MICRO-CHAMBER FILLING EXPERIMENTS FOR VALIDATION OF MACRO MODELS WITH APPLICATIONS IN CAPILLARY DRIVEN MICROFLUIDICS

A Thesis

by

STEPHEN BYRON GAUNTT

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2007

Major Subject: Mechanical Engineering

MICRO-CHAMBER FILLING EXPERIMENTS
FOR VALIDATION OF MACRO MODELS WITH
APPLICATIONS IN CAPILLARY DRIVEN MICROFLUIDICS

A Thesis

by

STEPHEN BYRON GAUNTT

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Debjyoti Banerjee
Committee Members,	Warren Heffington
	Yassin Hassan
Head of Department,	Dennis O'Neal

December 2007

Major Subject: Mechanical Engineering

ABSTRACT

Micro-Chamber Filling Experiments for Validation of Macro Models with Applications in Capillary Driven Microfluidics. (December 2007)

Stephen Byron Gauntt, B.S., Worcester Polytechnic Institute

Chair of Advisory Committee: Dr. Debjyoti Banerjee

Prediction of bubble formation during filling of microchambers is often critical for determining the efficacy of microfluidic devices in various applications. In this study experimental validation is performed to verify the predictions from a previously developed numerical model using lumped analyses for simulating bubble formation during the filling of microchambers. The lumped model is used to predict bubble formation in a micro-chamber as a function of the chamber geometry, fluid properties (i.e. viscosity and surface tension), surface condition (contact angle, surface roughness) and operational parameters (e.g., flow rate) as user defined inputs. Several microchambers with different geometries and surface properties were microfabricated. Experiments were performed to fill the microchambers with different liquids (e.g., water and alcohol) at various flow rates to study the conditions for bubble formation inside the microchambers. The experimental data are compared with numerical predictions to identify the limitations of the numerical model. Also, the comparison of the experimental data with the numerical results provides additional insight into the physics of the micro/nano-scale flow phenomena. The results indicate that contact angle plays a

significant role on properties of fluids confined within small geometries, such as in microfluidic devices.

DEDICATION

This thesis is dedicated to my parents, Stephen and Ann, whose support throughout my academic career has been unwavering and unprecedented. It is their continued support which has guided me to the point where this thesis is now possible. I would also like to thank my fiancée, Lindsay, who has been a loyal and loving companion to me since the beginning of my pursuit for this degree. Finally, I would like to recognize the influence of my late grandfather, Roland, who continues to inspire me to learn and pursue any goal I set forth for myself. Without the support of my family, I would not be where I am today.

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Debjyoti Banerjee, and my committee members, Dr. Warren Heffington, and Dr. Yassan Hassan, for their guidance and support throughout the course of this research. I would also like to thank Rodolfo DeLeon for his help in performing many of the experiments required of this thesis. Without his help this thesis would not have been possible.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience. I also want to extend my gratitude to the Materials Characterization Facility at Texas A&M which provided me access to the services required for the construction of my experimental equipment. Thanks goes out in particular to Dr. Jingyi Shen and Gregory Fernandez of the Chemical Engineering Department for their help in several aspects of this thesis.

Finally, thanks to my mother and father for their encouragement and to my fiancée for her patience and love.

NOMENCLATURE

HF	Hydrofluoric Acid
BOE	Buffered Oxide Etch
DI	Demineralized Water
ID	Inner Diameter
OD	Outer Diameter
OTS	Octadecyltrichlorosilane
u	velocity
μ	viscosity
ρ	density
σ	surface tension coefficient
θ_1	contact angle with top wall
θ_2	contact angle with bottom wall
θ_3	contact angle with side walls
L_1	length of micro-chamber
L_2	width of micro-chamber
$\mathrm{L_{e1}}$	Scan Length at Top of Microchamber
$\mathrm{L_{e2}}$	Scan Length at Bottom of Microchamber
$\mathrm{l_{s1}}$	Meniscus Position along $\mathrm{L_1}$ Direction
$\mathrm{l_{s2}}$	Meniscus Position along $\mathrm{L_2}$ Direction
$\mathrm{l_m}$	Position along central portion of the meniscus

Δl_{s1}	Measurement Error of l_{s1}
Δl_{s2}	Measurement Error of l_{s2}
Δl_{m}	Measurement Error of l_{m}
h	depth of micro-chamber
Q	volumetric flow rate
t	time
w	region of wall influence

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Page

Page

1. INTRODUCTION

Miniaturization of fluid handling technologies is known as “Microfluidics”. Microfluidics deals with the study of flow behavior, precise control and manipulation of small (e.g., microliter, nanoliter and picoliter) volumes of fluids. It is a multidisciplinary field intersecting engineering, physics, chemistry, micro/nano-fabrication and biotechnology. Explosive growth of microfluidics applications emerged in the 1990s with the development of lab-on-chip devices such as bio-chips (gene chips and protein chips for nucleic acid detection, i.e., DNA, RNA and proteins/ peptides as well as biochemical synthesis), micro-propulsion, micro-thermal technologies (Tsai et al. 2006; Wang 2004; Estes 2005, Orieux et al. 2002, Lewis et al. 2000). Behavior of fluids at the microscale can differ from 'macrofluidic' behavior due to predominance of surface effects (over volumetric forces) such as surface tension, energy dissipation, surface roughness and fluidic resistance. The field of microfluidics is aimed at studying how these behaviors change, and how they can be optimized or exploited for novel applications. Microfluidics technology enabled the “Human Genome” project to be completed ahead of schedule (Human Genome Project Information, 2007).

Microfluidic technologies confer several advantages – lower materials usage, faster operation (lower reaction times for chemical reactions), higher sensitivity as well precision for detection applications, less propensity for formation of impurities during

This thesis follows the style of Microfluid Nanofluid.

biochemical synthesis, development of portable (and hand-held) platforms and novel applications such as bio-chips (Tsai et al. 2006; Estes 2005).

Microfluidics devices are obtained by a combination of micro-chambers (or reservoirs), microchannels (or conduits for flow), and flow actuation devices (e.g., micro-pumps). Formation of bubbles during filling of micro-chambers and micro-channels is often considered to be a catastrophic failure of such devices. For example, micro-chambers are used as DNA hybridization chambers [Nanogen, Inc.]. DNA hybridization is prevented in the region occupied by a bubble and bubble formation is therefore undesirable.

Bubble formation can be predicted by various numerical models, e.g., Volume of Fluids (VOF) method (Menard, et al. 2007; Morel 2007; Banerjee et al. 2005), Level Set Method (Abe et al. 2007; Grob et al. 2006; Carrica et al. 2007), Front Tracking Method (Liu et al. 2007; Xu et al. 2007; Witteveen et al. 2007), etc. These models rely on discretization of the governing equations and boundary conditions (e.g., finite difference or finite volume) and require substantial computing resources (memory, computational steps, problem definition). As a consequence, a single computational run can require several days to a few weeks for completion even for simulating flow in simple geometries. Application of these techniques to microfluidics applications is also very challenging since microfluidics devices consist of high aspect ratio fluidic structures (e.g., micro-channels). Also, these numerical techniques (e.g., VOF) are very sensitive to the grid generation schemes (e.g., grid aspect ratio) as well as simulation parameters (e.g., numerical convergence and acceleration schemes) and are often susceptible to computational errors during execution – sometimes generating non-physical results if the problem definition is not properly implemented. Hence, these numerical techniques are not user-friendly, cumbersome, complicated to implement and often require substantial effort for development.

Lumped models (also known as “reduced-order-models”, “macro models”, “system models”, “compact models”, “Spice® models”, etc.) are behavioral models which can be very useful in the simulation of complex systems requiring minimal computational effort. Macromodels provide faster simulation schemes where the time required for a typical simulation can be reduced by a factor of 10-100 compared to the simulation times required by physical models that are based on discretization techniques. In such models, the system is described by behavioral (fitted) parameters. For example, the flow in a pipe can be described by the equivalent resistance-potential model. This enables a simplified implementation of the models, reduced model development effort, makes the tools user friendly, and is less susceptible to computational errors. Macromodels are ideally suited for simulating fluid behavior in microfluidic devices, especially for those with high aspect ratio fluidic structures. Macromodels have been used for simulating various microfluidics devices and systems. The development of such tools vastly simplifies the design procedure and also helps to minimize the necessity of CFD (Computational Fluid Dynamics) tools for parametric investigations. Hence, the macromodels reduce the time and effort required for parametric investigation as well as rapid exploration of the design space for design optimization. This makes macromodels ideally suited for the commercial environment. Such models can be developed using a wide range of programming languages including C++, Microsoft ® Visual Basic, Java, or even utilizing spreadsheet applications such as Microsoft ® Excel. Such programmed models are also ideally suited due to their low reliance on large amounts of computational resources, making them suitable for deployment on traditional desktop personal computers   Such a model, previously developed (Banerjee 2005) for simulating bubble formation during microchamber filling, will be validated experimentally and calibrated in this proposed study.

2. LITERATURE REVIEW

Simulation of microchamber filling has often focused on laborious CFD analysis. Such an analysis has been carried out extensively by Jensen (2002) who simulated bubble formation in both 2-D and 3-D channels. His findings, however, were not validated experimentally and focused on the utilization of various CFD packages and not the development of a quick and useful design tool. Additional CFD work has been conducted by Weber and Shandas (2007) who studied microbubble formation in microfluidic flow-focusing devices.

Some early work in the development of macromodels was conducted by Bourouina (1996), who presents such a model for rapid simulation of micropumps. This model is used to predict the flow-rates and pressure inside the pump chamber, comparing admirably to experimental results. Qiao, et al. (2002) presents a compact model to predict the flow rate, pressure distribution and other basic characteristics in microfluidic channels when the driving force is either an electric field or a combined electric field and pressure gradient, while also considering the effects of varying zeta potential. Their model was shown to give good results when compared to detailed numerical simulation, with errors around 8% for both flow types. Chatterjee, et al. (2005) further elaborated on the model to account for this error and significantly increased the models ability to capture the physics of the fluidic transport in much greater detail.

Macromodels can be used to model a variety of flow behaviors. Morris, et al. (2004) compared lumped-parameter expressions for the impedance of an incompressible viscous fluid subjected to harmonic oscillations in a channel with exact expressions based on the Navier-Stokes equations. He found, however, that these lumped-parameter expressions led to large errors, as high as 400% in some cases, and recommended that the exact solutions should be used. It is not uncommon for macromodels to lead to large errors such as these, which introduces the necessity of calibrating the expressions with experimental results. Magargle et al. (2004) and Mikulchenko et al. (2000) have used neural-network models for electrokinetic injection and a microflow sensor, respectively, which are parameterized by the device geometry and operational parameters (e.g., electric field and flow velocity). Jousse, et al. (2005) present another model used to describe the laminar flow of viscous multiphase fluids in microchannel networks, in which they use a "'incomplete Wheatstone bridge'' network to show how fluid repartition depends on the input parameters. Wang et al. (2004a, 2004b) have presented analytical models to study dispersion effects in electrokinetic flow induced by both turn geometry and Joule heating using a "method-of-moments" approach. These models effectively capture the effect of chip topology, separation element size, material properties, and electric field on the separation performance.

Turowski, et al. (2001) has suggested a design methodology for the generation of compact models of microfluidic elements which can be used with various system-level simulators such as SPICE (Simulation Program with Integrated Circuit Emphasis) and Saber, two circuit simulator programs. The specific example of a "Tesla Valve" was used to validate the procedure and comparisons were made with high-fidelity 3D simulations along with experimental results of the microfluidic device. The discrepancy between the generated compact models and 3D simulation results was shown to be less than 2% in the entire range for this particular example. Though these macromodels may not capture all the details elucidated by grid-based 3D modeling techniques, they are adequate enough to quickly and accurately capture the basic physical behavior of the system which can be used in the design of microfluidic systems.

Some work has also been done in revealing the behavior of fluids in micro and nano devices. Meinhart, et al. (2001) studied the validity of the common no-slip boundary condition for viscous flow at solid walls on the micro and nano scale. It was found that for hydrophilic surfaces this condition remained a reasonable assumption for micro and nano scale flows. However, for extremely hydrophobic surfaces, such as those treated with Octadecyltrichlorosilane, this assumption was no longer valid and it was found the velocity of the fluid at the wall is roughly 10% of the free-stream velocity. Hess et al. (1989) suggested that if the strain rate at the wall exceeds twice the molecular frequency scale, the no-slip boundary condition at the wall leads to incorrect modeling behavior. This assumption of slippage at the walls could be a possible explanation for a change in the region of wall influence for the macromodel used in this present study, which will be described in a later section.

Churaev et al (1971) found that the viscosity of water in glass capillaries of 80 nm diameter is approximately 40% elevated, and that this elevation decreases rapidly with increasing channel size. This was explained by a possible increased ordering of the polar water molecules near the channel walls, while Tas et al. (2004) attributes the change to electro-viscous effects. Quere (2001) studied the velocity of falling slugs in vertically mounted capillary tubing under both pre-wetting and dry conditions. The authors mentioned that the results can be explained by an apparent viscosity change due to a change in the falling slug’s velocity.

3. DESCRIPTION OF THE MACROMODEL

Banerjee (2005), described the development of a macromodel for predicting the filling of microchambers in capillary driven flows. The macromodel input parameters include geometric parameters (size of the chamber) and fluid properties (viscosity, contact angle, surface tension) as customizable inputs. Different geometries of liquid flow pathways may result in different capillary filling behavior such as filling time, and the possibility of bubble entrapment. Knowledge of the filling process can guide designers in arranging internal structures of the chip to avoid potential filling problems and achieve higher filling speeds. A brief explanation of the formulation and implementation of this numerical model will now be discussed.

Borrowing concepts from electrical engineering, fluid flow can often be modeled with the use of an equivalent electrical network. In the study conducted by Banerjee (2005), a Volume of Fluids (VOF) simulation was conducted to obtain a basic understanding of the fluid flow within a microchamber. The results show that near the wall – the wall effects cause a 3-D flow. This region where 3-D effects dominate is denoted by a region of width *w*, which is referred to as the “region of wall influence”, as illustrated in Figure 1.

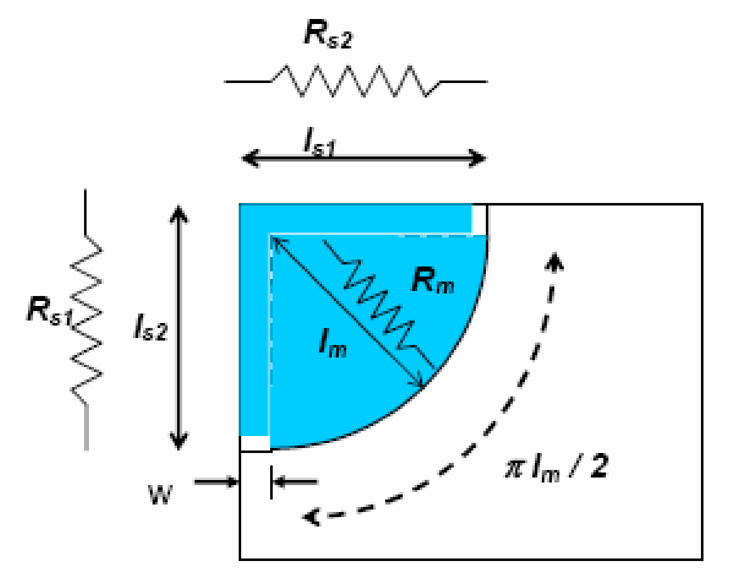


Figure 1: Schematic of the Macromodel for Microchamber Filling.

Away from the wall (outside *w*, or outside the region of wall influence) the velocity vectors demonstrate the characteristics similar to flow between infinite flat plates. Thus the majority of the flow may be modeled as flow between infinite flat plates. The variables R_{s1}, R_{s2}, and R_m denote the flow resistances along the walls (within the region of wall influence) and at the middle portion of the meniscus, respectively. The shape and position of the meniscus is specified by l_{s1}, l_{s2}, and l_m which denote the position along the walls and middle portion of the meniscus respectively. Figure 2 illustrates the equivalent electrical network used in defining the three different flow regimes.

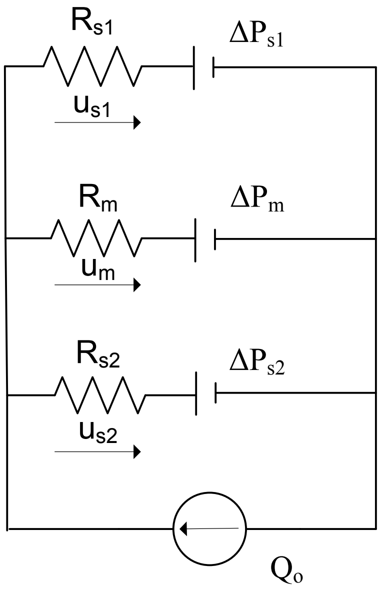


Figure 2: Schematic of the Resistance Network

The flow velocities of the meniscus (based on meniscus location) at the side walls and middle of the microchamber, are denoted by u_{s1}, u_{s2}, and u_m, respectively. Q_o represents the flow source and ΔP_{s1}, ΔP_{s2}, and ΔP_m denote the capillary pressure drops which are given in Eq.1 and Eq. 2.

$$\Delta P_m = \sigma(\cos\theta_1 + \cos\theta_2)/h \tag{1}$$

$$\Delta P_{s1} = \Delta P_{s2} = \sigma(\cos\theta_1 / w + \cos\theta_2 / w + \cos\theta_3 / h) \tag{2}$$

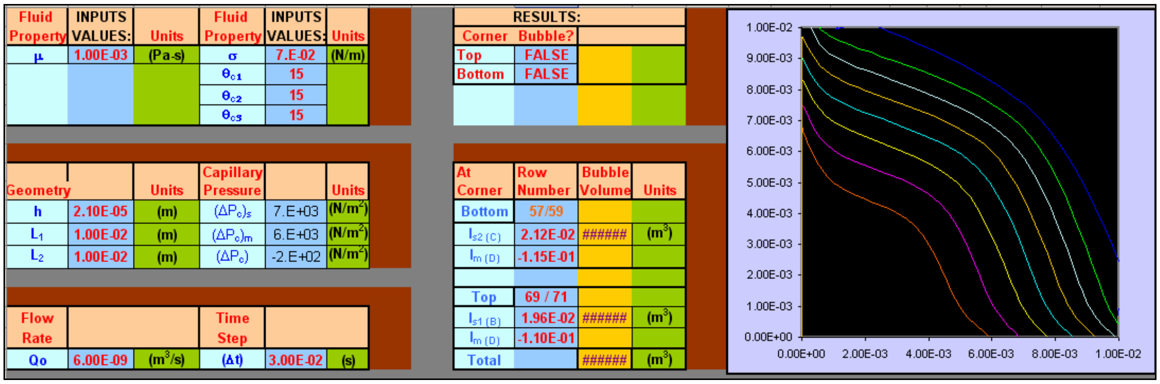
Kirchoff's law can be used to obtain the flow equations in the different legs of the fluidic circuit show in Figure 2. This will yield a system of 3 equations and 3 unknowns, as shown in Eq. 3 – 6. This system of equations can be used to obtain the unknown flow velocities at a particular instant of time.

$$R_{s1}u_{s1} + \Delta P_{s1} = R_m u_m + \Delta P_m \tag{3}$$

$$R_{s1}u_{s1} + \Delta P_{s1} = R_{s2}u_{s2} + \Delta P_{s2} \tag{4}$$

$$A_{s1}u_{s1} + A_m u_m + A_{s2}u_{s2} = Q_o \tag{5}$$

where A_{s1}, A_{s2}, and A_m are the flow areas in each flow region. The location of the three points located on the meniscus can then be obtained from the points on the previous time step by adding the product of the velocity and the chosen time differential (Δt). This algorithm has been incorporated into a Microsoft ® Excel ® spreadsheet for ease of development, implementation, distribution and use. Figure 3 shows a screenshot of the realized macromodel. In essence, this model describes the balance between surface forces (capillary forces, contact angle), viscous resistance and inertial forces (flow rate).



Fluid Property	INPUTS VALUES:	Units	Fluid Property	INPUTS VALUES:	Units
μ	1.00E-03	(Pa-s)	σ	7.E-02	(N/m)
			θ_{c1}	15	
			θ_{c2}	15	
			θ_{c3}	15	

Geometry		Units	Capillary Pressure		Units
h	2.10E-05	(m)	$(\Delta P_c)_s$	7.E+03	(N/m^2)
L_1	1.00E-02	(m)	$(\Delta P_c)_m$	6.E+03	(N/m^2)
L_2	1.00E-02	(m)	(ΔP_c)	-2.E+02	(N/m^2)

Flow Rate			Time Step		
Qo	6.00E-09	(m^3/s)	(Δt)	3.00E-02	(s)

RESULTS:			
Corner	Bubble?		
Top	FALSE		
Bottom	FALSE		

At Corner	Row Number	Bubble Volume	Units
Bottom	57/59		
$l_{s2\,(C)}$	2.12E-02	######	(m^3)
$l_{m\,(D)}$	-1.15E-01		
Top	69 / 71		
$l_{s1\,(B)}$	1.96E-02	######	(m^3)
$l_{m\,(D)}$	-1.10E-01		
Total		######	(m^3)

Figure 3: Screenshot of the Macromodel in Microsoft ® Excel ®

It is an open question whether fluid contact angle in small confined geometries (e.g., microfluidic devices) can affect fluid viscosity in the near wall region (e.g., the region of wall influence used in the macro-model). In such situations the surface forces are dominant and have the potential to alter flow behavior at the micro/nano-scale. This work will explore the effect of contact angle on viscosity. The effect of viscosity change can be discernible by the change in region of wall influence (*w*). Hence by studying the effect of the contact angle on the variation of the region of wall influence it can be conluded whether surface effects affect viscosity close to the wall. The variation of *w* will be obtained from the macromodel after calibration of the model with the experimental data. It is therefore hypothesized that a change in the size of this region (*w*), while leaving all other geometric and flow properties the same, can only happen if the viscosity of the fluid changes. The experimental validation of the numerical model will enable the verification of this hypothesis.

4. FABRICATION OF MICROCHAMBERS

Several microchambers of various geometric dimensions were microfabricated by etching glass substrates with depths between 20 and 50µm. The general layout of these chambers is shown in Figure 4.

