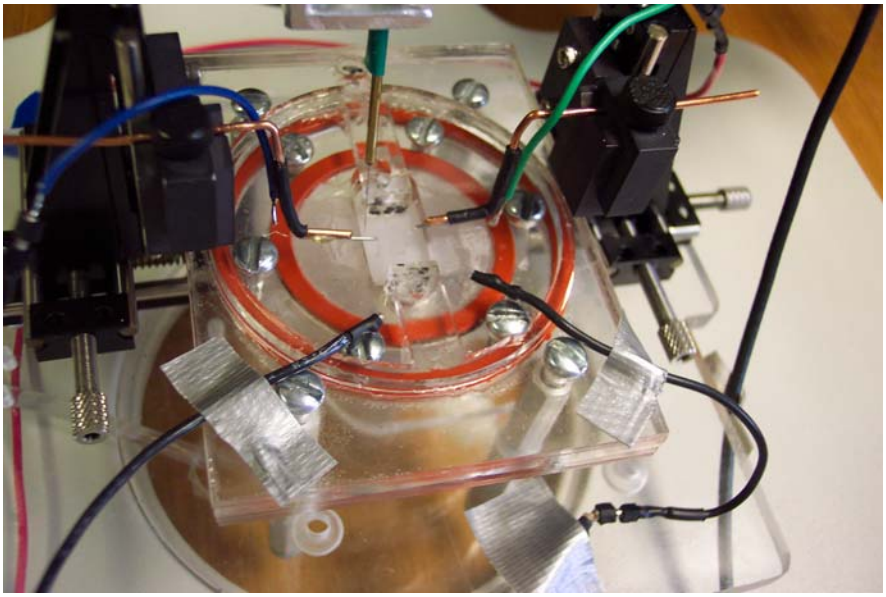


Figure 4-2 Fully assembled tissue bath

The chamber is composed of two separate pieces, a top and a bottom, which were held together with screws. The final chamber design was the seventh revolution from the initial design. All the parts and completed chambers were manufactured in Dr. Robert Dennis' laboratory. The first revolution was designed within SolidWorks (SolidWorks 6.1, SolidWorks Corporation, Concord, MA, USA) and fabricated with the FDM Titan (made by Stratasys Inc., Eden Prairie, Minnesota, USA) a fused deposition modeling rapid prototyping system. All subsequent chambers were designed in Corel Draw and fabricated by cutting parts out of sheets of acrylic with the X-660 laser cutter (made by Universal Laser Systems Inc., Scottsdale, Arizona, USA). The parts were assembled layer by layer and bonded together using methylene chloride, an instantly bonding solvent.

The bottom piece was composed of three layers of acrylic parts. As can be seen in Figure 4.2 the bottom houses two o-rings and the bottom half of the tissue bathing area. The inner o-ring serves two purposes. First it stretches and secures the nylon mesh upon

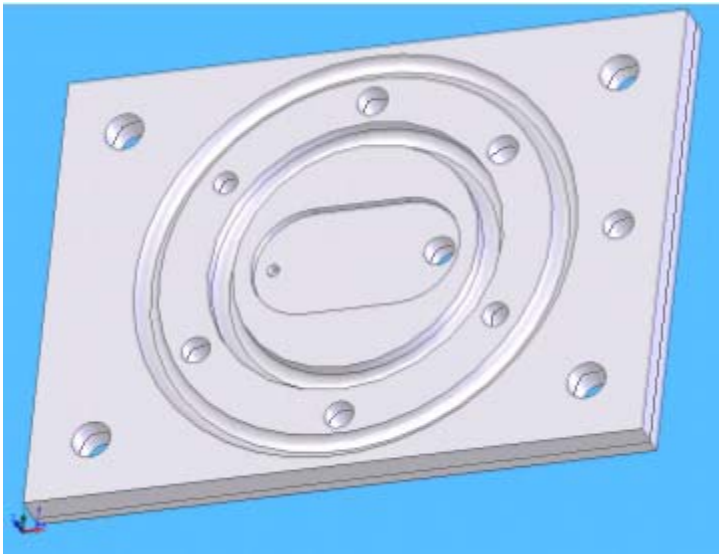


Figure 4-3 Diagram of chamber bottom

which the tissue rests. Secondly it provides a liquid tight seal preventing the profusing media from filling the area between the top and bottom parts of the chamber. Since the nylon mesh tended to wick fluid, the second o-ring was added to prevent any fluid loss from the entire chamber. The nylon mesh that was used in this project was made with 120 µm diameter nylon string with 185 µm open area between strings. Additionally a small strip of Vaseline was set around the bathing area to deter any fluid from leaking between the top and bottom pieces. The bottom piece additionally provided the inlet and outlet ports for the profusion media.

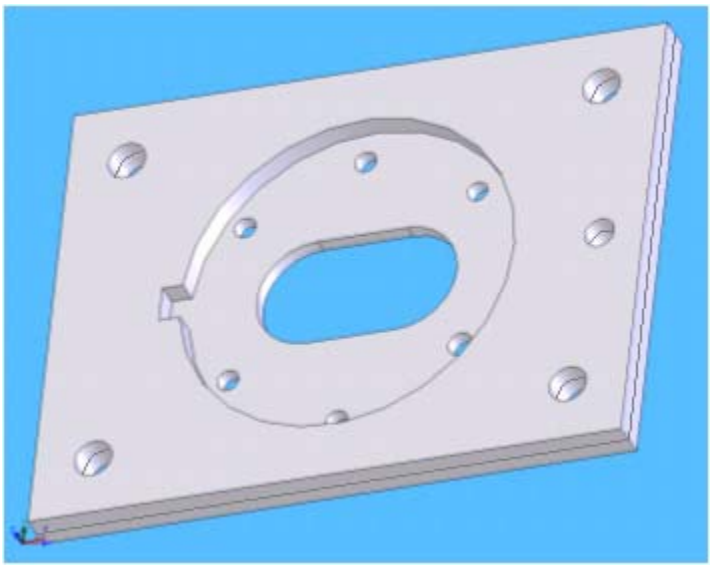


Figure 4-4 Diagram of chamber top

The top piece of the tissue chamber was much less complex then the bottom piece. It was composed of two layers of acrylic as seen in Figure 4.3. The bottom layer creates the top half of the tissue bathing area. The top layer acts as a guide to hold the harp in proper alignment. The harp, as depicted in Figure 4.4, was a circular piece of acrylic which was custom designed to prevent the tissue from moving within the bathing area.

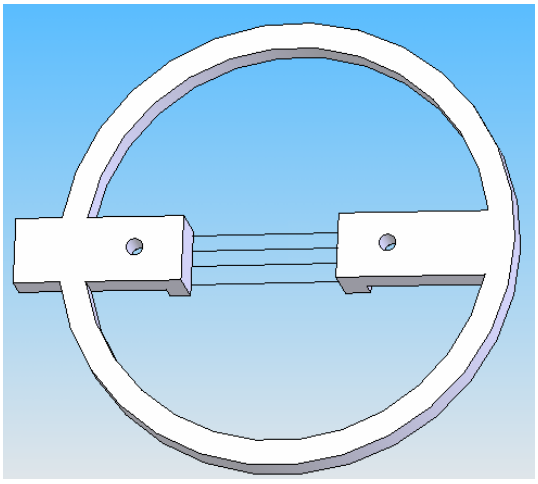


Figure 4-5 A diagram of the harp

This was accomplished by stringing 35 µm diameter nylon string between two posts which were lowered to the height of the mesh. The top piece of the tissue chamber in conjunction

with the screws arranged around the bathing area were used to apply the compression on the o-rings necessary for preventing fluid loss between the top and bottom pieces.

4.2.2 Perfusion System

Maintaining viability of the tissue slices was a crucial component of obtaining realistic and useful results. In order to control tissue viability a common buffering solution and temperature control system were setup. Fluid inflow was maintained with a gravity fed system while the rate was controlled by a resistive roll clamp at a rate of 0.7ml/min. The working volume of the chamber was one milliliter, but it would hold two milliliters before overflowing. The fluid filled bag was hanging five feet above the tissue chamber and flow was controlled by pinch clamp. Within the chamber a nylon stand pipe controlled the height of the fluid level, at low flow rates.

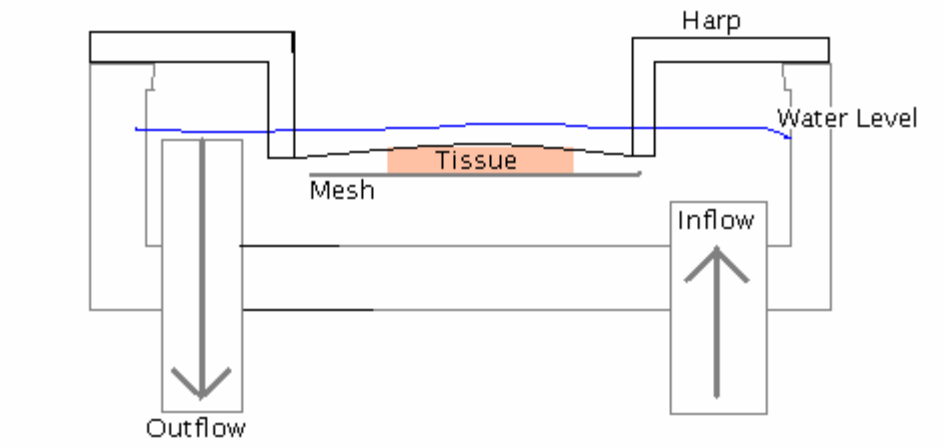


Figure 4-6 Schematic of tissue chamber

Filter paper was passed through the stand pipe to increase the outflow of fluid from the chamber and into the reservoir located directly below the tissue chamber. The reservoir was drained constantly through a vacuum chamber connected in series with a vacuum pump.

4.2.3 Micromanipulation Stages

The small size of the cochlear slices obtained, roughly 5mm in diameter, from the specimens and the nature of the measurements desired necessitated a precise electrode movement system. Due to the nature of the experiment one recording microelectrode was moved all around the tissue slice while two stimulating electrodes were stationary. The recording electrode was controlled by motorized micromanipulation stages while the stimulation electrodes were controlled by manual manipulators. Three linear actuators and three translational stages were connected in manner to provide the three-dimensionality needed for the recording electrode. The actuators (Zaber T-LA60) and translation stages (Zaber TSB) were purchased from Zaber Technologies (Zaber Technologies Inc., Richmond, British Columbia, Canada). The resolution of the actuators was 0.1 μm, providing the precise alignment for the recording electrode positioning. Two of the stages were stacked perpendicularly to each other to provide the x-direction and y-direction control. The third actuator and stage was mounted on a separate fixed support and oriented perpendicular to the x-y plane created by the first two actuators, thus controlling the vertical motion. Springs within the translation stages support the stages to keep them firmly against the end of the actuator's driving pin. However, for the vertical stage the spring could not support the weight of the stage and electrode holding apparatus, thus a pulley system was connected to provide the extra force necessary. The recording electrode was connected to the vertical translation stage with an aluminum rod onto which an electrode holder was mounted. This electrode holder provided support and was used to position the microelectrode that was used to make the recordings from the tissue bath as seen in the right hand portion of figure 4-7. The linear actuators were connected to a Dell Precision 420 Workstation (Dell Inc, Round Rock, Texas, USA) within which a custom GUI was created to control the precise movement

of the units. Figure 4.6 is a photograph of the micromanipulation stages which controlled the recording electrode. The stimulating electrodes, on the other hand, were simply mounted onto Sutter MM-3 manual micromanipulators (Sutter Instrument Company, Novato, CA, USA) and the manipulators were fixed to the base plate which also held the tissue chamber. The stimulating electrodes were also supported by electrode holders which reduced vibration and supported the electrodes to a position closer to the actual tissue.

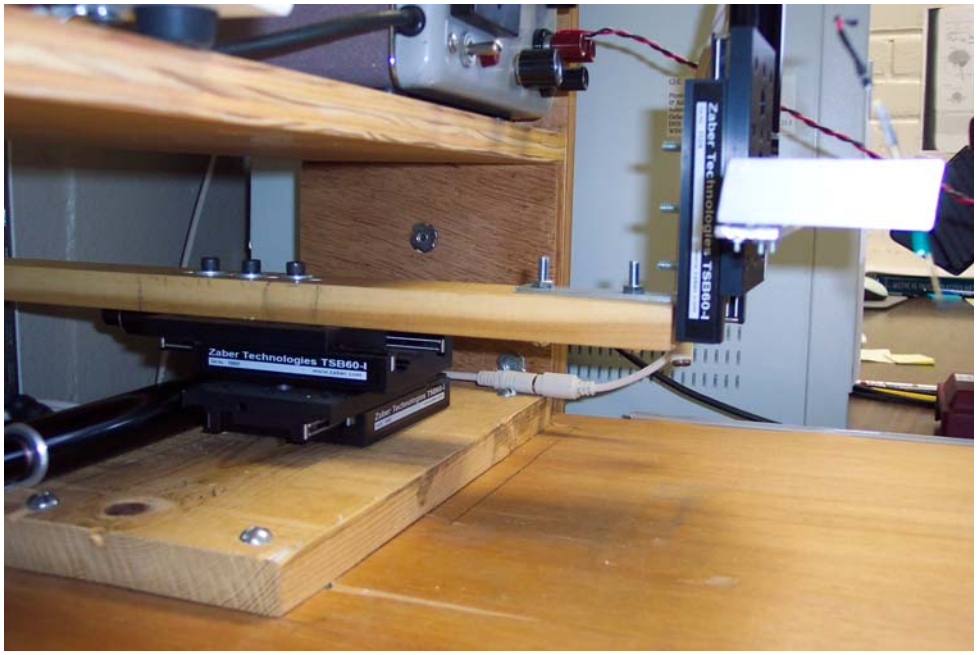


Figure 4-7 Photograph of manipulation Stage

4.2.4 Microscopy and Sample Imaging

The magnification for viewing the tissue slices for this experiment was provided through the use of the World Precision Instruments microscope model SSZ (WPI Inc. Sarasota, FL, USA). This model had a magnification range between 6.7x and 45x. Connected to the microscope was a Mintron MTV-7266ND CCD camera. The output from the camera was fed to a data translation video acquisition card installed in the Dell desktop computer. Software entitled DT Acquire Version 3.3.0 was provided with the video

acquisition card for viewing the imaged produced by the CCD camera, which allowed the user to continuously view the camera's output or to take still images and save them as a bit mapped graphic (bmp). The '.bmp' image would be captured within the DT Acquire software then saved to a file from which it would be read into the Matlab program responsible for controlling the movement of the electrodes.

4.2.5 Stimulating and Recording Electronics

The stimulation current delivered by the stimulating electrodes was controlled by a custom designed, high band width (>300kHz) current source that was built by Dr. Finley.

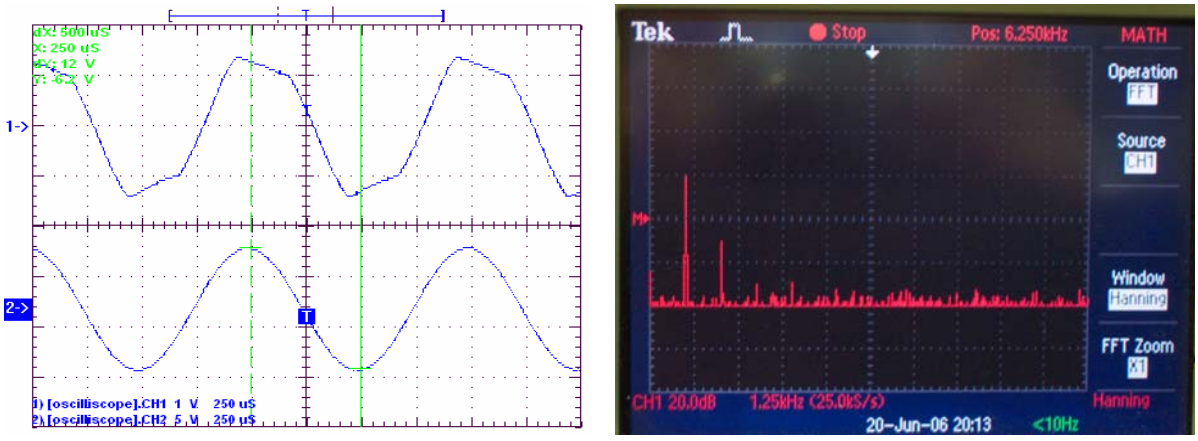


Figure 4-8 Output from current source

The recording amplifier was designed by myself and fellow lab member Punita Christopher, in completion of a separate project. The design was build around a TI INA116 instrumentation amplifier chip. This chip was chosen based on two of its distinct features. First, it provided for shielded guards, which serve to protect the lead wires from picking up environmental noise between the recording site and circuitry. This was of particular benefit since the small diameter of the stimulating electrodes only permitted voltages below 100mV. Figure 4-9 illustrates the fact that the stimulating electrodes could not operate above 100mV. The picture on the left illustrates a 150mV stimulus on top and 60mV stimulus on bottom. The small diameter stimulating electrodes constrained the level of unperturbed output that

could be sustained. The picture on the right hand side of figure 4-9 illustrates the harmonics that are created when stimulating above 100mV. Secondly, the INA116 functions with a particularly high common-mode rejection. The chip is designed to work with two inputs, a recording and a reference. The internal circuitry of the chip allows for the output of a high fidelity signal that is without noise that is common on both inputs. An additional advantage is that the complete amplifier circuit can be driven from a 9V battery; thereby eliminating noise that is typically associated with a wall-powered power sources. Additional noise reduction was gained from a lock-in amplifier, Princeton Applied Research Model 129. The lock-in is designed to accept a reference signal which is used to extract any signal with the same frequency from a noisy input signal. The bandwidth of this lock-in amplifier is 0.5Hz-100kHz.

4.2.6 Stimulation Ball Electrode Fabrication

Platinum-Iridium wires of 25 micron diameter (A-M Systems 776000) were melted with a MicroFlame Torch (Microflame 4000) to produce ball electrodes. The MicroFlame Torch combines butane and micronox to create a flame that is hotter than 1790 °C. To create the ball electrode the wire is first threaded through a 25 gauge needle for support. Melting the wire back from the tip creates a perfect ball that increases in diameter with the length of wire that is melted. Thus the length needed to create a specific ball diameter can easily be calculated and correlated to the length of wire that is melted based on equal volume of material. The balls used in this experiment were 200 microns in diameter, which is equivalent to a length of 8.53 mm. The fabrication technique described was obtained from a paper describing the fabrication process for identical electrodes to be used in a cochlear implant designed for an animal study (Sudharshana 2004).

4.2.7 Tissue Bath Recording Protocol

Experiments conducted with the tissue bath followed a similar protocol as that described in the earlier section for the simple syringe setup, with a few variations in hardware. The custom Visual Basic software controlled the output of the function generator, by sending the string "frequency (followed by the desired frequency as a number)" over the RS232 connection, which drove the current source. The current source output its signal onto platinum iridium ball electrodes which were positioned with manual micro manipulators within the bath. The head stage amplifier required three electrodes to be placed within the bath, two differential leads and a ground reference. The recording electrode was mounted to the three axes micromanipulation stage and was 125 μm diameter 12 degree tip tungsten micro electrode (A-M Systems 573210) with a parylene-C coating. The reference and ground electrodes that were needed for the head stage amplifier were custom made platinum iridium ball electrodes, see section 4.2.6. The output from the head stage amplifier was routed to a single input on the lock-in amplifier. The 'sync out' from the function generator was connected to the reference channel of the lock-in amplifier. The output of the lock-in was again set to give the magnitude and phase, which were read by the Visual Basic code through the DAQ card.

