TÉCNICO LISBOA

Contact Angle influence in Capillary Flows

Diogo Dias Navarro Ferreira

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. José Maria Campos da Silva André
Dr. Domenico Fiorini

Examination Committee

Chairperson: Prof. Afzal Suleman
Supervisor: Prof. José Maria Campos da Silva André
Member of the Committee: Prof. Miguel Alfonso Mendez
Member of the Committee: Prof. Ana Sofia Oliveira Henriques Moita

December 2022

To my grandfather.

Acknowledgments

I would like to first start by thanking the Von Karman Institute for the opportunity to work on this topic and for welcoming me. In this regard, I would like to extend my thanks to Domenico Fiorini and professor Miguel Mendez for all the help in tackling this topic.

In addition to this, I would really like to thank professor José Maria André for the guidance during the entire process and for the valuable inputs.

Not least, I would like to extend some personal thanks to Pedro Afonso Marques who helped me a lot during the entire experimental campaign as well.

Finally, but definitely not least of all, I would like to acknowledge the role my family and my girlfriend played in the entire process supporting me through it, without them none of this would have been possible.

Resumo

Esta tese estuda o impacto da tensão superficial num escoamento “quasi-capilar” dominado por forças inerciais. Para analisar isto foram realizadas experiências em três montagens diferentes com o objetivo de testar o impacto da tensão superficial em várias geometrias. A primeira experiência realizada utilizou a técnica conhecida como “sessile droplet test” para obter o ângulo de contacto entre uma interface liquida e uma superfície sólida. Em condições estáticas, o “ângulo de contacto estático” é uma propriedade da combinação do fluído e do sólido em questão, que no caso deste teste se tratava de HFE7200 (“hydrofluoroether”) e uma placa de quartzo. A segunda montagem é um tubo em forma de “U” onde testes sobre a deformação da interface após a aplicação de um sobrepressão foram efetuados numa campanha experimental anterior. Os resultados desta campanha experimental são comparados com os resultados obtidos com a terceira montagem experimental e por isso requerem que a distorção ótica que acontece em tubos cilíndricos seja tida em conta e corrigida. A terceira montagem experimental consistia em duas placas paralelas com um pequeno canal no meio por onde uma interface gás-líquido subia propulsionada por uma sobrepressão. Esta montagem permitiu estudar os modelos integrais desenvolvidos para escoamentos em tubos capilares numa nova geometria de modo a testar o impacto da tensão superficial em escoamentos “quasi-capilares” dominados por forças inerciais.

Palavras-chave: Escoamento Capilar, Angulo de Contacto, Distorção Ótica

Abstract

This work addresses the impact of surface tension forces on the dynamics of an inertia-driven capillary rise of a gas-liquid interface. Several experimental campaigns were conducted, cross-checking the impact of surface tension forces on different geometries. The experimental campaigns used three different facilities. The first facility carried out sessile droplet tests to obtain the angle of contact between a liquid interface and a solid surface. In static conditions, the ”static contact angle” is a property of the fluid/solid combination, which in this work concerned HFE7200 (hydrofluoroether) and quartz surfaces. The second facility is a quasi-capillary U-tube set-up where tests on the interface deformation following a pressure step were performed in a separate experimental campaign. The results from the latter were compared with the ones obtained by the third facility studied in this work, hence requiring the characterization of the optical distortion in cylindrical tubes and the development of a corrective routine. The third facility concerns two parallel plates in close vicinity where a gas-liquid interface is forced to rise by means of a pressure step. The facility allowed to test integral models developed for capillary tubes to explore the possibility of measuring the impact of surface tension and the role of the contact angle on the inertia-driven capillary rise between the parallel plates.

Keywords: Capillary Flow, Contact Angle, Optical Distortion

Contents

List of Tables

List of Figures

Nomenclature

θ_s	Static contact angle
δ	Quasi-capillary channel width
θ_0	Camera misalignment angle
Ω	Angle relative to the camera axis
$pix2mm$	Pixel to milimetre ratio
L	Distance between the centre line of the tube and the aperture of the camera
Ca	Capillary number
θ_d	Dynamic contact angle
HFE7200	Hydrofluoroether 7200
σ	Surface tension
g	Gravity acceleration
μ	Liquid viscosity
K_H	Hagenbach coefficient
K_C	Couette coefficient

1

Introduction

Contents

This master's thesis was developed during a Short Training Program at the Von Karman Institute For Fluid Dynamics. The project aimed to compare the motion and shape of the liquid-gas interface in different geometries, to gauge the impact that geometry has on the prevailing models for capillary follow.

Several configurations were considered, the first one was the quasi-capillary parallel plates. This experimental set-up consists of a channel formed by two parallel quartz plates at a distance of 5 mm and closed at both ends. This channel was immersed in a large liquid bath at one end and open to the atmosphere on the other. The reservoir where the liquid bath is housed is then pressurised forcing a liquid column to rise through the channel, where the interface is observed and measurements are taken.

After building up a database for the interface evolution, the data gathered was compared with a previous experimental campaign performed in the quasi-capillary U-tube. This facility is made of a U-shaped cylindrical quartz tube with an inner diameter of 8 mm. However, the images collected from this experimental campaign had to be corrected for optical distortion first, so the existing corrections for optical distortion in the literature were also tested during the experimental campaign conducted in this thesis.

Sessile droplet measurements were also carried out to compare the contact angle measured with this method to the one obtained in images of the static interface in the former experimental campaigns.

The contact angle parameter mentioned previously played a big role during the experimental campaigns conducted for this thesis. In static conditions, as is the case of a sessile droplet, the *equilibrium* contact angle, θ_{Young}, is given by Young's relation $\gamma_{SV} - \gamma_{SL} = \sigma cos(\theta_{Young})$, where γ_{SV} and γ_{SL} are the solid-vapor and solid-liquid interface energies respectively and σ is the surface tension. θ_{Young} varies largely depending on the characteristics of the substrate and the atmosphere surrounding the droplet, consequently experiments on sessile droplets characterise only the *static* contact angle θ_S. [1] [2]

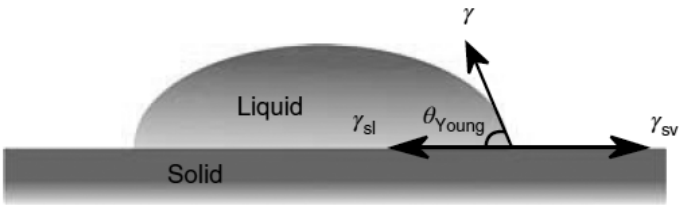


Figure 1.1: Visualisation of the *static* contact angle of a generic droplet [1]

However, this changes for a moving contact line, where the value of the *dynamic* contact angle, θ_d, will be a function of other parameters such as the velocity and acceleration. This dependence will be evaluated through the use of correlations found in the literature and developed over the course of other experimental campaigns conducted in the Von Karman Institute under *Project Slosh II* and more specifically Domenico Fiorini's PhD work. [3]

The objective behind testing these correlations is to ultimately test their accuracy in describing the

contact angle variation with time. Once this is achieved, they can be incorporated into the integral model for the motion of the interface to complement the term related to the surface tension, which depends on the contact angle. This opens up the possibility of characterising the motion of the interface by optimising the coefficients of the correlation to fit the interface height data. This will be looked at in greater detail in the chapter 5.