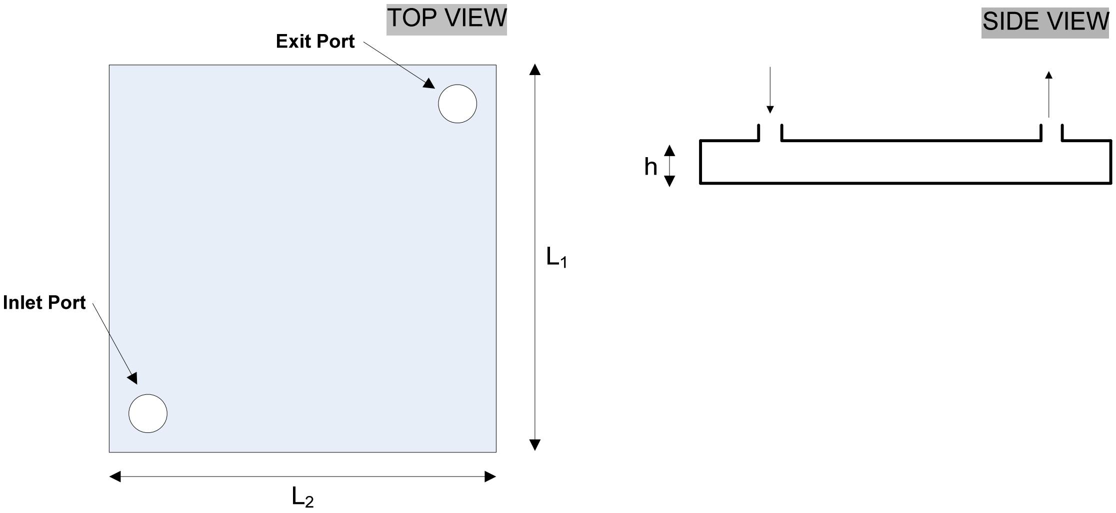


Figure 4: Typical Microchamber Setup

In Figure 4, L_1 and L_2 denote the lengths of the sides of the microchamber while h denotes the depth. Fabrication of microfluidic devices constructed from glass is typically done via Hydrofluoric Acid (HF) etching. With etching, a masking material is applied to the glass substrate which protects it from HF attack. This mask is typically either a photoresist which can defend against the acid or an inert metal such as gold. When the masked surface is brought into contact with a pool of liquid HF, only the exposed regions of glass are attacked (or etched) and, upon removal of the mask, the desired features are embossed. The following sections will outline the processing steps required for micro-chamber fabrication via this wet etching technique. Plain glass slides (manufactured by Fisher Scientific, catalog number 12-550A), were used for the microchamber assemblies.

4.1 Application of Masking Material

Masking layers help to resist unwanted attack of a substrate when brought into contact with an attacking liquid. The process of applying a photoresist masking layer via photolithography is simply illustrated in Figure 5. While this example is specifically for that of glass, the basic principle is applicable to a wide variety of substrate materials.

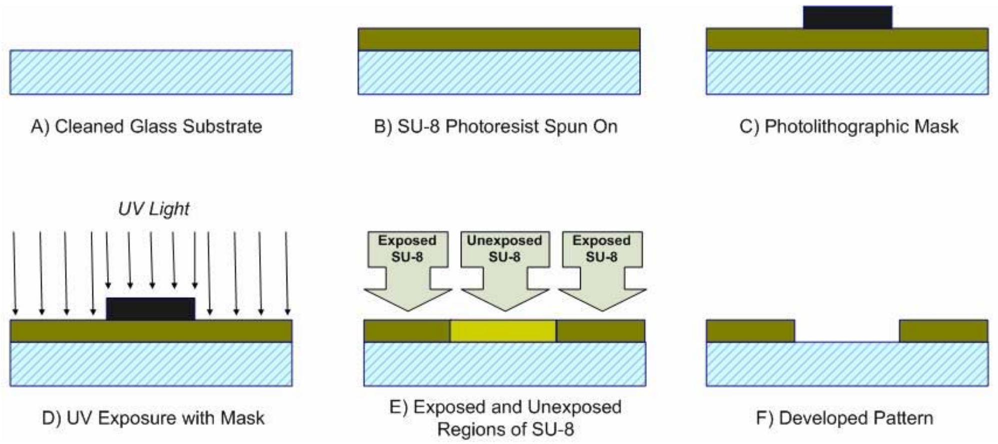

A) Cleaned Glass Substrate
B) SU-8 Photoresist Spun On
C) Photolithographic Mask
UV Light
D) UV Exposure with Mask
Exposed SU-8
Unexposed SU-8
Exposed SU-8
E) Exposed and Unexposed Regions of SU-8
F) Developed Pattern

Figure 5: Overview of the Photolithography Process

Application of the masking layer begins by cleaning a standard 0.15 mm thick glass cover slide in acetone, followed by rinsing in methanol and de-ionized water. The cover slide is then dehydrated at 200 ^{o}C for at least 5 min on a hotplate. Photo-curable epoxy, such as SU-8 2015 photoresist, is dispensed onto the cover slide and spun at 2000 rpm for 30 s, resulting in an SU-8 layer thickness of 15–20 μm. The coated cover slide is then soft-baked for 1 min at 65 ^{o}C, and then further soft-baked for 3 min at 95 ^{o}C. Next, the coated cover slide is exposed to UV through a photo-mask containing the micro-chamber pattern. After exposure, a hard-bake at 95 ^{o}C for 1 min is performed to cross-link the exposed SU-8 regions. The masking layer is complete after soaking in SU-8 developer for 3 min.

4.2 Hydrofluoric Acid Etching

Hydrofluoric Acid (HF) is the typical chemical etchant used in the fabrication of microfluidic devices constructed from glass. It is important to note that HF is an extremely hazardous chemical in almost any concentration, and should only be used if no other viable options are available. Special protective garments are required and should never be used by the operator in solitude, due to safety considerations.

The etchant used for the fabrication of microchambers was a Buffered Oxide Etch (BOE) in a 20:1 concentration, with surfactant, provided by J.T. Baker Company product number JT5568-3. Figure 6 illustrates the setup used for the actual etching process. In this setup, a small circular PVC stand with an ID of 7.62 cm and OD of 8.89 cm was used to support the glass cover slide while being etched. This was done to keep both sides clean and free of a rough etch since optical access were needed from the opposite sides in later experiments. Placed inside this stand was a magnetic stirring rod used to re-circulate the etchant and allow fresh BOE to come in contact with the glass surface. The support stand, stirrer, and glass slide were then placed into a standard plastic beaker and submerged in BOE etchant, as illustrated in the following figure.

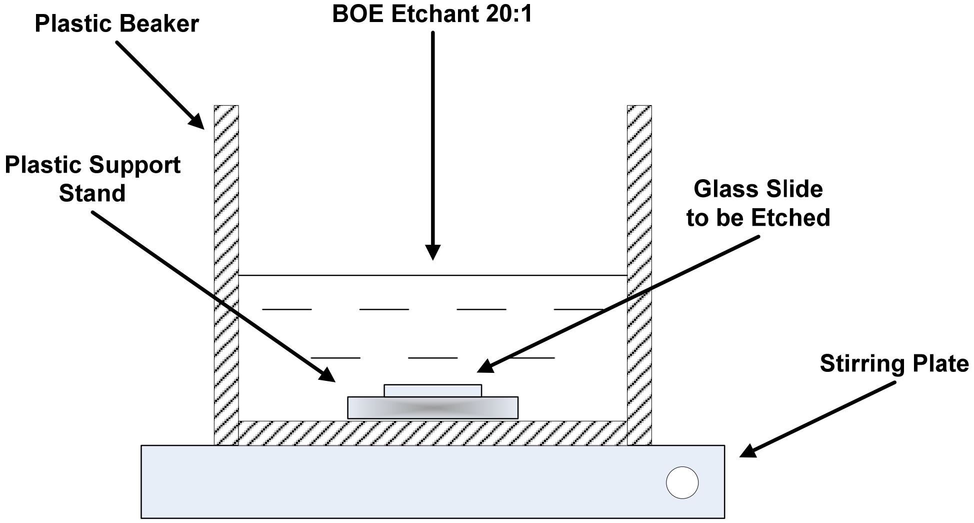


Figure 6: Schematic of Etching Setup

The apparatus was then placed onto a stirring plate used to circulate the stirring rod and set at a rate of around 60rpm. The etching rate of this solution has been found to be around 10μm/hour. Since a limited number of microchambers would need to be fabricated, etching speed was not of utmost importance. When the desired amount of time has elapsed, the glass slide is carefully removed and submerged in a beaker of DI water and subsequently rinsed again in water. The left over etchant solution is properly dispensed into a waste container labeled “HF Waste: Extremely Hazardous” and the PVC stand, beaker, and stirrer are then washed carefully with DI water. The photoresist mask is then removed by placing the glass slide in a beaker and submerging it in a small amount of PG remover. The beaker is then suspended in an ultrasonic cleaner for 20 minutes. Etching of the glass slide is then complete after removing the slide, disposing of the PG remover, and thoroughly cleaning all used materials. The result is a feature of roughly the same dimensions as the masking slide etched into the glass substrate with a depth defined by the etching time. The depth of each chamber was measured using a Veeco DekTak 3 Surface Profilometer for which the average depth was taken after exporting the data to a spreadsheet. The Dektak 3 Surface Profilometer is an instrument to measure the vertical profile of samples, thin film thickness, and other topographical features, such as film roughness or wafer bowing. Each chamber was scanned in six unique locations, at minimum, to ensure the etching depth was uniform. Example profiles of the resulting etches are shown in Figure 7 through Figure 9.

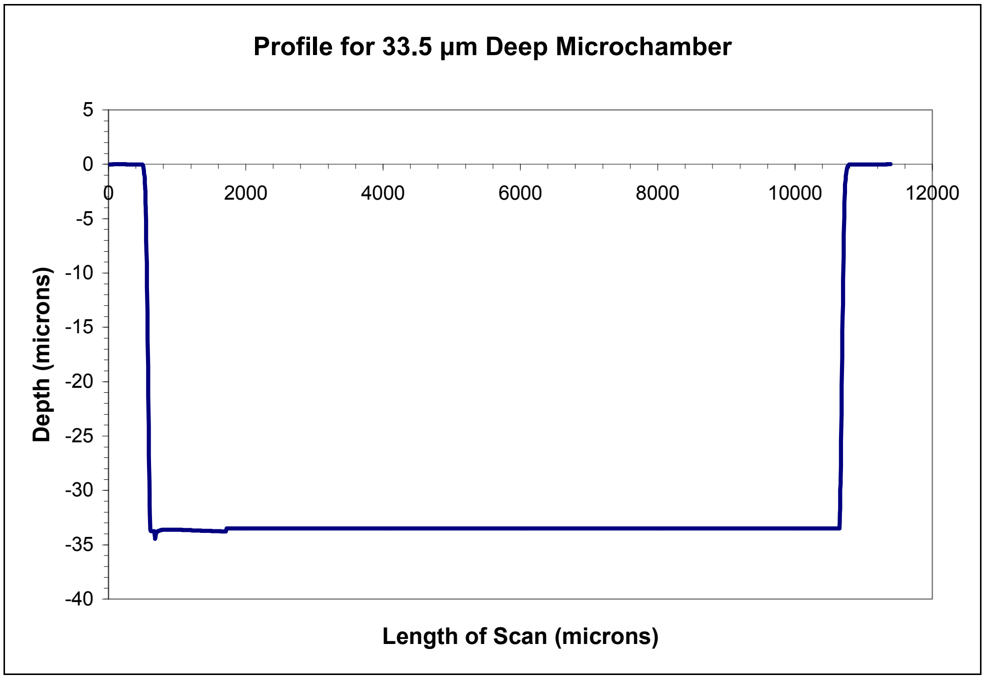


Figure 7: Example Profile for a 33.5µm Etch Depth

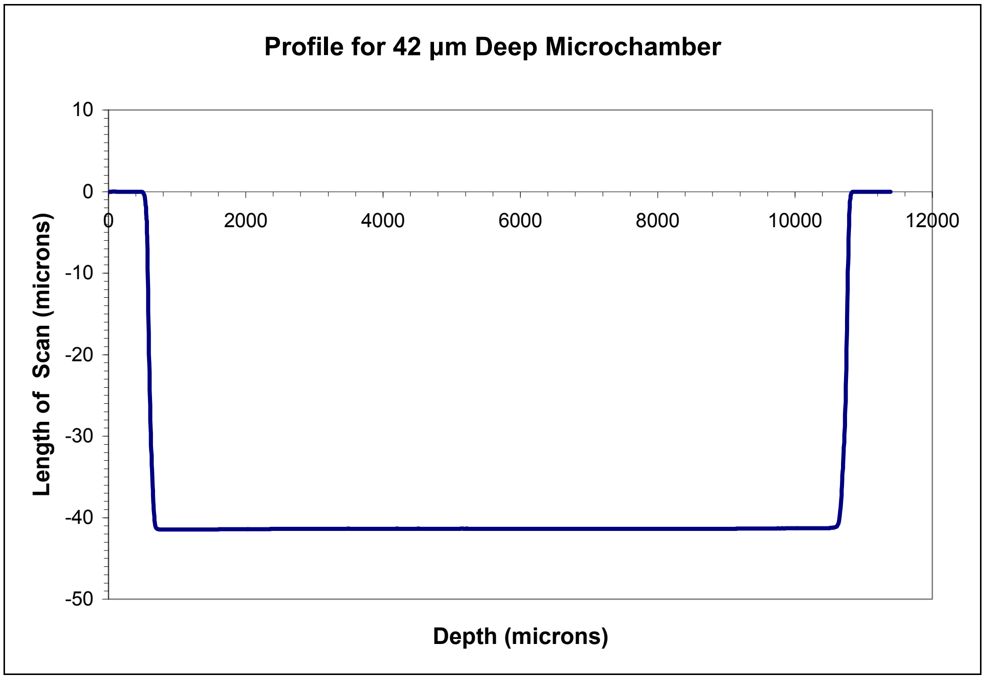


Figure 8: Example Profile for a 42µm Etch Depth

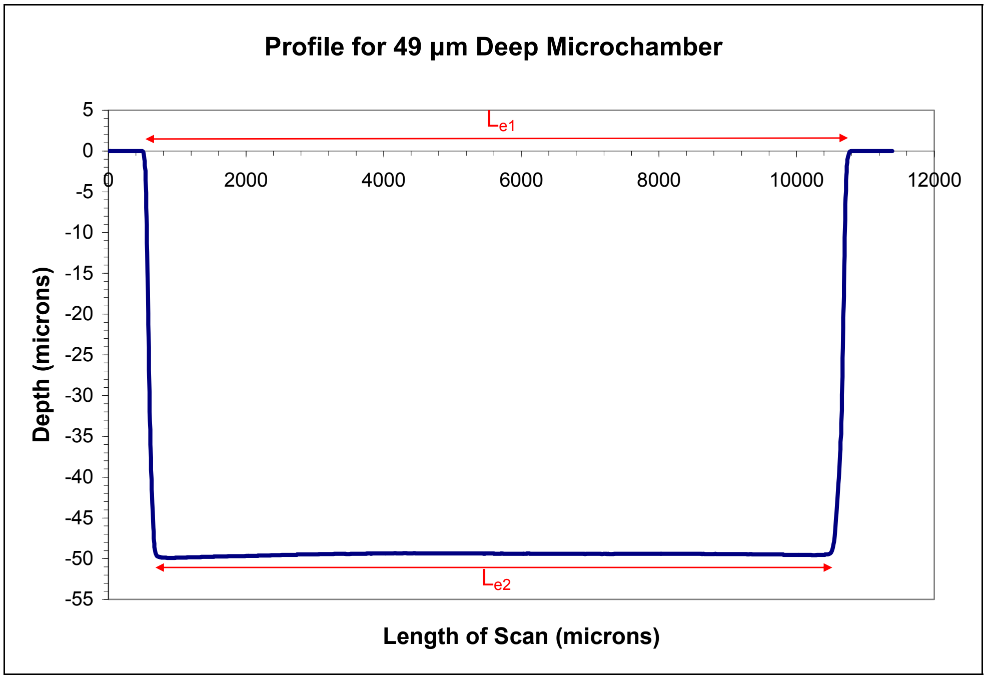


Figure 9: Example Profile for a 49µm Etch Depth

It can be seen from the preceding figures that etches appear smooth, with minimal undercutting occurring at the walls. In Figure 9, L_{e1} and L_{e2} represent the width of the microchamber at the top and bottom of the channel. These values will be slightly different due to undercutting, resulting in walls which are not perfectly vertical. This introduces an error into the measurement points for ls1 and ls2 which will be discussed in future sections. This error will be discussed in the section entitled “Comparison with Macromodel”.

4.3 Drilling Flow Ports

The top of the microchamber, hereby referred to as the glass cap, was constructed by drilling two, approximately 1mm diameter holes into a glass slide. This was done by utilizing a diamond plated solid thin drill bit provided by UKAM Industrial Superhard Tools, product number 4ED10. The drill bit was mounted on a Sears Craftsman 8” drill press set at a speed of 3100rpm. The locations of the holes were marked on the glass cap with a fine tipped marker. To reduce vibrations, a small piece of balsa wood was used as a cushion for the glass cap, both of which were placed in a shallow beaker to serve as a catch. While drilling, the bit was fed very slowly through the slide, occasionally withdrawing and dispensing water onto the bit and glass cap for cooling and flushing of the drilling area. Drilling is complete when the bit has fed through the entire thickness of the glass cap.

4.4 Chamber Assembly

The glass cap and slide containing the microchamber features were bonded using the method outlined by Fang (et al., 2004). The etched substrates and glass cap were washed sequentially with acetone, household dishwashing detergent, tap water at high flow rate (10-20 m/s), and ethanol to remove solid particles and organic contaminants from the glass surface. The cleaned slides were then further prepared by bathing in a Piranha solution (3:1 Sulfuric Acid to Hydrogen Peroxide) for a minimum of 40 minutes. Both slides were then dried before being soaked in concentrated sulfuric acid for 12 hours. Subsequently, the glass slides were aligned vertically and held with a space of 1-2 mm between the surfaces and washed again under a high flow of tap water for 5 min. The aligned slides were brought into close contact under a continuous stream of DI water flowing between them. The combined plates were then allowed to stand at room temperature for more than 3 hours to dry. This method of bonding proved superior to all other methods that were explored in this study which included various heat treatment schemes for bonding the glass substrates by melting. The selected bonding method also protected against leaking of the working fluid from the sides of the microchamber.

4.5 Final Assembly

Assembly of a microchamber is completed by installing a funnel which served as a connector for both supply side and exit side tubing. The tubing connections were required for pumping the working fluid into the microchamber, as shown in Figure 10.

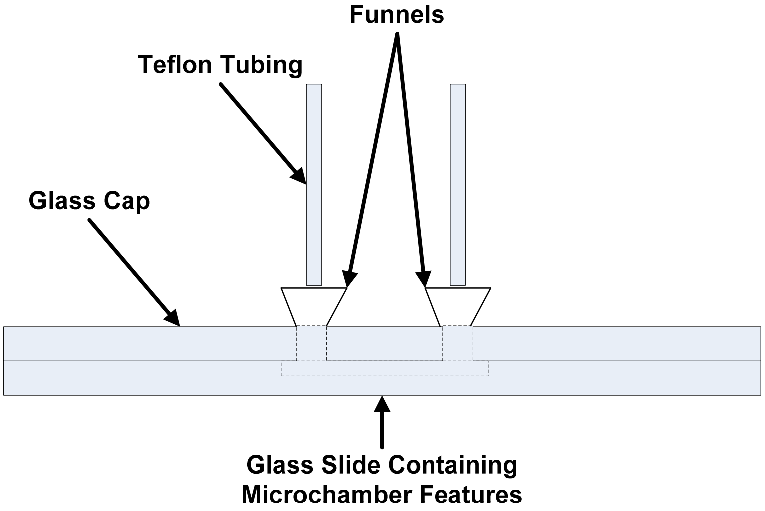


Figure 10: Final Microchamber Assembly

These connector funnels were made from pipette tips (supplied by VWR International, product number 53509-140). These pipette tips were cut with a razor blade to a length which allowed them to fit snugly in the holes of the glass cap. Teflon tubing of 0.0305 µm ID and 400 µm OD (provided by Upchurch Scientific, product number PM-1073), was fitted into the resulting funnel. This provided a good seal and minimized leaking of the working fluid at the entrance and exit of the microchamber. Figure 11 shows an image observed under a microscope as a representative sample of a completed chamber obtained using this process.

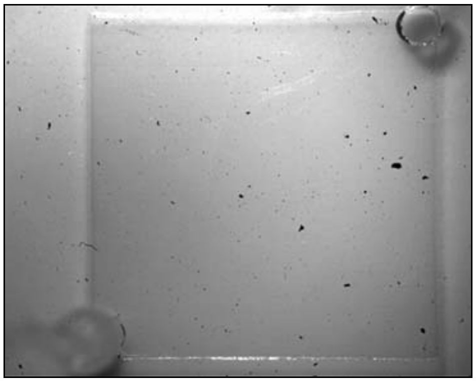


Figure 11: Magnified Image of a Representative Sample of an Assembled Microchamber

Microchambers were also fabricated utilizing SU-8 as an epoxy. In this method the photolithographic procedure was performed to imprint the desired pattern into SU-8 spun onto a glass substrate. The glass cap was then spin coated with a thin layer of SU-8 2002. The two pieces were then pressed together, with photoresist sides touching, and heated to 100^{o}C for at least 30 minutes. It was hoped that the resulting microchamber would provide hydrophobic side walls while leaving the top and bottom hydrophilic in contact with DI-H_2O. However, the bond proved to be rather weak and did not adequately protect against leaking. It's possible this method still warrants investigation since the spin coater used may not have evenly dispersed the photoresist and the resulting roughness could have impeded the bonding qualities sought after. To overcome the limitations of using SU-8 photoresist, fabrication of microchambers using double sided tape was also employed. In this method the double sided tape was applied to the surface of one cleaned class slide and a square of appropriate dimensions was cut into the tape and removed. Another glass slide containing the inlet and exit ports was aligned to this square and applied onto the exposed side of the double sided tape resulting in a sealed microchamber. The following sections will outline the experimental setup used with these microchambers and how it will be used to obtain the high speed images needed for comparison with the macromodel.

5. MEASUREMENT OF FLUID PROPERTIES AND SURFACE TREATMENT PROCEDURE

The two working fluids primarily used in this study are Isopropyl Alcohol and DI-H_2O, the relevant properties for which are shown in Table 1. In testing the macromodel, it was desired to prepare the glass cap and feature etched slides in such a way that the surface would be rendered either hydrophilic or hydrophobic with DI water. To do this an Octadecyltrichlorosilane (OTS) treatment was used. The following sections will discuss the hydrophilic and hydrophobic nature of glass slides before and after OTS treatment. Physical Properties for Isopropyl Alcohol were obtained from Shell Chemicals (Shell Chemicals, 2007). For DI-H_2O the surface tension was found using a Sessile Drop device as discussed in the following sections, while viscosity data was obtained from the online encyclopedia Wikipedia (Wikipedia, 2007).

Table 1: Fluid Properties Used with the Macromodel, taken at 20°C

Fluid	Property	Value	Units
DI-H_2O	Viscosity	1×10^{-3}	Pa.s
	Surface Tension	7×10^{-2}	$N.m^{-1}$
Isopropyl Alcohol	Viscosity	2.43×10^{-3}	Pa.s
	Surface Tension	2×10^{-2}	$N.m^{-1}$

5.1 No Treatment

The working fluids used in this experiment were analyzed for contact angle and surface tension prior to performing the experiments. The contact angle of DI water on plain untreated glass is shown in Figure 12.