4.2.8 Tissue Slicing Technique

Drs. Marc Bassim and Carlton Zdanski have completed work to establish at UNC the technique described by Jagger et al (2000) for preparing slices of extracted cochleae of neonatal rat pups. In this technique cochleae of neonatal rat pups (5 days old) are removed following intra-peritoneal overdose of sodium pentobarbitone and decapitation. The scala

tympani and scala vestibuli of the cochlea are perfused with a 35% solution of Pluronic gel,

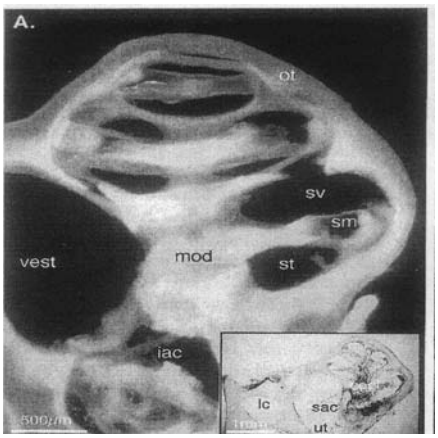


Figure 4-9 Image of cochlear slice

after opening the apex to permit free fluid flow from base to apex. The gel is a liquid at cold temperatures 4°C and is a solid at room temperature. Once perfused and warmed to room temperature, the open fluid space in the cochlea is mechanically stabilized with the solid gel, thus allowing sectioning with a vibrating microtome. Sections of 200-300 microns showing a mid modiolar cross section are retained. The mid modiolar slices are chilled to remove the gel and placed in an organ bath for microscopic visual observation and electrode probing. The quality of sections to date is considered of sufficient quality to justify the proposed measures of potential distributions, which will depend principally on bulk tissue properties as opposed to cellular viability. Figure 4.9 shows a typical section from the original paper by Jagger and colleagues.

4.3 Software

The use of a fully computer automated system was necessitated by the small size of the tissue and the type of data needed for proper analysis. Data was collected in a grid-wise manner in order that potential field plots could be constructed in a reliable manner. Two separate custom software programs were utilized for the completion of the project. The first

was written in Microsoft Visual Basic and controlled the collection of data from the recording electrode. The second was written in Matlab and controlled the movement of the recording electrode.

4.3.1 Electrode Movement Control Software

The custom software that controlled the movement of the recording electrode was written for Matlab version 7 (The MathWorks Inc, Natick, MA, USA). The functionality of the program was organized within a graphical user interface (GUI), which was designed to be easily navigated by any user. The GUI was setup to control the movement of the micromanipulation actuators described in section 4.2.3. The documentation from the manufacturer of the micromanipulation actuators provided a list of all the command strings that are recognized by the actuators as well as software written in C++ that could be used to control the movement of the actuators through the computer. Each command string sent to the actuators consisted of six bytes. The first byte is used to identify the actuator of choice which will receive the command embedded in the remaining five bytes.

The first and most important role that this code provided was a routine for scanning up to 100 points without driving the recording electrode into any obstructions. The first step that the user would take would display a saved image file of the tissue and all electrodes, stimulating and recording. This file would have been created by the Data Acquisition software described above. With the image displayed in the GUI the user could trace out a box around the tissue within which he would like to collect data from. Once this bounding box was drawn, a grid of all the points that would be used for data collection or imposed over the image. With all potential recording points displayed the user would need to decide which points should be excluded or recorded at a different level. For example, if a point landed on

top of the stimulating electrode it would need to be excluded from the recording, since putting the recording electrode down on the stimulating electrode would damage both. In this case the user would mark off the area which contained the stimulating electrodes with a box in which no recordings would be made. A second condition existed that could potentially damage the recording electrode was also protected by the computer code. This condition arose because cochlear tissue contains bone that could potentially damage the tip of the electrode. The condition was controlled similarly to the problem of the stimulating electrodes. Once the grid of all potential points was imposed the user would visually determine if any recording points would cause the recording electrode to contact bone. Since every grid point was numbered, the user would input the number of any problem points into a text field which would mark the points as bone points. During the recording routine any points designated as bone points would only drive the recording to electrode the surface of the tissue and not down into the tissue. Figure 4.10 is a screenshot of the Matlab GUI used to accomplish these tasks. In the image one can see the bounding box (green), No Zone (red box) and selected bone points (red points) as well as the 100 recording points.

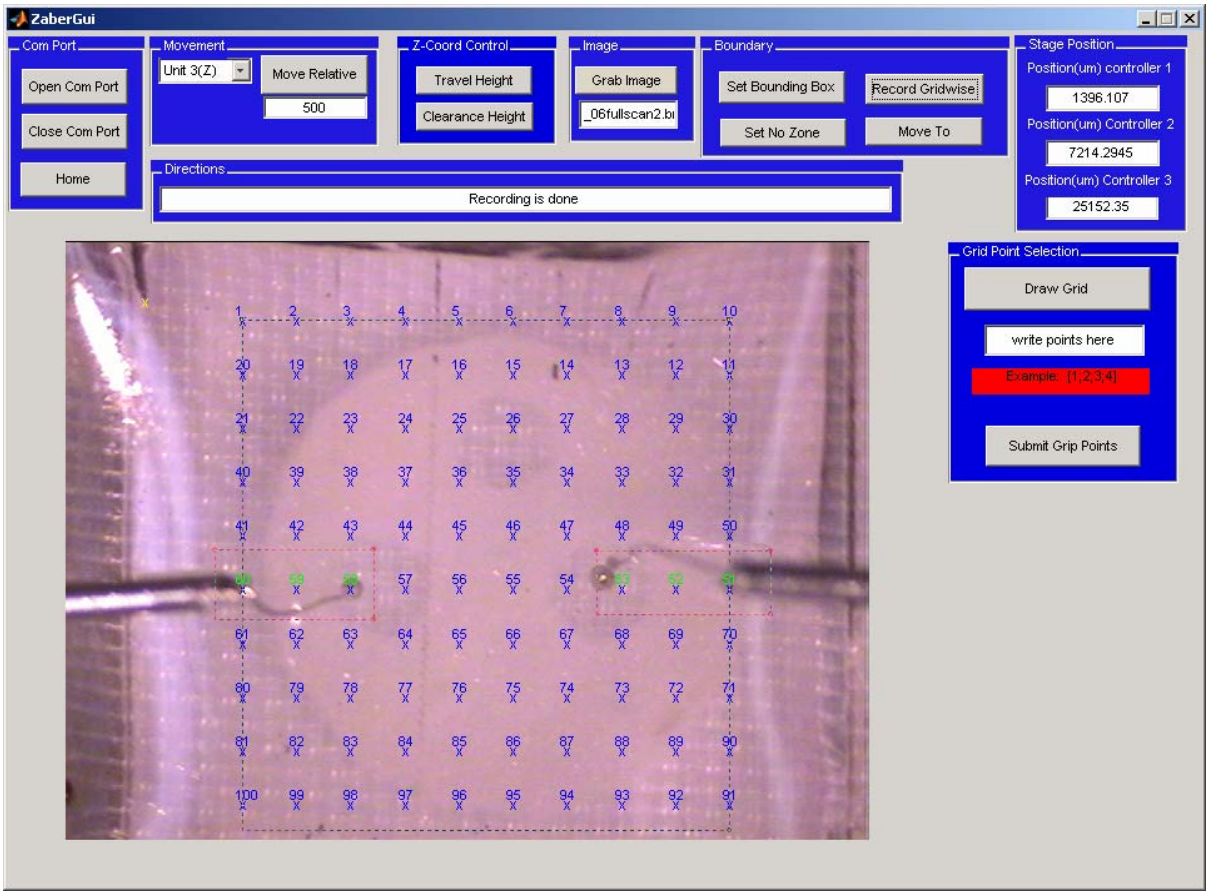


Figure 4-10 Screenshot of electrode movement control GUI

4.3.2 Data Collection and Stimulation Control Software

The program used to control the stimulation and record electric potential was written in Microsoft Visual Basic (VB). The program, named LockInAmp was constructed as an additional form to an already existing program written by Dr. Finley, called SurfacePotential. LockInAmp was added to SurfacePotential, because of SurfacePotential's robust interface with the existing data acquisition card (DAC), that was used to convert the physiological data. The DAC that was used for this project was a National Instruments (DAQCard-6062E, National Instruments Inc., Austin, TX, USA). The stimulation frequency was controlled by an Agilent 33120A function generator (Agilent Technologies, Inc. Palo Alto, CA USA). The LockInAmp program communicated with the function generator with the RS232 serial port of the computer.

Stimulation was produced through a current source constructed by Dr. Finley, while the frequency of the stimulation was controlled by the function generator. The software was written to set the function generator to every frequency on the logarithmic scale between 30 Hz and 100 kHz. These frequencies included 30, 40, 60, 80, 100, 200, 300, 400, 600, 800, 1000, 2000, 3000, 4000, 6000, 8000, 10000, 20000, 30000, 40000, 60000, 80000 and 100000Hz. All these values between these two frequencies were stored in a file which was read by the program and then passed to the function generator the change the stimulation frequency.

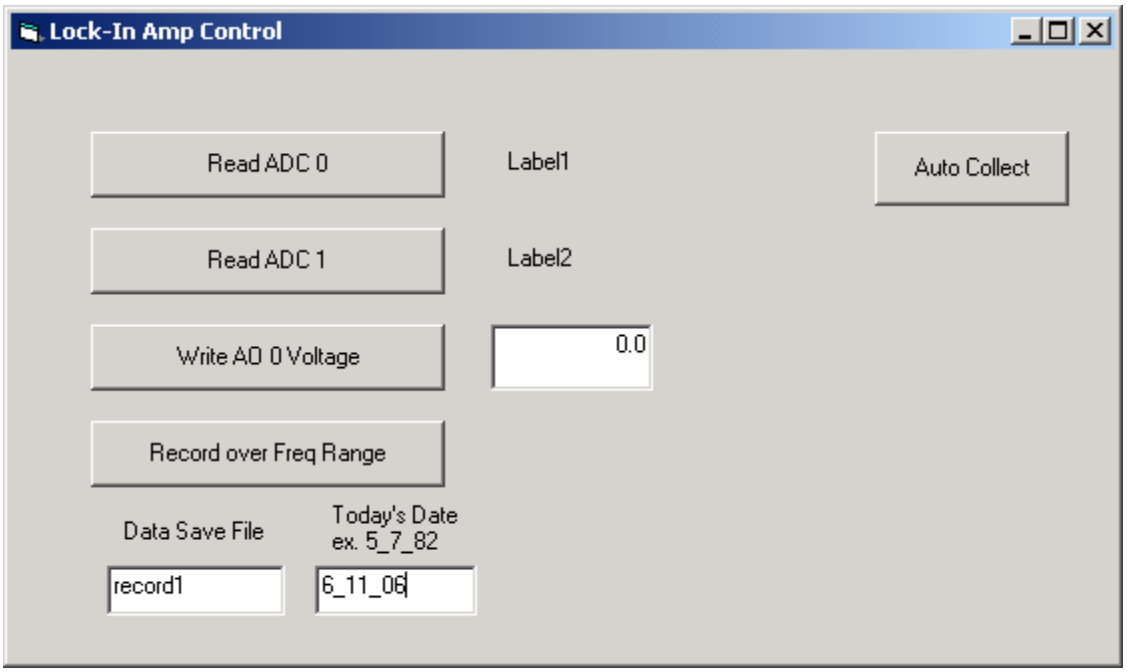


Figure 4-11 Screenshot of recording software GUI

At each frequency the software would perform two basic steps. The first was to retrieve the position of the recording electrode which was written to a file by the Matlab code described below, and write this information to a final data file. The second task was to monitor the lock-in amplifier till the readings had become stable at which point the program would record the values from the amplifier. Stability was achieved once the reading on the lock-in

amplifier was unchanged for 0.5 seconds. The values obtained from the lock-in amplifier, for magnitude and phase, were then written to the same data file as the coordinate positions. In this manner data files were created for each desired point in the field with the lock-in amplifier values at each frequency in the frequency range.

Timing between the two programs was controlled by creation and deletion of named files, and the subsequent polling of the status of said files. For example, once the data was recorded the VB code would create a file called “move.txt”, which would trigger the Matlab code to move the recording electrode to the next position. Once the electrode had arrived at the next position the Matlab code would create a file named “ready.txt”, which would trigger the VB code to begin collecting data. The “ready.txt” file would also contain the position of the electrode, which the VB code would read and combine with the collected data.

4.3.3 Data Analysis Software

Due to the vast volume of data that was collected for each trial, custom software was written to handle the data automatically. As trials were conducted the output values from the lock-in amplifier were read into the computer and written out to a tab delimitated matrix within a text file. Each point within the grid pattern would create its own separate data file with a corresponding number to the one seen on the screen shot in Figure 4.8. Each data file would contain the x-y-z coordinates of the recording electrode as well as the magnitude and phase values for all desired frequencies as recorded from the lock-in amplifier.

All the analysis programs were written in Matlab because it allows for easy matrix manipulation and construction of contours maps with sparse matrices. All programs were written to retrieve data from the saved text files, however each one differed in the output

display as well as working for a single point of frequency data or the entire grid. The first program, *singlepointbodeplot1.m*, would create a bode plot on a log-log scale from data collected at a single point. This program was used intensively during the testing phase of the tissue chamber in order to compare results with those obtained from the simple syringe setup. It was also widely used to determine the affect of stimulating and recording electrode depths. A second program, *go_bodeplot1singlegraph.m*, would expand the input capacity and display up to 10 Bode plots at a time. This was used to ensure that data was consistent and to illuminate outlier data that was not an accurate representation of the true values in the bath. The final program, *go_contourplot1.m*, constructed a contour plot from data collect in the 100 point grid pattern. The contour plot was create by retrieving the magnitude data for each point in the grid that corresponded with one desired frequency. In this manner as many contour plots could be created as there were stimulating frequencies. This program was used extensively for final data analysis. The contour plots displayed level of electrical potential at every recording point and draws equipotential lines which describe where the electrical potential declines and how fast it declines.

Chapter 5 RESULTS

5.1 Results from Simple Syringe Setup

Results from this section were used to test hardware and software capabilities as well as the characterization of test materials. For this section the results were collected to text-based files then analyzed within Microsoft excel. Graphs displayed in this section are reflections of the gain of the system, additionally the phase, real and imaginary parts of the system were collected but not displayed in this section.

5.1.1 Saline and Resistive Load Results

Results presented in this section were all obtained using the simple syringe setup described in section 4.1.2. The sample materials that were loaded in the 8cc syringe were cut using metal tubing of the same diameter as the syringe. Unless denoted otherwise all samples were three centimeters in length. Multiple slices of test material were pushed together to obtain the desired thickness. With the material in the syringe, the two syringe plungers,with electrodes on the end, were pushed towards each other until both electrodes were fully in contact with the test material. Tests denoted as ‘saline’ followed by a length, describe trials where the distance between the electrodes was the length noted, and saline filled the void between the electrodes. The syringe was placed in the metal box to ensure consistent environmental factors and electrical shielding. All testing was done at room

temperature. After data collection, the syringe and electrodes were cleaned, with q-tips, soap and water, before subsequent trials with different test materials.

The results presented in Figure 5.1 describe the behavior of the system to varying amount of saline within the syringe. The data for all the plots, except the old saline, were obtained sequentially on the same day. The 'old saline' had be recorded days prior with the same syringe, electrode and setup.

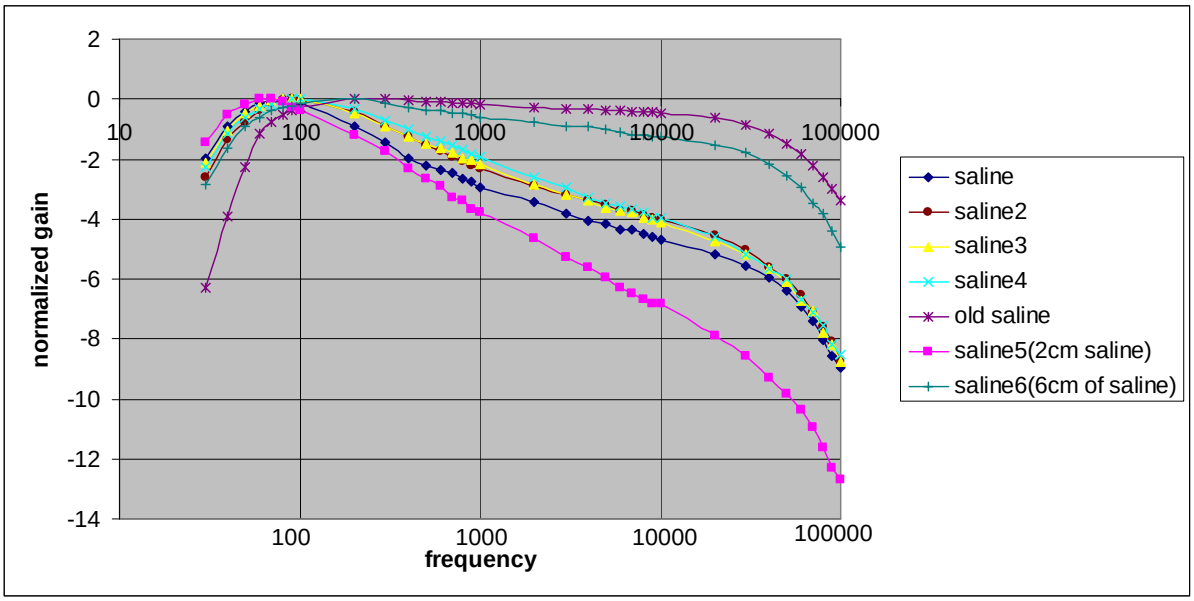


Figure 5-1 Results from pure saline in simple syringe

Figure 5.2 presents the Bode plots describing the system with a couple of different materials inside the syringe. All the results displayed were obtained on the same day with the same syringe and setup. The data plot labeled '1 slice pot' was obtained by inserting a single slice of a white potato, roughly 1 cm in length, into the syringe. 'Resistor in syringe' described the data that was obtained by inserting a 100k resistor in the syringe and contacting the resistor with the electrode tipped plungers.

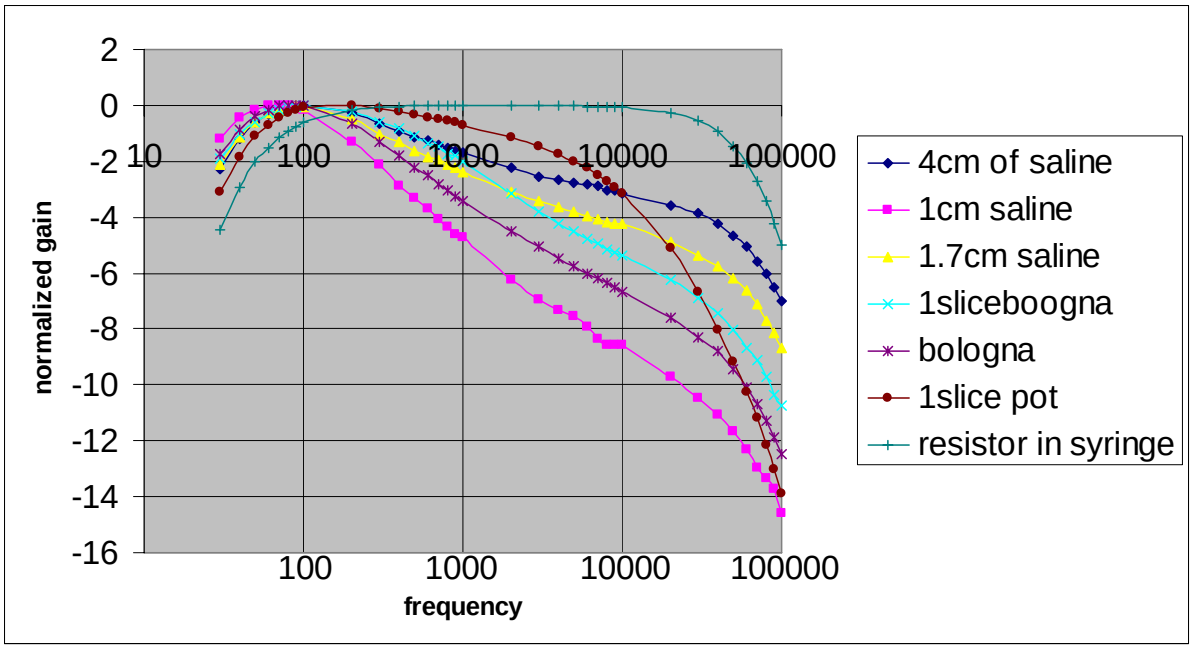


Figure 5-2 Additional results from the simple syringe setup

The data below in Figure 5.3 illustrates additional results, in the form of Bode plots, describing the system characteristics of the simple syringe with multiple materials within it. Data plots denoted as a material and saline describe trials in which a thin layer of saline was present on both sides of the material between the electrodes. The data plots denoted simply by a material name are those with three centimeter pieces of material directly contacting the electrodes. The data plot denoted as 'resistor' was obtained by directly connecting a resistor the BNC connectors of the box. Consequently neither the syringe nor the electrode tipped plungers were used for this trial.