The impact of better modelling for phenomena related to capillary flows and surface tension dominated motion can not be understated. The liquid sloshing motion studied at the Von Karman Institute in the *Project Slosh II* and *Project Minerva* has implications in diverse industrial fields.

One such example is liquid storage in overhead reservoir tanks where the load on supporting beams and structures can be up to three times higher for sloshing when compared to the hydrostatic load. Thus, better modelling for the vibration modes could help improve earthquake safety for such structures either in a residential or an industrial context. [4]

In the aerospace industry, these types of flow play an important role in orbital manoeuvres where the fuel and liquids stored within the spacecraft tanks are not bounded by gravity. This means that the liquid sloshing motion occurs in low Bond numbers, i.e. that the flow is capillary dominated and the prevailing forces are surface tension mechanisms, which can be modelled using the contact angle. A study was performed for the Ariane 5 A5ME upper stage where the controller struggled to cope with the time dependant inertia tensor due to the sloshing motion under a slew manoeuvre, this led to a high number of thrust activations. Better control and understanding of this phenomena would lead to quicker manoeuvres, settling times and higher fuel efficiency in these sorts of spacecraft. [5]

To round this off, capillary flow and the contact angle also play an important role in several common manufacturing and industrial processes, such as coatings, powder processing and fuel injector manufacturing, which would see an increase in efficiency with better modelling. [6] [7] [8]

1.1 Thesis Layout

This document is laid out to try to answer the underlying question of the influence geometry has on the contact angle parameter and, thus, on capillary flow.

Firstly, modelling is addressed. In chapter 2, a model based on the Young-Laplace equation for the liquid droplet shape is derived and discussed. In addition to this, the existing integral models for the motion of a liquid column within capillary tubes are also presented. Then, with the existing models in mind, an equation for the quasi-capillary parallel plates set-up is derived.

Chapter 3 ties both dynamic contact angle campaigns together, by discussing the optical corrections applied to the data of the quasi-capillary U-tube. In this chapter, the correction equations are presented; the influence of several parameters on the optical distortion is discussed as well as the predicted impact on the data collected from the U-tube experimental campaign.

In chapter 4 the experimental set-ups used to achieve the goals of this experimental campaign are presented as well as a brief overview of the methodology used to process the data collected and extract the results.

These results are then analysed in chapter 5, where a comparison between the different facilities is established and the correlations for the dynamic contact angle found in the literature and developed in the previous experimental campaign are presented and discussed. In this chapter, these correlations and the integral model developed in chapter 2 are combined and analysed in light of the experimental results.

This culminates in a conclusion where remarks are made about the results obtained by answering the questions:

- Did the prevailing model for optical correction yield the expected results?
- Is the static contact angle similar across different geometries?
- Are the prevailing integral models adequate to describe the motion of the liquid column with the Quasi-Capillary Parellel Plates?
- Are the capillary models sensitive to an optimisation based on the correlation's coefficients?

Finally, comments on future work to advance knowledge related with these research questions.

2

Modelling

Contents

During the work conducted for this thesis, both *static* and *dynamic* phenomena were analysed. Both need to be properly modelled to gauge the influence of each of the parameters on their behaviour, namely the dynamic contact angle and the correlations that describe its relation with the liquid column motion.

During this chapter, each model will be discussed, and the reasoning behind their use will be presented.

2.1 Sessile Droplet Test

During the course of the experimental work for this thesis, a sessile droplet test campaign was conducted, which aimed to determine the static contact angle for HFE7200, an abbreviation for hydrofluoroether. HFE7200 is a synthetic fluid mimicking the properties of cryogenic propellant (e.g. liquid oxygen). The tests consisted in recording a video of a drop of liquid on a quartz plate and then determining the contact angle through a fitting procedure. [1]

Thus, the equation to fit the droplet surface is derived with the appropriate parameters and forces that model its shape. The first force to be considered is the one produced by the hydrostatic pressure of the droplet. The second force is surface tension which balances the hydrostatic pressure. This balance is the macroscopic manifestation of the attractive and repulsive forces at a molecular level which gives the droplet and a meniscus its shape. The balance between these two quantities was established in equation 2.1. [9] [10]

$$\rho g z = \sigma \vec{\nabla} . \vec{n} \tag{2.1}$$

Equation 2.1 is known as the Young-Laplace equation, and it accounts for the pressure difference across the interface on the left side of the equation and balances this term on the right side of the equation with the surface tension.

Then, the normal vector to the surface must be derived to obtain the final form of equation 2.1. In this case, the coordinate system for the droplet is first defined as $z = h(r, \theta)$, where z, r and θ form a cylindrical coordinate system, which is described in image 2.1.

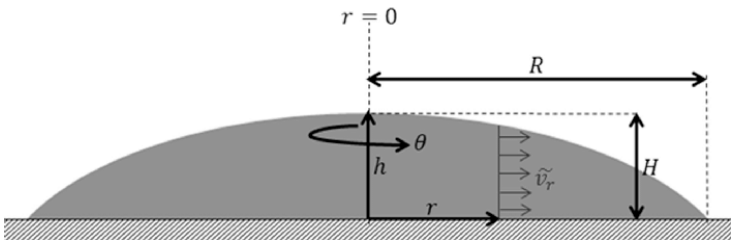


Figure 2.1: Coordinate system droplet [11]: h - coordinate for the distance between the liquid/gas interface and the solid surface; r - radial coordinate; θ - axial coordinate; R - radius of the droplet; H - height of the droplet

A functional that is equal to zero on the surface of the droplet was defined as $f(z, r, \theta) = z - h(r, \theta)$ and, thus, one can obtain the expression for the unit normal vector to the interface, using equation 2.2. [10]

$$\vec{n} = \frac{\vec{\nabla}.\vec{f}}{\left\|\vec{\nabla}.\vec{f}\right\|} \tag{2.2}$$

After setting up this equation in terms of its independent variables, one arrives at equation 2.3 for the curvature of the surface, $\vec{\nabla}.\vec{n}$. This equation also accounts for the axial-symmetric nature of the droplet interface; thus, all terms with respect to the variation of θ will disappear. All the steps to arrive at this equation are laid out in appendix 7.

$$\vec{\nabla}.\vec{n} = \frac{-rh_r - r^2 h_{rr}}{r^2(1 + h_r^2)^{3/2}} \tag{2.3}$$

Taking equation 2.3 and substituting it into equation 2.1, the differential equation for the surface shape of the droplet is obtained, where $l_c = \sqrt{\frac{\sigma}{\rho g}}$ and is defined as the capillary length. This quantity is directly linked to the balance between surface tension and gravity.

$$h_{rr} = \frac{h}{l_c^2}(1 + h_r^2)^{3/2} - \frac{h_r}{r} \tag{2.4}$$

From the Neumann-type boundary conditions, expressed in 2.5, we can obtain the static contact angle by fitting the experimental points of the surface with this equation. In this process, the boundary conditions in 2.5 are used to solve equation 2.4 numerically. The contact angle used in these boundary conditions will then be optimised to minimise the difference between the experimental data and this numerical solution. This will be elaborated on further in chapter 4.

$$\begin{cases} h_r = 0, & r = 0 \\ h_r = -tan(\theta_{static}), & r = R \end{cases} \tag{2.5}$$

Figure 2.2 shows a plot of a hypothetical fit to a droplet with a radius of $2.5mm$ using a contact angle of $7°$. The surface's slope near its hedges corresponds to the tangent of the static contact angle in the

plot show in figure 2.2.