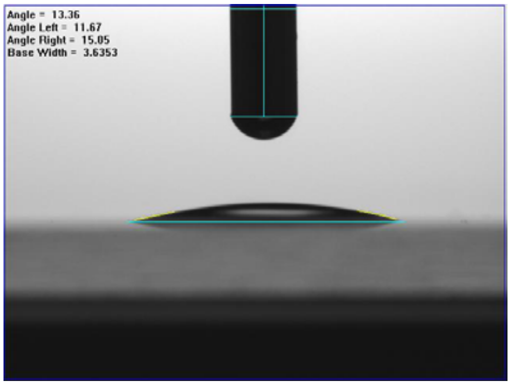


Figure 12: Contact Angle of DI Water on Plain Glass Slide Measured to Be 13.36^{o}

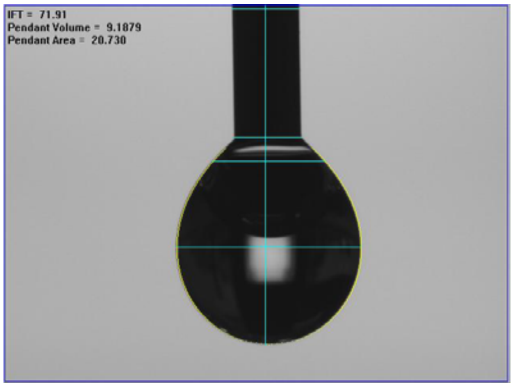


Figure 13: Surface Tension of DI Water Measured to Be 0.07191 N/m

With a contact angle less than 90^{o}, this provided the hydrophilic surface needed for testing. The surface tension of DI water was also measured to be roughly 0.07191 N/m using a pendant drop method (Figure 13).

5.2 OTS Treatment

To obtain a hydrophobic glass surface, the slides were coated with Octadecyltrichlorosilane (OTS). In this procedure OTS, Toulene, and Acetone (for washing) are used. First, the glass slides are thoroughly cleaned with a Piranha solution

(3:1 Sulfuric Acid to Hydrogen Peroxide solution) for 40 minutes and allowed to dry overnight by placing them in a dissicator to minimize surface adsorbed moisture on the substrate. Since OTS hydrolyses in moist environment, this is a necessary and important step. The cleaned glass slides are then immersed in Toulene using a Coplin Staining Jar and two drops of OTS are added. The jar containing the slides is then covered and placed inside a dissicator where it is allowed to sit undisturbed for six hours. Afterwards, the slides are immersed in a fresh pool of Toulene, sonicated for one minute, removed and immersed in Acetone, again sonicated for two minutes, removed and immersed in methanol, sonicated for two minutes, and finally removed and immersed in DI water where they are stored until they are to be used. The result of this treated surface and its effect on DI water contact angle is shown in Figure 14.

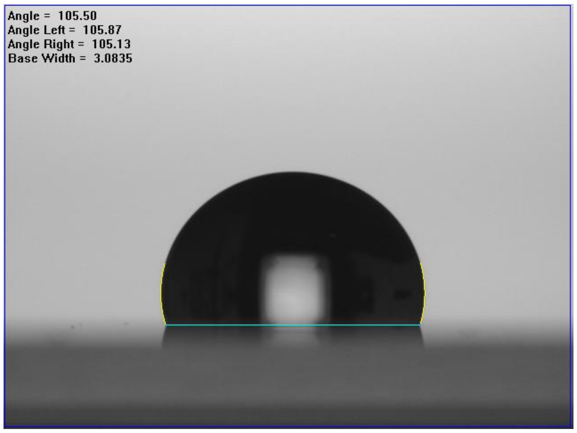


Figure 14: Water Droplet on OTS Treated Glass Surface

The trichlorosilane polar headgroups hydrolyze and convert the Si-Cl bonds to Si-OH (silanol) groups. The silanol groups, which are strongly attached to the oxidized hydrophilic surface, condense with the OH^- (hydroxyl) groups on the surface to form Si-O-Si (siloxane) links. The result is a monolayer in which the molecules are connected to each other on the surface by strong chemical bonds. This leaves a hydrophobic glass surface with water while also maintaining the same desired optical qualities of the original glass. Slides for which this surface treatment was applied show a measured contact angle of 105^o with DI water. Coating with OTS provides the hydrophobic condition needed for the contact angle parameters of θ_1, θ_2, and θ_3 as described by the macromodel. Chambers treated with OTS are here-to-forth referred to as OTS microchambers.

6. EXPERIMENTAL SETUP/PROCEDURE

The experimental setup was designed to provide optical access to the filling liquid and tracking of meniscus location. High speed digital image recording of the microchamber filling was obtained using this apparatus. An illustration of the basic equipment used in the experiments is shown in Figure 15. A 3D SolidWorks ® representation of the experimental setup is shown in Figure 16. Actual pictures of the experimental setup is shown in Figure 17 and a close up of a microchamber during testing is shown in Figure 18. Table 2 itemizes the various instruments used in the experiments.

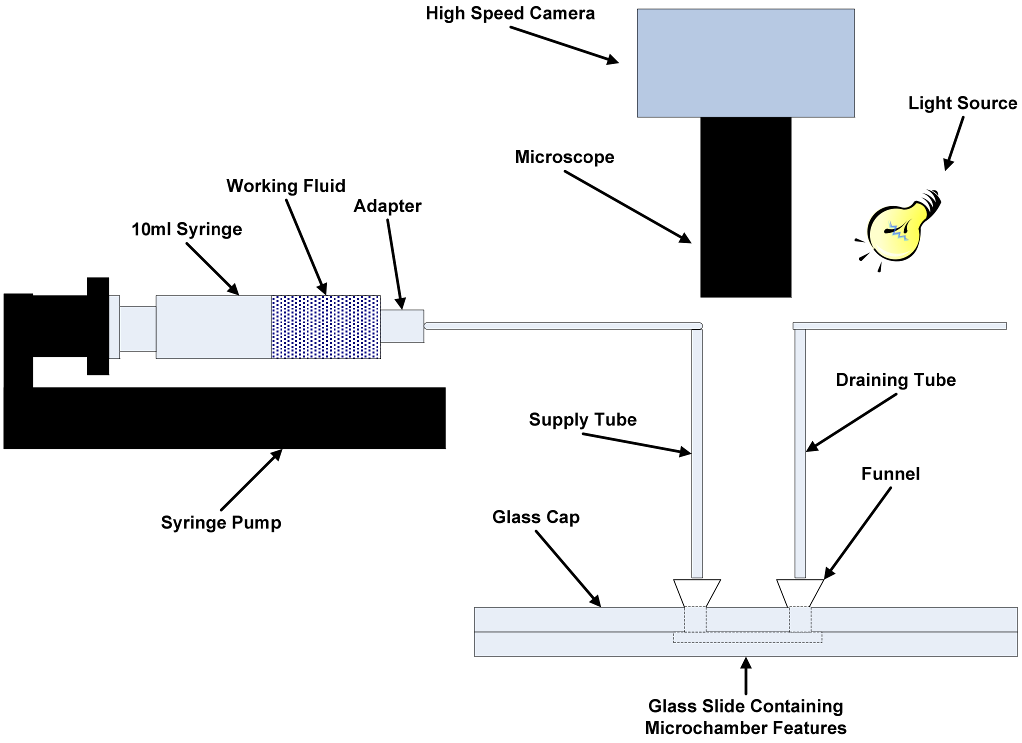


Figure 15: Experimental Setup

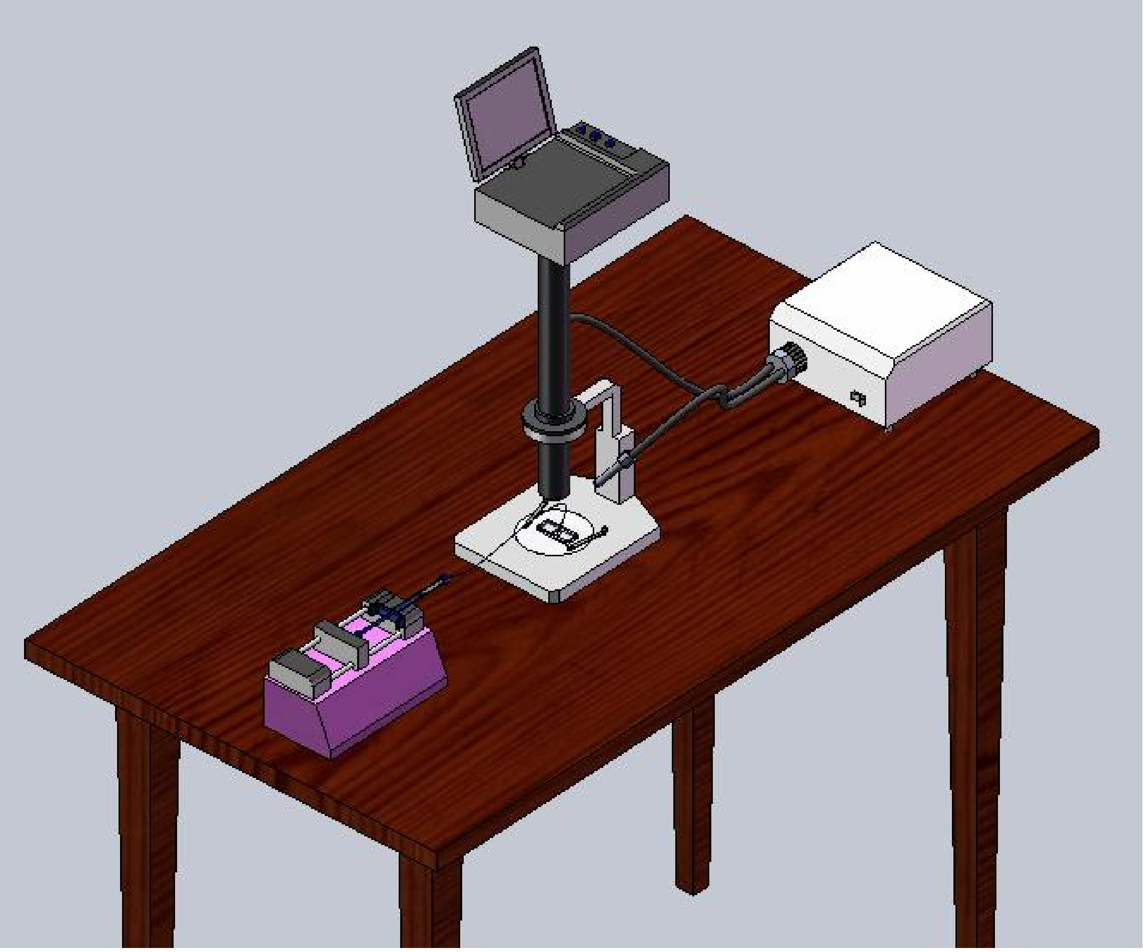


Figure 16: 3D Solidworks ® Model of Experimental Setup (Courtesy of Rodolfo DeLeon, Undergraduate Student in Mechanical Engineering, Texas A&M University)

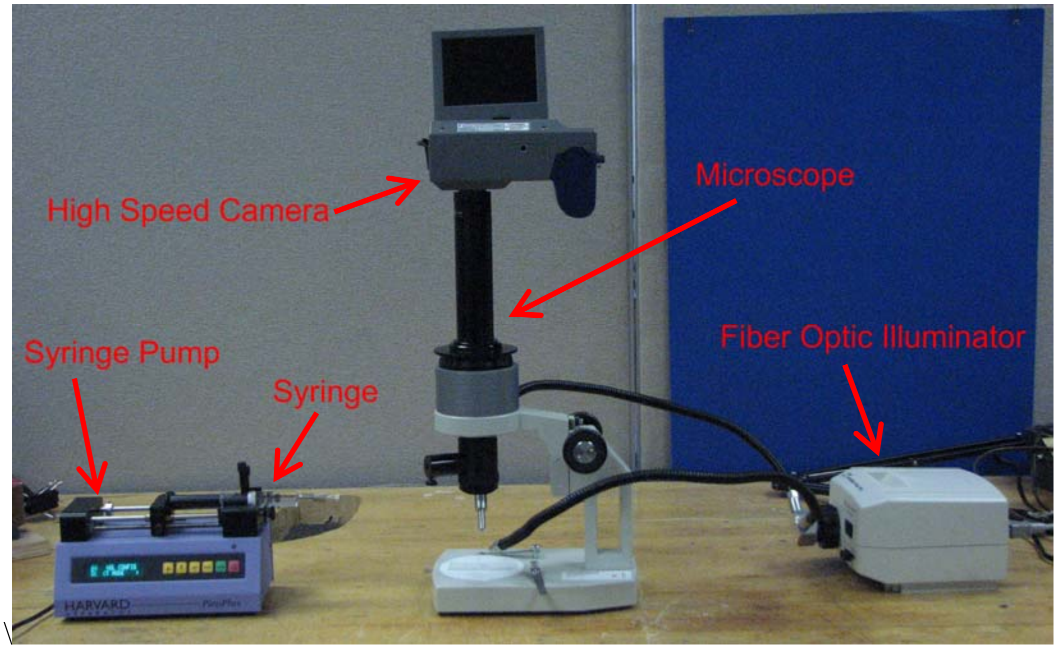


Figure 17: Actual Experimental Setup

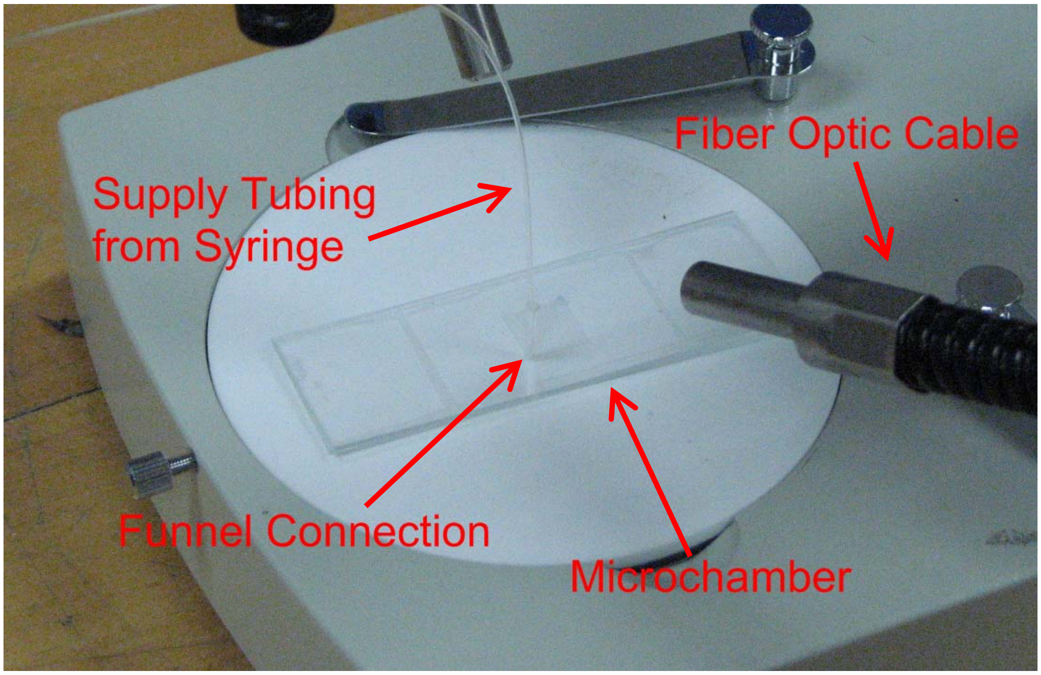


Figure 18: Close up of Microchamber during Experiments under the Microscope

Table 2: List of Equipment

Equipment	Manufacturer	Model	Quantity
Microscope	Navitar	1-6010	1
High Speed Camera	Fastec Imaging	TSHRMS	1
Syringe Pump	Harvard Apparatus	PicoPlus	1
Teflon Tubing	Upchurch Scientific	PM-1073	2
Fiber Optic Illuminator	Mille Luce	M1000	1
Adapter	Upchurch Scientific	P-659	1
10ml Syringe	Hamilton Company	1010TTL	1
Funnel	VWR International	53509-140	2

As part of the setup, a PicoPlus syringe pump (Harvard Aparatus) was used to provide a constant flow rate into the microchamber. With this pump, a 1010TTL 10ml luer tipped Hamilton Company Syringe was loaded and primed with the working fluid. Attached to the luer is an adapter provided by Upchurch Scientific which allowed Teflon tubing to be attached to the syringe. The other end of the tubing was connected to the funnels as explained earlier. To monitor the filling experiments, a Navitar microscope with a 12× magnification lens was used. A Fastec Imaging high speed camera was attached to the microscope for high speed digital image acquisition. Illumination of the microchamber during filling was achieved using a fiber optic illuminator. The Teflon tubing was flushed with the working fluid before starting an experiment to ensure that trapped air bubbles were eliminated from the supply line. In the case of DI-H_2O, the liquid was degassed by boiling for 20 minutes and sonicating for an additional 20 minutes. This setup allowed for high speed time lapsed digital images of the microchamber to be obtained during filling using the high speed camera. These images could then be analyzed and compared against values predicted by the macromodel, as will be outlined in the following sections.

7. COMPARISON WITH MACROMODEL

Several tests were performed using microchambers of various depths by pumping DI Water and Isopropyl Alcohol. Isopropyl Alcohol proved to be the most convenient working fluid of choice for performing the experiments due to its low value of surface tension compared to that of water. Customized software was written in Microsoft Visual Basic ® .NET 2005 to assist in the measurement of the meniscus points l_{s1}, l_{s2}, and l_m as described in the model. The software program uses a calibration procedure to select the number of pixels for a user specified distance. The code determines the conversion constant necessary to translate a specified pixel location with respect to the calibrated distance. Using this software, each filling video was discretized into individual component frames for analysis. The time step between images is dependent on the frame rate of capture set on the camera, typically 30 or 60 fps (frames per second). For each microchamber and flow rate to be analyzed, the frame images were loaded into the measurement program and the location of the three meniscus points were determined.

7.1 Error Analyses

As mentioned before, there will be an error associated with both etch undercutting arising from the variability in the microfabrication processes used and from the measurement program used to determine the meniscus points. This error is defined to be

$$\Delta l_{s1} = \pm \frac{\left(L_{e1} - L_{e2}\right) \cdot p_{ls1}}{L_1} \cdot l_{s1} \tag{6}$$

$$\Delta l_{s2} = \pm \frac{(L_{e1} - L_{e2}) \cdot p_{ls2}}{L_2} \cdot l_{s2} \tag{7}$$

$$\Delta l_m = \pm \sqrt{\left(\frac{\sigma_{lm}}{\bar{l}_m}\right)^2 + \left(\frac{(L_{e1} - L_{e2}) \cdot p_{ls1}}{L_1}\right)^2} \tag{8}$$

where Δl_{s1} and Δl_{s2} are the measurement errors in the L_1 and L_2 direction respectively while Δl_m is the measurement error for the central portion of the meniscus. In Equation 8, $\bar{l}_m$ is an average of three measured points along the central part of the meniscus and σ_{lm} is the standard deviation among those points. Again, L_{e1} and L_{e2} are the lengths associated with undercutting as discussed in previous sections. Here, the variable *p* represents the calibration constant for image analyses arising from the number of pixels from the image required to fill the distance (L_{e1} – L_{e2}) in the l_{s1} and l_{s2} directions, as denoted by the subscript. All points were then exported to a Microsoft Excel ® spreadsheet and plotted with respect to time while the same was done for the predicted points from the macromodel. Values for the region of wall influence, *w*, were varied iteratively until the error between the experimental and macromodel data was minimized. The following sections outline the results of this data analysis.

7.2 Isopropyl Alcohol Filling Experiments

As previously stated, the low value of surface tension of Isopropyl Alcohol enabled a more convenient experimental procedure and resulted in less complications during the experiments. Initially three square shaped microchambers with 1 cm sides and with depths of 33.5 μm, 41 μm, and 49 μm were used for the filling experiments.

Results obtained by using flow rates of 100, 200, and 300 μl/min are reported here. In each of the following graphs, the solid lines represent the predicted meniscus locations obtained from simulations performed using the macromodel, while the plotted points represent the measured distances from the filling experiments. For compactness, graphs for a microchamber with dimensions $L_1 \approx 1$ cm, $L_2 \approx 1$ cm, and h ≈ 33.5 μm are shown here. The remainder graphs are shown in Appendix A, B and C.