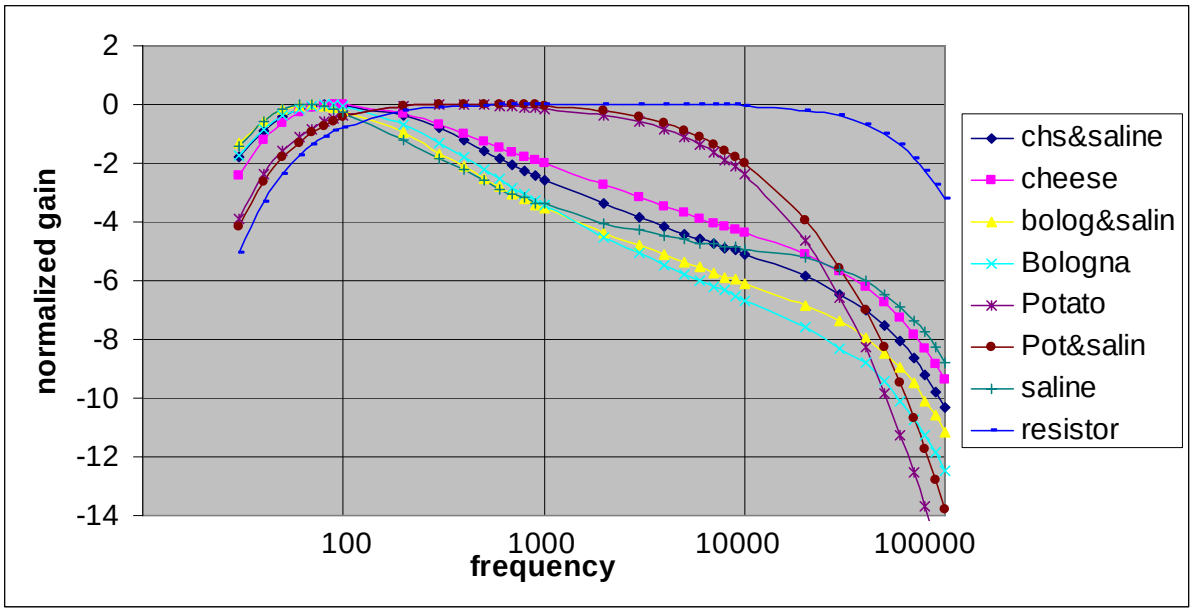


Figure 5-3 Results from alternate materials in the simple syringe setup

5.2 Results from Tissue Bath Setup

The results presented in this section were analyzed with Matlab software code described in section 4.2.3. The description of the hardware setup followed in collecting the data presented in this section can be found in section 4.1.10. As was the case in section 5.1 the results below only depict the magnitude response of the system; the phase, real and imaginary data were collected but not analyzed. All bode plots presented in this section are plotted as magnitude, as read off the lock-in amplifier, vs. frequency. It should also be noted that all the slices used in acquiring the results presented were cut by hand with a razor, so the thickness of slices had the potential to vary. However all slices were roughly 0.5 mm thick.

5.2.1 Single Point Repeatability Test Results

The sets of data presented in this section were both obtained while using the same protocol. However between the times at which the two sets of data were obtained multiple changes to tissue bath setup were executed. All three electrodes used by the head stage

amplifier were replaced and the gain setting on the amplifier was adjusted. Between each of the trials the recording electrode was removed from the bath and then placed back into the bath at the same position. During the trials no electrodes were moved or adjustments made to the hardware. Figure 5.4 presents the data collected from the tissue bath before the changes in electrodes were made. Figure 5.5 presents the data that was collected after the changes had been made.

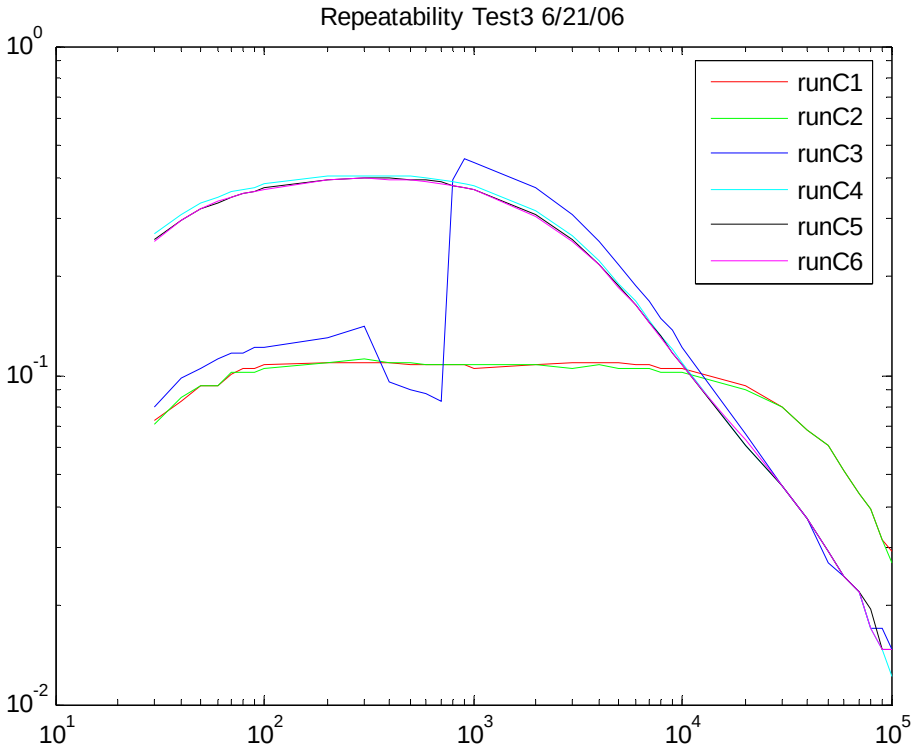


Figure 5-4 Bode plot of repeatability test A

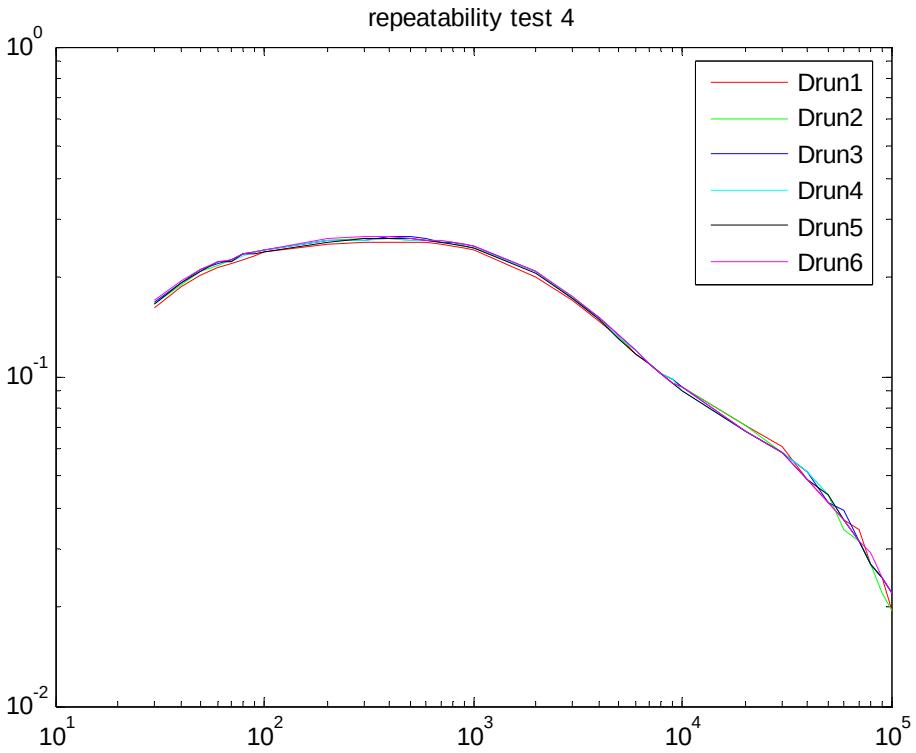


Figure 5-5 Bode Plot of repeatability test B

5.2.2 Single Point Depth Dependency Test

Results from this section were used to determine the effect caused by depth of electrode placement in multiple scenarios. The data entitled 'updown test' was a series of trials in which the recording electrode and stimulating electrodes were set at heights either above the surface of the tissue slice or within the plane of the tissue slice. In all these trials, holes were made in the slice for the stimulating electrodes, and the recording electrode was driven directly into the sample slice. Determining the height at which the electrodes were above or below the surface of the slice was confirmed through visual inspection. Duplication of the heights between tests was difficult, so the second set of trials entitle 'depth tests' were conducted in which the stimulating electrodes were stationary and recording electrode was solely moved. The results that are presented below were obtained from two sample types. The tests that are denoted as 'updown tests' were all acquired from bologna slices, while the

depth tests that are presented were acquired from white potato slices. 'Depth tests' were also made in bologna but are not presented as the same trends were reflected in the potato data.

The 'updown tests' were all conducted in the same order; the first reading was made with all electrodes above the sample, then the recording electrode was moved into the plane of the sample, next the stimulating electrodes were moved into the plane of the sample and the final trial was conducted with the stimulating electrodes in the plane of the sample and the recording electrode above the sample. Following this protocol allowed for the stimulating electrodes to be moved only once, this was ideal since they were manually controlled. Figures 5.6 and 5.7 show two such trials with four graphs each.

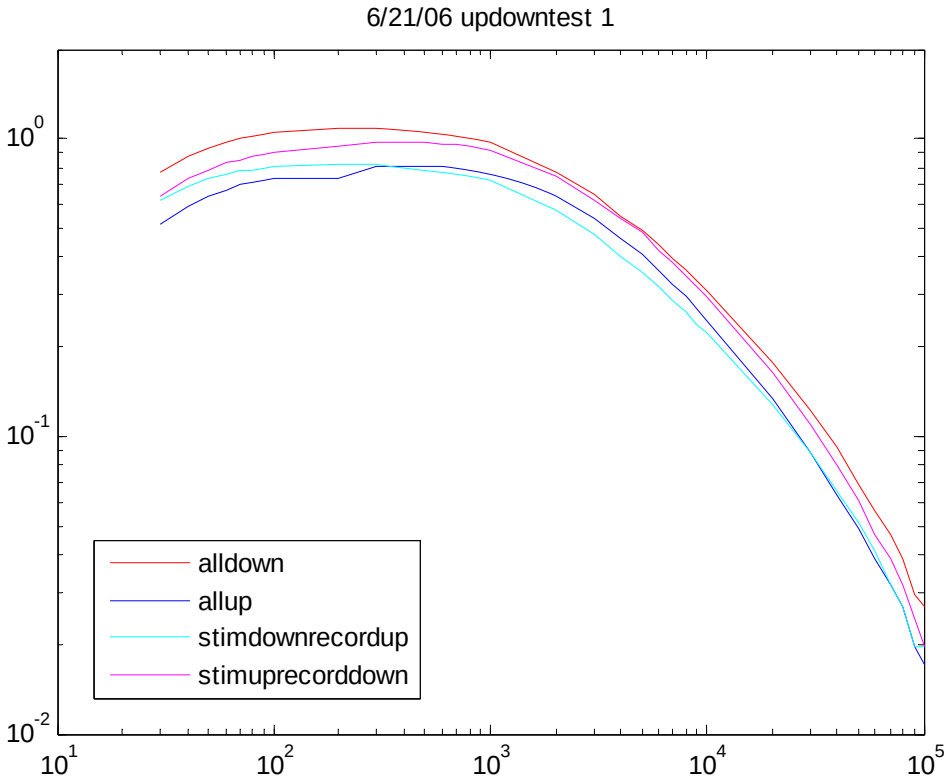


Figure 5-6 Bode plot of up down test1

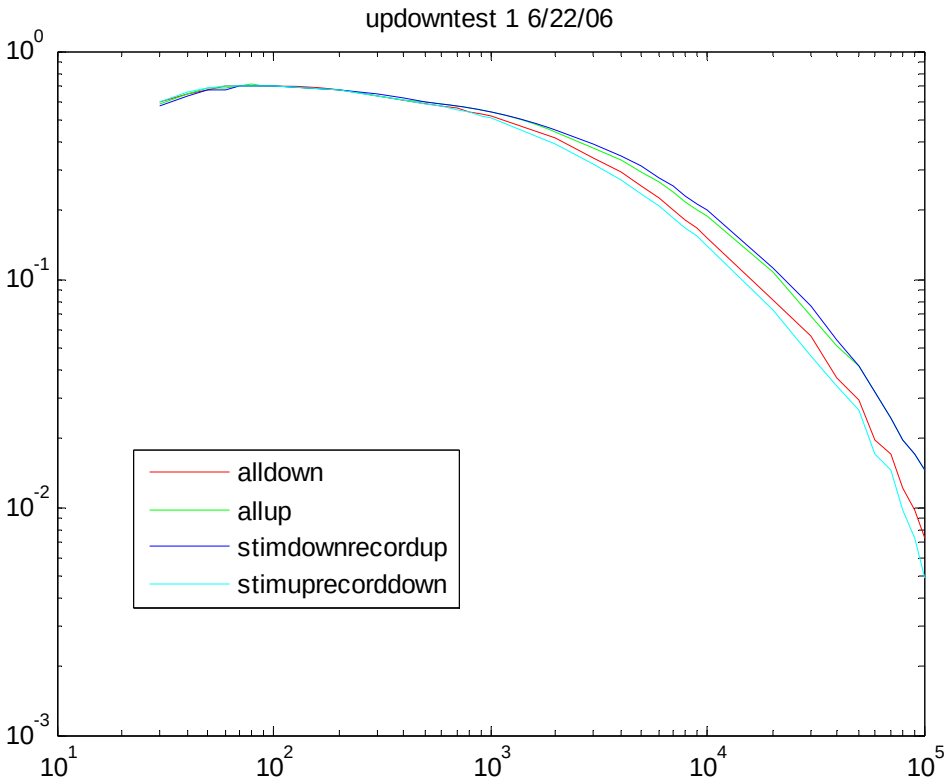


Figure 5-7 Bode plot of up down test 2

The data displayed in Figure 5.8 was collected in the same manner as the data in Figures 5.6 and 5.7, but after the four trials were collected the sample slice was removed and two more recordings were made with out the sample in the bath.

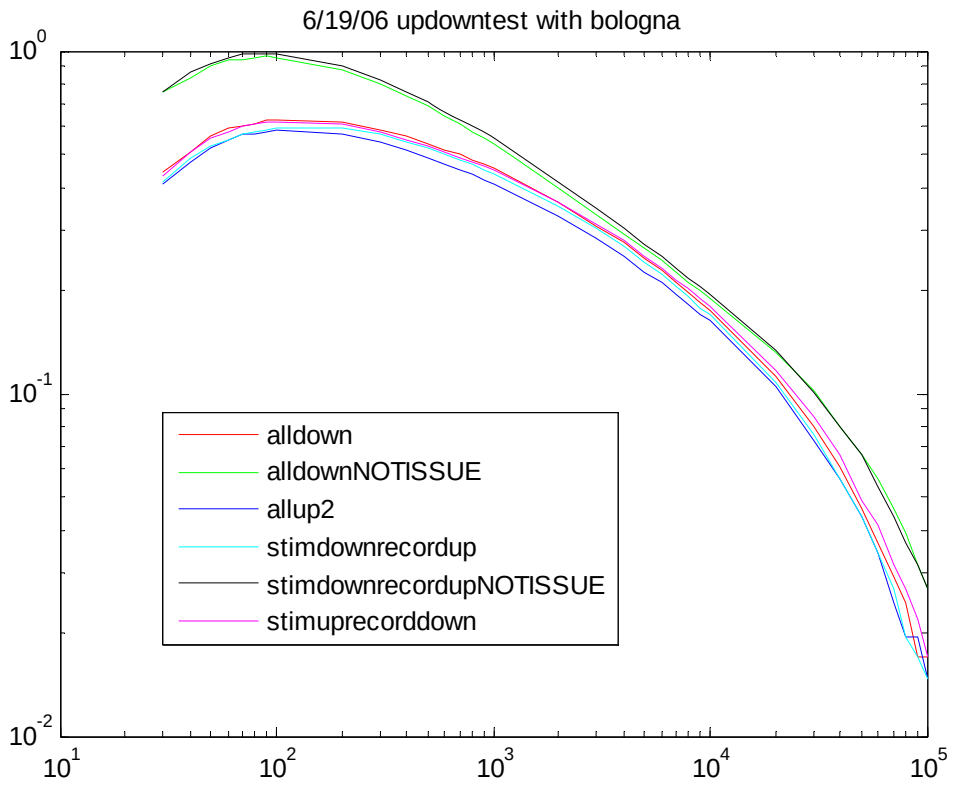


Figure 5-8 Bode plot of up down test with no sample

The variability inherent in the visual determination of electrode depth in the 'updown tests' led to the second method of testing the effect of electrode height. The data that was collected from this method was entitled 'depth test.' For these runs the depth was precisely measured using the micromanipulation stage. The titles given to each trial represent the distance the recording electrode has traveled from it fully retracted position. Thus larger numbers mean that the recording electrode is further down in the bath. In Figure 5.9, 5.10 and 5.11 the top plane of the sample slice was roughly 26600 microns. Figure 5.9 shows the resulting Bode plot when the recording electrode was driven directly into the sample, in this case it was a potato slice. The stimulating electrodes were positioned in holes, roughly 1mm in diameter, made in the slice and were resting on the mesh lining.

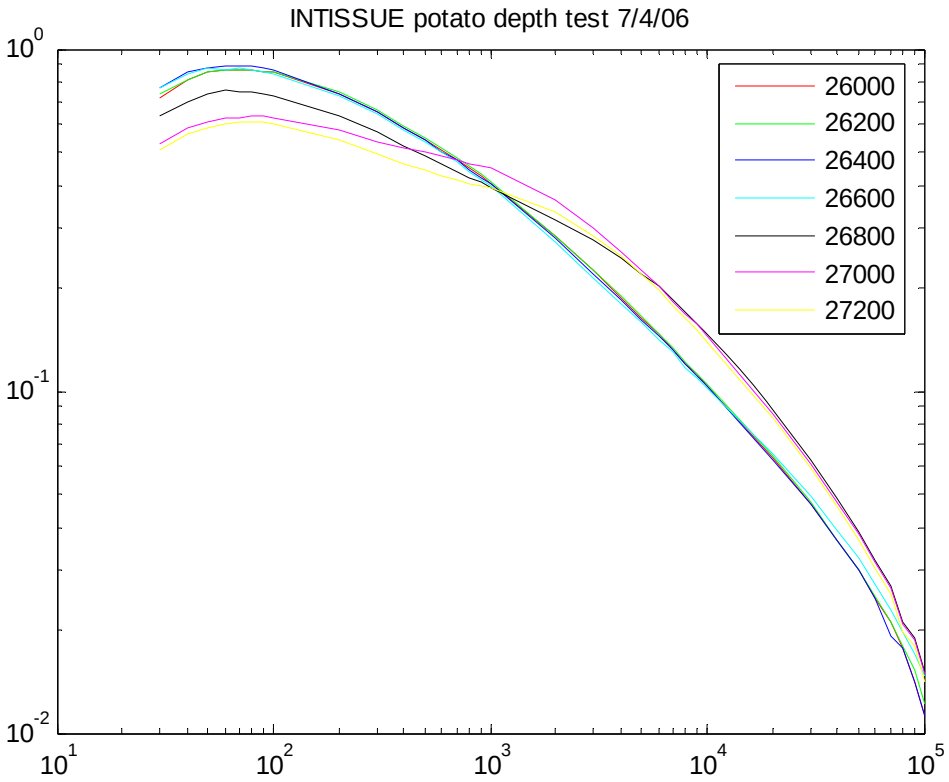


Figure 5-9 Bode plot of depth test INTISSUE

In comparison to Figure 5.9, the Bode plot seen in Figure 5.10 was created by driving the recording electrode into its own hole within the same potato slice. The degree of overlap seen in Figure 5.10 was not noticed in bologna slices under the same protocol procedures. The protocol that was established for 'depth tests' collected an additional set of measurements directly after the measurements that were used to produce Figures 5.9 and 5.10, in this final set of measurements the sample slice was removed and measurements were made at three sequential depths within the same vertical range as the previous measurements, as seen in Figure 5.11.