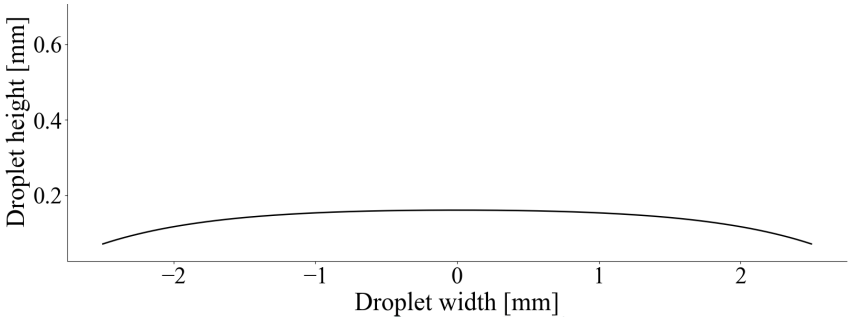


Figure 2.2: Example of the curve generated by the Young-Laplace equation using cylindrical coordinates

2.2 Symmetric meniscus

The same procedure is applied to the meniscus in the quasi-capillary U-tube, which shares a similar coordinate system described in figure 2.3.

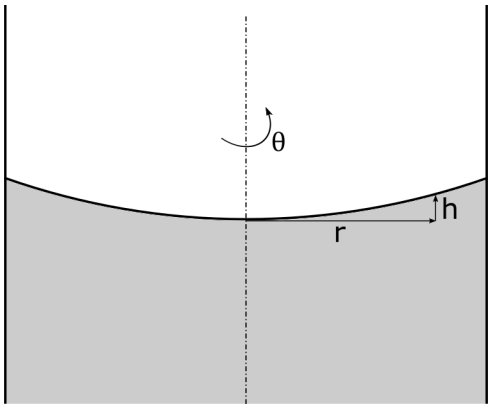


Figure 2.3: Coordinate system for the quasi-capillary U-tube interface (cylindrical coordinates): h is the coordinate for the height of a generic point with respect to the lowest point in the meniscus; r is the radial coordinate; θ is the axial coordinate

In this case, the boundary conditions are described in 2.6, where r_i represents the inner tube radius and h_r represents the spatial derivative with respect to r.

$$\begin{cases} h_r = 0, & r = 0 \\ h_r = 1/tan(\theta_{static}), & r = r_i \end{cases} \tag{2.6}$$

The quasi-capillary parallel plates rely on a Cartesian coordinate system to describe the bi-dimensional approximation of the interface. This coordinate system leads to a slightly different equation to describe its interface and boundary conditions to match it, as seen in 2.7 and 2.8, where w corresponds to the width of the channel.

$$h_{xx} = \frac{h}{l_c^2}(1 + h_x^2)^{3/2} \tag{2.7}$$

$$\begin{cases} h_x = 0, & x = 0 \\ h_x = 1/tan(\theta_{static}), & x = w/2 \end{cases} \tag{2.8}$$

The coordinate system used to derive this equation is described in figure 2.4.

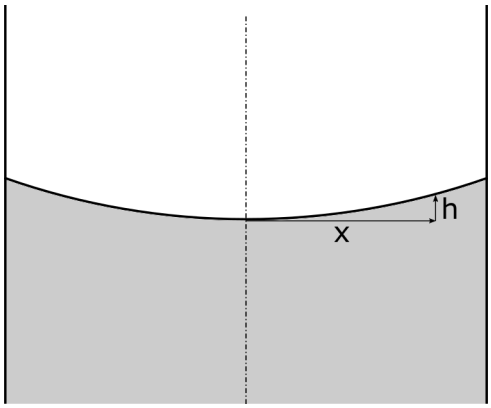


Figure 2.4: Coordinate system for the quasi-capillary parallel plates interface (bi-dimensional interface - fully developed flow along the length of the length of the channel): h is the coordinate for the height of a generic point with respect to the lowest point in the meniscus; x is the distance to the centre of the channel

2.3 Dynamic Capillary Motion

The dynamic formulation for capillary phenomena is much more complex than its static counterpart. One of the main difficulties lies in the fact that, as of the making of this thesis, there is no theoretical formulation to describe the shape of the dynamic two-phase flow interface.

Traditional models for the surface tension forces at the interface and the dynamic contact angle have characterised mainly small test-case applications dominated by capillary and viscous forces, such as capillary tubes [12] [13] [14]. However, emerging technologies for space applications require a deep understanding of free interface flows driven by the balance of capillary and inertial forces. Thus, the formulation of capillary-driven flows is extended for the case of forced pressure rise, and the impact of the contact angle within the surface tension term is evaluated on the interface rise.

2.3.1 Existing Models

One of the most commonly used models for capillary flow is the Lucas-Rideal-Washburn, equation 2.9, introduced in Washburn [15] and Deutsch [16]. This equation is of great significance to the work carried out during this thesis since all of the papers researched for establishing the model used reference this

equation, and most of its terms will appear in the final model.

$$\frac{dl}{dt} = \frac{1}{8\mu l}[\Delta P' - \rho g l] r_o^2 \tag{2.9}$$

In equation 2.9, l refers to the length of the liquid column within the capillary tube; μ is the fluid viscosity; $\Delta P'$ is the capillary pressure; ρ is the liquid density; g is the gravitational acceleration, and r_o is the tube radius.

This equation is derived from Poiseuille's Law, equation 2.10, by substituting its volumetric flow rate term by its cross-sectional area and the length of the liquid that has entered the capillary, expressed in equation 2.11.

$$\frac{dQ}{dt} = \frac{\pi \Delta P r_o}{8\mu l} \tag{2.10}$$

$$dQ = \pi r_o^2 \frac{dl}{dt} \tag{2.11}$$

In these equations, Q is the volumetric flow rate; ΔP is the pressure drop; r_o is the radius of the tube; μ is the viscosity of the fluid, and l is the length of the tube.

However, some authors have improved this formulation over the years to include a complete description of the phenomena. The most common issue with this equation is its inability to accurately account for the initial stages of the liquid rise in the tube since it results in an infinite velocity and acceleration, i.e. for $l \to 0$, $\frac{dl}{dt} \to \infty$. This overshoot is the effect of the steady-state approximation used to derive it. [12]

Several proposed solutions account for the entry flow from the reservoir and dissipative effects caused by circulation at the entrance to the capillary tube. J. Szekely, A. W. Neumann and Y. K. Chuang were the first to propose a solution incorporating all the terms the final model should include, in reference [13]. This model rests on an energy balance conducted to the control volume described in figure 2.5.