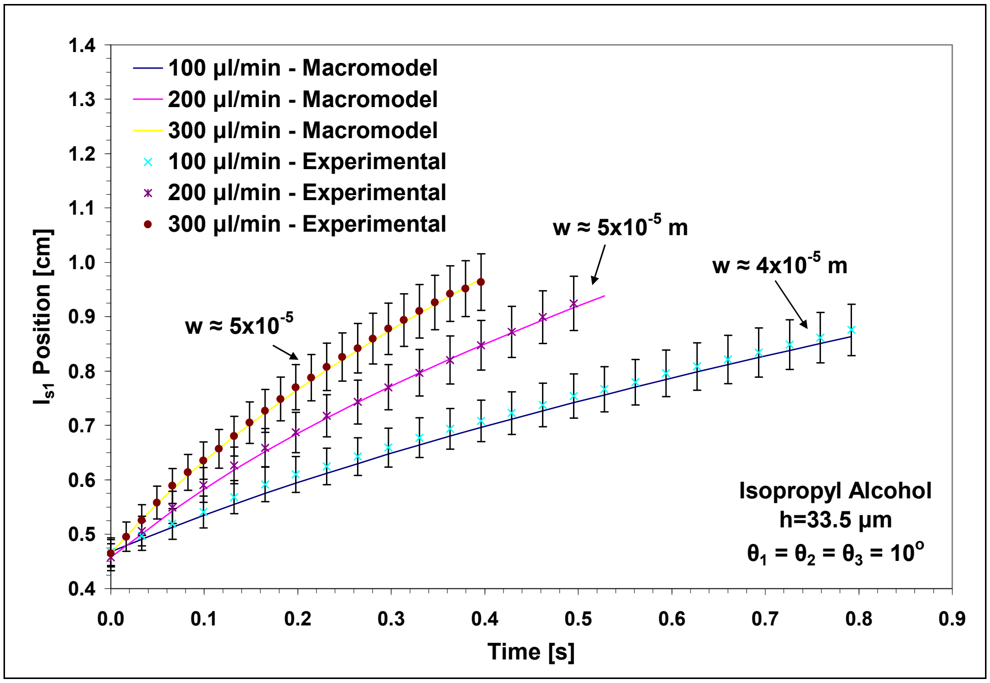


Figure 19: l_{s1} Meniscus Positions for Isopropyl Alcohol Filling Experiment with h=33.5 μm

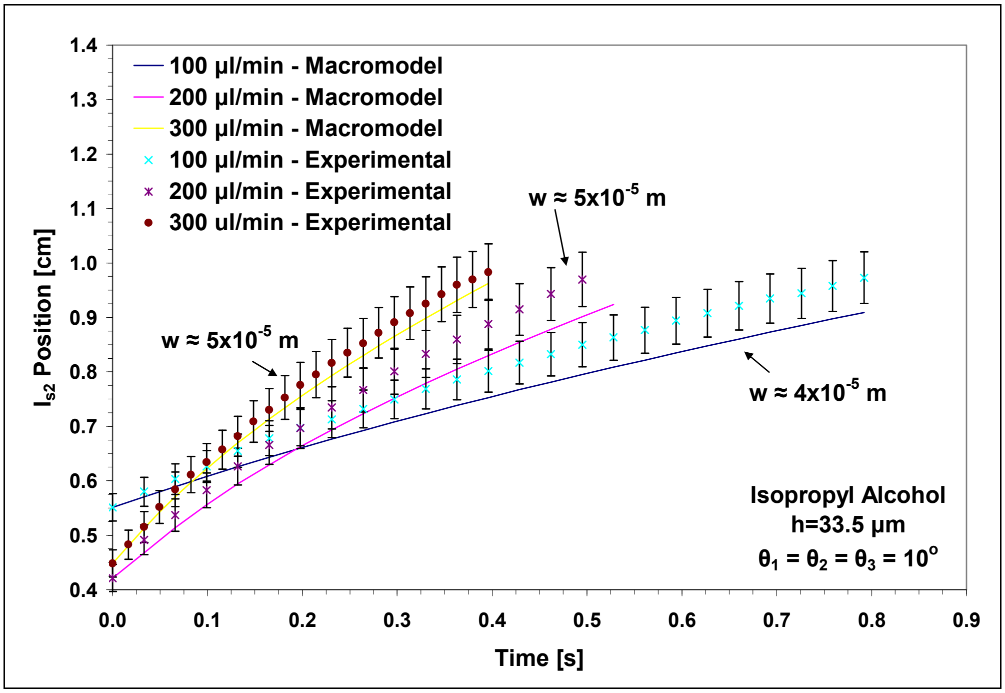


Figure 20: l_{s2} Meniscus Positions for Isopropyl Alcohol Filling Experiment with h=33.5 µm

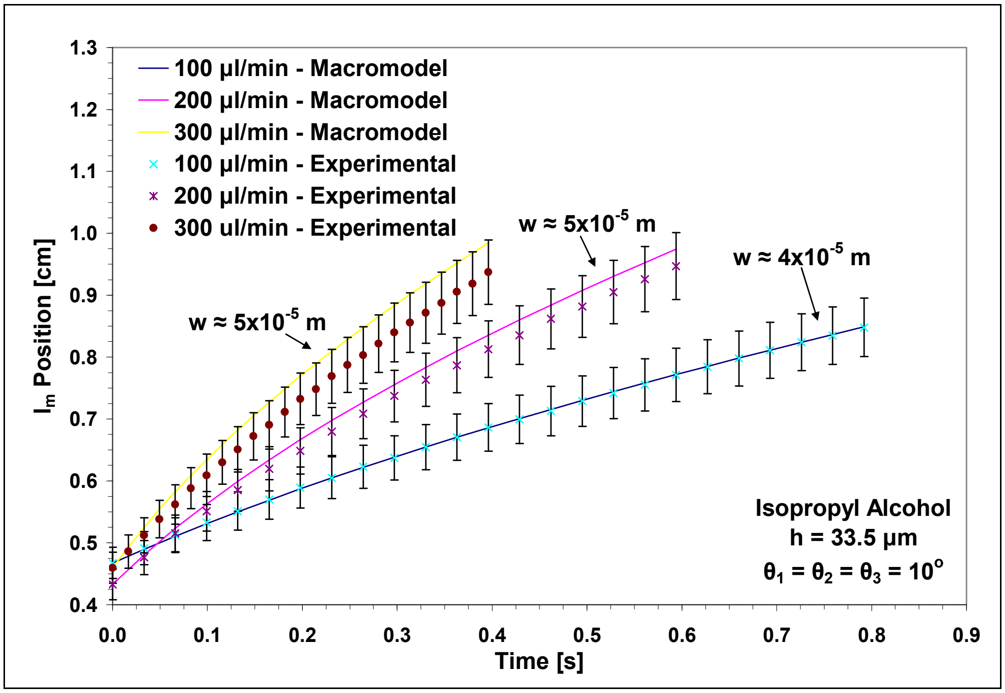


Figure 21: l_m Meniscus Positions for Isopropyl Alcohol Filling Experiment with h=33.5 µm

As shown in Figure 19 through Figure 21, the macromodel predictions are in very good agreement with experimental data and are within the bounds of the experimental uncertainties. Overlapping of data on the graphs is due to the uncertainty in locating the initial position of l_{s1}, l_{s2}, and l_m. The source of the uncertainties could vary depending on the experimental uncertainties and non-symmetric placement of the inlet flow port. Figure 22 shows calibrated values for the region of wall influence for several different microchambers of varying depths and fluid flow rates for isopropyl alcohol. For each of the microchambers represented in the graph, dimensions are $L_1 \approx 1$ cm, and $L_2 \approx 1$ cm.

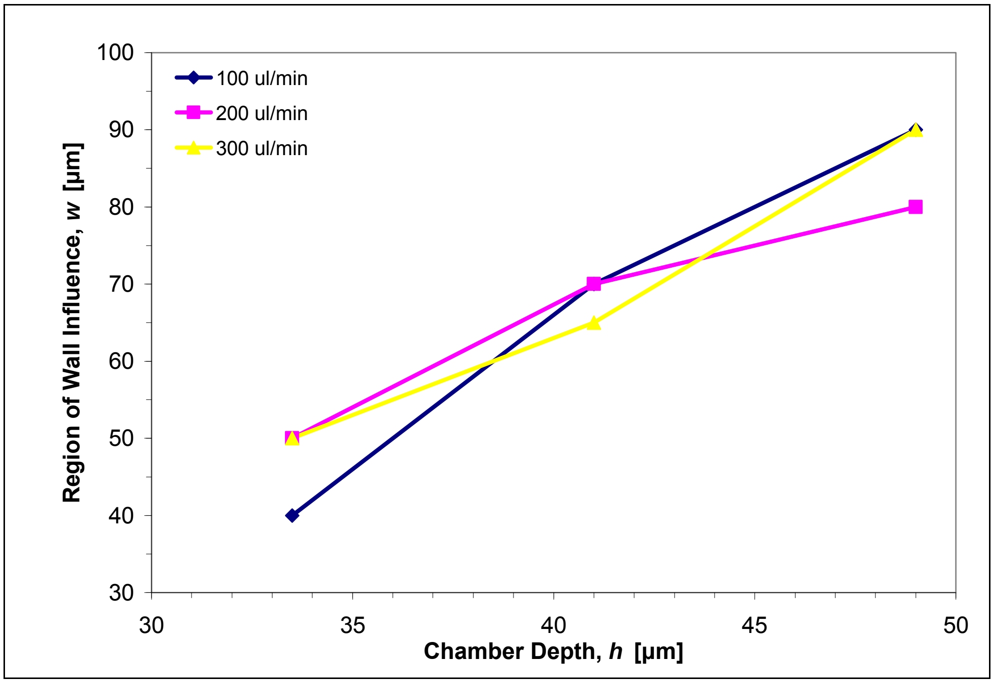


Figure 22: Dependence of *w* with *h* for Isopropyl Alcohol

As can be seen, the value of *w* is weakly sensitive to the flow rate (does not vary widely with flow rate for each microchamber) and is a strong function of chamber depth, *h*. The following sections will discuss how the size of this region responds to changes in fluid contact angle.

7.3 Effect of Contact Angle on Region of Wall Influence

To study the effect of fluid contact angle on the region of wall influence, a filling test was conducted using DI-H_2O as the working fluid on a microchamber with dimensions $L_1 \approx 1$ cm, $L_2 \approx 1$ cm, and h ≈ 31 µm. After testing, the microchamber was disassembled; the OTS treatment was applied and then reassembled. Figure 23 shows the comparison between predictions from the macromodel and experimental results for the meniscus positions l_{s1}, l_{s2}, and l_m before the OTS treatment was applied.

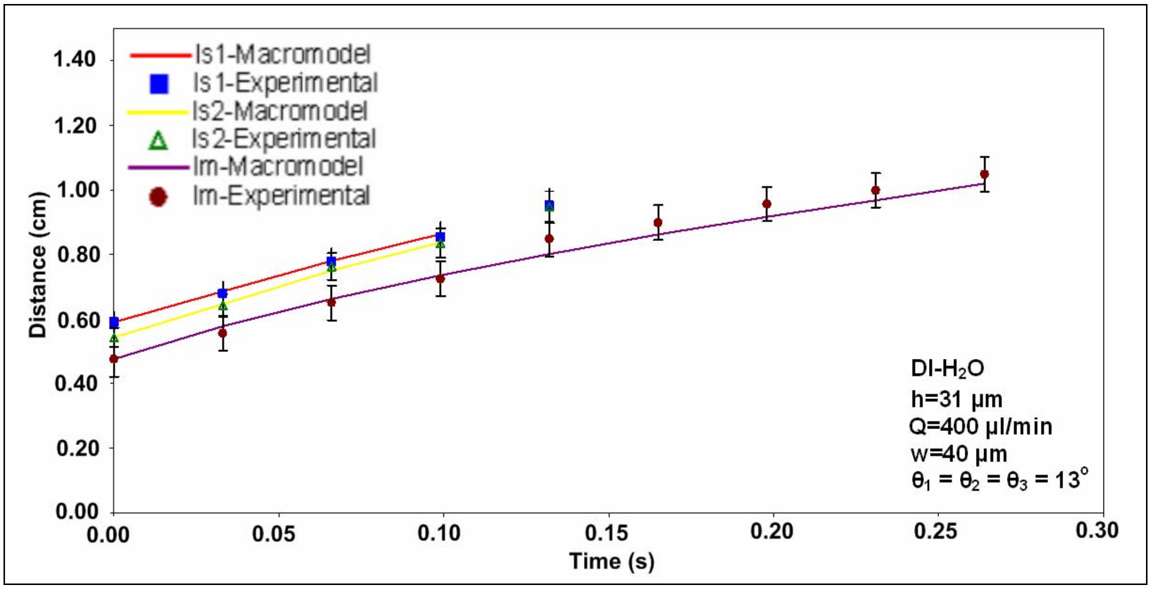


Figure 23: Meniscus Positions for DI-H_2O Filling Experiment before OTS Treatment

For this case, the region of wall influence was found to be around 40 µm. As can be seen from Figure 23, the macromodel still provides good correlation to experimental results, although not quite as accurate as the Isopropyl Alcohol cases. This could be due to the fact that DI-H_2O has a much higher value of surface tension than that of Isopropyl Alcohol, making its effect on filling much more susceptible to possible unevenness in microchamber depth due to irregular etching. Figure 24 shows the results from filling experiments after surface treatment using OTS.

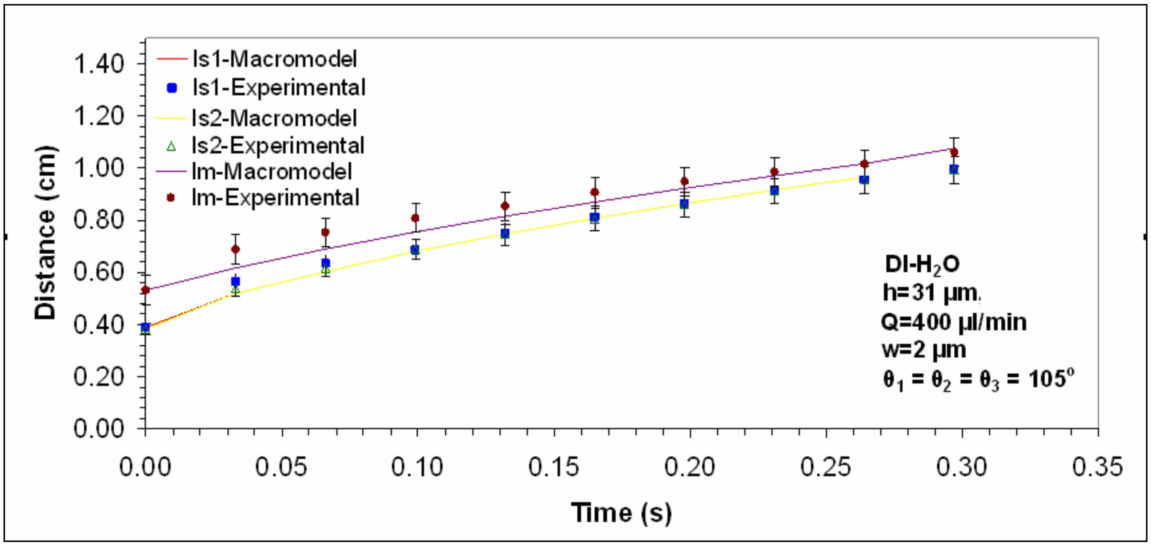


Figure 24: Meniscus Positions for DI-H_2O Filling Experiment after OTS Treatment

For this case, the region of wall influence was found to be around 2 µm, a dramatic decrease from the hydrophilic case. The figure shows that at this flow rate the capillary component is lesser than the inertial/viscous component for the pressure drop. Consequently, the bulk of the flow is along the center of the chamber than the walls. It is suggested that the observed change in the region of wall influence could only happen if the viscosity of the working fluid changes near the wall in response to a change in contact angle.

7.4 Meniscus Shapes and Bubble Formation

As previously mentioned, another important feature of the macromodel is whether it accurately predicts the formation of bubbles in the opposite corners from the flow inlet region of the microchamber. To do this, the calibrated regions of wall influence from the previous section will be used in the macromodel and the plotted meniscus shapes will be compared with those from experiments. Figure 25 shows the predicted shapes from the macromodel. Time difference images were extracted from the high speed camera footage and enhanced using Paint.NET ® which was also used to superimpose each image to produce Figure 26. Black spots observed on the images are due to dust on the microscope or camera lens and do not represent actual contaminates within the microchamber itself.

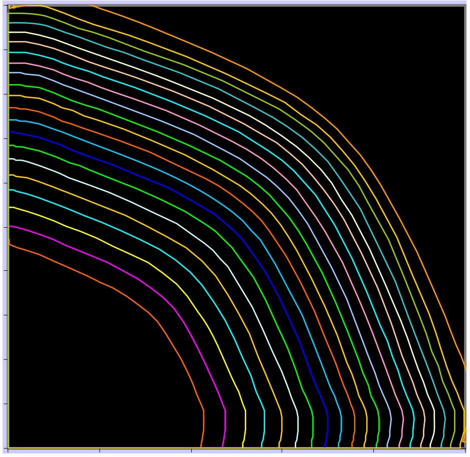


Figure 25: Meniscus Shapes Predicted by the Macromodel for h=33.5 µm and Q=200 µl/min Using Isopropyl Alcohol

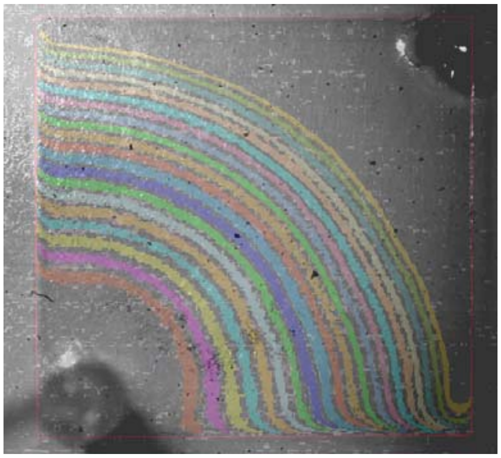


Figure 26: Meniscus Shapes from Experiments for h=33.5 µm and Q=200 µl/min Using Isopropyl Alcohol

The figures show that the predicted meniscus shapes are qualitatively in good agreement with meniscus shapes observed in the experiments. The macromodel also does not predict the formation of a bubble in the upper left or lower right corner of the microchamber, as was observed with the experiment. Similarly, the Macromodels ability to predict bubble formation in a hydrophobic microchamber, with DI-H_2O as the working fluid, was compared against experiments. In Figure 27 and Figure 28, DI-H_2O is pumped through a microchamber before undergoing the OTS treatment.

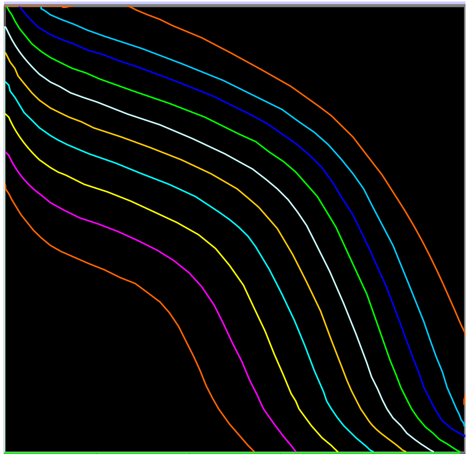


Figure 27: Meniscus Shapes Predicted by the Macromodel for a Hydrophilic Microchamber Using DI-H_2O, h=31 µm, and Q=400 µl/min

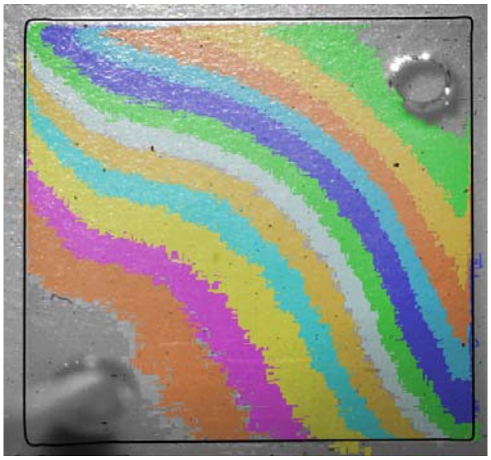


Figure 28: Meniscus Shapes from Experiments for a Hydrophobic Microchamber Using DI-H_2O , h=31 µm, and Q=400 µl/min

Comparison of the macromodel and experimental results for this case show a very good agreement for the meniscus shapes. At this flow rate the capillary component is greater than the inertial/viscous component for the pressure drop. As a result no bubble is trapped along the corners of the microchamber, which is in agreement with the predictions from the macromodel. This same microchamber was then subjected to surface treatment using OTS, rendering the bottom, top, and side walls hydrophobic with DI-H_2O having contact angles of $\theta_1 = \theta_2 = \theta_3 = 105^{\circ}$. Figure 29 and Figure 30 again illustrate a comparison between the macromodel and experimental results, respectively, for the hydrophobic case. The void area in the image is due to reflected light from the glass substrate.

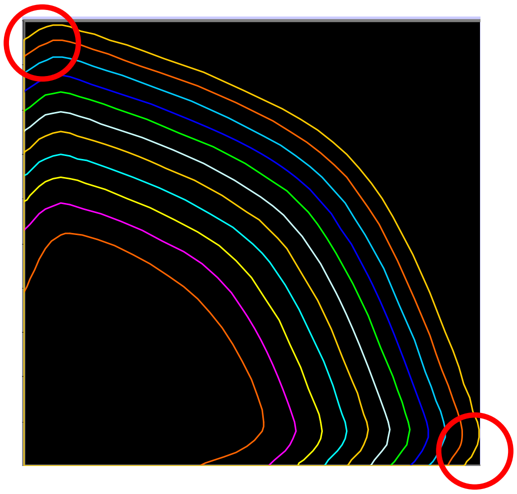


Figure 29: Meniscus Shapes Predicted by the Macromodel for a Hydrophobic Microchamber Using DI-H_2O, h=31 µm, and Q=300 µl/min

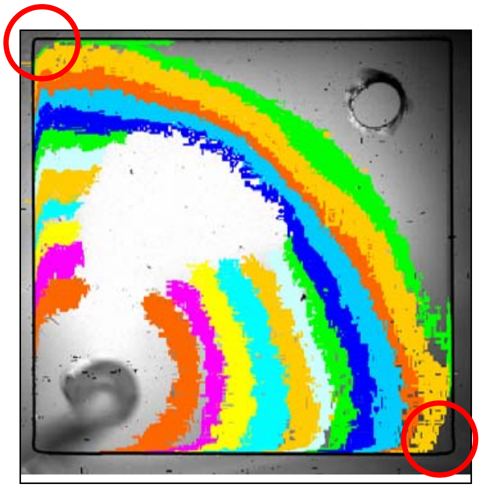


Figure 30: Meniscus Shapes from Experiments for a Hydrophobic Microchamber Using DI-H_2O , h=33.5 µm, and Q=300 µl/min

As explained earlier, these figures show that at this flow rate the capillary component is lesser than the inertial/viscous component for the pressure drop. Consequently, the bulk of the flow is along the center of the chamber than the walls. As a result bubbles are trapped along the corners of the microchamber. The presence of bubbles is highlighted by the red circles in each of the images and is confined to the opposite corners of the inlet flow port. Any spottiness in the experimental images is due to image processing done to help illuminate the meniscus positions. Again, the macromodel affords good accuracy in predicting meniscus shapes for the hydrophobic case, while also accurately predicting the formation of bubble entrapment. As an attempt to observe the effects of hydrophobic side walls ($\theta_3 > 90^o$) while leaving the top and bottom of the chamber hydrophilic ($\theta_1 = \theta_2 < 90^o$), microchambers were constructed from double sided tape, as previous mentioned. Figure 31 shows a typical contact angle measurement using DI-H_2O on the non-adhesive sides of the double sided tape.

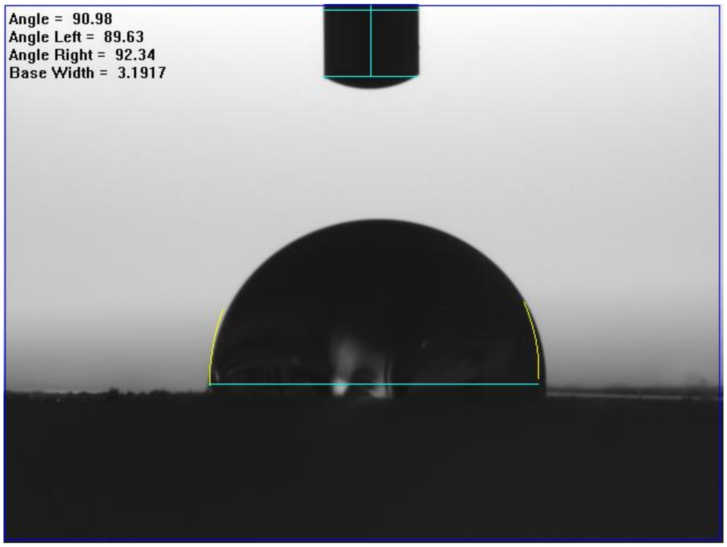


Figure 31: Contact Angle of DI-H_2O on the Non-Adhesive Side of Double Sided Tape

As a result of cutting the tape, the edges of the microchamber end up jagged, making measurements of the meniscus points difficult and prone to errors for comparison to the macromodel. For this reason, only the meniscus shapes are compared in Figure 32 and Figure 33.