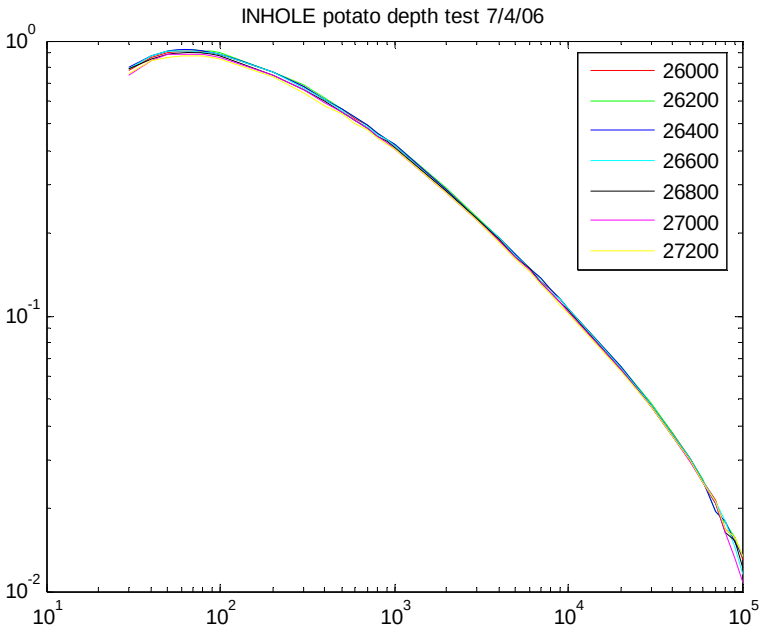


Figure 5-10 Bode plot of depth test IN HOLE

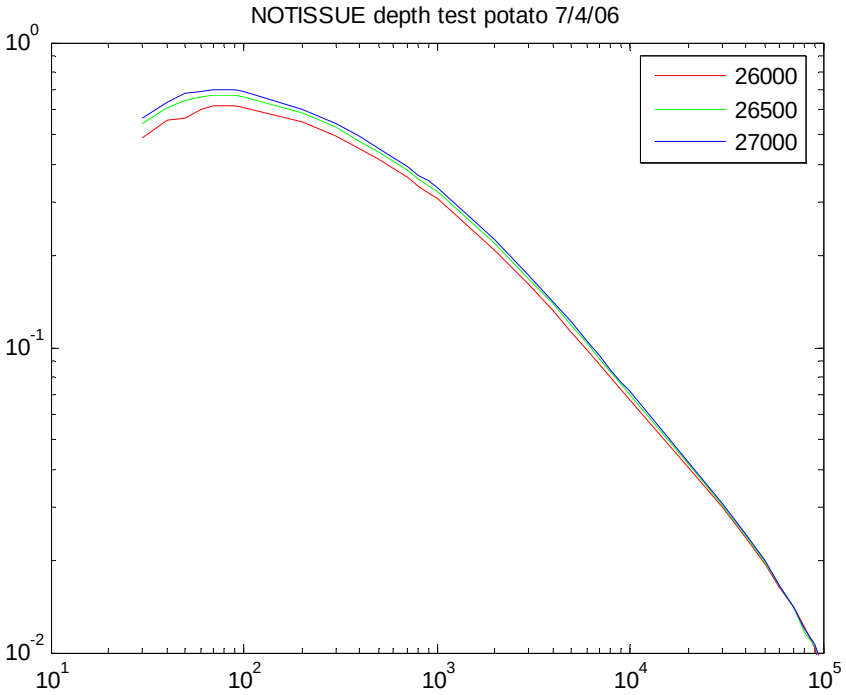


Figure 5-11 Bode plot of depth test without sample slice

5.2.3 One Hundred Point Grid Tests

The results that are presented in this section demonstrate the feasibility of the approach taken to create field potential plots of the tissue bath and its contents. The results from the previous two sections helped improve the process of recording from the tissue bath, however the final trials with actual cochlear tissue will be conducted under the protocol that produced the results in this section. By combining hardware and software described in the previous chapter, an automated system collected the frequency dependent data needed to produce the Bode plots seen in the previous section for nearly all one hundred points. As described in section 4.2.2 a 'no zone' around the stimulating electrodes was established which eliminated six data points. Interpolation for these data points was found by averaging the two points vertically on either side of the missing point. Two types of graphs were produced from the data collected. The contour plots are 2D plots in which the lines represent equal valued areas. The surface plot is a 3D plot representing the same data in a relief map manner. Figures 5.12 and 5.13 are the contour and surface plots obtain by sampling the Bode plot at every point at 200 hertz.

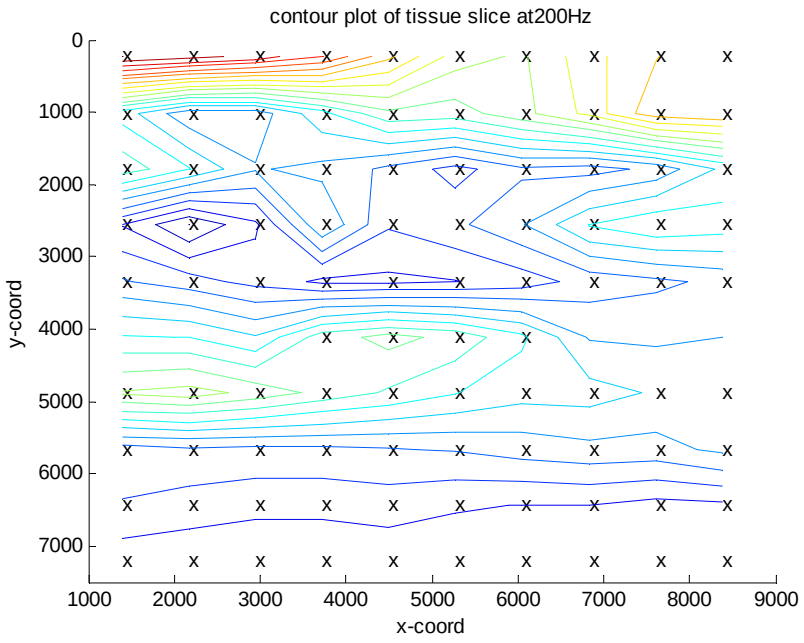


Figure 5-12 Contour plot at 200 hertz

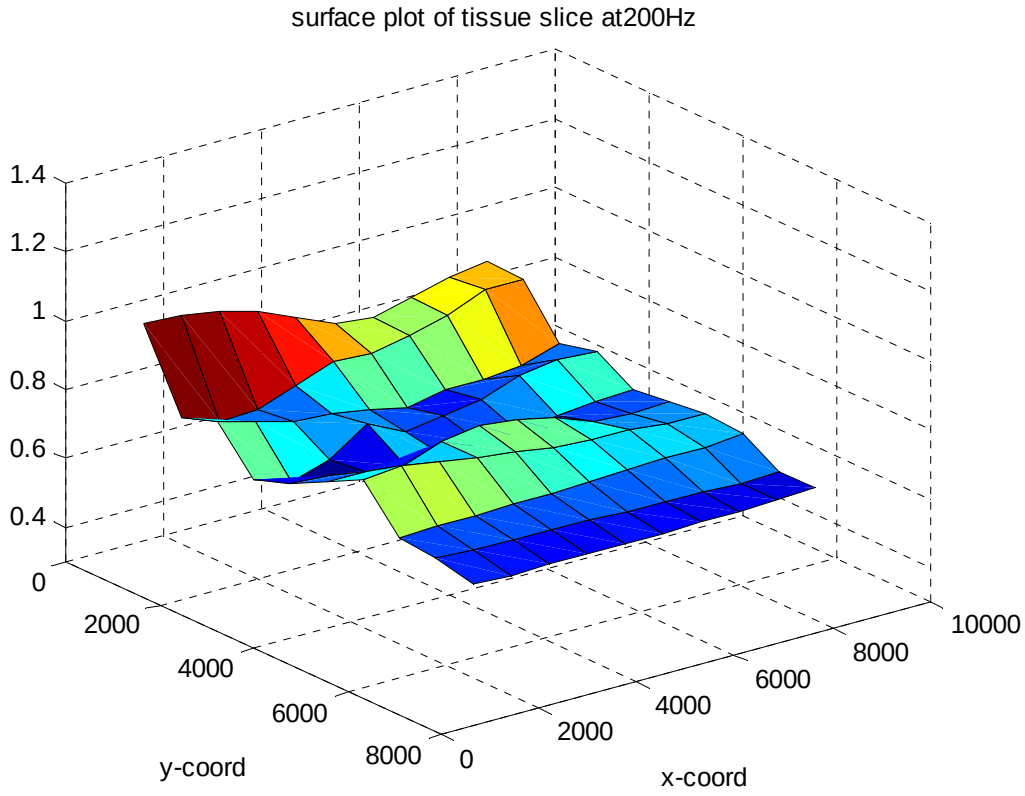


Figure 5-13 Surface plot at 200 hertz

The middle of the frequency range over which this data was collected is represented in Figures 5.14 and 5.15. These Figures present the contour and surface plots of the data sampled at 2000 hertz.

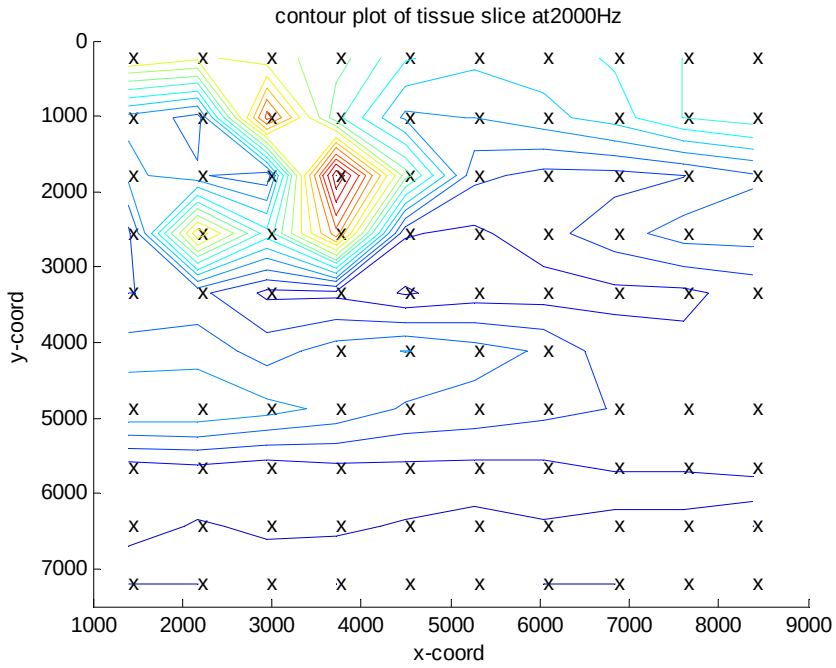


Figure 5-14 Contour plot at 2000 hertz

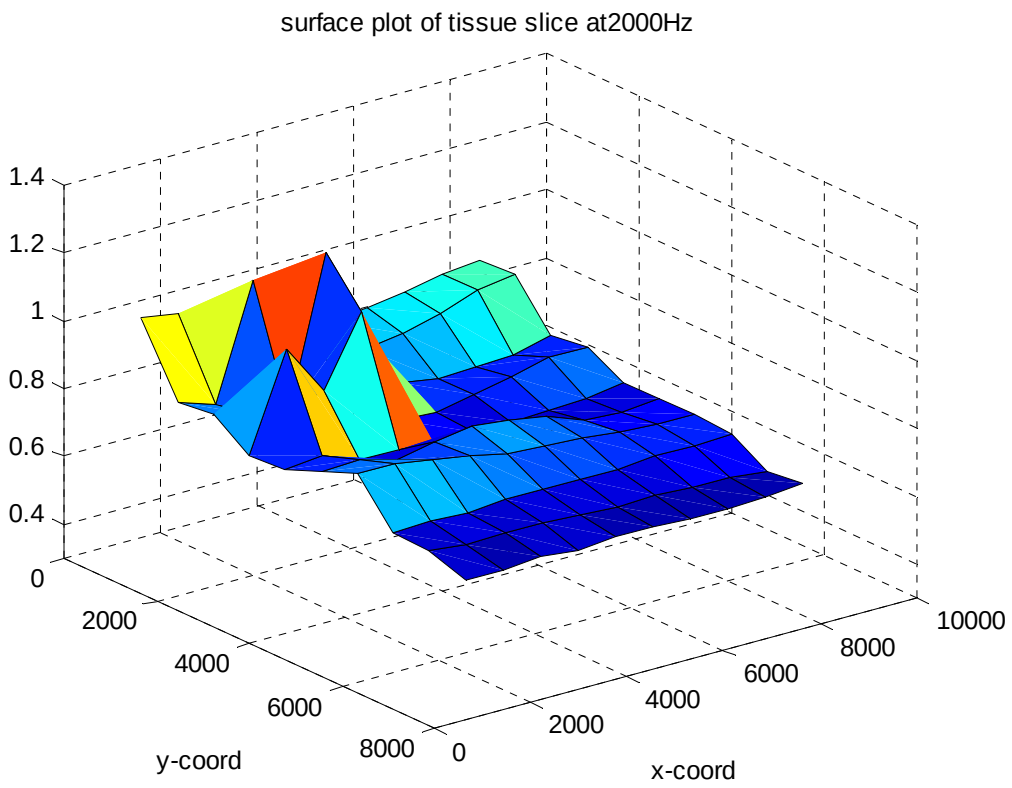


Figure 5-15 Surface plot at 2000 hertz

The high frequency data is represented in Figure 5.16 and 5.17, which are the contour and surface plots sampled at 80000 hertz. Figure 5.18 is the screen shot from the Matlab GUI that controlled the electrode movement and displays the points at which data was collected in the tissue bath.

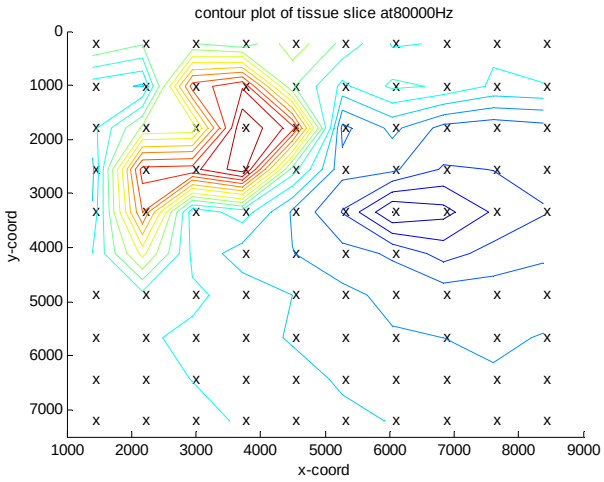


Figure 5-16 Contour plot at 80000 hertz

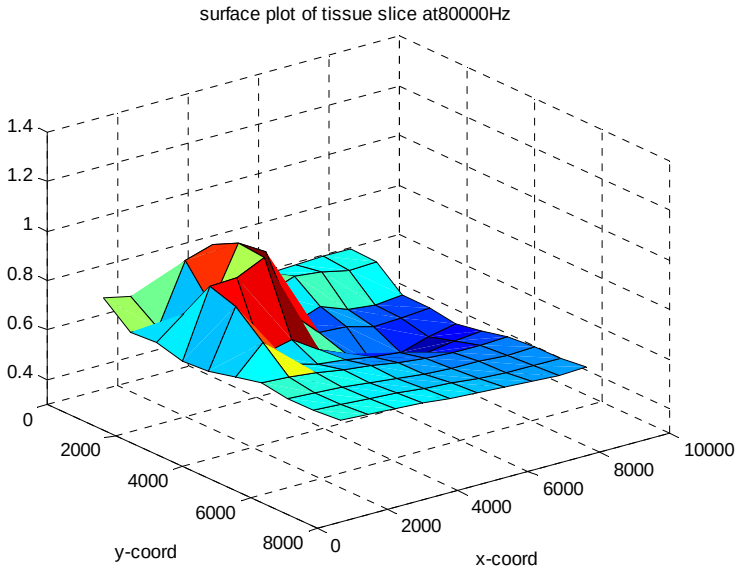


Figure 5-17 Surface plot at 80000 hertz

The contour and surface plots in figures 5-12 through 5-17 were all constructed by sampling the bode plots at each individual point. Figures 5-18 through 5-20 present the Bode plots across thirty of the 100 points, in order to illustrate were the fluctuations seen in the contour and surface plots. Figure 5-18 depicts the first ten points that were collect and are reflected at the top row on points on the contour plots. Figure 5-19 depicts points twenty one

through 30, which correspond to the third row of point in the contour and surface plots. This corresponds to the row in which the large mound can be found in the surface plots. Figure 5-20 depicts the last ten points of the scan, or the bottom row of the contour and surface plots. The overlapping of the Bode plots explains the flatness of the contour and surface plots on the bottom row.