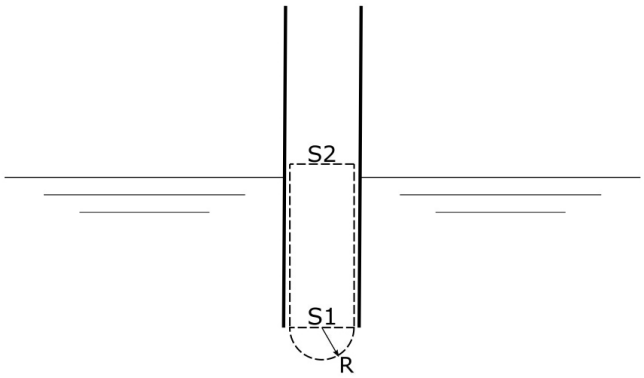


Figure 2.5: Control volume used to formulate the model for the flow within the capillary tube: capped by surface *S1* and *S2* and limited laterally by the tube walls

The control volume used to model the liquid rise in the capillary tube represented in figure 2.5 is limited by the walls of the tube and the surfaces *S1* and *S2*. The semi-circle designed at the bottom of the tube serves as the control volume, used to obtain the pressure at *S1*. In the context of the quasi-capillary parallel plates, this semi-circle represents a cylinder cut in half in its longitudinal direction. For the equations and coefficients derived for the the following models, it represents the bottom half of a sphere since the geometries used are capillary tubes.

The integral energy balance is postulated as between *[rate of change of kinetic energy + potential energy of the control volume]* and *[net input of kinetic energy - potential energy + pressure energy] - [rate of work done on the surroundings] + [rate of work dissipated irreversibly]*. This energy balance results in equation 2.12.

$$(h + 7/6r_o)\frac{d^2h}{dt^2} + 1.225(\frac{dh}{dt})^2 + \frac{8\mu}{\rho r_o^2}h\frac{dh}{dt} = \frac{1}{\rho}[\Delta P - \rho g h] \tag{2.12}$$

In equation 2.12, l is replaced by h as the length of the liquid column in the capillary. The first term of equation 2.12 on the left side of equation 2.12 is related to the rate of change of kinetic energy. This term also includes an adjustment for the flow converging on the entrance coming from the reservoir, this equates to adding an effective height increase of $\frac{7}{6}$ of the width of the tube. This term also solves the previously mentioned problem of the Lucas-Rideal-Washburn equation since it prevents the existence of an infinite velocity/acceleration when h tends to zero. While the second term is related to the pressure loss due to circulation at the entrance of the tube. The third term is associated with viscous forces' work on the tube walls. Finally, the term on the right side is related to the work done by the gravity force and surface tension.

However, this model is used for higher Reynolds numbers than those generally encountered in experimental campaigns on capillaries due to the formulation used for its dissipative term and some authors

have pointed out that the added quantity due to flow coming from the reservoir is not correct.

Given this, the model presented will be based on the work done by S. Levine, P. Reed and E. J. Watson to come up with a different equation that more accurately describes the rise through the tube. [12]

To get the model, it becomes essential to understand the origin of its coefficients and how they are obtained.

For this derivation, the same control volume was applied to a capillary tube, but in this instance, a momentum balance will be used. With this model, a similar equation is obtained. However, the differences lie in the coefficients used for the term concerning the flow converging into the tube and the dissipation term due to the *vena contracta* phenomena. This paper will also be a reference since it outlays all the steps to reach the coefficients used in the model equation.

For the term regarding the converging flow, a momentum balance is established in the semi-sphere where the change in momentum balances the pressure acting on this frontier.

The momentum balance established can be written as such: *[rate of change of total momentum in the system] = [flux of momentum entering] - [flux of momentum leaving]- [pressuring acting in the vertical direction at h = 0] + [pressure acting in the vertical direction at R =* r_o*]*.

The objective of establishing this balance is to obtain the previously mentioned pressure at surface $S1$, i.e. $h = 0$, represented in figure 2.5. Thus, two boundary conditions are imposed to get an expression for the radial velocity. The first is that the radial velocity will tend to zero as the sphere's radius at the bottom of the tube is expanded. This condition makes sense since it is equivalent to stating that the velocity will be zero far away from the tube entrance. Secondly, for a radius equal to that of the tube, the influx coming into the tube must be equal to that coming into the sphere, which is achieved in equation 2.13.

$$-2\pi r_o^2 \int_0^{\frac{\pi}{2}} sin\theta [radial\ velocity]_{R=r_o} d\theta = \pi r_o^2 \frac{dh}{dt} \tag{2.13}$$

With these conditions, it is possible to solve the Navier-Stokes equation for the radial velocity, which only depends on the radius and time. This will, in turn, result in an expression for the pressure term across the semi-sphere. This term accounts for the term concerning the *[pressure acting in the vertical direction at R =* r_o*]* in the previously established balance.

An average acceleration is considered to get the term *[rate of change of total momentum in the system]*. To achieve this, an average between the average acceleration calculated for the tube entrance at $h = 0$, using Poiseuille flow as the velocity profile within the capillary tube, and the average acceleration for the hemisphere at $R = a$, taking into account the previous expression obtained for the radial velocity are considered.

For the term concerning the fluxes of momentum through the control volume, *[flux of momentum*

entering] - [flux of momentum leaving], similar considerations are taken into account.

This is the basis for the added mass term that is used to model the motion for the capillary channel and is adapted to include only a term for the over-pressure, which is not included in these exclusively capillary-driven flows. This approximation will be later evaluated and discussed in chapter 5 when the influence of the parameters is discussed.

The result of all these operations is a change in the coefficient regarding the inertia of the fluid near the entrance of the tube from $\frac{7}{6}r_o$ to $\frac{37}{36}r_o$. More recent works, such as Stange et al. [14], have suggested that this term should be $\frac{73}{60}$. This value will be used to model the flow behaviour in the capillary channel.

Another difference between the two previously mentioned models introduced in references Levine et al. [12] and Szekely et al. [13] is the dissipative terms used. Both formulations use the Hagenbach and Couette corrections which relate the behaviour of the liquid and the contraction ratio upstream of the tube's entrance. In Szekely et al. [13], only the Hagenbach coefficient, K_H, is used. This is a good approximation for higher Reynolds numbers since the Couette coefficient becomes less significant, as seen in equation 2.14. [17] [18]

$$\frac{\Delta p}{\rho V^2/g} = K_H + \frac{K_C}{Re} \tag{2.14}$$

However, for the range of Reynolds Numbers typical of capillary-driven flows, the Couette correction, K_C, becomes quite significant and can not be ignored.

These considerations result in a final model for the motion in a capillary tube, represented in equation 2.15, which is the basis for the one obtained in the following section.

$$\rho r_o^2(h + \frac{37}{36}r_o)\frac{d^2h}{dt} = 2r_o\gamma - 8\mu h\frac{dh}{dt} - \rho r_o^2 gh - \frac{1}{2}\rho r_o^2\left[K_H\dot{h}^2 + \frac{\mu K_C}{\rho r_o}\dot{h}\right] \tag{2.15}$$

To compare the Lucas-Rideal-Washburn equation with the model suggested in equation 2.15, both of these equations were solved using the properties of Di-propylene Glycol and considering a capillary tube with a radius of $0.05mm$. This radius is of the same order of magnitude as the capillary length, which means that capillary phenomena will dominate this flow. The results for these equations are displayed in figure 2.6.