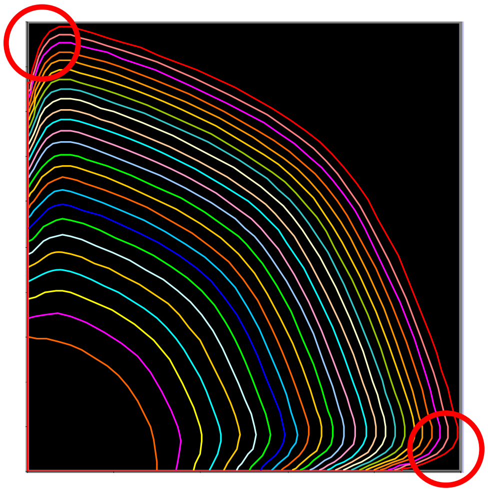


Figure 32: Meniscus Shapes Predicted by the Macromodel for Hydrophobic Sidewalls Using DI-H_2O, h=50 µm, and Q=400 µl/min

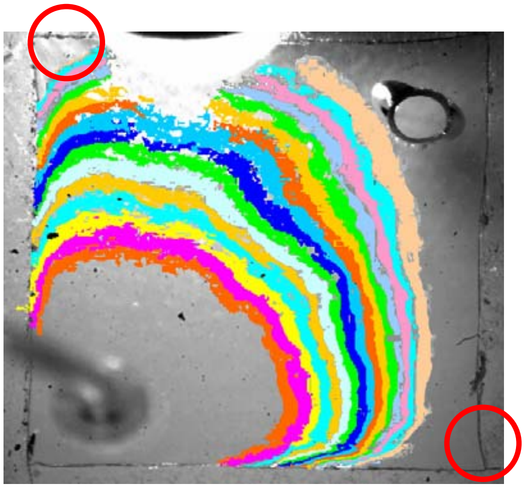


Figure 33: Meniscus Shapes from Experiments for Hydrophobic Sidewalls Using DI-H_2O , h=50 µm, and Q=400 µl/min

Since the meniscus points could not be easily measured for this case, the region of wall influence was taken to be equal to the chamber depth of 50µm. It can be seen from the comparison of meniscus shapes that the macromodel still affords reasonable accuracy in predicting bubble formation within a microchamber of this configuration.

7.5 Limitations of the Macromodel

Microchambers rectangular in shape, where $L_1 < L_2$, were also tested against the macromodel. However, these did not perform as well as those of equal distances of L_1 and L_2. Figure 34 shows the experimental meniscus shapes for a rectangular microchamber with dimensions $L_1 \approx 0.5$ cm, $L_2 \approx 1$ cm, and h ≈ 52 µm filling with Isopropyl Alcohol at 200 µl/min. Again, time difference images were extracted from the high speed camera footage and edited using Paint.NET ® to combine each image into one, as seen in Figure 34.

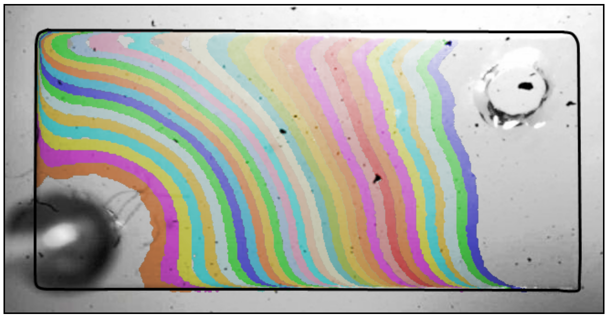


Figure 34: Experimental Meniscus Shapes for Rectangular Microchambers with Δt=0.017s, h=52μm, and Q=200μl/min

Inputting the geometric and flow property data for this microchamber into the macromodel yields mixed results. Figure 35 shows predicted meniscus shapes from the macromodel for the flow conditions of the microchamber in Figure 34. It can be seen that the macromodel shows very good agreement with the experimental data.

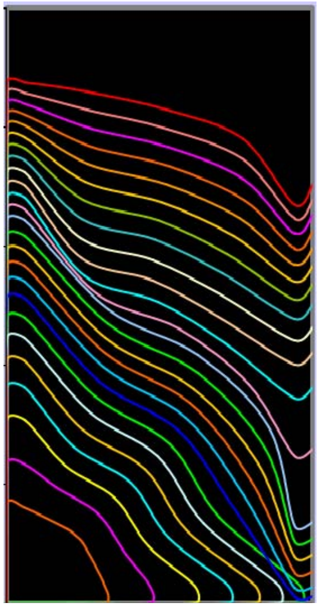


Figure 35: Predicted Meniscus Shapes for a Rectangular Microchamber with Δt=0.017s, h=52μm, and Q=200μl/min

Figure 36, Figure 37 and Figure 38 show a comparison between the macromodel and experimental filling data for the chamber again depicted in Figure 34.

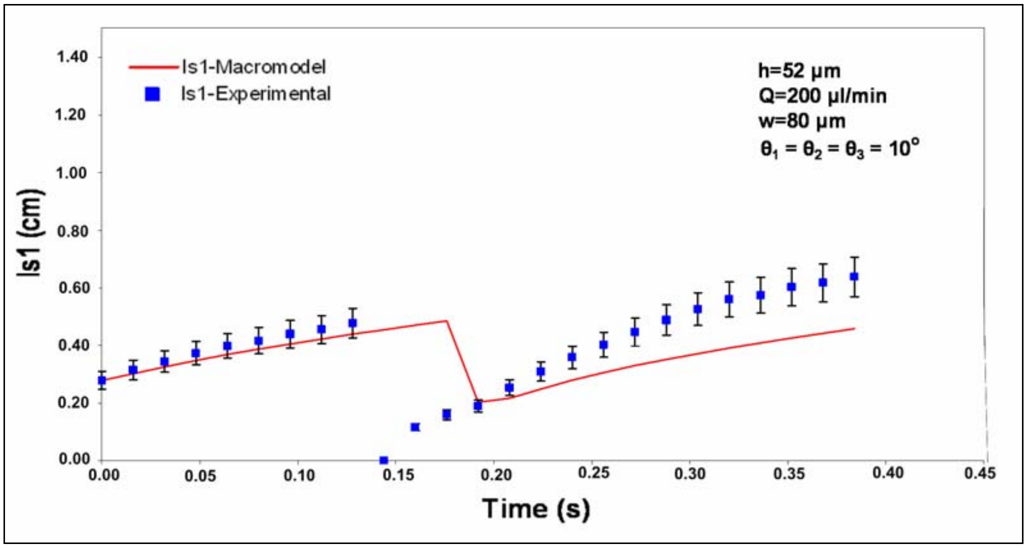


Figure 36: l_{s1} vs. Time Comparison of Macromodel and Experimental Points for a Rectangular Microchamber

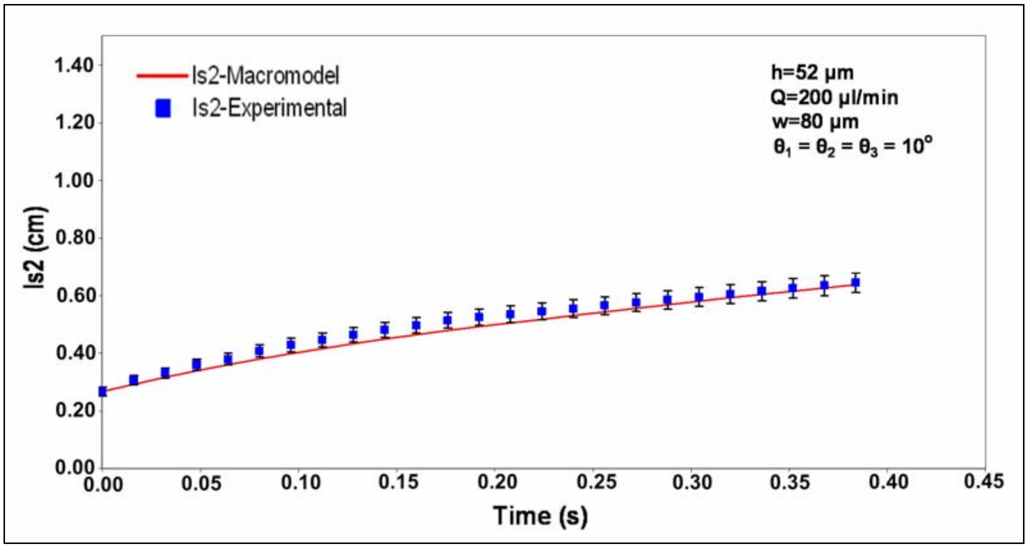


Figure 37: l_{s2} vs. Time Comparison of Macromodel and Experimental Points for a Rectangular Microchamber, h=52µm

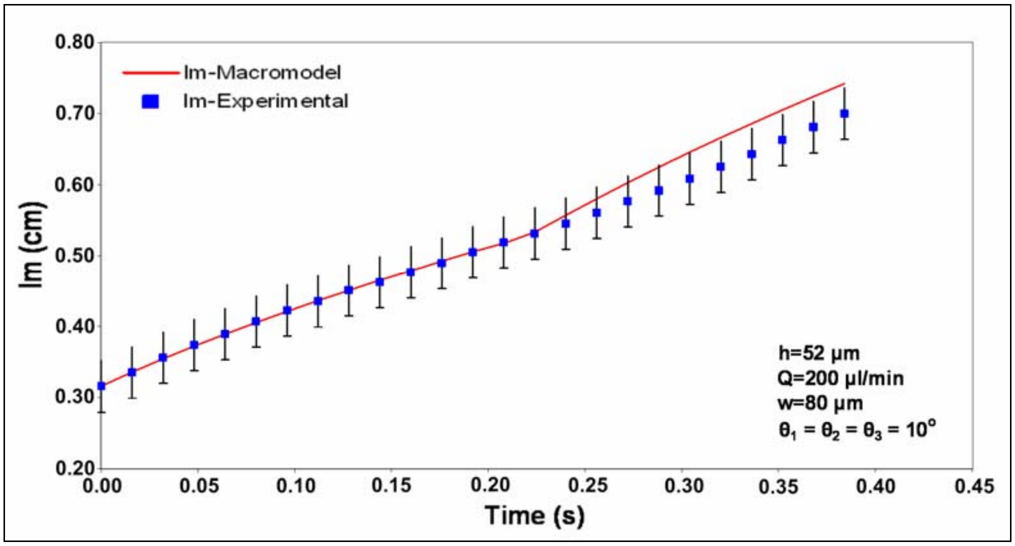


Figure 38: l_m vs. Time Comparison of Macromodel and Experimental Points for a Rectangular Microchamber, h=52µm

The macromodel does not accurately predict the filling behavior once the flow reaches the opposite corners from the flow inlet port, however still affords reasonable accuracy up to that point. Based on these results, the macromodel could afford from further development in predicting the effects on the fluid flow when the meniscus has reached the opposite corners from the inlet flow port, however still reliably fulfills one of its primary functions; the prediction of bubble entrapment during filling.

8. CONCLUSION AND FUTURE DIRECTION

It has been shown that the macromodel proposed by Banerjee (2005) accurately predicts capillary driven flow behavior inside microchambers while bypassing otherwise computationally intensive methods to model such flow behavior. It has been shown that the region of wall influence increases with microchamber depth, while remaining relatively insensitive to fluid flow rate. This region also becomes a function of fluid contact angle, and decreases in length with hydrophobic surfaces. This implies that the viscosity of the working fluid changes in response to this change in contact angle. Hence variation of fluid geometries in small confined spaces can be different compared to the macroscopic situations and becomes a significant factor in capillary filling behavior, while often neglected in macroscale applications. The macromodel predictions for resulting meniscus shapes as well as bubble entrapment are found to be in good agreement with experimental. Limitations of the macromodel were observed for predicting meniscus shapes and locations when the meniscus reaches the opposite corners of the microchamber from the inlet flow port.

This study has demonstrated that further calibration and development of the macromodel is required. Precision of the experiments could be enhanced by obtaining a more uniform etch when using Hydroflouric Acid with glass. Microchambers could be constructed of alternate materials or utilize other manufacturing techniques which may yield better controlled tolerances of chamber depth. Further investigation into constructing microchambers utilizing SU-8 photoresist as an epoxy to essentially “glue” two glass substrates together might also be warranted. This would have the advantage of hydrophobic side walls while leaving the top and bottom of the microchamber hydrophilic. It may prove beneficial to also pump the working fluids at even higher flow rates to explore if this parameter might influence bubble formation.

REFERENCES

Abe K, Kazama S, Koro K (2007) A boundary element approach for topology optimization problem using the level set method. Commun. Numer. Meth. Engng. 23:405-416

Banerjee D (2005) Experimental Validation of Macromodels for Simulating Capillary Driven Multiphase Flows used for Microchamber Filling. The ASME/Pacific Rim Technical Conference and Exhibition on Integration and Packaging of MEMS, NEMS and Electronic Systems, INTERPACK2005-73412

Banerjee D, Amro NA, Fragala J (2005) Optimization of microfluidic ink-delivery apparatus for Dip Pen Nanolithography™. Journal of Microlithography, Microfabrication and Microsystems ("JM3") SPIE publication

Bourouina T, Grandchamp JP (1996) Modeling micropumps with electrical equivalent networks. J. Micromech. Microeng. 6:398–404

Carrica PM, Wilson RV, Stern F (2007) An unsteady single-phase level set method for viscous free surface flows. Int. J. Numer. Meth. Fluids. 53:229-256

Chatterjee A, Aluru NR (2005) Combined circuit/device modeling and simulation of integrated microfluidic systems. Journal of Microelectromechanical Systems. 14:1

Churaev, NV, Sobolev VD, Zorin ZM. (1971) Special discussion on thin liquid films and boundary layers, Wiley, New York, p. 213–220

Estes R (2005) Semiconductor packaging technologies advance DNA analysis systems. IVD Technology Magazine. Retrieved 03:08, September 30, 2007, from <http://www.devicelink.com/ivdt/archive/00/07/006.html>

Fang Q, Jia Z, Fang Z (2004) Bonding of glass microfluidic chips at room temperatures. Anal. Chem. 76:5597-5602

Grob S, Reichelt V, Reusken A (2006) A finite element based level set method for two-phase incompressible flows. Comput. Visual Sci. 9:239-257

Hess S, Loose W (1989) Rheology of dense fluids via nonequlibrium molecular hydrodynamics: Shear thinning and ordering transition. Rheologica Acta. 48:91–101

Human Genome Project Information (2007) The Human Genome Program of the U.S. Department of Energy Office of Science. Retrieved 01:08, September 30, 2007, from <http://www.ornl.gov/sci/techresources/Human_Genome/home.shtml>

Jensen MJ (2002) Bubbles in microfluidics. Master's Thesis, Technical University of Denmark, Lyngby

Jousee F, Lian G, Janes R, Melrose J (2005) Compact model for multi-phase liquid-liquid flows in micro-fluidic devices. The Royal Society of Chemistry, Lab Chip. 5:646-656

Lewis DH, Janson SW, Cohen RB, Antonsson EK (2000) Digital MicroPropulsion. Sensors and Actuators A 80:143-154

Liu X, Yuanhua L, Glimm J, Li XL (2007) A front tracking algorithm for limited mass diffusion. Journal of Computational Physics. 222:644-653

Magargle, R, Hoburg JF, Mukherjee T (2004) Microfluidic injector models based on neural networks. Tech. Proc. NSTI Nanotechnology Conf. and Trade Show (Nanotech), Anaheim, CA, p. 616–619

Meinhart C, Tretheway D (2001) Apparent fluid slip at hydrophobic microchannel walls. Physics of Fluids. 14(3):L9-L12

Menard T, Tanguy S, Berlemont A (2007) Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet. International Journal of Multiphase Flow. 33:510-524

Mikulchenko A, Rasmussen A, Mayaram K (2000) A neural network based macromodel for microflow sensors. Tech. Proc. Int. Conf. Modeling and Simulation Microsystems (MSM), San Diego, CA, p. 540–543

Morel C (2007) Modeling approaches for strongly non-homogenous two-phase flows. Nuclear Engineering and Design. 237:1107-1127

Morris CJ, Forster FK (2004) Oscillatory flow in microchannels. Experiments in Fluids. 36:928-937

Nanogen, Inc. (2007) San Diego, California 92121. Retrieved 02:15, September 30, 2007, from <http://www.nanogen.com/aboutus>

Orieux S, Rossi C, Esteve D (2002) Compact model based on a lumped parameter approach for the prediction of solid propellant micro-rocket performance. Sensor and Actuators A 101:383-391

Qiao R, Aluru NR (2002) A compact model for electroosmotic flow in microfluidic device. Journal of Micromechanics and Microengineering. 12:625-635

Quere D (2001) Falling Slugs. Journal of Colloid and Interface Science. 243:262-264

Shell Chemicals, Chemical data sheet for isopropyl alcohol Retrieved 02:08, September 30, 2007, from <http://www.shellchemicals.com/chemicals/pdf/solvents/chemical/alcohols/ipa_na_216.pdf?section=our_products>

Tas NR, Haneveld J, Jansen HV, Elwenspoek M, Berg AV (2004) Capillary filling speed of water in nanochannels. APL. 85:3274-3276

Tsai N, Sue C (2006) SU-8 based continuous-flow RT-PCR bio-chips under high-precision temperature control. Biosensors & Bioelectronics 22:313-317

Turowski M, Chen Z, Przekwas A (2001) Automated generation of compact models for fluidic microsystems. Analog Integrated Circuits and Signal Processing. 29:27-36

Wang C (2004) The principle of micro thermal analysis using atomic force microscope. Thermochimica Acta. 423:89-97

Wang Y, Lin Q, Mukerjee T (2004a) A model for Joule heating-induced dispersion in microchip electrophoresis. Lab Chip. 4(6):625–631

Wang Y, Lin Q, Mukherjee T (2004b) System-oriented dispersion models of general-shaped electrophoresis microchannels. Lab Chip. 4(5):453–463

Water (molecule). In *Wikipedia, The Free Encyclopedia*

Retrieved 01:08, September 30, 2007, from <http://en.wikipedia.org/w/index.php?title=Water_%28molecule%29&oldid=160607576>

Weber MW, Shandas R (2007) Computational fluid dynamics analysis of microbubble formation in microfluidic flow-focusing devices. Microfluid Nanofluid. 3:195-206

Witteveen JAS, Koren B, Bakker PG (2007) An improved front tracking method for the Euler equations. Journal of Computational Physics. 224:712-728

Xu Z, Glimm J, Zhang Y, Liu X (2007) A multiscale front tracking method for compressible free surface flows. Chemical Engineering Science. 62:3538-3548