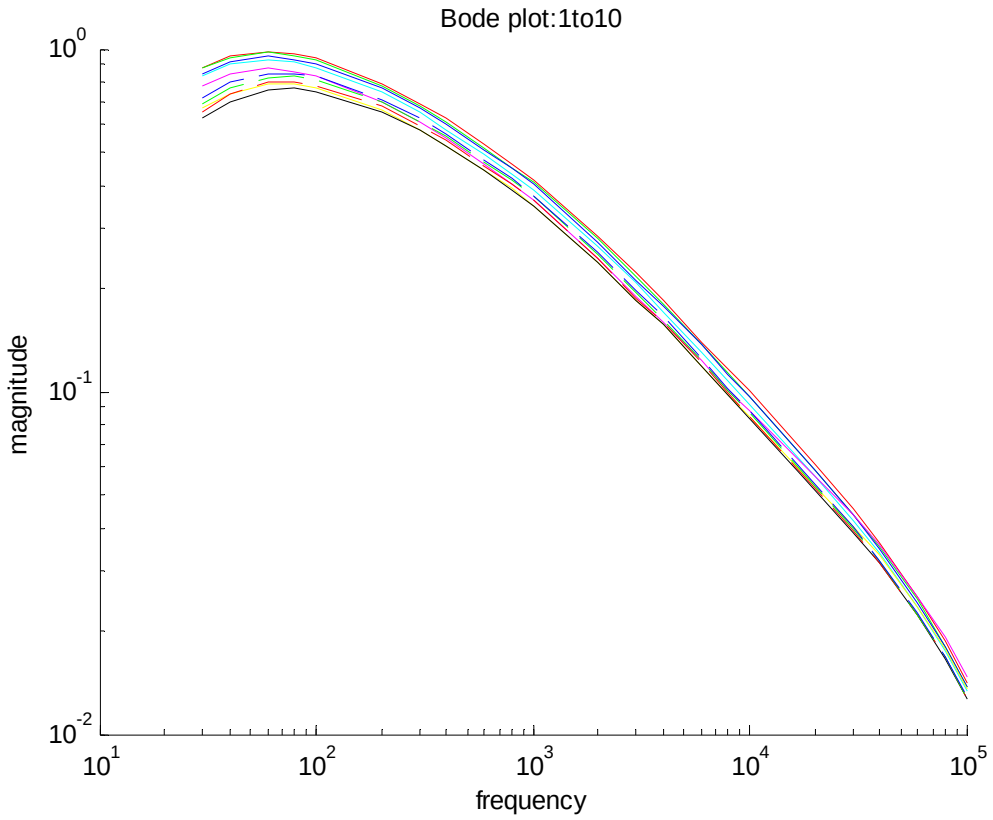


Figure 5-18 Bode plot of first ten points of full scan

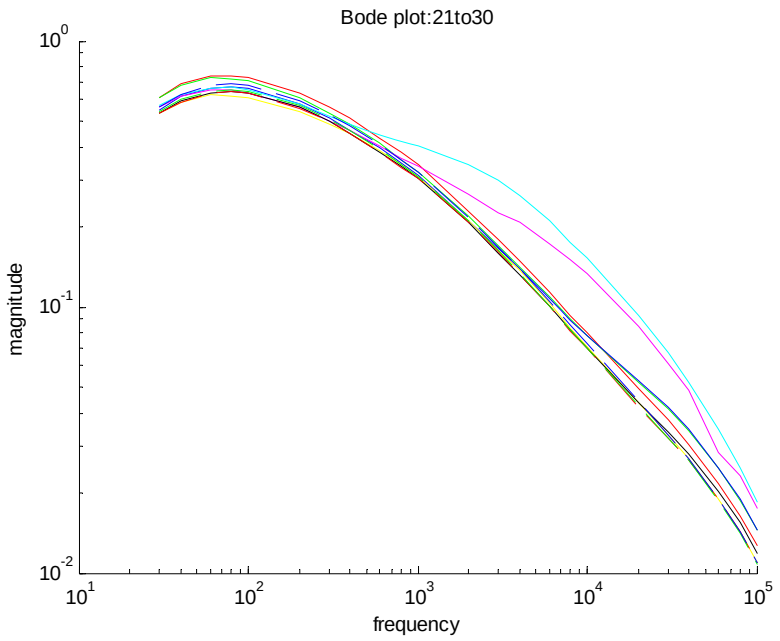


Figure 5-19 Bode plot of points 21-30 of full scan

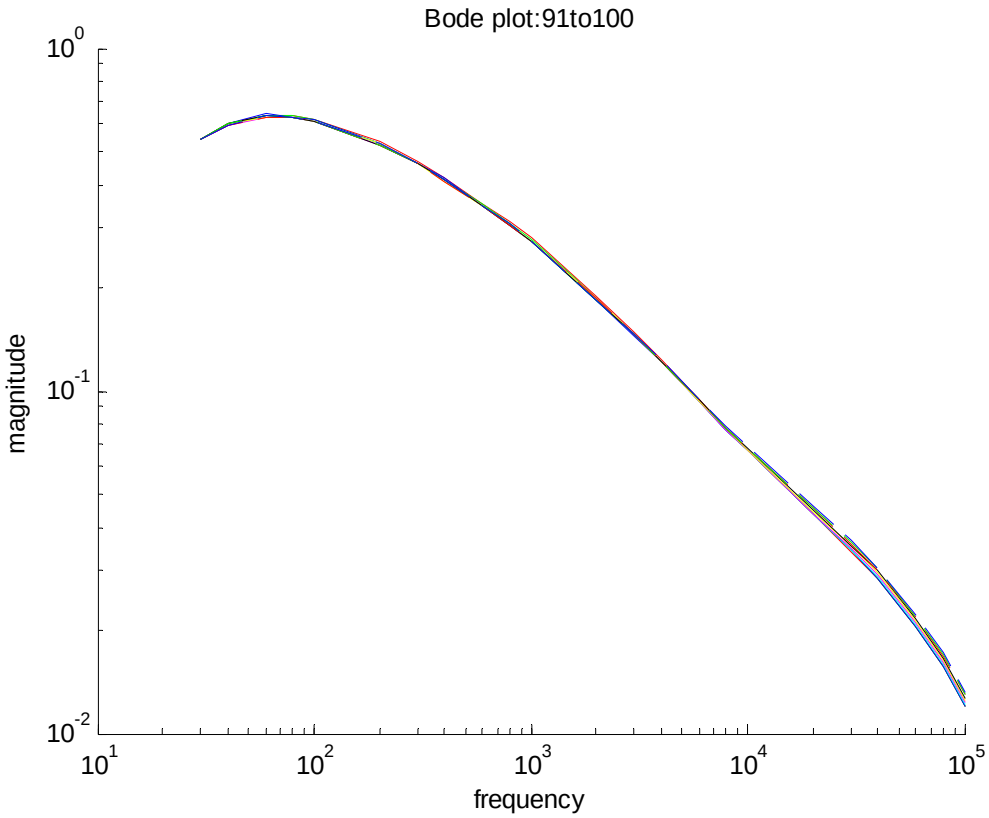


Figure 5-20 Bode plot of points 91 through 100 of full scan

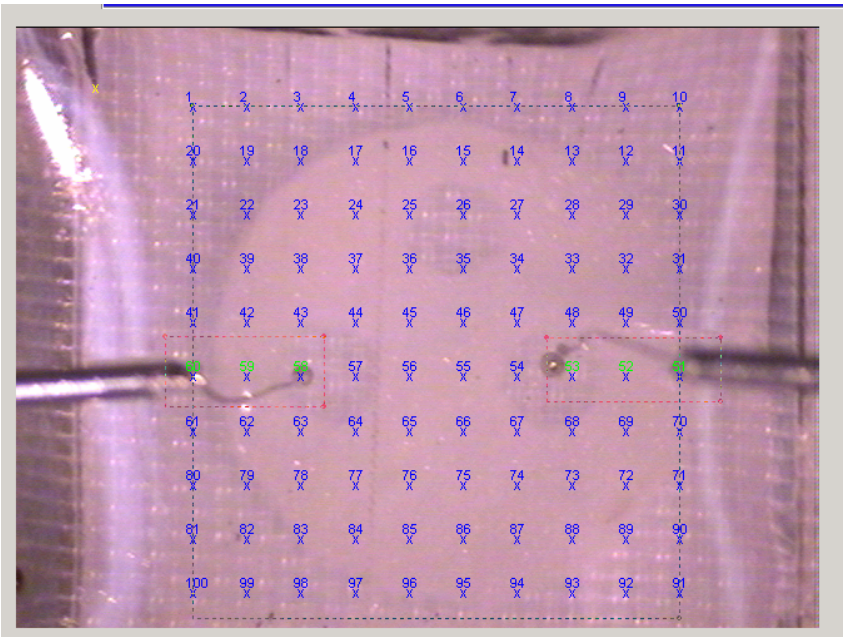


Figure 5-21 Screen shot of scanned slice

Chapter 6 DISCUSSION

6.1 Proof of Setup Performance

The goal of producing a tissue perfusion chamber and measuring the potential throughout was successfully accomplished. The results described by Figures 5.12 through 5.17 demonstrate the capability to produce the contour plots that could potentially describe the characteristics inherent in a tissue slice. The time to acquire all one hundred points was roughly three hours. During which time the fluid level in the bath was constant and did not have to be adjusted. While the contour and surface plots obtained were not able to distinguish between points in the slice and out of the slice or in the hole, previous results attest to the system's ability to accomplish such tasks. For example, the bode plot in Figure 5.8 clearly displayed a difference between the presence and lack of a sample slice, with the NOTISSUE curves clearly distinguishable from the rest. Additionally, the difference between Figure 5.9 and 5.10 demonstrate that a noticeable difference can be identified between the recording tip in a hole and in the tissue, below a certain depth, which corresponded to the top plane of the tissue. All these signs point to the conclusion that this approach ought to be able to characterize a material placed in the tissue chamber.

Every point of data that has been collected has given insight into improvements that can be made to the system. The repeatability test presented in Figure 5.4 clearly demonstrates the system's ability to replicate results from one trial to another, but more importantly it demonstrated that the system was not perfectly stable. As can be seen in the

dramatic change in ‘runC3’ indicating that a shift in magnitude occurred and then was constant for the remainder of the trials. It was not discovered what caused the change but the results did inspire the addition of greater control over positioning of all electrodes. All the ‘updown tests’ that were conducted led to the conclusion that the height of the stimulating electrode played a small role in the determining the results that were obtained. Consequently, when the protocol for the ‘depth tests’ was designed the height of the stimulating was left constant and the focus was shifted to the recording electrode. The first ‘depth tests’ quickly demonstrated that the recording electrode that was being used was in fact damaged and was subsequently replaced. Fluctuation between trials lead to the observation that movement of the reference electrode could greatly affect the results that were obtained, so the reference electrode was then firmly attached to the tissue chamber to avoid movement. Step by step the process has been refined until a complete 100 point scan could be conducted with a complete belief that obtaining reliable results was achievable.

The contour plots and surface plots displayed in the previous chapter are not what were expected, but when reviewed, these figures shine light on where the project needs to head in the future. At the low frequencies we notice that the first 5 points of the scan, starting from the bottom right corner of figures 5-12 and 5-13, are abnormally high, indicating that the low frequency level signal was dropping off rapidly after these points. It was noted in separate experiments that if the system was left alone at a low frequency for variable amounts of time the amplitude of the signal would fluctuate. This can also be contributed to the milder hump in the middle of the surface plot, best seen in figures 5-15 and 5-16, after the electrode had been removed from the bath to clear the stimulating electrode and the move back down to the bath. Additionally, the mound that can be seen in the high

frequency surface plots can likely be contributed to an un-level sample slice. It was noticed that when the recording electrode was moved at a constant height above the tissue it would snag on the tissue at the top left corner where the mound is located on the surface plot. This observation correlated with the bode plot in Figure 5.9 would agree that the further the electrode is down in the tissue the higher the amplitude would be at higher frequencies. It can be proposed that the results that prove that this approach ought to work also raise many issues that need to be resolved before it provides the desired results.

6.2 Limitations of the Approach

While there are still minor obstacles to be overcome, only two foreseeable limitations have yet to be explored. The first is how long the flow rate and design of the tissue chamber will support a live tissue slice. While the flow rate is within the window of ranges used by other researchers, the design of the chamber was completely unique. The position of the inlet and outlet ports may or may not provide even distribution. If it was discovered that the chamber could not maintain a tissue slice for a long enough period of time to make the necessary recording, it would not doom the research. None of the hardware is specific to the chamber so a chamber redesign would not preclude the use of any of the hardware.

The second limitation and most troubling is the effect of shunt resistance. The trouble arises with the fact that current will follow the path of least resistance. Consequently, the path along which the current travels needs to be understood. Figure 6.1 presents an idealized circuit that can be used to understand the complications of shunt resistance.

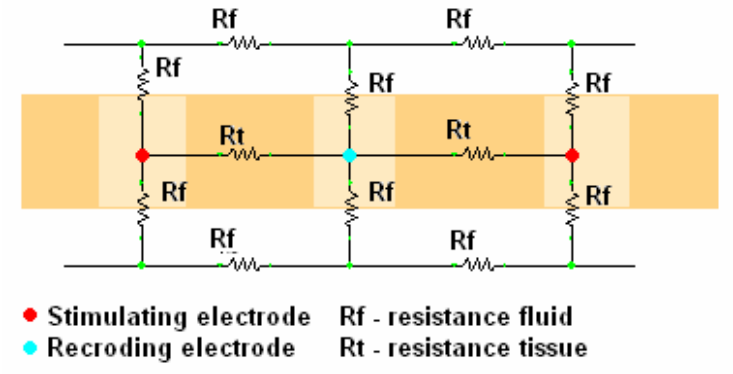


Figure 6-1 Resistor simulation of tissue in bath

Further testing will have to be completed to determine the amount of current that is picked up by the recording electrode is due to current passed through the tissue or current that is shunted through the saline around the tissue. If the recording electrode is overwhelmed by shunted current, then information about the tissue slice will not be observed. Such a situation would occur if, as seen in figure 6-1, the resistance of the tissue (Rt) was much greater than the surrounding fluid (Rf). This question will have to be answered through many controlled experiments that can test the effect the bathing solution level has on the recording. While the method that has been established may or may not be affected by shunt resistances, the method does provide an easy manner for detecting changes in magnitude of the desired signals. Since the results are displayed with a contour plot or surface plot, the baseline signal is virtually eliminated so the only results that are observed are those variations from the baseline. As a result the system does not have to be perfectly identical between experiments; as long as it is stable over one experiment, results from one experiment can be compared to another.

6.3 Future Work

The goal as stated in the introduction to characterize cochlear tissue slices is the pinnacle of this line of research. In order to achieve this goal steps will have to be taken in a

coupled directions. First, identifying and eliminating any sources of fluctuation in signal levels within a trial must be eliminated. That is the first step to establishing a truly accurate baseline. Secondly, more trials need to be conducted to determine the exact affect of holes within the samples. Along these same lines, since cochlear tissue contains bone the system should be tested for its ability to make readings in or around bone and the specific effects bone might have on the field distributions. More work will be necessary to determine and correct for the limitations described above. Finally, testing needs to be done with the actual bathing solution that will be used with the cochlear slice to determine if the chamber and electrodes operate in the same manner as they have in saline.

Chapter 7 CONCLUSIONS

This project has succeeded in developing a system for perfusing tissue slices and recording electrical potentials within the perfusing area. Software was written that automated the collection process as well as the analysis. While the ultimate goal of the research, characterizing a cochlear tissue slice, was not reached, the results that have been recorded have provided a wealth of knowledge for the progression towards that goal. The problem of shunt resistance, which could previously only be hypothesized about, is nearly ready to be tested on a physical system, where real results can speak for themselves. The system's capabilities were demonstrated on a potato slice in an experiment that provided constant profusion for roughly three hours. All results to date indicate that the system and process for producing the contour plots will provide informative results that may one day improve the effectiveness of cochlear implants.

APPENDICES

The following Appendices contain source code listings of MatLab and Visual Basic software developed by the author to facilitate data collection and data analysis. These Appendices include:

Appendix A: MatLab code for controlling the movement of the recording electrode
Appendix B: Visual Basic code for recording and storing data
Appendix C: MatLab code for extracting data from text files
Appendix D: MatLab code for creating potential field plots
Appendix E: MatLab code for creating bode plots of collected data
Appendix F: Corel Draw files for tissue bath