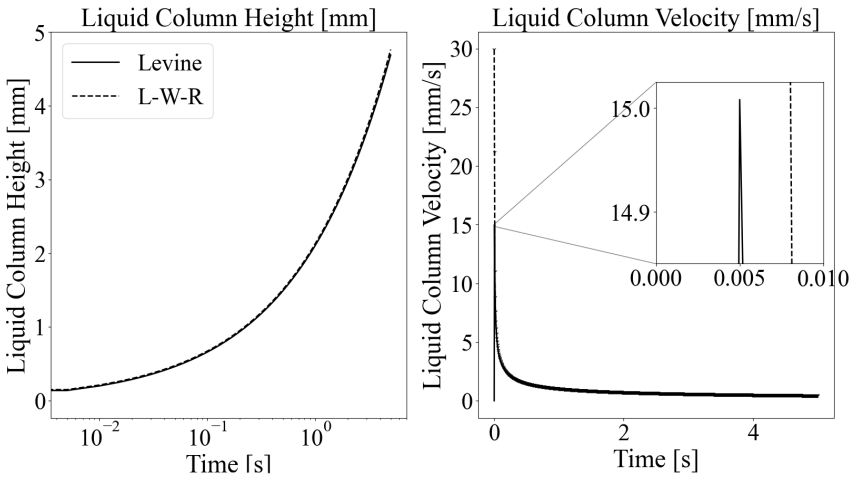


Figure 2.6: Solution of the Lucas-Rideal-Washburn and Levine equations: a logarithmic scale was applied the time axis for the plot of the interface height; both plots share the same curve labelling

Figure 2.6 displays similar results for both models. Despite this, the role of the added mass term is observed in the difference between both models in the initial stages of the motion. While the initial velocity of the Lucas-Rideal-Washburn equation tends to be infinite, the velocity predicted by Levine's equation is bounded. This singularity leads to an initial height offset being considered for the Lucas-Rideal-Washburn equation. In addition, the added term of the pressure dissipation at the entrance to the tube ensures that the liquid column height predicted by Levine's model is ever so slightly smaller.

2.3.2 Model Equation for the Quasi-Capillary Channel

Based on the models described in the previous section, it is now possible to establish that the control volume used will be represented in figure 2.5. The axis for this figure will be at the bottom of the channel, as illustrated in figure 2.7.

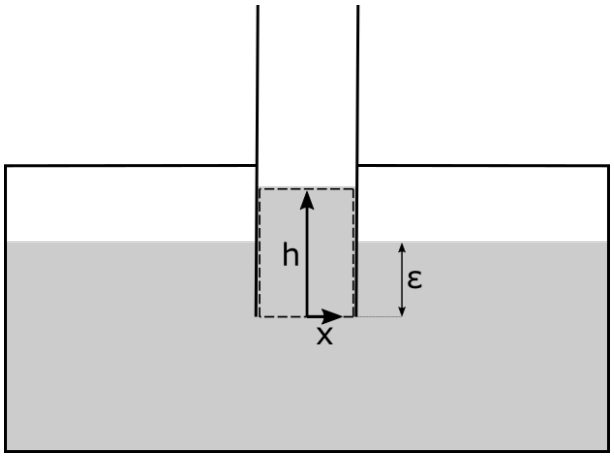


Figure 2.7: Model for the quasi-capillary parallel plates: x is the coordinate for the width of the channel; h is the coordinate that accounts for the distance from the bottom of the channel; ϵ accounts for the difference between the surface of the liquid and the entrance to the channel

Equation 2.16 defines an integral force balance that is applied to the control volume defined by the dashed lines in figure 2.7.

$$\frac{d}{dt}(m\dot{h}) = \sum_i \vec{F}_i \tag{2.16}$$

Firstly, the term on the left-hand side of equation 2.16 must be analysed. There are two ways to consider this term. It can either be admitted that the infinitesimal volume of fluid, $dV = \delta L dh$, acquires its velocity instantaneously upon entry into the control volume, which would result in equation 2.17, or that this infinitesimal volume added to the control volume at time Δt has the fully developed velocity $\dot{h}$, which results in equation 2.18.

$$\rho\delta L \lim_{\Delta t\to 0}\left[\frac{(h+\Delta h)\frac{d}{dt}(h+\Delta h) - h\frac{dh}{dt}}{\Delta t}\right] = \rho\delta L\frac{d}{dt}(h\frac{dh}{dt}) \tag{2.17}$$

$$\rho\delta L \lim_{\Delta h\to 0}(h+\Delta h)\times\lim_{\Delta t\to 0}\left[\frac{\frac{d}{dt}(h+\Delta h) - \frac{dh}{dt}}{\Delta t}\right] = \rho\delta Lh(\frac{d^2h}{dt^2}) \tag{2.18}$$

The latter formulation was chosen for the following modelling since it is featured in both Lucas-Rideal-Washburn and Levine's models. [16] [12]

As discussed previously, the forces that must be considered for this balance are the pressure force acting on the lower surface, *S1*; the pressure force applied on the upper surface, *S2*; the surface tension acting on surface *S2*; the weight of the liquid column; the viscous forces on the walls and the pressure loss due to a sudden contraction of the flow coming in from the reservoir.

The pressure on the lower surface is related to the term derived for the converging flow on the tube entrance. The inclusion of the term for the over-pressure in the reservoir in this formulation is achieved

by changing the pressure at the surface of the reservoir to include it. This change results in the same formulation seen in the previous section, where the deviation from the hydrostatic pressure on the tube entrance will be offset by an effective increase in the liquid column inertia. However, in this instance and since the pressure at the reservoir's surface is different from the one on top of the liquid column this term concerning the pressure difference between the two appears in the final equation. Thus, an increasing the inertia by $\frac{73}{60}\frac{\delta}{2}$, is observed which is included in term derived in equation 2.18 changing it to 2.19. With this, the deviation from hydrostatic pressure is accounted for, and it is possible to obtain the pressure for surface *S1* as the hydrostatic pressure in equation 2.20.

$$\frac{d}{dt}(m\dot{h}) = \rho\delta L(h + \frac{73}{60}\frac{\delta}{2})\ddot{h} \tag{2.19}$$

$$F_{pressureS1} = \delta L(p_a + \Delta p + (\rho g\epsilon)) \tag{2.20}$$

In equation 2.20, ϵ represents the distance to the surface, seen in figure 2.7, which is considered to be approximately constant and p_a represents the ambient pressure.

The pressure acting on the upper surface is much easier to obtain since this portion was open, and it can be considered to be at ambient pressure. Thus, the force acting on the upper surface due to ambient pressure is described in equation 2.21.

$$F_{pressureS2} = -\delta L(p_a) \tag{2.21}$$

A circular shape will approximate the meniscus surface for the surface tension term. This approximation is the most common for the meniscus shape, and together with the static formulation for the meniscus using the Young-Laplace equation, it yields equation 2.22. In this equation, θ is the contact angle, and σ is the surface tension. [19] [20]

$$F_{SurfaceTension} = (\delta L)(\frac{2}{\delta}\sigma cos(\theta)) = 2L\sigma cos(\theta) \tag{2.22}$$

This formulation is used since there was no available dynamic formulation for the meniscus. Furthermore, in reference Petrov and Sedev [9], it is argued that a fit of the meniscus with the static formulation based on the Young-Laplace equation would yield good dynamic contact angle results.

The weight of the liquid column is readily obtained using the liquid density, ρ, in equation 2.23.

$$F_{WeightColumn} = -\rho L\delta gh \tag{2.23}$$

Then, the velocity profile has to be determined to calculate the forces acting on the fluid due to the viscous forces on the wall. In capillary tubes, the velocity profile is typically assumed to be that of a Poiseuille flow.