APPENDIX A

PLOTS OF l_{s1} FOR MICROCHAMBERS OF VARIOUS DEPTHS AND FLOW CONDITIONS

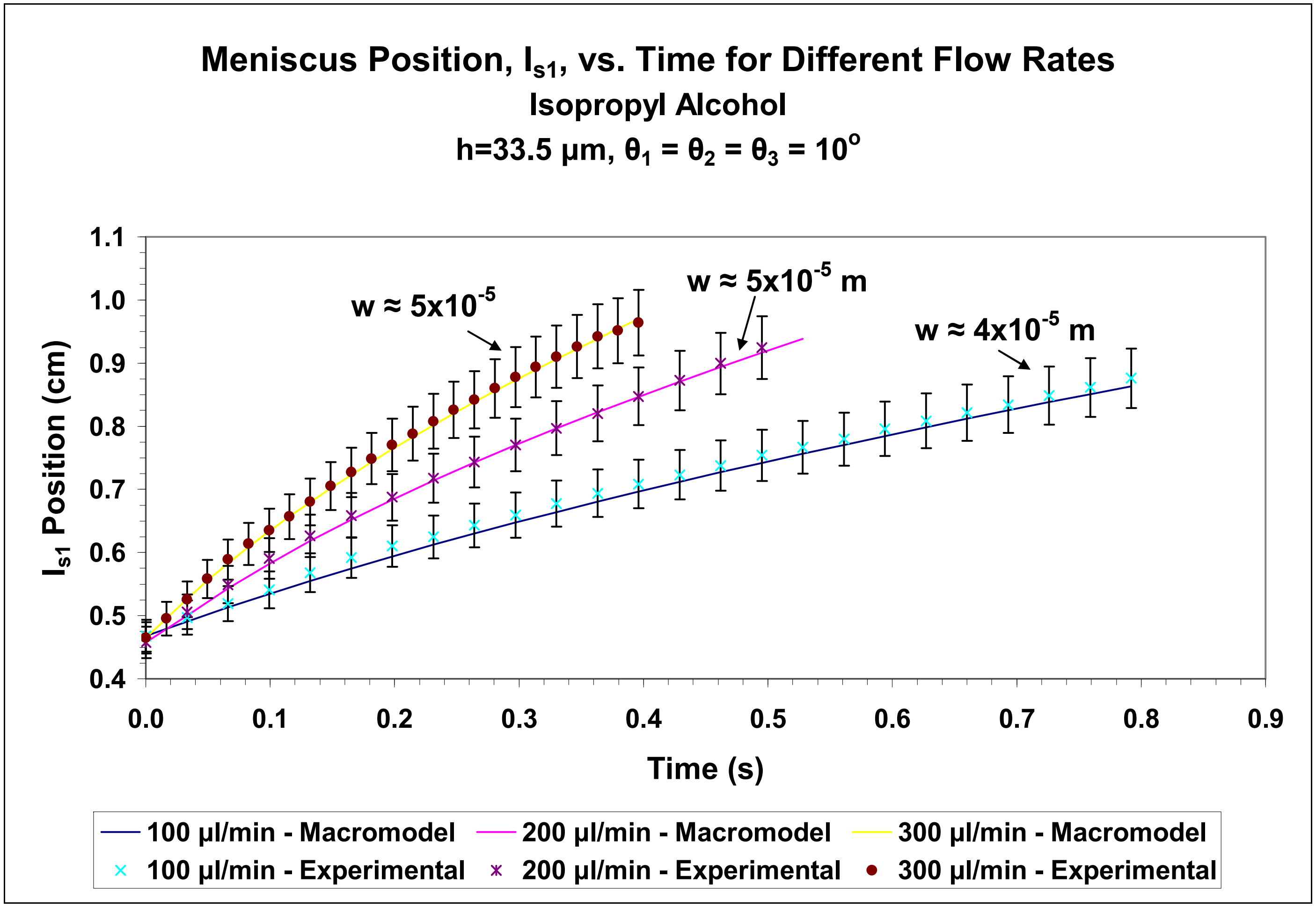


Figure 39: l_{s1} vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈33.5μm

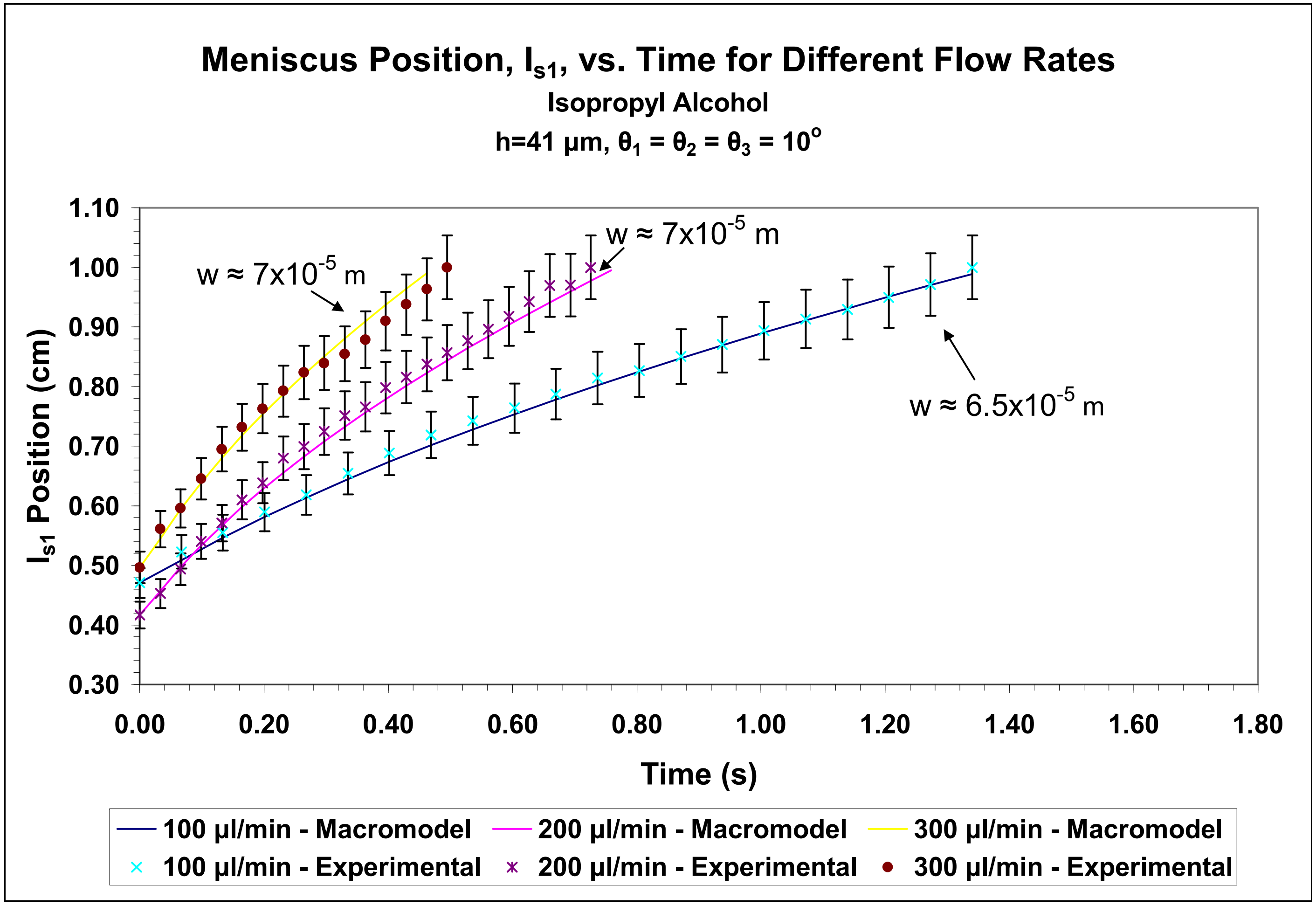


Figure 40: l_{s1} vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈41μm

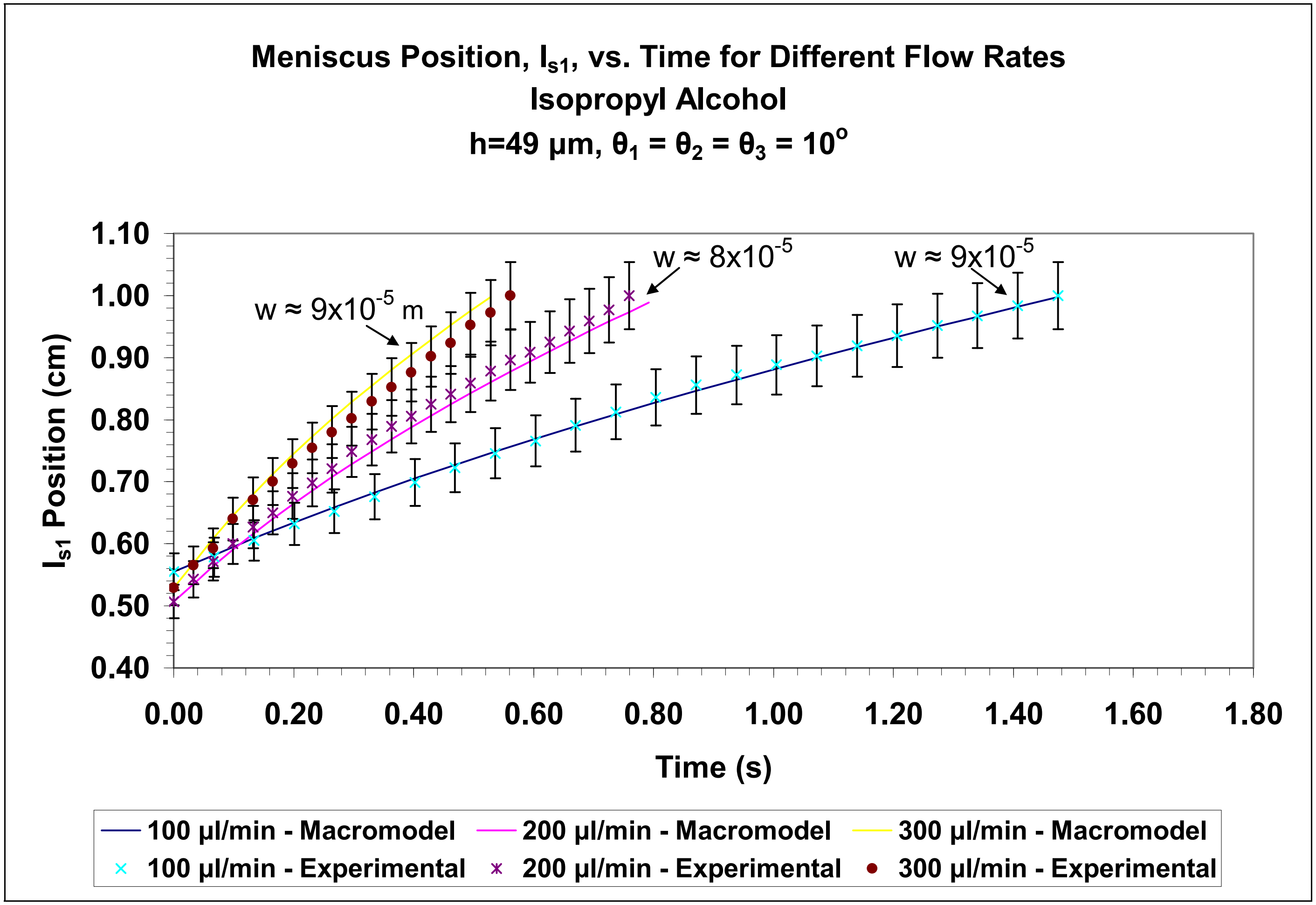


Figure 41: l_{s1} vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈49μm

APPENDIX B

PLOTS OF l_{s2} FOR MICROCHAMBERS OF VARIOUS DEPTHS AND FLOW CONDITIONS

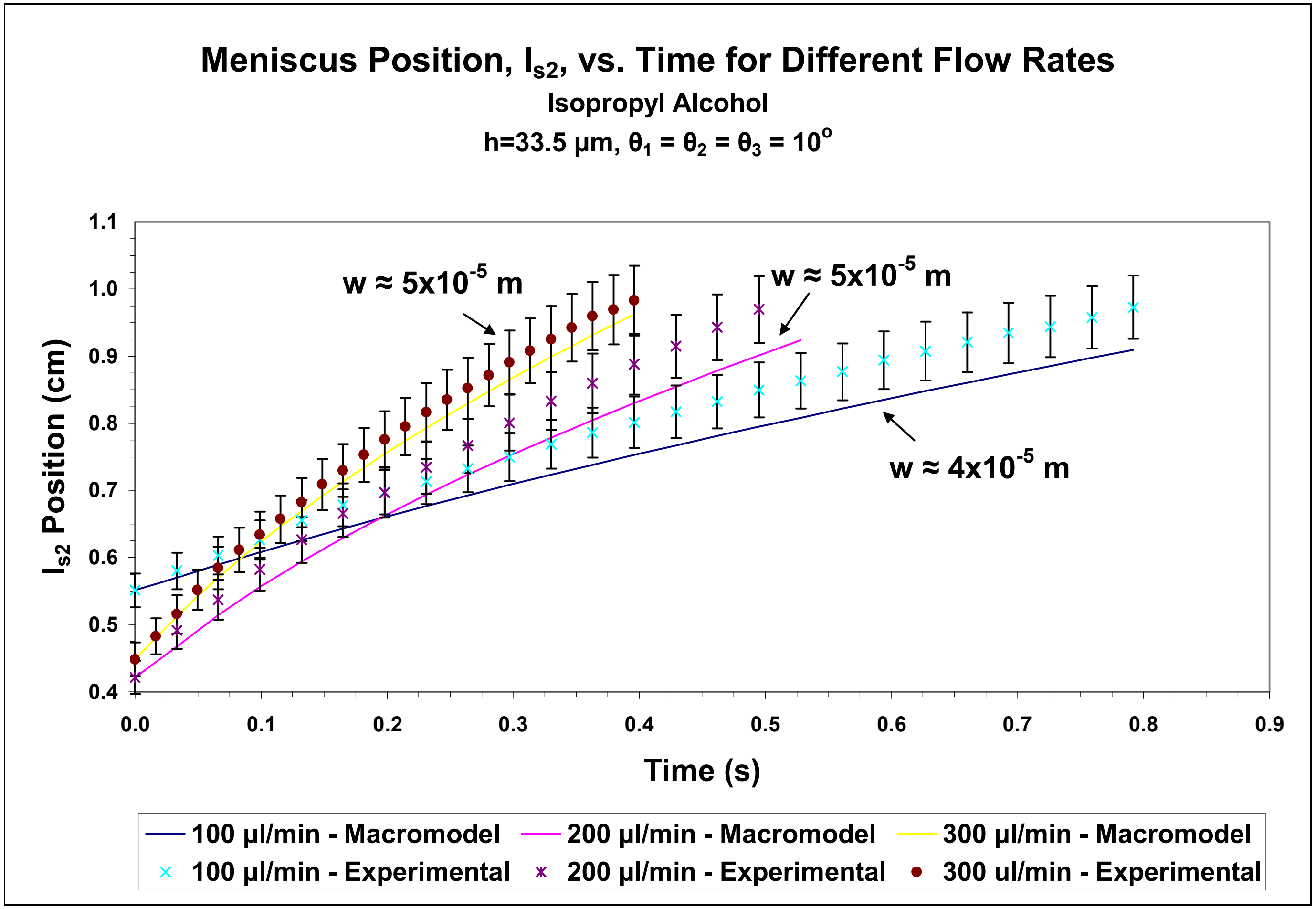


Figure 42: l_{s2} vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈33.5µm

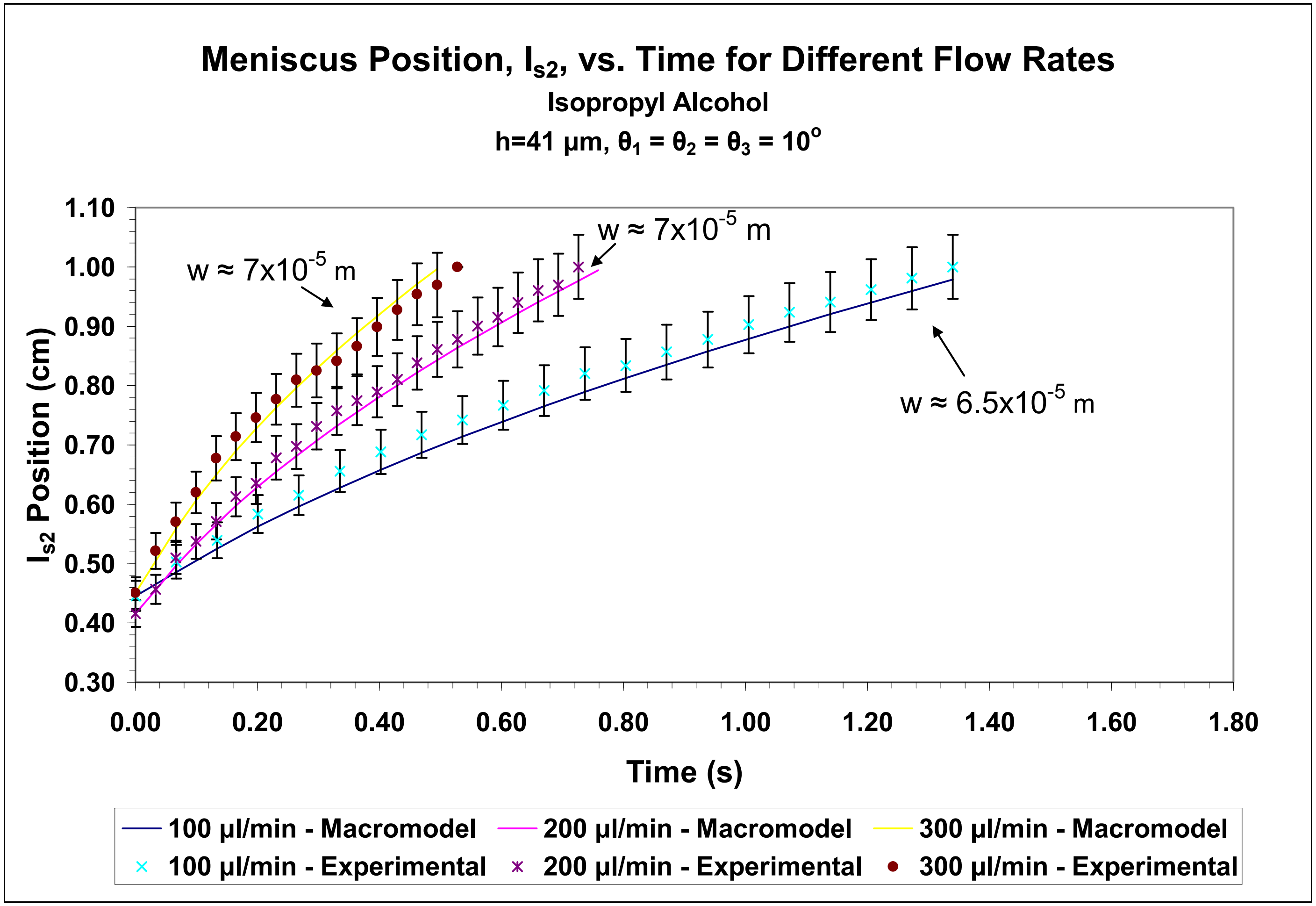


Figure 43: l_{s2} vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈41μm

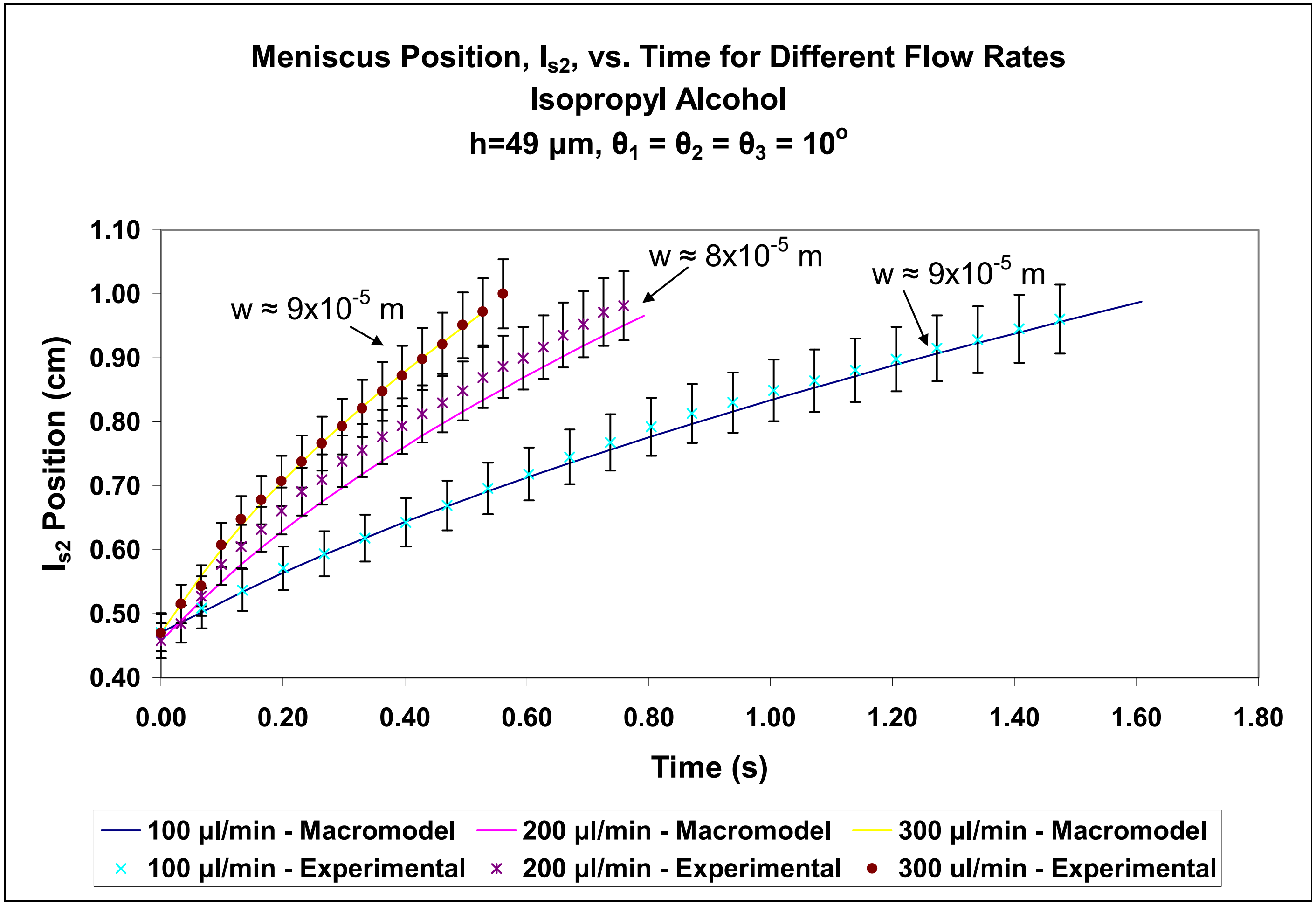


Figure 44: l_{s2} **vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions** L_1**≈1cm,** L_2**≈1cm, and h≈49µm**