Appendix A: MatLab code for controlling the movement of the recording electrode

This software controlled all the movement of the recording electrode. It allowed the user to import an image of the tissue to be scanned and move the recording electrode to any spot in the image. The code also worked in conjunction with the visual basic code in Appendix B to automatically record data from 100 points within the imported image. With the imaged displayed the user is able to box off the area of interest and select the points within that area that the recording electrode can record from. The code presented below must be opened in conjunction with a MatLab figure file which is the template for the GUI interface.

```
function varargout = ZaberGui(varargin)
% ZABERGUI M-file for ZaberGui.fig
%      ZABERGUI, by itself, creates a new ZABERGUI or raises the existing
%      singleton*.
%
%      H = ZABERGUI returns the handle to a new ZABERGUI or the handle to
%      the existing singleton*.
%
%      ZABERGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in ZABERGUI.M with the given input arguments.
%
%      ZABERGUI('Property','Value',...) creates a new ZABERGUI or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before ZaberGui_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to ZaberGui_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help ZaberGui

% Last Modified by GUIDE v2.5 10-Mar-2006 15:28:54

% Begin initialization code - DO NOT EDIT
%----------------global variable-------------------------------------
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ZaberGui_OpeningFcn, ...
```

```
                'gui_OutputFcn',  @ZaberGui_OutputFcn, ...
                'gui_LayoutFcn',  [] , ...
                'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ZaberGui is made visible.----------------------
function ZaberGui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ZaberGui (see VARARGIN)
handles.axis_number = 1;
 %start count for No Zone boxes
% Choose default command line output for ZaberGui
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ZaberGui wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.--------
function varargout = ZaberGui_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.----------------------------
```

```
%this button grab the desired picture and displays it on the axis
function pushbutton1_Callback(hObject, eventdata, handles)
set(hObject, 'Units', 'pixels');
imagefile = get(handles.edit4,'string');
handles.picture = imread(char(imagefile)); % Read the image file banner.jpg
info = imfinfo(char(imagefile)); % Determine the size of the image file
position = get(handles.axes1, 'Position');

%set(hObject, 'Position', [position(1:2) info.Width + 100
%info.Height + 100]);
axes(handles.axes1);
image(handles.picture);
handles.NoBoxCoordXmin = 0; %reset the NO ZONE box coordinates
handles.NoBoxCoordXmax = 0;
handles.NoBoxCoordYmin = 0;
handles.NoBoxCoordYmax = 0;
for i = 1:handles.axis_number-1
   cla(handles.drawn_axis(i))
end
set(handles.axes1, ...
   'Visible', 'off', ...
   'Units', 'pixels');
handles.boxnumber = 1;
set(handles.edit6,'string','Double click on the tip of the probe, or single click and hit enter');
[handles.zprobex,handles.zprobey,p]=impixel
text(handles.zprobex-3,handles.zprobey-1,'x','color','y'); % CCF mark the probe zero spot

guidata(hObject,handles);
guidata(gcbo,handles);
set(handles.edit6,'string','');
% --- Executes on button press in pushbutton4.-------------------------
function pushbutton4_Callback(hObject, eventdata, handles)
%subroutine to send all units home
s1 = handles.current_port;

home = 0; %this puts the stages at fully contracted position
bytemovestagex = micro2byte(home);
bytemovestagey = micro2byte(home);
bytemoveZ = micro2byte(home);
fwriteQx(s1,[1 20 bytemovestagex(3) bytemovestagex(4) bytemovestagex(5)
bytemovestagex(6)],handles);
fwriteQy(s1,[2 20 bytemovestagey(3) bytemovestagey(4) bytemovestagey(5)
bytemovestagey(6)],handles);
fwriteQz(s1,[3 20 bytemoveZ(3) bytemoveZ(4) bytemoveZ(5) bytemoveZ(6)],handles);
while abs(str2num(get(handles.edit1,'string')))>0|abs(str2num(get(handles.edit2,'string')))>0
```

```
end

set(handles.edit6,'string','Units are home');
% --- Executes on button press in pushbutton3.--------------------------
function pushbutton3_Callback(hObject, eventdata, handles)
%subroutine to close the com port
s1=handles.current_port;
set(handles.edit6,'string','wait to close program');
%%%%%%%%%%%%send stages home before shut down%%%%%%%%%%%%
pause(0.01);home = 0; %this puts the stages at home
bytemovestagex = micro2byte(home);
bytemovestagey = micro2byte(home);
bytemoveZ = micro2byte(home);
fwriteQx(s1,[1 20 bytemovestagex(3) bytemovestagex(4) bytemovestagex(5)
bytemovestagex(6)],handles);
fwriteQy(s1,[2 20 bytemovestagey(3) bytemovestagey(4) bytemovestagey(5)
bytemovestagey(6)],handles);
fwriteQz(s1,[3 20 bytemoveZ(3) bytemoveZ(4) bytemoveZ(5) bytemoveZ(6)],handles);
while
abs(str2num(get(handles.edit1,'string')))>0|abs(str2num(get(handles.edit2,'string')))>0|abs(str
2num(get(handles.edit5,'string')))>0

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
M = [0 0 0];
M(1)= str2num(get(handles.edit1,'string'));
M(2)= str2num(get(handles.edit2,'string'));
M(3)= str2num(get(handles.edit5,'string'));

csvwrite('ZaberGuilastPosition.txt',M);
set(handles.edit6,'string','safe to close program');
guidata(hObject,handles)
guidata(gcbo,handles);
fclose(s1);
delete(s1);
clear s1;


% --- Executes on button press in pushbutton2.--------------------------
function pushbutton2_Callback(hObject, eventdata, handles)
%subroutine to open the com Port

s1 = serial('com2', 'baudrate',9600);
s1.BytesAvailableFcnMode = 'byte';
s1.BytesAvailableFcnCount = 6;
s1.BytesAvailableFcn = {@datareturned,handles};
```

```
fopen(s1);
set(handles.edit6,'string','wait for communication to be established');
fwriteQx(s1,[0 2 0 0 0 0],handles);
set(handles.edit6,'string','communication was established');
pause(2)
M = csvread('ZaberguilastPosition.txt');
handles.lastXpos = M(1);
handles.lastYpos = M(2);
handles.lastZpos = M(3);
bytemovestagex = micro2byte(M(1));
bytemovestagey = micro2byte(M(2));
bytemoveZ = micro2byte(M(3));
fwriteQx(s1,[1 45 bytemovestagex(3) bytemovestagex(4) bytemovestagex(5)
bytemovestagex(6)],handles);
fwriteQy(s1,[2 45 bytemovestagey(3) bytemovestagey(4) bytemovestagey(5)
bytemovestagey(6)],handles);
fwriteQz(s1,[3 45 bytemoveZ(3) bytemoveZ(4) bytemoveZ(5) bytemoveZ(6)],handles);
set(handles.edit6,'string','Stage is ready');
handles.CONVERT = 17;
handles.current_port = s1;
%guidata(hObject,handles);
guidata(gcbo,handles);

%--------------------------------------------------------------------
%function that will scan the bytesavailable and trigger a callback when the
%positioner sends back data of any kind
function datareturned(s1, bytesavailable, handles)
returneddata = fread(s1,6);
%handles.returneddata = returneddata;
if returneddata(2)==21|20|45;
   currentposition = byte2micro(returneddata);
   if returneddata(1) ==1;
      set(handles.edit1,'string',num2str(currentposition));
   elseif returneddata(1)==2;
      set(handles.edit2,'string',num2str(currentposition));
   else returneddata(1)==3;
      set(handles.edit5,'string',num2str(currentposition));
   end
elseif returneddata(2)==1;
   currentposition = byte2micro(returneddata)
   if returneddata(1) ==1;
      set(handles.edit1,'string',num2str(currentposition))
   elseif returneddata(1)==2;
      set(handles.edit2,'string',num2str(currentposition))
   elseif returneddata(1)==3;
      set(handles.edit5,'string',num2str(currentposition))
```

```
   end
elseif returneddata(2)==60;
   currentposition = byte2micro(returneddata);
   if returneddata(1) ==1;
      handles.lastXpos = currentposition;
   elseif returneddata(1)==2;
      handles.lastYpos = currentposition;
   else returneddata(1)==3;
      handles.lastZpos = currentposition;
   end
   guidata(gcbo,handles);
end

% --------------------------------------------------------------------
%converts data given in a microstep format to its byte equivalent
function [movecommand]=micro2byte(position);
movecommand = [0 0 0 0 0 0];
position= round(position *64 /6.35);
if position < 0
   position = 4294967296 + position;
end
movecommand(6) = floor(position/(256*256*256));
position = position- 256*256*256*movecommand(6);
movecommand(5) = floor(position/(256*256));
position = position - 256*256*movecommand(5);
movecommand(4) = floor(position /256);
position = position - 256*movecommand(4);
movecommand(3) = floor(position);
movecommand;


%converts data in the byte format into its microstep value----------
function [microposition]= byte2micro(bytevalue);
   if bytevalue(6) >127 %then value negative
      microposition = -
4294967296+(256*256*256*bytevalue(6))+(256*256*bytevalue(5))+(256*bytevalue(4))+by
tevalue(3);
   else
      microposition =
(256*256*256*bytevalue(6))+(256*256*bytevalue(5))+(256*bytevalue(4))+bytevalue(3);
   end
   microposition = microposition*6.35/64;

% --- Executes on button press in pushbutton5.-------------------------
function pushbutton5_Callback(hObject, eventdata, handles)
%subroutine to controll the relative movement of the unit
```

```
% Stub for Callback of the uicontrol handles.pushbutton5.
desired_position = str2num(get(handles.edit3,'string'));
%%%%convert microposition to byte string%%%%%%%%%%
   position = desired_position;
   movecommand = micro2byte(position);
   Unitchoice = get(handles.popupmenu2, 'value');
   movecommand(1) = Unitchoice; %define which motor is moving
   movecommand(2) = 21; %define if it is relative or absolute move
s1 = handles.current_port;
fwrite(s1,movecommand);
guidata(gcbo,handles);

% --------------------------------------------------------------------
function edit1_Callback(hObject, eventdata, handles)
function edit2_Callback(hObject, eventdata, handles)
function edit3_Callback(hObject, eventdata, handles)
function popupmenu2_Callback(hObject, eventdata, handles)
function edit4_Callback(hObject, eventdata, handles)
function edit5_Callback(hObject, eventdata, handles)

% --- boundary box button- set the boundary around whole slice------------
function pushbutton6_Callback(hObject, eventdata, handles)
set(handles.edit6,'string','Single left click at top left point of desired entire slice area, move to
bottom right and single left click');
clear xpoints;%reset parameters so most recent boundary will be used
clear ypoints;
[xpoints,ypoints] = rubberband('-anim','xor','-return','vectors')
ypointstransformed = 480-ypoints;
handles.xstart = xpoints(1);
handles.ystart = 480-ypoints(1);
handles.xstop = xpoints(2);
handles.ystop = 480-ypoints(2);
handles.drawn_axis(handles.axis_number)= gca;
handles.axis_number = handles.axis_number +1;
guidata(hObject,handles);
guidata(gcbo,handles);
set(handles.edit6,'string','');
if handles.zprobex >= handles.xstart  %CCF
   msgbox('Notice: zprobex is >= to xstart for the box.  Pick the bounding box again');
end
if handles.zprobey >= (480-handles.ystart)  %CCF
   msgbox('Notice: zprobey is >= to ystart for the box.  Pick the bounding box again');
end

% ---box out the area where you do not want to record--------------------
function pushbutton8_Callback(hObject, eventdata, handles)
```

```
set(handles.edit6,'string','Single left click at top left point of "no recording area", move to
bottom right and single left click');
[xpoints,ypoints] = rubberband('-anim','xor','-return','vectors', '-color', 'g');
box_number = handles.boxnumber;
handles.NoBoxCoordXmin(box_number) = xpoints(1);
handles.NoBoxCoordXmax(box_number) = xpoints(2);%store the boxes into an array
handles.NoBoxCoordYmin(box_number) = ypoints(2);
handles.NoBoxCoordYmax(box_number) = ypoints(1);
handles.drawn_axis(handles.axis_number)= gca;
handles.axis_number = handles.axis_number +1;

guidata(hObject,handles);
guidata(gcbo,handles);
handles.boxnumber = handles.boxnumber + 1;
guidata(hObject,handles);
guidata(gcbo,handles);
set(handles.edit6,'string','');
%set condition so that electrode will not go down into bath


% --- move function to move in a grid like manner-----------------------
%the left,right, top and bottom boundaries are set by the boundary box
%the routine will move the electrode to the top left corner and will mover
%to the right till it hits it boundary then it will drop down and begin
%moving left till it his the left boundary till and continue back and forth
%and dropping till it passes the bottom boundary
function pushbutton7_Callback(hObject, eventdata, handles)
TissueBathPath = 'E:\ChrisDillon Thesis Data\ScanResults\'; %path that communicates to
VB code
currentpointx = handles.xstart;
currentpointy = handles.ystart;
s1 = handles.current_port;
set(handles.edit6,'string','');
%move the stage so that the z-probe is in position of top left corner of
%bouding box
handles.Pixmovestagex = abs(handles.zprobex - currentpointx);%define how for in pixels the
stage must move in xdir
handles.Pixmovestagey = abs(handles.zprobey - currentpointy); %define how far in pixels
the stage must move proper zprobe position
%*****************************************************************

Micromovestagex = (handles.Pixmovestagex*handles.CONVERT);
Micromovestagey = (handles.Pixmovestagey*handles.CONVERT);
%calculate the step size here
   boxsize = (handles.xstop - handles.xstart)*handles.CONVERT;
   handles.divisions = 9; %change this to change the step size
```

```
   Micronstepsize = boxsize/handles.divisions;
   Pixelstepsize = (Micronstepsize/handles.CONVERT);
%*****************************************************************

Bytemovestagex = micro2byte(Micromovestagex);
Bytemovestagey = micro2byte(Micromovestagey);
Bytemovestagez = micro2byte(200);
fwriteQx(s1,[1 21 Bytemovestagex(3) Bytemovestagex(4) Bytemovestagex(5)
Bytemovestagex(6)],handles);
fwriteQy(s1,[2 21 Bytemovestagey(3) Bytemovestagey(4) Bytemovestagey(5)
Bytemovestagey(6)],handles);
fwriteQz(s1,[3 21 Bytemovestagez(3) Bytemovestagez(4) Bytemovestagez(5)
Bytemovestagez(6)],handles);
text((currentpointx-3),(480-currentpointy),'x','color', 'r');
text((currentpointx-5),(480-currentpointy+7),'1','color','r','fontsize',8);%number the point
set(handles.edit6,'string','Stage is ready');
while
abs(str2num(get(handles.edit1,'string')))==0|abs(str2num(get(handles.edit2,'string')))==0
   %waits till units move
end
yy= [str2num(get(handles.edit1,'string')) str2num(get(handles.edit2,'string'))];
dlmwrite(strcat(TissueBathPath, 'ready.txt'),yy,'precision','%.2f','newline','pc')
pause(.01)
while exist(strcat(TissueBathPath, 'start.txt'))==0
   %do nothing until the start command is sent
end
delete(strcat(TissueBathPath, 'start.txt'));
xdir = 'positive';%set the routing moving right to begin

%wait for VB to make first recording and send first move
while exist(strcat(TissueBathPath, 'move.txt'))==0
   %hold the electrodes in place
end
%retract the z probe
Bytemovestagez = micro2byte(-200);
fwriteQz(s1,[3 21 Bytemovestagez(3) Bytemovestagez(4) Bytemovestagez(5)
Bytemovestagez(6)],handles);
pause(2) %give time for electrode to be retracted

%loop to go through till whole grid is covered
pause(0.01)
whileloopcount = 1;
while exist(strcat(TissueBathPath, 'finished.txt'))==0
   if exist(strcat(TissueBathPath, 'move.txt')) == 2 & (currentpointy < handles.ystop);
      delete((strcat(TissueBathPath, 'move.txt')));
```

```
    if currentpointx <= handles.xstop - Pixelstepsize+2 & xdir=='positive';%moving to right
but not at edge
        a = (str2num(get(handles.edit1,'string')));
        movegridwise('positive','none',s1,handles);
        while abs((str2num(get(handles.edit1,'string'))))==a

        end
        currentpointx = currentpointx + Pixelstepsize;
        text((currentpointx-3),(480-currentpointy),'x','color', 'r');
        pause(.01)
        if InNoZone(currentpointx,480-currentpointy,handles) == true
           %do not move Z-controler down to bath
           dlmwrite(strcat(TissueBathPath, 'ready.txt'),120);%tells VB that electrode is in No
Zone
           while exist(strcat(TissueBathPath, 'move.txt'))==0
              %wait till VB is ready for next position
           end
           pause(.1)
           if currentpointx > handles.xstop-Pixelstepsize+2 &xdir=='positive'& currentpointy
+ Pixelstepsize >handles.ystop;%at last point of recording
              set(handles.edit6,'string','Recording is done');
              csvwrite(strcat(TissueBathPath, 'finished.txt'),0);
              csvwrite(strcat(TissueBathPath, 'ready.txt'),101);%send an additional ready for
VB to finish
           end
        else
           downstroke = micro2byte(200);
           fwriteQz(s1,[3 21 downstroke(3) downstroke(4) downstroke(5)
downstroke(6)],handles);
           pause(2)%give time for electrode to desend into bath
           yy = [str2num(get(handles.edit1,'string')) str2num(get(handles.edit2,'string'))];
           dlmwrite(strcat(TissueBathPath, 'ready.txt'),yy,'precision','%.2f','newline','pc');
           whileloopcount = whileloopcount +1;%count used to label the points
           text((currentpointx-5),(480-
currentpointy+7),int2str(whileloopcount),'color','r','FontSize',8);%number the point
           while exist(strcat(TissueBathPath, 'move.txt'))==0
              %VB is recording data from the tissue
           end
           %check to see if recording is done
           if currentpointx > handles.xstop-Pixelstepsize+2 &xdir=='positive'& currentpointy
+ Pixelstepsize >handles.ystop;%at last point of recording
              set(handles.edit6,'string','Recording is done');
              csvwrite(strcat(TissueBathPath, 'finished.txt'),0);
              csvwrite(strcat(TissueBathPath, 'ready.txt'),101);%send an additional ready for
VB to finish
           end
```

```
            upstroke = micro2byte(-200);
            fwriteQz(s1,[3 21 upstroke(3) upstroke(4) upstroke(5) upstroke(6)],handles);
            pause(2);%so electrode is up before next move
        end

    elseif currentpointx > handles.xstop-Pixelstepsize+2 &xdir=='positive';%at right edge
        b = (str2num(get(handles.edit2,'string')));
        movegridwise('none', 'positive',s1,handles);
        while abs((str2num(get(handles.edit2,'string'))))==b;

        end
        currentpointy = currentpointy + Pixelstepsize;
        text((currentpointx-3),(480-currentpointy),'x','color', 'r');
        pause(.01)
        if InNoZone(currentpointx,480-currentpointy,handles) == true
            %do not move Z-controler down to bath
            dlmwrite(strcat(TissueBathPath, 'ready.txt'),120);%tells VB that electrode is in No
Zone
            while exist(strcat(TissueBathPath, 'move.txt'))==0
                %wait till VB is ready for next position
            end
            pause(.1)
        else
            downstroke = micro2byte(200);
            fwriteQz(s1,[3 21 downstroke(3) downstroke(4) downstroke(5)
downstroke(6)],handles);
            pause(2)
            yy = [str2num(get(handles.edit1,'string')) str2num(get(handles.edit2,'string'))];
            dlmwrite(strcat(TissueBathPath, 'ready.txt'),yy,'precision','%.2f','newline','pc')
            whileloopcount = whileloopcount +1;%count used to label the points
            text((currentpointx-5),(480-
currentpointy+7),int2str(whileloopcount),'color','r','FontSize',8);%number the point
            while exist(strcat(TissueBathPath, 'move.txt'))==0
                %VB is recording data from the tissue
            end
            upstroke = micro2byte(-200);
            fwriteQz(s1,[3 21 upstroke(3) upstroke(4) upstroke(5) upstroke(6)],handles);
            pause(2);%so electrode is up before next move
        end
        xdir = 'negative';

    elseif currentpointx >=handles.xstart+Pixelstepsize-2 & xdir =='negative';%moving left
but not at edge
        c = (str2num(get(handles.edit1,'string')));
        movegridwise('negative','none',s1,handles);
        while abs((str2num(get(handles.edit1,'string'))))==c;
```

```
        end
        currentpointx = currentpointx-Pixelstepsize;
        text((currentpointx-3),(480-currentpointy),'x','color', 'r');
        pause(.01)
        if InNoZone(currentpointx,480 - currentpointy,handles) == true
          %do not move Z-controler down to bath
          dlmwrite(strcat(TissueBathPath, 'ready.txt'),120);%tells VB that electrode is in No
Zone
          while exist(strcat(TissueBathPath, 'move.txt'))==0
            %wait till VB is ready for next position
          end
          pause(.1)
          if currentpointx > handles.xstop-Pixelstepsize+2 &xdir=='positive'& currentpointy
+ Pixelstepsize >handles.ystop;%at last point of recording
            set(handles.edit6,'string','Recording is done');
            csvwrite(strcat(TissueBathPath, 'finished.txt'),0);
            csvwrite(strcat(TissueBathPath, 'ready.txt'),101);%send an additional ready for
VB to finish
          end
        else
          downstroke = micro2byte(200);
          fwriteQz(s1,[3 21 downstroke(3) downstroke(4) downstroke(5)
downstroke(6)],handles);
          pause(2)
          yy = [str2num(get(handles.edit1,'string')) str2num(get(handles.edit2,'string'))];
          dlmwrite(strcat(TissueBathPath, 'ready.txt'),yy,'precision','%.2f','newline','pc')
          whileloopcount = whileloopcount +1;%count used to label the points
          text((currentpointx-5),(480-
currentpointy+7),int2str(whileloopcount),'color','r','FontSize',8);%number the point
          while exist(strcat(TissueBathPath, 'move.txt'))==0
            %VB is recording data from the tissue
          end
          if currentpointx < handles.xstart +Pixelstepsize-2 & xdir =='negative'
&currentpointy +Pixelstepsize > handles.ystop;
            csvwrite(strcat(TissueBathPath, 'finished.txt'),0);
            set(handles.edit6,'string','Recording is done');
            csvwrite(strcat(TissueBathPath, 'ready.txt'),101);%send additional ready so that
VB can finish
          end
          upstroke = micro2byte(-200);
          fwriteQz(s1,[3 21 upstroke(3) upstroke(4) upstroke(5) upstroke(6)],handles);
          pause(2);%so electrode is up before next move
        end

      elseif currentpointx < handles.xstart +Pixelstepsize & xdir =='negative' ;%at left edge
```

```
        d = (str2num(get(handles.edit2,'string')));
        movegridwise('none','positive',s1,handles);
        while abs((str2num(get(handles.edit2,'string'))))==d

        end
        currentpointy = currentpointy + Pixelstepsize;
        text((currentpointx-3),(480-currentpointy),'x','color', 'r');
        pause(.01)
        if InNoZone(currentpointx,currentpointy,handles) == true
           %do not move Z-controler down to bath
           dlmwrite(strcat(TissueBathPath, 'ready.txt'),120);%tells VB that electrode is in No
Zone
           while exist(strcat(TissueBathPath, 'move.txt'))==0
              %wait till VB is ready for next position
           end
        pause(.1)
        else
           downstroke = micro2byte(200);
           fwriteQz(s1,[3 21 downstroke(3) downstroke(4) downstroke(5)
downstroke(6)],handles);
           pause(2)
           yy = [str2num(get(handles.edit1,'string')) str2num(get(handles.edit2,'string'))];
           dlmwrite(strcat(TissueBathPath, 'ready.txt'),yy,'precision','%.2f','newline','pc')
           whileloopcount = whileloopcount +1;%count used to label the points
           text((currentpointx-5),(480-
currentpointy+7),int2str(whileloopcount),'color','r','FontSize',8);%number the point
           while exist(strcat(TissueBathPath, 'move.txt'))==0
              %VB is recording data from the tissue
           end
           upstroke = micro2byte(-200);
           fwriteQz(s1,[3 21 upstroke(3) upstroke(4) upstroke(5) upstroke(6)],handles);
           pause(2); %so electrode is up before movement to next spot
        end
        xdir = 'positive';
      end
   elseif currentpointy > handles.ystop
        csvwrite(strcat(TissueBathPath, 'finished.txt'),0);
   end
end

%function that will move in a grid manner dempending on input
function movegridwise(xdir, ydir,s1,handles)
   %calculate the step size here
   boxsize = ((handles.xstop - handles.xstart)*handles.CONVERT);

   ustepsize = boxsize/handles.divisions;
```

```
   if (strcmp(xdir,'positive'))==1;
      newcoord = micro2byte(ustepsize);
      movecommand = [1 21 newcoord(3) newcoord(4) newcoord(5) newcoord(6)];
      fwriteQx(s1,movecommand,handles);
   elseif (strcmp(xdir,'negative')) ==1;
      newcoord = micro2byte(-ustepsize);
      movecommand = [1 21 newcoord(3) newcoord(4) newcoord(5) newcoord(6)];
      fwriteQy(s1,movecommand,handles);
   elseif (strcmp(ydir,'positive'))==1;
      newcoord = micro2byte(ustepsize);
      movecommand = [2 21 newcoord(3) newcoord(4) newcoord(5) newcoord(6)];
      fwriteQx(s1,movecommand,handles);
   elseif (strcmp(ydir,'negative'))==1;
      newcoord = micro2byte(-ustepsize);
      movecommand = [2 21 newcoord(3) newcoord(4) newcoord(5) newcoord(6)];
      fwriteQy(s1,movecommand,handles);
   end

 %function to determine if the current point is within a no zone box
 function [status]=InNoZone (currentx,currenty,handles);
    for n=1:handles.boxnumber-1;
       status = 0;
       if (currentx > handles.NoBoxCoordXmin(n)) &
(currentx<handles.NoBoxCoordXmax(n))
&(currenty>handles.NoBoxCoordYmin(n))&(currenty<handles.NoBoxCoordYmax(n));
          status = 1;
       end
    end
%%%%%%%%try/catch loops the write serial data to
controls%%%%%%%%%%%%%%%%%%%%%%%5
function fwriteQx(s1,movecommand,handles)
   try,
      c=(str2num(get(handles.edit2,'string')));
      fwrite(s1,movecommand);
      catch,
         try,
            pause(0.5)
            cat = 0
            if abs(str2num(get(handles.edit2,'string'))) ==c
               fwrite(s1,movecommand);
            end
         catch,
            try,
               pause(0.5)
               cat = 1
               if abs(str2num(get(handles.edit2,'string'))) ==c
```

```
            fwrite(s1,movecommand);
        end
    catch
        try,
            pause(0.5)
            cat = 2
            if abs(str2num(get(handles.edit2,'string'))) ==c
                fwrite(s1,movecommand);
            end
        catch
            try,
                pause(0.5)
                cat = 3
                if abs(str2num(get(handles.edit2,'string'))) ==c
                    fwrite(s1,movecommand);
                end
            catch
                try,
                    pause(0.5)
                    cat = 4
                    if abs(str2num(get(handles.edit2,'string'))) ==c
                        fwrite(s1,movecommand);
                    end
                catch
                    pause(0.5)
                    cat = 5
                    if abs(str2num(get(handles.edit2,'string'))) ==c
                        fwrite(s1,movecommand);
                    end
                end
            end
        end
    end
end
function fwriteQy(s1,movecommand,handles)
    try,
        d=(str2num(get(handles.edit3,'string')));
        fwrite(s1,movecommand);
        catch,
            try,
                pause(0.5)
                cat = 10
                if abs(str2num(get(handles.edit3,'string'))) ==d
                    fwrite(s1,movecommand);
                end
```

```
        catch,
           try,
              pause(0.5)
              cat = 11
              if abs(str2num(get(handles.edit3,'string'))) ==d
                 fwrite(s1,movecommand);
              end
           catch
              try,
                 pause(0.5)
                 cat = 12
                 if abs(str2num(get(handles.edit3,'string'))) ==d
                    fwrite(s1,movecommand);
                 end
              catch
                 try,
                    pause(0.5)
                    cat = 13
                    if abs(str2num(get(handles.edit3,'string'))) ==d
                       fwrite(s1,movecommand);
                    end
                 catch
                    try,
                       pause(0.5)
                       cat = 14
                       if abs(str2num(get(handles.edit3,'string'))) ==d
                          fwrite(s1,movecommand);
                       end
                    catch
                       pause(0.5)
                       cat = 15
                       if abs(str2num(get(handles.edit3,'string'))) ==d
                          fwrite(s1,movecommand);
                       end
                    end
                 end
              end
           end
        end
  end
function fwriteQz(s1,movecommand,handles)
  try,
     e=(str2num(get(handles.edit5,'string')));
     fwrite(s1,movecommand);
     catch,
        try,
```

```
            pause(0.5)
            cat = 20
            if abs(str2num(get(handles.edit5,'string'))) ==e
               fwrite(s1,movecommand);
            end
         catch,
            try,
               pause(0.5)
               cat = 21
               if abs(str2num(get(handles.edit5,'string'))) ==e
                  fwrite(s1,movecommand);
               end
            catch
               try,
                  pause(0.5)
                  cat = 22
                  if abs(str2num(get(handles.edit5,'string'))) ==e
                     fwrite(s1,movecommand);
                  end
               catch
                  try,
                     pause(0.5)
                     cat = 23
                     if abs(str2num(get(handles.edit5,'string'))) ==e
                        fwrite(s1,movecommand);
                     end
                  catch
                     try,
                        pause(0.5)
                        cat = 24
                        if abs(str2num(get(handles.edit5,'string'))) ==e
                           fwrite(s1,movecommand);
                        end
                     catch
                        pause(0.5)
                        cat = 25
                        if abs(str2num(get(handles.edit5,'string'))) ==e
                           fwrite(s1,movecommand);
                        end
                     end
                  end
               end
            end
         end
   end
%----------------Create function build by matlab gui guide--------------
```

```
% --- Executes during object creation, after setting all properties.----
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
% --- Executes during object creation, after setting all properties.-------
function edit4_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
% --- Executes during object creation, after setting all properties.------
function popupmenu2_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
% --- Executes during object creation, after setting all properties.----
function edit2_CreateFcn(hObject, eventdata, handles)
```

```
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6 as a double


% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% --- Executes on button press in pushbutton10.
function pushbutton10_Callback(hObject, eventdata, handles)
s1=handles.current_port;
set(hObject, 'Units', 'pixels');%use pixels values and not corrdinate values from the grid
imagefile = get(handles.edit4,'string');%retreive picture file of the implant
handles.picture = imread(char(imagefile)); % Read the image file banner.jpg
info = imfinfo(char(imagefile)); % Determine the size of the image file
position = get(handles.axes1, 'Position');
```

```
axes(handles.axes1);
image(handles.picture);%display the picture on the axes in the gui
set(handles.axes1, ...
    'Visible', 'off', ...
    'Units', 'pixels');
handles.boxnumber = 1;
set(handles.edit6,'string','Double click where you want the probe to go');
text(handles.zprobex-3,handles.zprobey-1,'x','color','y'); % CCF mark the probe zero spot
[devicepointx,devicepointy,p]=impixel%displays a cursor on the screen to be used to select
position of probe
currentxpoint = abs(str2num(get(handles.edit1,'string')));%get the current position of the
micro motors
currentypoint = abs(str2num(get(handles.edit2,'string')));%get the current position of the
micro motors
PixelMoveStageX = ((devicepointx-handles.zprobex)*handles.CONVERT) -
(currentxpoint)%subtract b/c the zero point on the micron scale is at the probe
PixelMoveStageY = ((devicepointy-handles.zprobey)*handles.CONVERT) -
(currentypoint)%subtract  b/c the zero point on the micron scale is at the probe
ByteMoveStageX = micro2byte(PixelMoveStageX);%convert the micron measure to bytes to
be sent to the stage
ByteMoveStageY = micro2byte(PixelMoveStageY);%convert the micron measure to bytes to
be sent to the stage
text(devicepointx-3,( devicepointy)-1,'x','color','r');
fwriteQx(s1,[1 21 ByteMoveStageX(3) ByteMoveStageX(4) ByteMoveStageX(5)
ByteMoveStageX(6)],handles);
fwriteQy(s1,[2 21 ByteMoveStageY(3) ByteMoveStageY(4) ByteMoveStageY(5)
ByteMoveStageY(6)],handles);
while
abs(str2num(get(handles.edit2,'string')))==currentxpoint|abs(str2num(get(handles.edit2,'strin
g')))==currentypoint

end
set(handles.edit6,'string','correct alignment if nessecary');
guidata(hObject,handles);
guidata(gcbo,handles);



% % --- Button used to calibrate the pixel to micron conversion
% function pushbutton11_Callback(hObject, eventdata, handles)
% % note you must have the calibrate image showing on the gui screen before
% % clicking on this button
% set(handles.edit6,'string','Single left click at top of the top line, then click at the top of the
bottom line');
% [xpoints,ypoints] = rubberband('-anim','xor','-return','vectors');
% box_number = handles.boxnumber;
```

```
% handles.NoBoxCoordXmin(box_number) = xpoints(1);
% handles.NoBoxCoordXmax(box_number) = xpoints(2);%store the boxes into an array
% handles.NoBoxCoordYmin(box_number) = ypoints(2);
% handles.NoBoxCoordYmax(box_number) = ypoints(1);
% handles.convert = abs(7500/(ypoints(2)-ypoints(1)))  %18.52 microns/pixel is found by
5000 microns for nut size/pixels found using IMPIXEL
% %conversion between pixels and microns
% csvwrite('C:\VBMatlab\calibrate.txt',handles.convert);
% handles.drawn_axis(handles.axis_number)= gca;
% handles.axis_number = handles.axis_number +1;
%
% guidata(hObject,handles);
% guidata(gcbo,handles);
% handles.boxnumber = handles.boxnumber + 1;
% guidata(hObject,handles);
% guidata(gcbo,handles);
% set(handles.edit6,'string','');
```