Thus, the viscous force, assuming that HFE 7200 is a Newtonian fluid, is given by equation 2.24. [21]

$$F_{ViscousForce} = (Lh)\mu\frac{du}{dx}(x = \frac{\delta}{2}) \tag{2.24}$$

The velocity profile for a Couette flow is given by equation 2.25. [22]

$$u_y = u_{max}(1 - \frac{x^2}{(\delta/2)^2}) \tag{2.25}$$

To include this in the model for the capillary channel, this velocity profile must be related to the average velocity for the liquid column. This is achieved through equation 2.26.

$$\dot{h} = \frac{1}{\delta}\int_{-\frac{\delta}{2}}^{\frac{\delta}{2}} u_{max}(1 - \frac{x^2}{(\delta/2)^2})dx = \frac{2}{3}u_{max} \tag{2.26}$$

Thus, the final expression for the viscous force is obtained in equation 2.27.

$$F_{ViscousForce} = (2Lh)\mu\frac{du}{dx}(x = \frac{\delta}{2}) = -(2Lh)\mu u_{max}(\frac{2\frac{\delta}{2}}{(\delta/2)^2}) = -\frac{3(2Lh)\mu\dot{h}}{2}(\frac{\delta}{(\delta/2)^2}) = -\frac{12\mu L}{\delta}h\dot{h} \tag{2.27}$$

The pressure loss due to the sudden contraction must also be accounted for in this case. However, most of the formulation for this phenomenon found in the literature centres around the usage of the Hagenbach and Couette corrections described before, which are used for capillary tubes, which have a different geometry from the capillary channel. Nonetheless, this correction is introduced in equation 2.28 as represented in Dutt [23].

$$\frac{\Delta p_{contraction}}{\rho v^2/2} = K_H + \frac{K_C}{Re} \tag{2.28}$$

In equation 2.28, v is the mean velocity of the liquid column entering the tube; K_H is the Hagenbach correction coefficient; K_C is the Couette correction coefficient, and Re is the Reynolds number.

Using these definitions, it is possible to calculate the force acting on the control volume caused by this pressure loss. This is done in equation 2.29, where v is substituted by $\dot{h}$, the Reynolds number is broken down into its components, and the pressure loss is multiplied by the channel area.

$$F_{contraction} = \delta L\Delta p_{contraction} = \delta L(\rho\dot{h}^2/2)(K_H + \frac{K_C}{\frac{\rho\dot{h}\delta}{\mu}}) = \frac{\rho\delta L}{2}(K_H\dot{h} \mid \dot{h} \mid + \frac{\mu K_C}{\delta\rho}\dot{h}) \tag{2.29}$$

As an initial approximation, the coefficients considered for this term will be the ones used in Levine's equation as the values used in those equations match correlations used in later works such as Dutt's work Dutt [23]. So, the Hagenbach correction used is $K_H = \frac{7}{3}$ and the Couette correction is $K_C = 4$. [12]

With all the forces taken into account, it is now possible to get a complete model for the interface motion in the capillary channel, equation 2.30.

$$(h + \frac{73}{60}\frac{\delta}{2})\ddot{h} = \frac{\Delta p + \rho g \epsilon}{\rho} + \frac{2\sigma cos\theta}{\rho\delta} - gh - \frac{12\mu}{\rho\delta^2}h\dot{h} - \frac{1}{2}(K_H\dot{h} \mid \dot{h} \mid + \frac{\mu K_C}{\delta\rho}\dot{h}) \tag{2.30}$$

This model is a non-linear second order differential equation and in the interest of predicting its behaviour pressure steps corresponding to the experimental points considered in chapter 4. In addition, all the pertinent properties for HFE 7200 were considered, and a static contact angle obtained from prior experimental campaign of 27.5 degrees was taken into account. The results are represented in figure 2.8. [21]

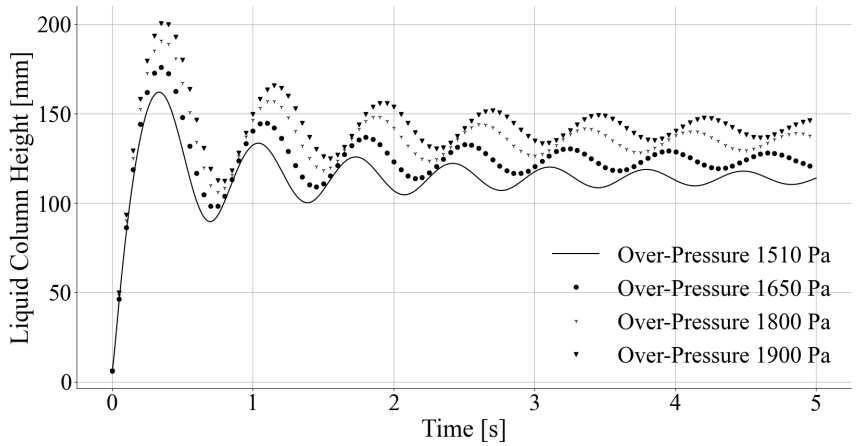


Figure 2.8: Model response pressure-step inputs of $1510Pa$, $1650Pa$, $1800Pa$ and $1900Pa$

Figure 2.8 shows that this system exhibits the same behaviour as a second-order under-damped system when subjected to a step impulse, which makes sense when considering the nature of the pressure impulse and the viscous effects. [24] [25]

3

Optical Deformations and Corrections for Liquid Images in Straight Channels

Contents

In everyday life, the apparent deformation of a drinking straw inside a glass of water represents the most common optical deformation due to the different refractive indexes of the materials involved (gas, liquid and solid) and the curvature of the glass. In this case, the initial straight straw, once immersed in water, appears to be bent at the contact point with the liquid interface.

This phenomenon also happens when observing a meniscus inside a straight tube, as seen in figure 3.1, where a small piece of plastic with a grid-like pattern was inserted. Upon closer observation, one can see that the vertical line number 6 is being distorted going through the fluid.

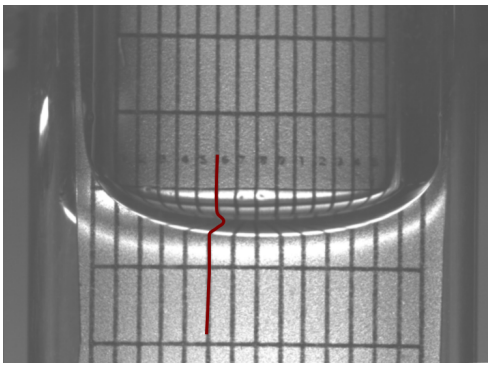


Figure 3.1: Image of the straight section of U-tube facility where a grid has been placed and distorted by its curvature

In the experimental campaigns conducted, the shape of the gas-liquid interface of a liquid (meniscus) was observed, and just like in the case of the drinking straw in the glass, the form of the tube distorts the meniscus image in the radial direction of the channel itself. Thus, ignoring distortion effects would lead to an incorrect meniscus shape and, consequently, the wrong contact angle with the solid surface.

3.1 Context

The results of the experimental campaign conducted in the quasi-capillary parallel plates will be compared with those obtained from the experimental campaign conducted in a facility that consisted of a cylindrical tube shaped like a ”U”, henceforth referred to as ”U Tube”. Given the problem mentioned above of optical distortions in cylindrical channels, this chapter focuses on the experimental campaign conducted within this tube, described in chapter 4.

During this experimental campaign, the meniscus movement inside the U Tube is observed by a camera perpendicular to the U Tube, which is illuminated by a uniform, diffused light source on the opposite side of the camera. This results in the observed meniscus being a blacked-out area on the image, which is cast to the tube wall closer to the camera. This shadow is distorted due to the different refractive indexes of HFE, quartz and air and the curvature of the tube wall.