APPENDIX C

PLOTS OF l_m FOR MICROCHAMBERS OF VARIOUS DEPTHS AND FLOW CONDITIONS

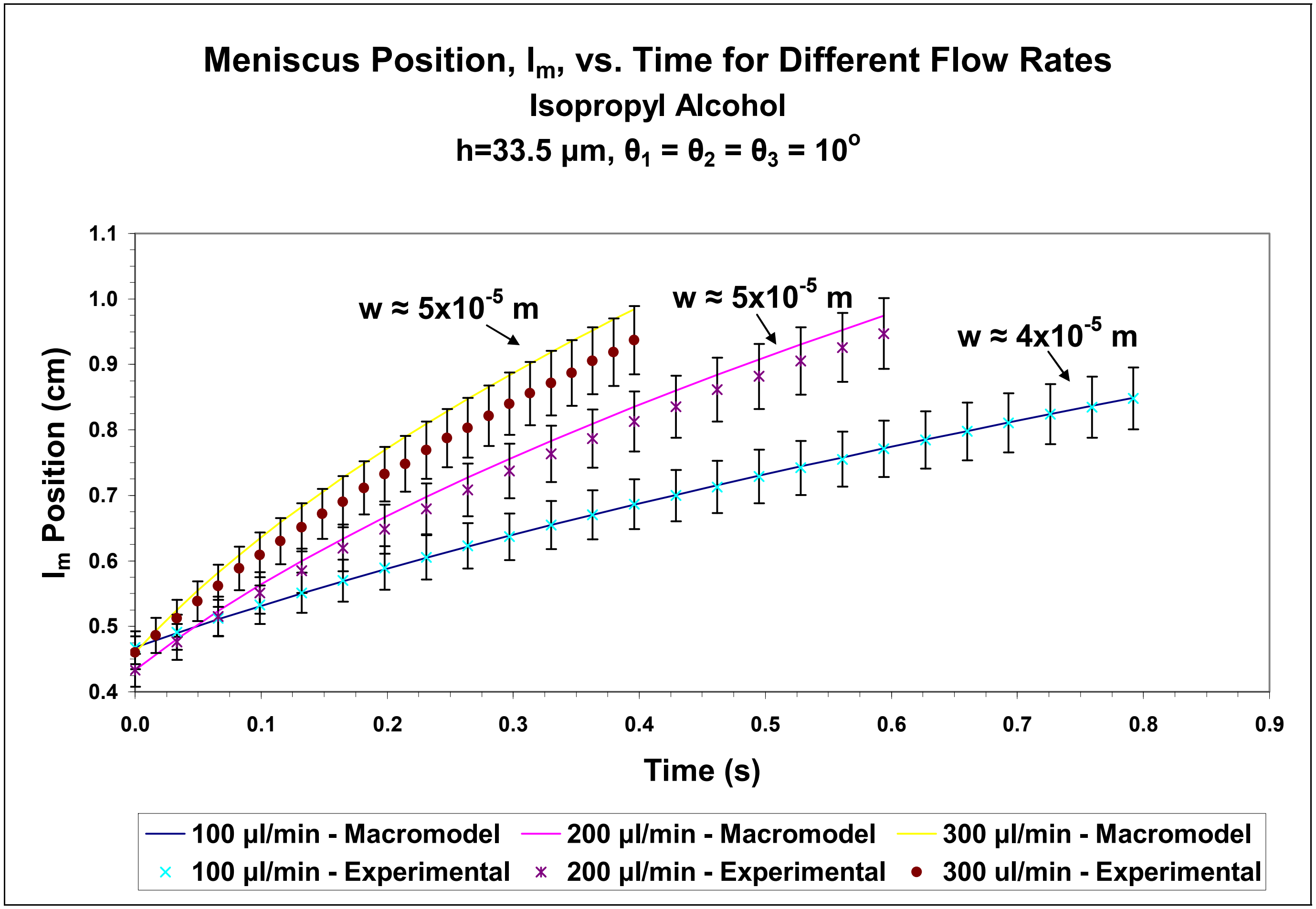


Figure 45: l_m vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈33.5μm

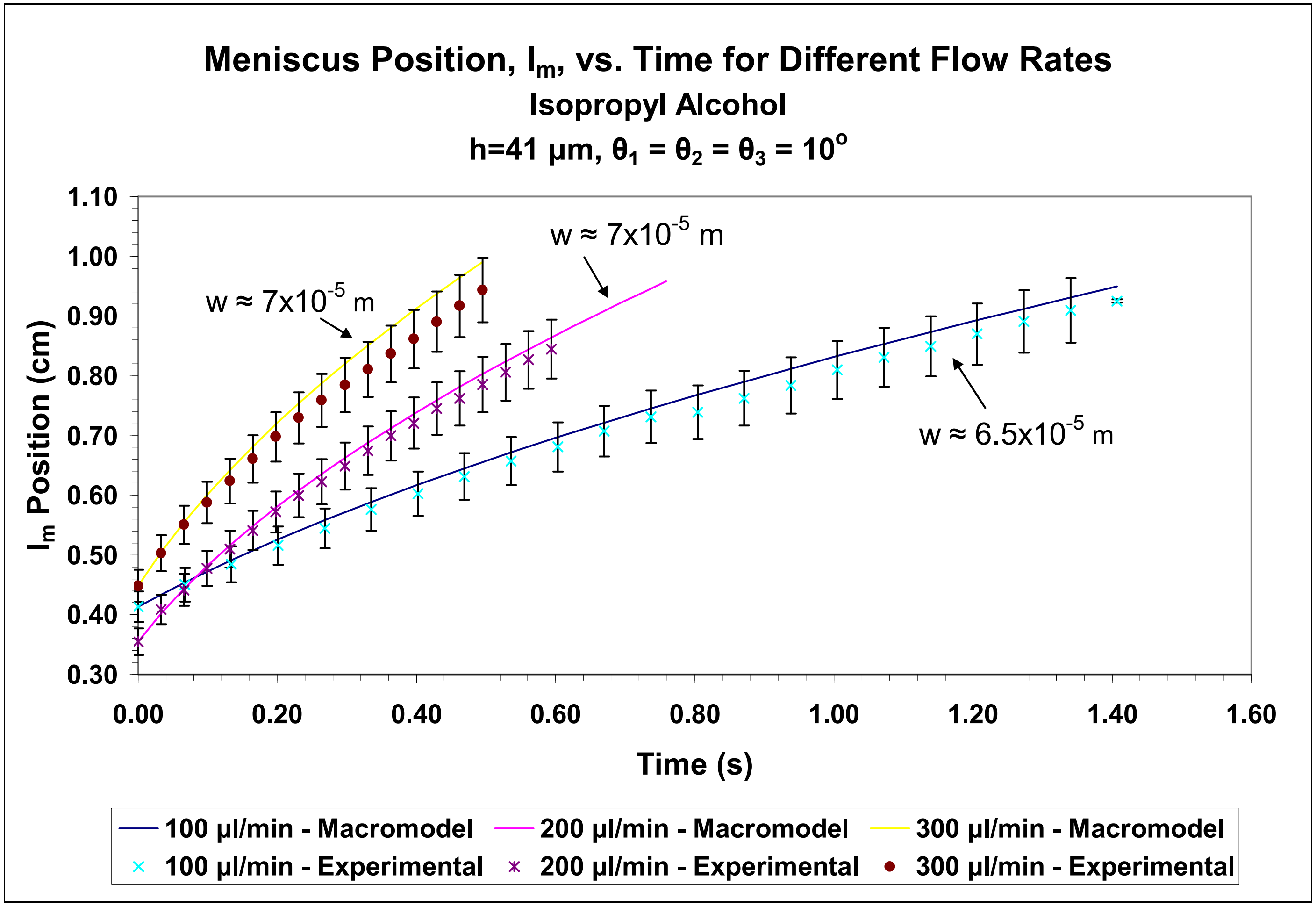


Figure 46: l_m vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈41µm

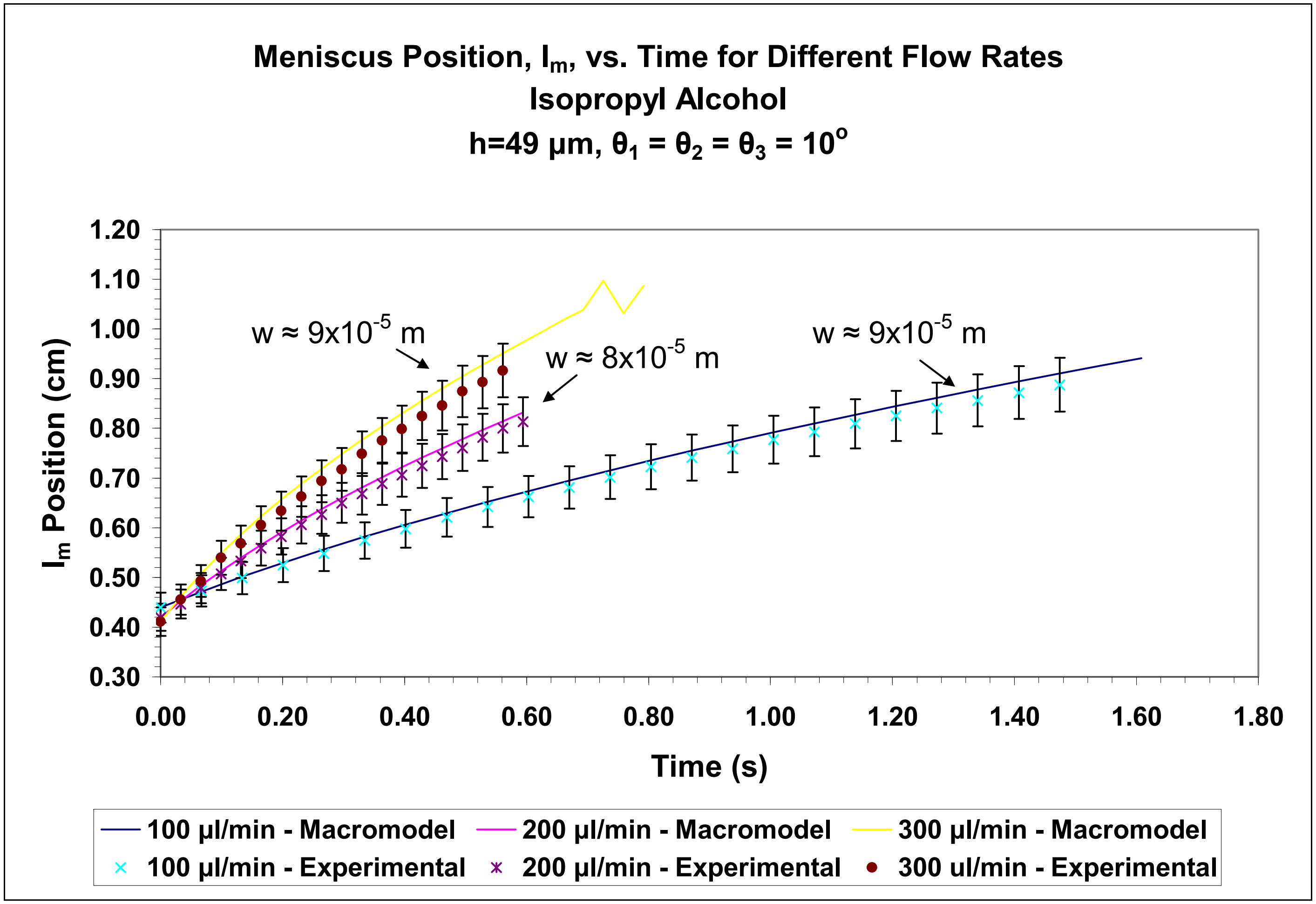


Figure 47: l_m vs. Time for Isopropyl Alcohol Filling Experiments at Various Flow Rates with Chamber Dimensions L_1≈1cm, L_2≈1cm, and h≈49μm

APPENDIX D

VISUAL BASIC .NET CODE FOR MENISCUS MEASUREMENTS

```
Imports System.Math
Public Class Form1
    Dim DrawSquare As Boolean
    Dim LineX1 As Integer
    Dim LineY1 As Integer
    Dim LineX2 As Integer
    Dim LineY2 As Integer
    Dim SizePoint As Integer
    Dim FirstSave As Boolean = True
    Dim CurSpline As Integer = 0
    'Dim SplinePoints(200) As Point
    Private Sub PictureBox1_MouseClick(ByVal sender As Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles PictureBox1.MouseClick
        If e.Button = Windows.Forms.MouseButtons.Right Then
            If ListBox1.Items.Count <> 0 Then
                'Delete all the points, but not the outlined square
                ListBox1.Items.RemoveAt(ListBox1.Items.Count - 1)
                PictureBox1.Refresh()
                DrawRectangle(0, 0)
                Exit Sub
            Else 'Reset to beginning
                LineX1 = 0
                LineY1 = 0
                LineX2 = 0
                LineY2 = 0
                TextBox3.Text = ""
                TextBox6.Text = ""
                PictureBox1.Refresh()
                DrawSquare = False
                'DrawRectangle(0, 0)
                Exit Sub
            End If
        End If
        'Check if this is the first click to define a square
        If LineX1 = 0 And LineY1 = 0 Then
            DrawSquare = True
            LineX1 = e.X
            LineY1 = e.Y
            TextBox3.Text = LineX1 & "," & LineY1
        ElseIf LineX2 = 0 And LineY2 = 0 And DrawSquare = True Then 'This is the last click to define the square
            DrawSquare = False
            LineX2 = e.X
            LineY2 = e.Y
            DrawRectangle(e.X, e.Y)
            'Some of the blow code may not be needed, it was inteded to make it easier to click on the boundary of the square
        ElseIf LineX1 <> 0 And LineY1 <> 0 And LineX2 <> 0 And LineY2 <> 0 Then 'We are defining points
            'If possible, lets put the point directly on the line of the square
            'Check if they're clicking on the vertical axis
            If Abs((((e.X - LineX1) / e.X) * 100)) < 1 And e.Y > LineY1 Then 'The difference between the click and the actual line is small
                'Draw a circle on the line
                'Save the points
                ListBox1.Items.Add(LineX1 & "," & e.Y & "   0 " & ComboBox1.Text & ", " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text)
                Dim Pen2 As New System.Drawing.Pen(Color.Blue, 0.5)
                Dim PointDraw As System.Drawing.Graphics
                PictureBox1.Refresh()
                DrawRectangle(0, 0)
                PointDraw = PictureBox1.CreateGraphics
                'Vertical Axis
                PointDraw.DrawEllipse(Pens.Blue, New Rectangle((LineX1 - (SizePoint / 2)), (e.Y - (SizePoint / 2)), SizePoint, SizePoint))
                If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString("0 " & ComboBox1.Text & ", " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text, Me.Font, Brushes.Black, e.X, e.Y)
            'Now update the text box to show the altered position
            'This is the vertical axis
            Label6.Text = "Radial Distance: " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text
            Label1.Text = "X Position: 0 " & ComboBox1.Text
            Label2.Text = "Y Position: " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text
            Exit Sub
        ElseIf Abs((((e.Y - LineY2) / e.Y) * 100)) < 1 And e.X < LineX2 Then 'The difference between the click and the actual line is small
            'Save the points
            ListBox1.Items.Add(e.X & "," & LineY2 & "   " & Math.Round(1 - (((LineX2 - e.X) / (LineX2 - LineX1))) * TextBox4.Text, 4) & " " & ComboBox1.Text & ", 0 " & ComboBox1.Text)
            'CheckHorizontal Axis
            'Draw a circle on the line
            Dim Pen2 As New System.Drawing.Pen(Color.Blue, 0.5)
            Dim PointDraw As System.Drawing.Graphics
            PictureBox1.Refresh()
            DrawRectangle(0, 0)
            PointDraw = PictureBox1.CreateGraphics
            'Horizontal Axis
            PointDraw.DrawEllipse(Pens.Blue, New Rectangle((e.X - (SizePoint / 2)), (LineY2 - (SizePoint / 2)), SizePoint, SizePoint))
            If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString(Math.Round(TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4) & " " & ComboBox1.Text & ", 0 " & ComboBox1.Text, Me.Font, Brushes.Black, e.X, e.Y)
            'Now update the text box to show the altered position
            'This is the vertical axis
            Label6.Text = "Radial Distance: " & TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4) & " " & ComboBox1.Text
            Label1.Text = "X Position: " & Math.Round(TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4) & " " & ComboBox1.Text
            Label2.Text = "Y Position: 0 " & ComboBox1.Text
            Exit Sub
        ElseIf Abs((((e.Y - LineY1) / e.Y) * 100)) < 1 And e.Y < LineY2 Then 'The difference between the click and the actual line is small
            'Save the points
            ListBox1.Items.Add(e.X & "," & LineY1 & "   " & Math.Round(1 - (((LineX2 - e.X) / (LineX2 - LineX1))) * TextBox4.Text, 4) & " " & ComboBox1.Text & ", " & TextBox2.Text & " " & ComboBox1.Text)
            'CheckHorizontal Axis
            'Draw a circle on the line
            Dim Pen2 As New System.Drawing.Pen(Color.Blue, 0.5)
            Dim PointDraw As System.Drawing.Graphics
            PictureBox1.Refresh()
            DrawRectangle(0, 0)
            PointDraw = PictureBox1.CreateGraphics
            'Horizontal Axis
            PointDraw.DrawEllipse(Pens.Blue, New Rectangle((e.X - (SizePoint / 2)), (LineY1 - (SizePoint / 2)), SizePoint, SizePoint))
            If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString(Math.Round(TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4) & " " & ComboBox1.Text & ", " & TextBox2.Text & " " & ComboBox1.Text, Me.Font, Brushes.Black, e.X, e.Y)
            'Now update the text box to show the altered position
            'This is the vertical axis
            Label6.Text = "Radial Distance: " & TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4) & " " & ComboBox1.Text
            Label1.Text = "X Position: " & Math.Round(TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4) & " " & ComboBox1.Text
            Label2.Text = "Y Position: " & TextBox2.Text & " " & ComboBox1.Text
            Exit Sub
        ElseIf Abs((((e.X - LineX2) / e.X) * 100)) < 1 And e.X > LineX1 Then 'The difference between the click and the actual line is small
            'Save the points

            ListBox1.Items.Add(LineX2 & "," & e.Y & "   " & TextBox2.Text & " " & ComboBox1.Text & ", " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text)
            'CheckHorizontal Axis
            'Draw a circle on the line
            Dim Pen2 As New System.Drawing.Pen(Color.Blue, 0.5)
            Dim PointDraw As System.Drawing.Graphics
            PictureBox1.Refresh()
            DrawRectangle(0, 0)
            PointDraw = PictureBox1.CreateGraphics
            'Horizontal Axis
            PointDraw.DrawEllipse(Pens.Blue, New Rectangle((LineX2 - (SizePoint / 2)), (e.Y - (SizePoint / 2)), SizePoint, SizePoint))
            If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString(TextBox4.Text & " " & ComboBox1.Text & ", " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text, Me.Font, Brushes.Black, e.X, e.Y)
            'Now update the text box to show the altered position
            'This is the vertical axis
            Label6.Text = "Radial Distance: " & TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4) & " " & ComboBox1.Text
            Label1.Text = "X Position: " & TextBox4.Text & " " & ComboBox1.Text
            Label2.Text = "Y Position: " & Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4) & " " & ComboBox1.Text
            Exit Sub
        ElseIf e.X > LineX2 Then
            Exit Sub
        ElseIf e.Y > LineY2 Then
            Exit Sub
        ElseIf e.Y < LineY1 Then
            Exit Sub
        ElseIf e.X < LineX1 Then
            Exit Sub
        Else
            'Anywhere else on the image
            'Draw a circle anywhere else on the image
            'Save the points
            Dim XCoordinate As Double
            Dim YCoordinate As Double
            XCoordinate = Math.Round(TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
            YCoordinate = Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4)
            ListBox1.Items.Add(e.X & "," & e.Y & "   " & XCoordinate & " " & ComboBox1.Text & ", " & YCoordinate & " " & ComboBox1.Text)
            Dim Pen2 As New System.Drawing.Pen(Color.Blue, 0.5)
            Dim PointDraw As System.Drawing.Graphics
            PictureBox1.Refresh()
            DrawRectangle(0, 0)
            PointDraw = PictureBox1.CreateGraphics
            'Anywhere else along the vertical
            PointDraw.DrawEllipse(Pens.Blue, New Rectangle((e.X - (SizePoint / 2)), (e.Y - (SizePoint / 2)), SizePoint, SizePoint))
            If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString(XCoordinate & " " & ComboBox1.Text & ", " & YCoordinate & " " & ComboBox1.Text, Me.Font, Brushes.Black, e.X, e.Y)
            Label6.Text = "Radial Distance: " & Math.Round(Sqrt((XCoordinate * XCoordinate) + (YCoordinate * YCoordinate)), 4) & " " & ComboBox1.Text
            Label1.Text = "X Position: " & XCoordinate & " " & ComboBox1.Text
            Label2.Text = "Y Position: " & YCoordinate & " " & ComboBox1.Text
        End If
    End If
End Sub

```

```
Private Sub PictureBox1_MouseMove(ByVal sender As Object, ByVal e As System.Windows.Forms.MouseEventArgs) Handles PictureBox1.MouseMove
    If DrawSquare = True Then
        DrawRectangle(e.X, e.Y)
        TextBox6.Text = e.X & "," & e.Y
    ElseIf LineX1 = 0 And LineY1 = 0 Then
        TextBox3.Text = e.X & "," & e.Y
    End If
    If LineX1 <> 0 And LineY1 <> 0 And LineX2 <> 0 And LineY2 <> 0 Then
        Dim XCoordinate As Double
        Dim YCoordinate As Double
        XCoordinate = Math.Round(TextBox4.Text - Math.Round(((LineX2 - e.X) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
        YCoordinate = Math.Round(((LineY2 - e.Y) / (LineY2 - LineY1)) * TextBox2.Text, 4)
        If e.X > LineX1 And e.X < LineX2 And e.Y > LineY1 And e.Y < LineY2 Then
            Label6.Text = "Radial Distance: " & Math.Round(Sqrt((XCoordinate * XCoordinate) + (YCoordinate * YCoordinate)), 4) & " " & ComboBox1.Text
            Label1.Text = "X Position: " & XCoordinate & " " & ComboBox1.Text
            Label2.Text = "Y Position: " & YCoordinate & " " & ComboBox1.Text
        End If
    End If
End Sub
Sub DrawRectangle(ByVal CurX, ByVal CurY)
    Dim Pen2 As New System.Drawing.Pen(Color.Tomato, 0.5)
    Dim LineDraw As System.Drawing.Graphics
    PictureBox1.Refresh()
    LineDraw = PictureBox1.CreateGraphics

    'Draw the data points
    Dim PointDraw As System.Drawing.Graphics, PointString As String, CurPoint As Integer
    For CurPoint = 0 To (ListBox1.Items.Count - 1)
        PointDraw = PictureBox1.CreateGraphics
        PointString = ListBox1.Items(CurPoint).ToString
        PointString = Strings.Left(PointString, Strings.InStr(PointString, " ") - 1)
        PointDraw.DrawEllipse(Pens.Blue, New Rectangle((Strings.Left(PointString, Strings.InStr(PointString, ",") - 1) - (SizePoint / 2)), (Strings.Right(PointString, Strings.Len(PointString) - Strings.InStrRev(PointString, ",")) - (SizePoint / 2)), SizePoint, SizePoint))
        If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString(Strings.Right(ListBox1.Items(CurPoint).ToString, (Strings.Len(ListBox1.Items(CurPoint).ToString) - (Strings.Len(PointString) + 3))), Me.Font, Brushes.Black, (Strings.Left(PointString, Strings.InStr(PointString, ",") - 1)), (Strings.Right(PointString, Strings.Len(PointString) - Strings.InStrRev(PointString, ","))))
    Next


    If LineX2 = 0 And LineY2 = 0 Then
        If CurX < LineX1 Or CurY < LineY1 Then
            LineX1 = 0
            LineY1 = 0
            LineX2 = 0
            LineY2 = 0
            DrawSquare = False
            Exit Sub
        End If
        LineDraw.DrawLine(Pen2, LineX1, LineY1, LineX1, CurY) 'Vertical Left Side
        LineDraw.DrawLine(Pen2, CurX, LineY1, CurX, CurY) 'Vertical Right Side
        LineDraw.DrawLine(Pen2, LineX1, LineY1, CurX, LineY1) 'Horizontal Line
        LineDraw.DrawLine(Pen2, LineX1, CurY, CurX, CurY)
    Else
        LineDraw.DrawLine(Pen2, LineX1, LineY1, LineX1, LineY2) 'Vertical Left Side
        LineDraw.DrawLine(Pen2, LineX2, LineY1, LineX2, LineY2) 'Vertical Right Side
        LineDraw.DrawLine(Pen2, LineX1, LineY1, LineX2, LineY1) 'Horizontal Line
        LineDraw.DrawLine(Pen2, LineX1, LineY2, LineX2, LineY2)
    End If
End Sub