Appendix B: Visual Basic code for recording and storing data

The software was responsible for collecting the data from the lock-in amplifier and writing the data to text files that could be analyzed later. The software allowed for two methods of recording data. The “Auto Collect” option worked in conjunction with the MatLab code in Appendix A to collect data from 100 grid points. The “Record over Freq Range” option records over the same frequency range as the “Auto Collect” option, however it only take the reading at one point and doesn’t interact with the MatLab code. Thus the recording electrode must be placed by the user with the MatLab code before conducting a “Record over Freq Range.”

```
Private Sub AutoColl_Click()
'this routine collects data over the frequency range at each of 100 points
'in a square grid controlled by Chris Dillon's "Zaber gui" written in matlab

Dim FinishedFlg As Boolean
   Dim ReadyFlg As Boolean
   Dim TissueBathPath As String
   Dim pathstrfinished As String
   Dim pathstrready As String
   Dim pathstrstart As String
   Dim pathstrmove As String
   Dim file_name1 As String
   Dim DataFile As String
   Dim startfile As TextStream
   Dim readyfile As TextStream
   Dim movefile As TextStream
   Dim msgstr As String
   Dim strCoordinates As String
   Dim strX As String
   Dim strY As String
   Dim strZ As String
   Dim commapos1 As Long
   Dim commapos2 As Long
   Dim Magnitude(35) As Double
   Dim Phase(35) As Double
   Dim MagCompare As String
   Dim PhaseCompare As String
   Dim RealPart(35) As Double
   Dim ImagPart(35) As Double
   Dim SendFreq As String
   Dim FreqRange(35) As Double
   Dim strMagnitude As String
   Dim strPhase As String
   Dim strRealPart As String
```

```
Dim strImagPart As String
Dim PointCount As Integer
Dim response As Integer
Dim phaseA As Double
Dim MeterPhase(35) As Double
Dim expandadjust As Double

TissueBathPath = "E:\ChrisDillon Thesis Data\ScanResults"
pathstrready = TissueBathPath + "\ready.txt"
pathstrmove = TissueBathPath + "\move.txt"
pathstrstart = TissueBathPath + "\start.txt"
pathstrfinished = TissueBathPath + "\finished.txt"

'check for save folder creation
If Not Dir(TissueBathPath + "\" + Text2.Text + "\") = "" Then
   'no need to do anything
Else
    MkDir (TissueBathPath + "\" + Text2.Text + "\")
End If

FinishedFlg = False
ReadyFlg = False

'Clear Finished flag
If Not Dir(pathstrfinished) = "" Then
   Kill (pathstrfinished)
End If

'Clear ready flag
If Not Dir(pathstrready) = "" Then
   Kill (pathstrready)
End If

'Clear move flag
If Not Dir(pathstrmove) = "" Then
   Kill (pathstrmove)
End If

'clear start flag
If Not Dir(pathstrstart) = "" Then
   Kill (pathstrstart)
End If

'read the file with the frequency range values
file_name1 = TissueBathPath + "\FreqRange.txt"
Open file_name1 For Input As #1
```

```
    For k = 0 To (23 - 1) Step 1
        Line Input #1, Value
        FreqRange(k) = CDbl(Value)
    Next k
    Close 1



'*******setup communication between the function generator***********
    Dim mgr As AgilentRMLib.SRMCls
    Dim instrument As VisaComLib.FormattedIO488

    Set mgr = New AgilentRMLib.SRMCls
    Set instrument = New VisaComLib.FormattedIO488
    Set instrument.IO = mgr.Open("ASRL1::INSTR")

    Dim sfc As VisaComLib.ISerial
    Set sfc = instrument.IO
    sfc.BaudRate = 9600
    sfc.FlowControl = ASRL_FLOW_DTR_DSR
    instrument.IO.TerminationCharacter = 10
    instrument.IO.TerminationCharacterEnabled = True
'**********done setting up communication*******************************

    MsgBox "ready to send initial start. Select OK when ready"

    ' Send initial start
    Open pathstrstart For Output As #55
    Close #55

    KeyDownFlg = False
    strCoordinates = ""
    PointCount = 0

    While Not FinishedFlg
       msg = "type the phase setting" + Chr(13) + "0 90 180 or 270" 'display phase selection
       respond = InputBox(msg)   'prompt the user for the phase setting
       calcphaseadjust respond, ztop, phaseA

       'check for a key press
       If KeyDownFlg Then
          KeyDownFlg = False
          Exit Sub
       End If

       'if probe is in NO Zone skip the recording
```

```
        Open pathstrready For Input As #55
        Seek #55, 1
        Do Until EOF(55)
           Line Input #55, strNOZONEcheck
        Loop
        Close #55
        ReadyFlg = True

        If strNOZONEcheck = 120 Then
           ScanStatus.Caption = "in NO ZONE"
           PointCount = PointCount + 1
           Pause (1)
           'finish housekeeping and send the move command
           While Not Dir(pathstrready) = ""
              Kill (pathstrready)
              ScanStatus.Caption = "ready kill"
           Wend
           Open pathstrmove For Output As #55
           Close #55
           ScanStatus.Caption = "move sent"
           'wait for next ready flag to be sent
           While Not ReadyFlg
              If Not Dir(pathstrready) = "" Then
                 ReadyFlg = True
                 ScanStatus.Caption = "found ready"
              End If
           Wend
           ReadyFlg = False
        End If

        If ReadyFlg Then
           'do housekeeping first
           ReadyFlg = False
           Open pathstrready For Input As #55
           ScanStatus.Caption = "ready open"
           Input #55, strX, strY, strZ
           Close #55

        'start a counter to be used to name data files
        PointCount = PointCount + 1

        'write the header for the file that will contain the data
        ScanStatus.Caption = PointCount
        On Error Resume Next
        DataFile = TissueBathPath + "\" + Text2.Text + "\" + Text1.Text + "_" +
ScanStatus.Caption + ".txt"
```

```
    Open DataFile For Append As #3
      Print #3, "X-coord"; Tab; strX; Tab; "Y-coord"; Tab; strY; Tab; "Z-coord"; Tab; strZ
'write the coordinates at the top of data file
      Print #3, "Freq"; Tab; "Magnitude"; Tab; "Phase"; Tab; "Real"; Tab; "Imag" 'writes
header for data file
    Close #3
     expandadjust = 1
    ' perform potential data collection now
    ScanStatus.Caption = "Scanning"                'tells the user the scan is running
    For k = 0 To 22 Step 1


      SendFreq = "frequency" + Str(FreqRange(k))    'set freq on function generator
      instrument.WriteString SendFreq
      Pause (0.5)
      iStatus% = AI_VRead(1, 0, 1, Magnitude(k))    'read magnitude of lock in
      iStatus% = AI_VRead(1, 1, 1, Phase(k))        'read phase from lock in

      If Magnitude(k) < 0.11 Then
        msg = "enter the expand adjust value x10 or x100"
        expandadjust = InputBox(msg)
      End If

      MagCompare = 0
      PhaseCompare = 0
      While Abs(Abs(MagCompare) - Abs(Magnitude(k))) > (0.001 / expandadjust) Or
Abs(Abs(PhaseCompare) - Abs(Phase(k))) > 0.05
        Pause (0.6)      'makes the program not read data unless value is stable for 0.5 sec.
        MagCompare = Magnitude(k)
        iStatus% = AI_VRead(1, 0, 1, Magnitude(k))    'read magnitude of lock in
        Magnitude(k) = (Magnitude(k) / expandadjust)
        lADC0.Caption = Format(Magnitude(k), "#.####")  'store magnitude into array
        PhaseCompare = lADC1.Caption
        If MeterPhase(k) < -0.9 Or MeterPhase(k) > 0.9 Then
          msg = "type the phase setting" + Chr(13) + "0 90 180 or 270" 'display phase
selection
          respond = InputBox(msg)   'prompt the user for the phase setting
          calcphaseadjust respond, ztop, phaseA
          iStatus% = AI_VRead(1, 1, 1, Phase(k))        'read phase from lock in
          MeterPhase(k) = Phase(k)
          Phase(k) = Phase(k) + phaseA
          lADC1.Caption = Format(Phase(k), "#.####")    'store phase into array
        Else
          iStatus% = AI_VRead(1, 1, 1, Phase(k))        'read phase from lock in
          MeterPhase(k) = Phase(k)
          Phase(k) = Phase(k) + phaseA
```

```
                lADC1.Caption = Format(Phase(k), "#.####")    'store phase into array
              End If
            Wend
'           iStatus% = AI_VRead(1, 0, 1, Magnitude(k))
'           lADC0.Caption = "ADC 0 Voltage =" + Format(dRefVoltage, "#.####")
'           iStatus% = AI_VRead(1, 1, 1, Phase(k))
'           lADC1.Caption = "ADC 1 Voltage =" + Format(dRefVoltage, "#.####")
            RealPart(k) = Magnitude(k) * Cos(Phase(k))
            ImagPart(k) = Magnitude(k) * Sin(Phase(k))
            strMagnitude = Format(Magnitude(k), "#.######")
            strPhase = Format(Phase(k), "#.######")
            strRealPart = Format(RealPart(k), "#.######")
            strImagPart = Format(ImagPart(k), "#.######")

        ScanStatus.Caption = PointCount
        Open DataFile For Append As #3
          Print #3, FreqRange(k); Tab; strMagnitude; Tab; strPhase; Tab; strRealPart; Tab;
strImagPart; Tab; respond 'write all data to a file
        Close #3
        Next k

        ScanStatus.Caption = "Single Point Scan is Done"
        'finish housekeeping and send the move command
        While Not Dir(pathstrready) = ""
          Kill (pathstrready)
          ScanStatus.Caption = "ready kill"
        Wend
        Open pathstrmove For Output As #55
        Close #55
          ScanStatus.Caption = "move sent"
        'check for Readyflg and process if set, else loop
        While Not ReadyFlg
          If Not Dir(pathstrready) = "" Then
            ReadyFlg = True
            ScanStatus.Caption = "found ready"
          End If
        Wend
        Open pathstrready For Input As #55
        Seek #55, 1
        Do Until EOF(55)
          Line Input #55, strNOZONEcheck
        Loop
        Close #55
      End If
      While Not ReadyFlg
        If Not Dir(pathstrready) = "" Then
```

```
            ReadyFlg = True
            ScanStatus.Caption = "found ready"
        End If
    Wend
    'step the frequncy back down to 30Hz
    SendFreq = "frequency" + Str(30)
    instrument.WriteString SendFreq
    Pause (0.5)
    msg = "Reset all setting on lock-in!!"
    expandadjust = MsgBox(msg, vbOKOnly)
    'check for FinishedFlg and exit if set (via while loop check)
    If Not Dir(pathstrfinished) = "" Then
        FinishedFlg = True
        ScanStatus.Caption = "Recording Done"
    End If
Wend

    MsgBox "finished data collection"
End Sub

Private Sub cmdReadADC0_Click()
    Dim dRefVoltage As Double
    iStatus% = AI_VRead(1, 0, 1, dRefVoltage)
    lADC0.Caption = "ADC 0 Voltage =" + Format(dRefVoltage, "#.####")
End Sub

Private Sub cmdReadADC1_Click()
    Dim dRefVoltage As Double
    iStatus% = AI_VRead(1, 1, 1, dRefVoltage)
    lADC1.Caption = "ADC 1 Voltage =" + Format(dRefVoltage, "#.####")
End Sub

Private Sub Command1_Click()
    Dim FreqRange(35) As Double
    Dim DVMRange(35) As Double
    Dim file_name1 As String
    Dim file_name2 As String
    Dim Magnitude(35) As Double
    Dim Phase(35) As Double
    Dim DataFile As String
    Dim folderpath As String
    Dim RealPart(35) As Double
    Dim ImagPart(35) As Double
    Dim SendFreq As String
    Dim strMagnitude As String
    Dim strPhase As String
```

```
    Dim strRealPart As String
    Dim strImagPart As String
    Dim PointCount As Integer
    Dim MagCompare As Double
    Dim PhaseCompare As Double
    Dim msg As String
    Dim respond As Double
    Dim phaseA As Double
    Dim MeterPhase(35) As Double
    Dim ztop As Integer
    Dim expandadjust As Double

    folderpath = "E:\ChrisDillon Thesis Data\SinglePointResults\"
    file_name1 = folderpath + "FreqRange.txt"
    'check for save folder creating
    If Not Dir(folderpath + Text2.Text + "\") = "" Then
       'no need to do anything
    Else
        MkDir (folderpath + Text2.Text + "\")
    End If

'*******setup communication between the function generator***********
    Dim mgr As AgilentRMLib.SRMCls
    Dim instrument As VisaComLib.FormattedIO488

    Set mgr = New AgilentRMLib.SRMCls
    Set instrument = New VisaComLib.FormattedIO488
    Set instrument.IO = mgr.Open("ASRL1::INSTR")

    Dim sfc As VisaComLib.ISerial
    Set sfc = instrument.IO
    sfc.BaudRate = 9600
    sfc.FlowControl = ASRL_FLOW_DTR_DSR
    instrument.IO.TerminationCharacter = 10
    instrument.IO.TerminationCharacterEnabled = True
'**********done setting up communication*******************************
    instrument.WriteString "frequency" + Str(30)
    msg = "type the phase setting" + Chr(13) + "0 90 180 or 270" 'display phase selection
    respond = InputBox(msg)   'prompt the user for the phase setting
    calcphaseadjust respond, ztop, phaseA

    Open file_name1 For Input As #1
    For k = 0 To (35 - 1) Step 1
          Line Input #1, Value
          FreqRange(k) = CDbl(Value)
    Next k
```

```
   Close 1
   DataFile = folderpath + Text2.Text + "\" + Text1.Text + ".txt"
   Open DataFile For Append As #3
      Print #3, "Freq"; Tab; "Magnitude"; Tab; "Phase"; Tab; "Real"; Tab; "Imag" 'writes
header for data file
   Close #3
   expandadjust = 1        'set the expand adjust variable to 1

   ScanStatus.Caption = "Scanning"                    'tells the user the scan is running
   For k = 0 To 34 Step 1
      SendFreq = "frequency" + Str(FreqRange(k))      'set freq on function generator
      instrument.WriteString SendFreq
      Pause (0.5)   'give time to reading to move from last point
      iStatus% = AI_VRead(1, 0, 1, Magnitude(k))      'read magnitude of lock in
      iStatus% = AI_VRead(1, 1, 1, Phase(k))          'read phase from lock in
      If Magnitude(k) < 0.11 Then
         msg = "enter the expand adjust value x10 or x100"
         expandadjust = InputBox(msg)
      End If
      MagCompare = 0
      PhaseCompare = 0
      While Abs(Abs(MagCompare) - Abs(Magnitude(k))) > (0.001 / expandadjust) Or
Abs(Abs(PhaseCompare) - Abs(Phase(k))) > 0.05
         Pause (0.6)       'makes the program not read data unless value is stable for 0.5 sec.
         MagCompare = Magnitude(k)
         iStatus% = AI_VRead(1, 0, 1, Magnitude(k))      'read magnitude of lock in
         Magnitude(k) = Magnitude(k) / expandadjust
         lADC0.Caption = Format(Magnitude(k), "#.####")  'store magnitude into array
         PhaseCompare = Phase(k)
         If MeterPhase(k) < -0.9 Or MeterPhase(k) > 0.9 Then
               msg = "type the phase setting" + Chr(13) + "0 90 180 or 270" 'display phase
selection
               respond = InputBox(msg)   'prompt the user for the phase setting
               calcphaseadjust respond, ztop, phaseA
               iStatus% = AI_VRead(1, 1, 1, Phase(k))          'read phase from lock in
               MeterPhase(k) = Phase(k)
               Phase(k) = Phase(k) + phaseA
               lADC1.Caption = Format(Phase(k), "#.####")    'store phase into array
         Else
            iStatus% = AI_VRead(1, 1, 1, Phase(k))          'read phase from lock in
            MeterPhase(k) = Phase(k)
            Phase(k) = Phase(k) + phaseA
            lADC1.Caption = Format(Phase(k), "#.####")    'store phase into array
         End If
      Wend
```

```
        RealPart(k) = Magnitude(k) * Cos(Phase(k))
        ImagPart(k) = Magnitude(k) * Sin(Phase(k))
        strMagnitude = Format(Magnitude(k), "#.######")
        strPhase = Format(Phase(k), "#.######")
        strRealPart = Format(RealPart(k), "#.######")
        strImagPart = Format(ImagPart(k), "#.######")

    Open DataFile For Append As #3
        Print #3, FreqRange(k); Tab; strMagnitude; Tab; strPhase; Tab; strRealPart; Tab;
strImagPart; Tab; respond 'write all data to a file
    Close #3


    Next k
    ScanStatus.Caption = "Scan is Done"



End Sub

Public Sub calcphaseadjust(response, zztop, phaseadjust As Double) 'subroutine to decide the
'the amount of ajustment to make to the reading depending on phase dial setting
    If response = 270 Then
        phaseadjust = -0.9
        zztop = 1
        ElseIf response = 90 Then
        phaseadjust = 0.9
        zztop = 2
        ElseIf response = 180 And zztop = 1 Then
        phaseadjust = -1.8
        ElseIf response = 180 And zztop = 2 Then
        phaseadjust = 1.8
        Else
        phaseadjust = 0
    End If



End Sub

Private Sub Command3_Click()
    Dim DesiredFreq As String
    DesiredFreq = txtAO0Value.Text
'*******setup communication between the function generator**********
    Dim mgr As AgilentRMLib.SRMCls
    Dim instrument As VisaComLib.FormattedIO488
```

```
    Set mgr = New AgilentRMLib.SRMCls
    Set instrument = New VisaComLib.FormattedIO488
    Set instrument.IO = mgr.Open("ASRL1::INSTR")

    Dim sfc As VisaComLib.ISerial
    Set sfc = instrument.IO
    sfc.BaudRate = 9600
    sfc.FlowControl = ASRL_FLOW_DTR_DSR
    instrument.IO.TerminationCharacter = 10
    instrument.IO.TerminationCharacterEnabled = True
'**********done setting up communication*******************************
  DesiredFreq = "frequency " + DesiredFreq
   instrument.WriteString DesiredFreq


End Sub

Public Sub Pause(NbSec As Single)
 Dim Finish As Single
 Finish = Timer + NbSec
 DoEvents
 Do Until Timer >= Finish
 Loop
 End Sub
```