Since the shape of the meniscus interface corresponds to this shadow, it is imperative to correct it

to obtain the right contact angle. Furthermore, it is essential to characterise optical distortion to get this contact angle. The ray-tracing techniques employed in Darzi and Park [26] and in Lowe and Kutt [27] will be compared against the results obtained from correcting distorted geometries within the tube.

3.2 Problem Geometry

In the U-tube experiment, a generic light ray belonging to a diffused light source comes through the air into the tube, where it goes through the glass and then into the fluid, then finally it reaches the camera after having gone through glass and air again, as illustrated in figure 3.2.

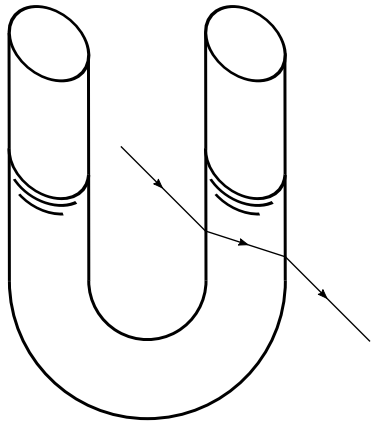


Figure 3.2: Light ray passing through the U Tube facility

Notably, the path of the rays distorts the image's proportions in the radial direction. While in the axial direction, the image conserves its proportions. Even if any distortion due to differing refraction indexes would occur, it would simply magnify or shrink the image since there is no curvature in this direction. This behaviour is explained in section B of Darzi and Park [26].

To trace the path of a generic light ray through a transverse section of the U tube depicted in figure 3.2. This path is traced from the light source, represented on the left side of figure 3.3, to the camera on the right side.

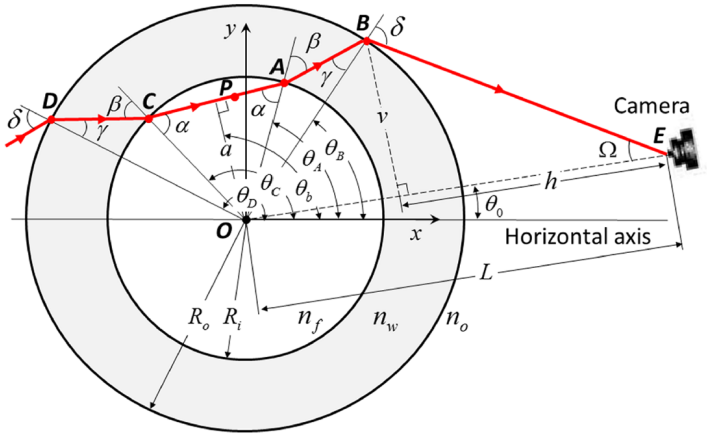


Figure 3.3: Generic light ray passing through a transversal section of cylindrical tube, where the corresponding refraction and angles with the horizontal axis are represented, figure from [26]

Figure 3.3 considers the camera sensor as a single point, which allows each light ray to be associated with its angle relative to the camera axis, Ω. Thus, through Snell's Law each of these angles has a set of accompanying refraction angles(δ, γ, α and β) that is unique to each value of Ω. Given this, it is possible to conclude that in this formulation, each point that is observed along the surface of the cylindrical tube has only one light ray coming through it that reaches the camera.

Figure 3.3 makes it clear that to account for the effect of optical distortion in each of these points, point B, which will be the position of the meniscus captured by the camera, has to be transformed to point A, that corresponds to the position of the meniscus on the inner tube wall. However, the images represent a bi-dimensional projection of this geometry. Thus one can not simply compute the distance between points A and B but the projection of this distance onto the plane of the camera sensor. This projected distance is calculated by relating the position of points A and B through the refraction angles, distance from the camera sensor and the tube dimensions to obtain the length of segment $\overline{BM}$ as represented in figure 3.4. The length of segment BM represents the distance between points A and B projected onto the plane of the camera sensor.

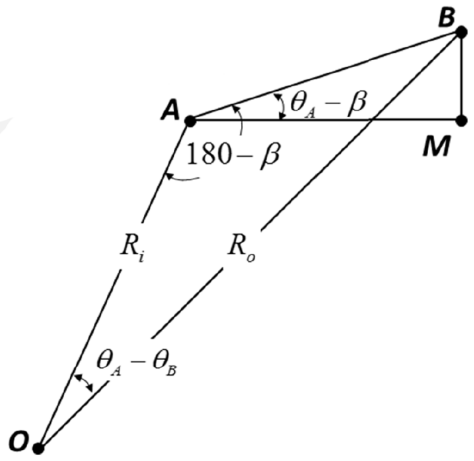


Figure 3.4: Geometric construct relating points A and B (0 represents the center of tube) with their projection onto the camera's aperture plane, figure from [26]

Then, considering both triangles formed in this geometrical construct and the sine law, the distance $\overline{AB}$ can be calculated and then projected onto $\overline{BM}$, which will be the distance projected onto the camera plane sensor, as explained in equations 3.1 and 3.2.

$$AB = R_o \frac{sin(\theta_A - \theta_B)}{sin(180 - \beta)} \tag{3.1}$$

$$BM = ABsin(\theta_A - \theta_B) \tag{3.2}$$

The parameters in equations 3.1 and 3.2 are obtained using Snell's law and trigonometric relations.

The refraction angles are derived from the prior knowledge of the distance from the centre of the tube to the camera sensor and Ω, as well as the inner and outer radius of the tube.

The angles with the horizontal axis, θ_A and θ_B can be derived through trigonometric relations, and their equations depend only on the previously obtained refraction angles, Ω and the camera misalignment angle, θ_0 which is the angle between the normal vector the plane formed by the camera sensor and the one from the diffuse light source.

All derivations of the expressions for these angles can be found in 8.

The correction can then be applied to each point along the surface of the tube. First, to use this correction, the distance between points B and M must be calculated for each point along the meniscus surface, as described in equation 3.2. Then, each meniscus point is shifted horizontally by adding this quantity to its horizontal coordinate.

For example, considering a generic point along the meniscus of coordinates (x_i, y_i), where x_i is its horizontal coordinate and y_i is its vertical coordinate, the corrected position of this point will be given by (x_f, y_f), where $x_f = x_i + \overline{BM}$ and $y_f = y_i$.

3.3 Predicted Impact on Observations

´ The correction process described in the previous section allows some predictions to be made regarding how this correction will shift the points that belong to the meniscus and how some of the parameters will impact this shift.

In reference Darzi and Park [26], the observed effect of the correction was a shift of all the points on the meniscus to the centre of the tube, as seen on figure 3.5.