```

```
    Private Sub Form1_Activated(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Activated
        ComboBox1.Text = "cm"
    End Sub
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        DrawSquare = False
        LineX1 = 0
        LineY1 = 0
        PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
        SizePoint = 4
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        On Error GoTo ErrorHandler
        OpenFileDialog1.ShowDialog()
        TextBox1.Text = OpenFileDialog1.FileName
        PictureBox1.ImageLocation = TextBox1.Text
        PictureBox1.Load()
        GroupBox4.Visible = False
        picSource.ImageLocation = TextBox1.Text
        picSource.Load()
        FirstSave = True
        RotatePic(False)
        Exit Sub
ErrorHandler:
        MsgBox("You must select an appropriate image file to continue")
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        TextBox3.Text = LineX2 - LineX1
        LineX1 = 0
        LineY1 = 0
        LineX2 = 0
        LineY2 = 0
    End Sub

    Private Sub TextBox5_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles txtAngle.KeyPress
        If Asc(e.KeyChar) = 13 Then
            RotatePic(False)
        End If
    End Sub
    Sub RotatePic(ByVal Original As Boolean)
        On Error GoTo errorhandler
        Dim strAngle As Double
        If Original = True Then strAngle = 0 Else strAngle = txtAngle.Text
        Dim bm_in As New Bitmap(picSource.Image)
        Dim wid As Single = bm_in.Width
        Dim hgt As Single = bm_in.Height
        Dim corners As Point() = {New Point(0, 0), New Point(wid, 0), New Point(0, hgt), New Point(wid, hgt)}
        Dim cx As Single = wid / 2
        Dim cy As Single = hgt / 2
        Dim i As Long
        For i = 0 To 3
            corners(i).X -= cx
            corners(i).Y -= cy
        Next i
        Dim theta As Single = Single.Parse(strAngle) * PI / 180.0
        Dim sin_theta As Single = Sin(theta)
        Dim cos_theta As Single = Cos(theta)
        Dim X As Single
        Dim Y As Single
        For i = 0 To 3
            X = corners(i).X

            Y = corners(i).Y
            corners(i).X = X * cos_theta + Y * sin_theta
            corners(i).Y = -X * sin_theta + Y * cos_theta
        Next i
        Dim xmin As Single = corners(0).X
        Dim ymin As Single = corners(0).Y
        For i = 1 To 3
            If xmin > corners(i).X Then xmin = corners(i).X
            If ymin > corners(i).Y Then ymin = corners(i).Y
        Next i
        For i = 0 To 3
            corners(i).X -= xmin
            corners(i).Y -= ymin
        Next i
        Dim bm_out As New Bitmap(CInt(-2 * xmin), CInt(-2 * ymin))
        Dim gr_out As Graphics = Graphics.FromImage(bm_out)
        ReDim Preserve corners(2)
        gr_out.DrawImage(bm_in, corners)
        PictureBox1.Image = bm_out
        DrawRectangle(0, 0)
        Exit Sub
errorhandler:
        MsgBox("No working image has been loaded")
    End Sub
    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged
        Label4.Text = ComboBox1.Text
        Label11.Text = ComboBox1.Text
        ListBox1.Items.Clear()
    End Sub

    Private Sub TextBox3_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles TextBox3.KeyPress
        If Asc(e.KeyChar) = 13 Then
            LineX1 = Strings.Left(TextBox3.Text, Strings.InStr(TextBox3.Text, ",") - 1)
            LineY1 = Strings.Right(TextBox3.Text, Strings.Len(TextBox3.Text) - Strings.InStrRev(TextBox3.Text, ","))
            DrawRectangle(0, 0)
        End If
    End Sub

    Private Sub TextBox6_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles TextBox6.KeyPress
        If Asc(e.KeyChar) = 13 Then
            If CDbl(Strings.Left(TextBox6.Text, Strings.InStr(TextBox6.Text, ",") - 1)) < LineX1 Then
                MsgBox("X Coordinate Ending Point cannot be less than the X Coordinate Starting Point")
                TextBox6.Text = LineX2 & "," & LineY2
                Exit Sub
            End If
            If CDbl(Strings.Right(TextBox6.Text, Strings.Len(TextBox6.Text) - Strings.InStrRev(TextBox6.Text, ","))) < LineY1 Then
                MsgBox("Y Coordinate Ending Point cannot be less than the Y Coordinate Starting Point")
                TextBox6.Text = LineX2 & "," & LineY2
                Exit Sub
            End If
            LineX2 = Strings.Left(TextBox6.Text, Strings.InStr(TextBox6.Text, ",") - 1)
            LineY2 = Strings.Right(TextBox6.Text, Strings.Len(TextBox6.Text) - Strings.InStrRev(TextBox6.Text, ","))
            DrawRectangle(0, 0)
        End If
    End Sub
```

```
Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles CheckBox1.CheckedChanged
    DrawRectangle(0, 0)
End Sub

Private Sub ListBox1_MouseDown(ByVal sender As Object, ByVal e As System.Windows.Forms.
MouseEventArgs) Handles ListBox1.MouseDown
    If e.Button = Windows.Forms.MouseButtons.Right Then
        ListBox1.SelectedIndex = -1
    End If
End Sub

Private Sub ListBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles ListBox1.SelectedIndexChanged
    'Dim Pen2 As New System.Drawing.Pen(Color.GreenYellow, 0.5)
    On Error Resume Next
    Dim PointDraw As System.Drawing.Graphics, PointString As String
    PictureBox1.Refresh()
    DrawRectangle(0, 0)
    PointDraw = PictureBox1.CreateGraphics
    PointString = ListBox1.Items(ListBox1.SelectedIndex).ToString
    PointString = Strings.Left(PointString, Strings.InStr(PointString, " ") - 1)
    Dim point1 As New Point(Strings.Left(PointString, Strings.InStr(PointString, ",") - 1),
Strings.Right(PointString, Strings.Len(PointString) - Strings.InStrRev(PointString, ",")))
    PointDraw.DrawEllipse(Pens.Red, New Rectangle((Strings.Left(PointString, Strings.InStr
(PointString, ",") - 1) - (SizePoint * 2 / 2)), (Strings.Right(PointString, Strings.Len
(PointString) - Strings.InStrRev(PointString, ",")) - (SizePoint * 2 / 2)), SizePoint * 2,
SizePoint * 2))
    If CheckBox2.Checked = True Then PictureBox1.CreateGraphics.DrawString(Strings.Right
(ListBox1.Items(ListBox1.SelectedIndex).ToString, (Strings.Len(ListBox1.Items(ListBox1.
SelectedIndex).ToString) - (Strings.Len(PointString) + 3))), Me.Font, Brushes.Black,
(Strings.Left(PointString, Strings.InStr(PointString, ",") - 1)), (Strings.Right(PointString
, Strings.Len(PointString) - Strings.InStrRev(PointString, ","))))
End Sub

Private Sub Button2_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
    Static SaveFileName As String
    Dim CurDataRecord As Integer
    'Save the Data
    SaveFileDialog1.Filter = "Comma Delimited Data (*.csv)|*.csv"
    SaveFileDialog1.ShowDialog()
    If SaveFileDialog1.FileName = "" Then
        MsgBox("Data was not saved.")
        Exit Sub
    End If
    SaveFileName = SaveFileDialog1.FileName
    FirstSave = False
    If Dir(SaveFileDialog1.FileName) = "" Then
        IO.File.AppendAllText(SaveFileName, "ls1-x (" & ComboBox1.Text & "),ls1-y (" &
ComboBox1.Text & "),ls1-Radial (" & ComboBox1.Text & "),lm-x (" & ComboBox1.Text & "),lm-y
(" & ComboBox1.Text & "),lm-Radial (" & ComboBox1.Text & "),ls2-x (" & ComboBox1.Text & "),
ls2-y (" & ComboBox1.Text & "),ls2-Radial (" & ComboBox1.Text & ")" & vbCrLf)
    End If
    Dim xCoordinate As Double, yCoordinate As Double, DataPoints As String
    Dim TempString As String, NumSaveRecords As Integer
    NumSaveRecords = 0
    '**Right now this save routine pretty much assumes only 3 points are being saved, and
are being labeled as such.
    'This can be changed, but right now it's for the microfluidics project.
    For CurDataRecord = 0 To (ListBox1.Items.Count - 1)
        DataPoints = Strings.Trim(Strings.Right(ListBox1.Items(CurDataRecord).ToString,
Strings.Len(ListBox1.Items(CurDataRecord).ToString) - Strings.InStr(ListBox1.Items
(CurDataRecord).ToString, "   ")))
        TempString = Strings.Left(DataPoints, Strings.InStr(DataPoints, ",") - 1)
        xCoordinate = Strings.Left(TempString, Strings.InStr(TempString, " ") - 1)

```

```
            TempString = Strings.Trim(Strings.Right(DataPoints, Strings.Len(DataPoints) - Strings.InStrRev(DataPoints, ",")))
            yCoordinate = Strings.Left(TempString, Strings.InStr(TempString, " ") - 1)
            'Save as x,y,radial
            IO.File.AppendAllText(SaveFileName, xCoordinate & "," & yCoordinate & "," & Math.Sqrt((xCoordinate * xCoordinate) + (yCoordinate * yCoordinate)) & ",")
        Next
        IO.File.AppendAllText(SaveFileName, vbCrLf)
        Beep()
        DrawRectangle(0, 0)
        'ListBox1.Items.Clear()
        CurSpline = 0
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
        On Error Resume Next
        PictureBox1.Image = Clipboard.GetImage
        'ListBox1.Items.Clear()
        SaveFileDialog1.Filter = "JPEG (*.jpg)|*.jpg"
        SaveFileDialog1.ShowDialog()
        Do Until SaveFileDialog1.FileName <> ""
            SaveFileDialog1.ShowDialog()
        Loop
        PictureBox1.Image.Save(SaveFileDialog1.FileName)
        On Error GoTo ErrorHandler
        TextBox1.Text = SaveFileDialog1.FileName
        PictureBox1.ImageLocation = TextBox1.Text
        PictureBox1.Load()
        GroupBox4.Visible = False
        picSource.ImageLocation = TextBox1.Text
        picSource.Load()
        FirstSave = True
        RotatePic(False)
        Exit Sub
ErrorHandler:
        MsgBox("You must select an appropriate image file to continue")
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
        If GroupBox4.Visible = True Then GroupBox4.Visible = False Else GroupBox4.Visible = True
    End Sub

    Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles CheckBox2.CheckedChanged
        DrawRectangle(0, 0)
    End Sub

    Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button6.Click
        IncreasePoints(True, True)
    End Sub
    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click
        DecreasePoints(True, True)
    End Sub

```

```
Sub DecreasePoints(ByVal xpoint As Boolean, ByVal ypoint As Boolean)
    'Incrementpoints and decrementpoints could be combined into one sub
    Dim CurDataRecord As Integer
    Dim DataPoints As String, TempString As String, xCoordinate As Double, yCoordinate As Double
    Dim xCoordinateMeasure As Double, yCoordinateMeasure As Double
    Dim CurrentListCount As Integer
    CurrentListCount = (ListBox1.Items.Count - 1)
    For CurDataRecord = 0 To CurrentListCount
        DataPoints = Strings.Trim(Strings.Left(ListBox1.Items(CurDataRecord).ToString, Strings.InStr(ListBox1.Items(CurDataRecord.ToString), " ") - 1))
        TempString = Strings.Left(DataPoints, Strings.InStr(DataPoints, ",") - 1)
        xCoordinate = TempString
        TempString = Strings.Trim(Strings.Right(DataPoints, Strings.Len(DataPoints) - Strings.InStrRev(DataPoints, ",")))
        yCoordinate = TempString
        If xCoordinate = LineX1 Or xCoordinate = LineX2 Then 'Only increment the y coordinate
            If (yCoordinate + 1) < LineY1 Then 'Max out y and increment x
                If ypoint = True Then yCoordinate = LineY1
                If xpoint = True Then xCoordinate = xCoordinate - 1
            ElseIf (yCoordinate + 1) > LineY2 Then
                If xpoint = True Then xCoordinate = LineX2 - 1
                If ypoint = True Then yCoordinate = LineY2
            Else
                If ypoint = True Then yCoordinate = yCoordinate + 1
                If xpoint = True And (xCoordinate - 1) <= LineX2 And (xCoordinate - 1) >= LineX1 And xCoordinate <> LineX1 And xCoordinate <> LineX2 Then xCoordinate = xCoordinate - 1
            End If
        ElseIf (yCoordinate + 1) < LineY1 And (xCoordinate - 1) > LineX2 Then
            If xpoint = True Then xCoordinate = LineX2
            If ypoint = True Then yCoordinate = LineY1
        ElseIf yCoordinate = LineY2 Or yCoordinate = LineY1 Then 'Only increment the x coordinate
            If xpoint = True Then xCoordinate = xCoordinate - 1
        Else 'Increment Both
            If xpoint = True Then xCoordinate = xCoordinate - 1
            If ypoint = True Then yCoordinate = yCoordinate + 1
        End If
        'Translate to measured distance
        If ListBox1.SelectedIndex = -1 Then
            xCoordinateMeasure = Math.Round(TextBox4.Text - Math.Round(((LineX2 - xCoordinate) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
            yCoordinateMeasure = Math.Round(((LineY2 - yCoordinate) / (LineY2 - LineY1)) * TextBox2.Text, 4)
            ListBox1.Items.Add(xCoordinate & "," & yCoordinate & "    " & xCoordinateMeasure & " " & ComboBox1.Text & ", " & yCoordinateMeasure & " " & ComboBox1.Text)
        ElseIf ListBox1.SelectedIndex = CurDataRecord Then
            xCoordinateMeasure = Math.Round(TextBox4.Text - Math.Round(((LineX2 - xCoordinate) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
            yCoordinateMeasure = Math.Round(((LineY2 - yCoordinate) / (LineY2 - LineY1)) * TextBox2.Text, 4)
            ListBox1.Items.Insert(CurDataRecord + 1, xCoordinate & "," & yCoordinate & "    " & xCoordinateMeasure & " " & ComboBox1.Text & ", " & yCoordinateMeasure & " " & ComboBox1.Text)
            ListBox1.Items.RemoveAt(CurDataRecord)
            ListBox1.SelectedIndex = CurDataRecord
        End If
    Next
    If ListBox1.SelectedIndex = -1 Then
        For CurDataRecord = 0 To CurrentListCount
            ListBox1.Items.RemoveAt(0)
        Next
    End If
    DrawRectangle(0, 0)
End Sub
```

```
Sub UpdatePoints()
    Dim CurDataRecord As Integer
    Dim DataPoints As String, TempString As String, xCoordinate As Double, yCoordinate As Double
    Dim xCoordinateMeasure As Double, yCoordinateMeasure As Double
    Dim CurrentListCount As Integer
    CurrentListCount = (ListBox1.Items.Count - 1)
    For CurDataRecord = 0 To CurrentListCount
        DataPoints = Strings.Trim(Strings.Left(ListBox1.Items(CurDataRecord).ToString, Strings.InStr(ListBox1.Items(CurDataRecord.ToString), " ") - 1))
        TempString = Strings.Left(DataPoints, Strings.InStr(DataPoints, ",") - 1)
        xCoordinate = TempString
        TempString = Strings.Trim(Strings.Right(DataPoints, Strings.Len(DataPoints) - Strings.InStrRev(DataPoints, ",")))
        yCoordinate = TempString
        'Translate to measured distance
        If ListBox1.SelectedIndex = -1 Then
            xCoordinateMeasure = Math.Round(TextBox4.Text - Math.Round(((LineX2 - xCoordinate) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
            yCoordinateMeasure = Math.Round(((LineY2 - yCoordinate) / (LineY2 - LineY1)) * TextBox2.Text, 4)
            ListBox1.Items.Add(xCoordinate & "," & yCoordinate & "   " & xCoordinateMeasure & " " & ComboBox1.Text & ", " & yCoordinateMeasure & " " & ComboBox1.Text)
        ElseIf ListBox1.SelectedIndex = CurDataRecord Then
            xCoordinateMeasure = Math.Round(TextBox4.Text - Math.Round(((LineX2 - xCoordinate) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
            yCoordinateMeasure = Math.Round(((LineY2 - yCoordinate) / (LineY2 - LineY1)) * TextBox2.Text, 4)
            ListBox1.Items.Insert(CurDataRecord + 1, xCoordinate & "," & yCoordinate & "   " & xCoordinateMeasure & " " & ComboBox1.Text & ", " & yCoordinateMeasure & " " & ComboBox1.Text)
            ListBox1.Items.RemoveAt(CurDataRecord)
            ListBox1.SelectedIndex = CurDataRecord
        End If
    Next
    If ListBox1.SelectedIndex = -1 Then
        For CurDataRecord = 0 To CurrentListCount
            ListBox1.Items.RemoveAt(0)
        Next
    End If
    DrawRectangle(0, 0)
End Sub
Sub IncreasePoints(ByVal xpoint As Boolean, ByVal ypoint As Boolean)
    Dim CurDataRecord As Integer
    Dim DataPoints As String, TempString As String, xCoordinate As Double, yCoordinate As Double
    Dim xCoordinateMeasure As Double, yCoordinateMeasure As Double
    Dim CurrentListCount As Integer
    CurrentListCount = (ListBox1.Items.Count - 1)
    For CurDataRecord = 0 To CurrentListCount
        DataPoints = Strings.Trim(Strings.Left(ListBox1.Items(CurDataRecord).ToString, Strings.InStr(ListBox1.Items(CurDataRecord.ToString), " ") - 1))
        TempString = Strings.Left(DataPoints, Strings.InStr(DataPoints, ",") - 1)
        xCoordinate = TempString
        TempString = Strings.Trim(Strings.Right(DataPoints, Strings.Len(DataPoints) - Strings.InStrRev(DataPoints, ",")))
        yCoordinate = TempString
        If xCoordinate = LineX1 Or xCoordinate = LineX2 Then 'Only increment the y coordinate
            If (yCoordinate - 1) < LineY1 Then 'Max out y and increment x
                yCoordinate = LineY1
                If (xCoordinate + 1) > LineX2 Then
                    If xpoint = True Then xCoordinate = LineX2
                Else
                    If xpoint = True Then xCoordinate = xCoordinate + 1
                End If

                Else
                    If ypoint = True Then yCoordinate = yCoordinate - 1
                    If xpoint = True And (xCoordinate + 1) <= LineX2 And (xCoordinate + 1) >= LineX1 And xCoordinate <> LineX1 And xCoordinate <> LineX2 Then xCoordinate = xCoordinate + 1
                End If
            ElseIf (yCoordinate - 1) < LineY1 And (xCoordinate + 1) > LineX2 Then
                If xpoint = True Then xCoordinate = LineX2
                If ypoint = True Then yCoordinate = LineY1
            ElseIf yCoordinate = LineY2 Or yCoordinate = LineY1 Then 'Only increment the x coordinate
                If xpoint = True Then xCoordinate = xCoordinate + 1
            Else 'Increment Both
                If xpoint = True Then xCoordinate = xCoordinate + 1
                If ypoint = True Then yCoordinate = yCoordinate - 1
            End If
            'Translate to measured distance
            If ListBox1.SelectedIndex = -1 Then
                xCoordinateMeasure = Math.Round(TextBox4.Text - Math.Round(((LineX2 - xCoordinate) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
                yCoordinateMeasure = Math.Round(((LineY2 - yCoordinate) / (LineY2 - LineY1)) * TextBox2.Text, 4)
                ListBox1.Items.Add(xCoordinate & "," & yCoordinate & "   " & xCoordinateMeasure & " " & ComboBox1.Text & ", " & yCoordinateMeasure & " " & ComboBox1.Text)
            ElseIf ListBox1.SelectedIndex = CurDataRecord Then
                xCoordinateMeasure = Math.Round(TextBox4.Text - Math.Round(((LineX2 - xCoordinate) / (LineX2 - LineX1)) * TextBox4.Text, 4), 4)
                yCoordinateMeasure = Math.Round(((LineY2 - yCoordinate) / (LineY2 - LineY1)) * TextBox2.Text, 4)
                ListBox1.Items.Insert(CurDataRecord + 1, xCoordinate & "," & yCoordinate & "   " & xCoordinateMeasure & " " & ComboBox1.Text & ", " & yCoordinateMeasure & " " & ComboBox1.Text)
                ListBox1.Items.RemoveAt(CurDataRecord)
                ListBox1.SelectedIndex = CurDataRecord
            End If
        Next
        If ListBox1.SelectedIndex = -1 Then
            For CurDataRecord = 0 To CurrentListCount
                ListBox1.Items.RemoveAt(0)
            Next
        End If
        DrawRectangle(0, 0)
    End Sub

    Private Sub Button7_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button7.Click
        If Button7.Text = ">>" Then 'Play
            Timer1.Enabled = True
            Button7.Text = Chr(8) 'stop
        Else
            Button7.Text = ">>"
        End If
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        If Button7.Text = Chr(8) Then
            IncreasePoints(True, True)
        Else
            Timer1.Enabled = False
        End If
    End Sub

    Private Sub Button8_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button8.Click
        DecreasePoints(True, False)
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)       ↙
    Handles Button9.Click
        IncreasePoints(True, False)
    End Sub

    Private Sub Button11_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)      ↙
    Handles Button11.Click
        DecreasePoints(False, True)
    End Sub

    Private Sub Button10_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)      ↙
    Handles Button10.Click
        IncreasePoints(False, True)
    End Sub

    Private Sub TextBox2_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.      ↙
    KeyPressEventArgs) Handles TextBox2.KeyPress
        If Asc(e.KeyChar) = 13 Then
            UpdatePoints()
        End If
    End Sub

    Private Sub TextBox4_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.      ↙
    KeyPressEventArgs) Handles TextBox4.KeyPress
        If Asc(e.KeyChar) = 13 Then
            UpdatePoints()
        End If
    End Sub

    Private Sub Button12_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)      ↙
    Handles Button12.Click
        ListBox1.Items.Clear()
        DrawRectangle(0, 0)
    End Sub

    Private Sub Button13_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)      ↙
    Handles Button13.Click
        Dim PointString As String
        Dim xCoordinate1 As Double, yCoordinate1 As Double, xCoordinate2 As Double, yCoordinate2↙
     As Double
        If ListBox1.Items.Count = 2 Then
            'Supports only horizontal leveling
            PointString = Strings.Left(ListBox1.Items(0).ToString, Strings.InStr(ListBox1.Items ↙
    (0).ToString, " ") - 1)
            xCoordinate1 = Strings.Left(PointString, Strings.InStr(PointString, ",") - 1)
            yCoordinate1 = Strings.Right(PointString, Strings.Len(PointString) - (Strings.Len   ↙
    (CStr(xCoordinate1)) + 1))
            PointString = Strings.Left(ListBox1.Items(1).ToString, Strings.InStr(ListBox1.Items ↙
    (1).ToString, " ") - 1)
            xCoordinate2 = Strings.Left(PointString, Strings.InStr(PointString, ",") - 1)
            yCoordinate2 = Strings.Right(PointString, Strings.Len(PointString) - (Strings.Len   ↙
    (CStr(xCoordinate1)) + 1))
            Dim OppLen As Double, AdjLen As Double
            OppLen = yCoordinate2 - yCoordinate1
            AdjLen = xCoordinate2 - xCoordinate1
            'RotatePic(True)
            'Application.DoEvents()
            txtAngle.Text = Math.Tanh(OppLen / AdjLen) * (180 / Math.PI)
            RotatePic(False)
        Else
            MsgBox("Please only specify two points for leveling or enter a angle manually and   ↙
    press enter")
        End If
    End Sub
End Class

```

VITA

Stephen Byron Gauntt received his Bachelor of Science degree in Mechanical Engineering from Worcester Polytechnic Institute in February 2005. Upon graduation he entered the Mechanical Engineering graduate program at Texas A&M University. He also holds several software development certifications, primarily in the Microsoft ® Visual Basic ® programming language. He plans on finding a career which can utilize his expertise in Mechanical Engineering and software development.

Mr. Gauntt may be reached at 1114 Brookview Drive, Allen, TX 75002. His email address is sgauntt@alum.wpi.edu.