Appendix C: MatLab code for extracting data from text-based data storage files

The following two programs were written to retrieve the data that were collected and written by the Visual Basic code in Appendix B. The first program, *getbathdata1.m*, reads the text files and stores the data in the text file into one large MatLab matrix. The second program, *get_all_mag_data1.m*, reads the magnitude data from the matrix created by getbathdata1.m. It then creates 2 n-by-n matrices; one for the *x* coordinate values and one for the *y* coordinate values. The magnitude data is put into its own matrix to be retrieved in a subsequent program.

PROGRAM 1:

```
function [x,y,z,d] = getbathdata1(filename)
%  retrieve x,y,z and measurement data for all freqs from a single specified file
%
format long
fid = fopen(filename);
a = fgets(fid);
b = sscanf(a,'%*s%f%*s%f%*s%f');
x = b(1);
y = b(2);
z = b(3);
a = fgets(fid);
d = [];
for i=1:35

   tline = fgets(fid);
   if ~ischar(tline)
      fclose(fid);
      break
   end
   c = sscanf(tline,'%f%f%f%f%f%f',[6 inf]);
   d = [d c];

end
d= d';
% x_data = d(:,1);
% y_data = d(:,2);
% figure;
% loglog(x_data,y_data);
% tstring = strcat('Bode plot:', filename);
% title(tstring);

% figure;
% % mesh(X,Y,Z);  % for a mesh plot
% contour(X,Y,Z,35) % for a contour plot
```

```
% axis tight;
% hold on
% plot3(x,y,z,'.','MarkerSize',15) % nonuniform
% xlabel('x');
% ylabel('y');
% zlabel('z');
```

PROGRAM 2:

```
function [X,Y,Z,D] = get_all_mag_data1(root,first,last)
%example call:
%get_all_mag_data1('C:\ChrisDillon ThesisData\ScanResults\6_8_06\fulltest_',1,10)
X = [];
Y = [];
Z = [];
D = [];
columncount = 1;
rowcount = 1;
fillcolumndirection = 1;  %1 = down  0=up
for i=first:last
   istr = num2str(i);
   filenamestr = strcat(root,istr,'.txt');
   if fopen(filenamestr)> -1
      [x,y,z,d] = getbathdata1(filenamestr);
      X(rowcount,columncount) = x;
      Y(rowcount,columncount) = y;
      Z(rowcount,columncount) = z;
      [row col] = size(d);
      irow = linspace(i,i,row);
      icol = irow';
      id = [icol d];
      D = [D id];
   else
      X(rowcount,columncount) = 0;
      Y(rowcount,columncount) = 0;
      Z(rowcount,columncount) = 0;
      [row col] = size(d);
      irow = linspace(i,i,row);
      icol = irow';
      id = [icol (d.*0)];
      D = [D id];
   end
   %determine if column fill is up or down and direct next fill according
   if fillcolumndirection ==1
      rowcount = rowcount+1;
   else
```

```
        rowcount = rowcount-1;
    end
    %determine if filling is at bottom or top of column and move over 1
    if i/5 == ceil(i/5) & fillcolumndirection ==1
        columncount = columncount+1;
        rowcount = 5;  %this needs to be changed with the # of grid points
        fillcolumndirection = 0;
    elseif i/5 == ceil(i/5) & fillcolumndirection ==0
        columncount = columncount+1;
        rowcount = 1;
        fillcolumndirection = 1;
    end

end
```

Appendix D: MatLab code for creating potential field plots

This software, *go_contourplot1.m,* is higher level code which calls on both programs in Appendix C. The user specifies the frequency of interest and the program will compute a contour plot based on the magnitude date at the desired frequency at all 100 points. The program reads the magnitude data from the matrix created by the program get_all_mag_data1.m and creates an n-by-n matrix where each magnitude value is in the same position as its x and y coordinate values in their respective matrices.

```
function go_contourplot1(rootfile,first,last,freq)
% go_contourplot1.m
% plot a contour map of a set of data collected from Chris' bath for a single test frequency
%   'first' is the beginning file number
%   'last' is the final file number
%   'rootfile' is the basic name for the experimental run
%   'freq' is the test frequency in Hz to display
%
%first = 1;
%last = 11;
%rootfile = 'test2_';
%freq = 40;

% go collect all of the data for the files
[X,Y,Z,D] = get_all_mag_data1(rootfile,first,last);
X
Y

% now begin the plotting process

% first select the appropriate index for the desired test frequency

switch freq
case 30
   k = 1;
case 40
   k = 2;
case 50
   k = 3;
case 60
   k = 4;
case 70
   k = 5;
case 80
   k = 6;
case 90
   k = 7;
```

```
case 100
   k = 8;
case 200
   k = 9;
case 300
   k = 10;
case 400
   k = 11;
case 500
   k = 12;
case 600
   k = 13;
case 700
   k = 14;
case 800
   k = 15;
case 900
   k = 16;
case 1000
   k = 17;
case 2000
   k = 18;
case 3000
   k = 19;
case 4000
   k = 20;
case 5000
   k = 21;
case 6000
   k = 22;
case 7000
   k = 23;
case 8000
   k = 24;
case 9000
   k = 25;
case 10000
   k = 26;
case 20000
   k = 27;
case 30000
   k = 28;
case 40000
   k = 29;
case 50000
   k = 30;
```

```
case 60000
   k = 31;
case 70000
   k = 32;
case 80000
   k = 33;
case 90000
   k = 34;
case 100000
   k = 35;
otherwise
   echo ' error in freq specification; check code listing'
end

DAT = [];
columncount = 1;
rowcount = 1;
fillcolumndirection = 1;  %1= down   0= up
for m=first:last
   q = 1;  %  q = 1, 2, 3 or 4 for magnitude, phase, real or imag
   colptr = (7*(m-1))+(2+q);
   DAT(rowcount,columncount) =  D(k,colptr);
%determine if column fill is up or down and direct next fill according
   if fillcolumndirection ==1
      rowcount = rowcount+1;
   else
      rowcount = rowcount-1;
   end
 %determine if filling is at bottom or top of column and move over 1
   if m/5 == ceil(m/5) & fillcolumndirection ==1
      columncount = columncount+1;
      rowcount = 5;  %this needs to be changed with the # of grid points
      fillcolumndirection = 0;
   elseif m/5 == ceil(m/5) & fillcolumndirection ==0
      columncount = columncount+1;
      rowcount = 1;
      fillcolumndirection = 1;
   end

end

DAT(1,3) = 0.734863;
DAT(4,3) = 0.734863;
DAT(5,3) = 0.734863
size(X);
size(Y);
```

```
size(DAT);

% now do contour plot
%  test data to test contour plotting ==>  [X,Y,DAT] = PEAKS;
contour(X,Y,DAT);
%pause;
```

Appendix E: MatLab code for creating bode plots of collected data

These programs allow the user to visualize the bode plots of up to ten points at once. The first program, *go_bodeplot1.m,* creates as many subplots as indicated by the user. It calls on the lower program, *bodeplot1.m,* to retrieve the data from the text file. These programs allowed the user to quickly determine if any of the 100 scanned points produced inconsistent with the general trends of the bode plot.

PROGRAM 1:

```
% plot the magnitude data in bode format
function go_bodeplot1(root,first,last)
for i=first:last
   istr = num2str(i);
   filenamestr = strcat(root,istr,'.txt');
   if exist(filenamestr)==2
      if (last/2) == floor(last/2)
         subplot(ceil((last-first)/2),2, i-(first-1));
         bodeplot1(filenamestr);
         tstring = strcat('Bode plot:',num2str(i));
         title(tstring)
      else
         subplot(ceil((last-first)/2)+1,2, i-(first-1));
         bodeplot1(filenamestr);
         tstring = strcat('Bode plot:',num2str(i));
         title(tstring)
      end
   else
      %don't plot

   end
end
```

PROGRAM 2:

```
function bodeplot1(filename)
%  produce a bode plot of the magnitude data in the specified file
%       the plot is loglog mag vs. freq
%   phase is not plotted in the present version
%
format long
fid = fopen(filename);
a = fgets(fid);
b = sscanf(a,'%*s%f%*s%f%*s%f');
x = b(1);
y = b(2);
```

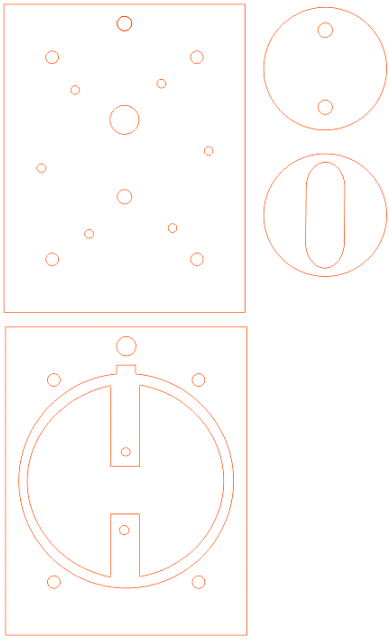
```
z = b(3);
a = fgets(fid);
d = [];
%for i=1:34
%    a = fgets(fid);
while 1
   tline = fgetl(fid);
   if ~ischar(tline)
      fclose(fid);
      break
   end
   c = sscanf(tline,'%f%f%f%f%f',[5 inf]);
   d = [d c];
end

d= d';
f_data = d(:,1);
mag_data = d(:,2);
%mag_data = 20*log(mag_data/.1);
%figure;
loglog(f_data,mag_data);
%tstring = strcat('Bode plot:', filename);
%title(tstring);
end
% figure;
% % mesh(X,Y,Z);  % for a mesh plot
% contour(X,Y,Z,35) % for a contour plot
% axis tight;
% hold on
% plot3(x,y,z,'.','MarkerSize',15) % nonuniform
% xlabel('x');
% ylabel('y');
% zlabel('z');
```

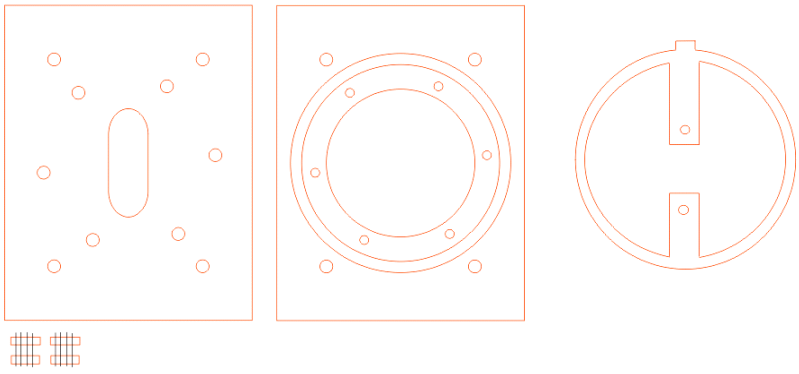
Appendix F: Corel Draw files for tissue bath

The images below were created in Corel Draw for the fabrication of the acrylic pieces that were used to make the tissue bath. The pieces labeled thick pieces were cut from 3mm thick pieces of acrylic. The pieces labeled thin pieces were cut from 1.5 mm thick pieces of acrylic. Components were directly machined from cast acrylic sheets of the desired thickness using a Universal Laser Systems x660 dual-laser 120W CO2 laser engraving and machining system.

Thin Parts:



Thick Parts:



REFERENCES

National Institute on Deafness and Communication Disorders (NIDCD) (2002) http://www.nidcd.nih.gov/health/statistics/hearing.asp

Spelman FA, Clopton BM and Pfingst BE (1982). Tissue impedance and current flow in the implanted ear. Implications for cochlear prosthesis. Ann Otol Rhinol Laryngol Suppl. Sep-Oct ; 93. 3-8

Susserman MF and Spelman FA (1993). Quantitative In Vivo measurements of inner ear resistivities: I. In Vitro characterization. IEEE Transactions on Biomedical Engineering, 40 (10): 1032-1046.

Clopton BM and Spelman FA (1982). Neural mechanisms relevant to the design of an auditory prosthesis. Location and electrical characteristics. Ann Otol Rhinol Laryngol Suppl. Sep-Oct ; 98. 9-14

Spelman FA, Pfingst BE, Clopton BM, Jolly CN, Rodenhiser KL (1995). Effects of electrical current configuration on potential fields in the electrically stimulated cochlea: field models and measurements. Ann Otol Rhinol Laryngol Suppl. Sep; 166. 131-6

Ni D, Shepherd RK, Seldon HL, Xu SA, Clark GM, Millard RE (1992). Cochlear pathology following chronic electrical stimulation of the auditory nerve. I: Normal hearing kittens. Hearing Research. 62(1): 63-81

Loizou, P (1998). Mimicking the Human Ear. IEEE Signal Processing Magazine, 15(5): 101-130.

Bess FH, Humes LE (1995), Audiology, The Fundamentals

Seshadri, Sudharshana. Design and Fabrication of a multi-channel micro-molded cochlear implant for animal studies (2004). http://forms.gradsch.psu.edu/equity/sroppapers/2004/SeshadriSudharshana.pdf