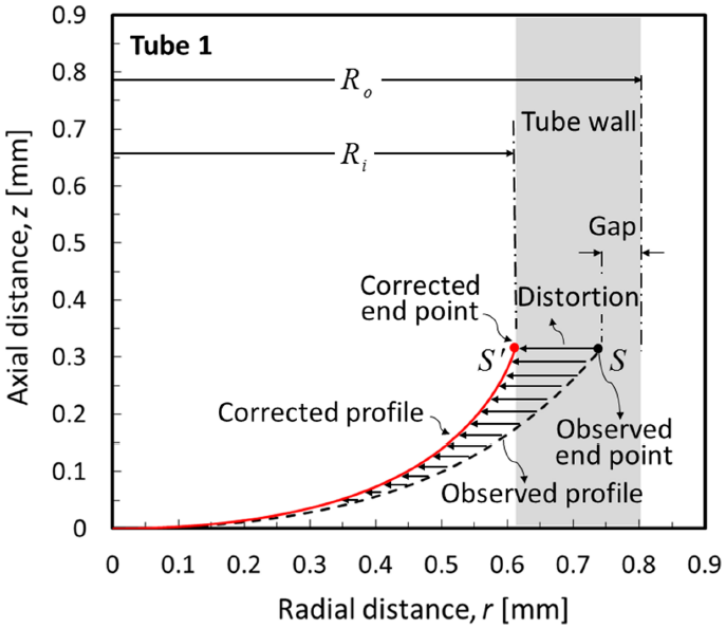


Figure 3.5: Effect of the correction on the meniscus, figure from [26]

Figure 3.5's prediction agrees with what was observed during the experimental campaigns since the meniscus seemingly penetrates the inner wall of the tube, as can be observed in figure 3.6. In figure 3.6, the inner wall of the tube is marked with two vertical blue lines and its position is obtained by relating the outer radius of the tube with its inner radius. Since the position of the outer wall is easily obtained, one can obtain the pixel to millimetre ratio and, thus, obtain the position of the inner wall by offsetting each of the outer walls by half of the difference between the outer and inner radius, i.e. $x_{InnerWall} = x_{OuterWall} \pm (R_o - R_i) \times pix2mm$.

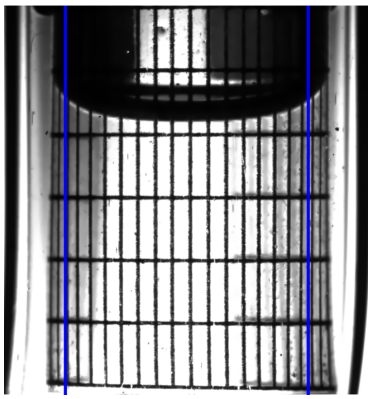
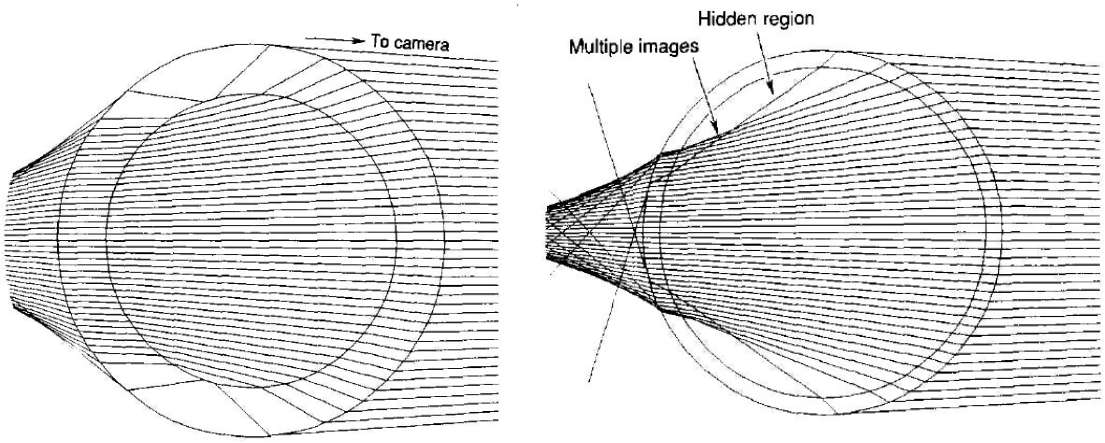


Figure 3.6: Penetration of the meniscus of the inner wall of the tube represented with the vertical blue lines

To further test this, in reference Darzi and Park [26], a plug gauge with a known diameter was inserted in the tube and the difference between the known and corrected diameter was found to be less than 3%. This result indicates that this correction will accurately account for the effect of optical distortion and, thus, produce a correct contact angle. Furthermore, in the study mentioned, more tests were conducted using this correction method for cylindrical tubes. These tests had static contact angles for water on untreated borosilicate glass, which showed a good agreement with the reference values for sessile droplet tests of this material. These comparisons will also be established between the results obtained for the experimental campaigns conducted during this thesis.

In addition, reference Lowe and Kutt [27] also supports the use of this method for this facility. The tube's diameter and thickness are investigated in this paper. These parameters are instrumental in controlling the behaviour of the light rays within the tube. The behaviour of these light rays is studied extensively using ray tracing algorithms applied to a plexiglass tube filled with water.



(a) Ray tracing for higher q values

(b) Ray tracing for lower q values

Figure 3.7: Ray tracing algorithm from reference Lowe and Kutt [27]

In figure 3.7, the effect of different inner to outer diameter ratios, $q = R_o/R_i$, can be observed. For higher values of q that can be observed in image 3.7(a), typically above 1.33, rays within the tube run practically parallel to each other in the middle, which will mean that an object placed in the middle of the tube will suffer little to no distortion. While closer to the edges of the tube, rays will be more affected by distortion, which will lead to bigger values for the correction.

The same can be said for image 3.7(b), which depicts what happens for lower values of q, typically less than 1.11, where distortion effects are significant, and there are even regions where particles located within do not even appear in the captured image.

For the experimental campaign conducted within the scope of this thesis, the value of q is 1.5, which would put it in the bracket of larger q values. However, the refractive indexes for both the liquid and the tube material are slightly different at $n_f = 1.28$ for HFE and $n_w = 1.46$ for the quartz UTube, when compared to the $n_f = 1.33$ and $n_w = 1.49$ for plexiglas, which represent differences that will be assumed to have negligible impact on the brackets for the behaviour of the rays.

4

Experimental Set-Ups and Methods

Contents

This chapter describes the experimental set-ups used during this thesis' work. Dynamic contact angle experiments have been performed using a quasi-capillary parallel plate facility, where the flow is not exclusively driven by capillary forces but also by a pressure gradient applied by a pressurised tank. The results are compared with the ones obtained by a separate campaign performed in quasi capillary U-tube set-up with similar conditions. The U-tube set-up has been characterised in terms of optical distortions to compare the two sets of data obtained from these experiments. As further validation, the interfaces obtained by the two set-ups in static conditions are compared with a more traditional sessile droplet test through a separate test set-up. In the following sections, the three facilities are further detailed.

4.1 Quasi-Capillary Parallel Plate Facility

This facility offered a different geometry to test the models previously developed for quasi-capillary flow in the U-Tube setup. The former has a rectangular cross-section and the latter a circular one. This opens up the opportunity to compare models for capillary flow, which are commonly developed for cylindrical capillary tubes with new geometry and see how well they describe it.

4.1.1 Facility Description

To achieve this parity, the facility has a similar operation to the U-Tube facility. A tank also pressurises it, and the objective here is to record the motion of the interface and its shape as the pressure is released and it moves between the two parallel plates. So, the air is let on to the system via an electronically operated pressure valve which also records the time of the pressure release. This time record allows the images of the liquid motion to be synchronised with the pressure release. The motion of the interface is driven by capillary forces and the pressure applied to the reservoir. It then seeks to balance this pressure with the atmospheric pressure in the opening between the two plates. Figure 4.1 demonstrates this facility, and its components.

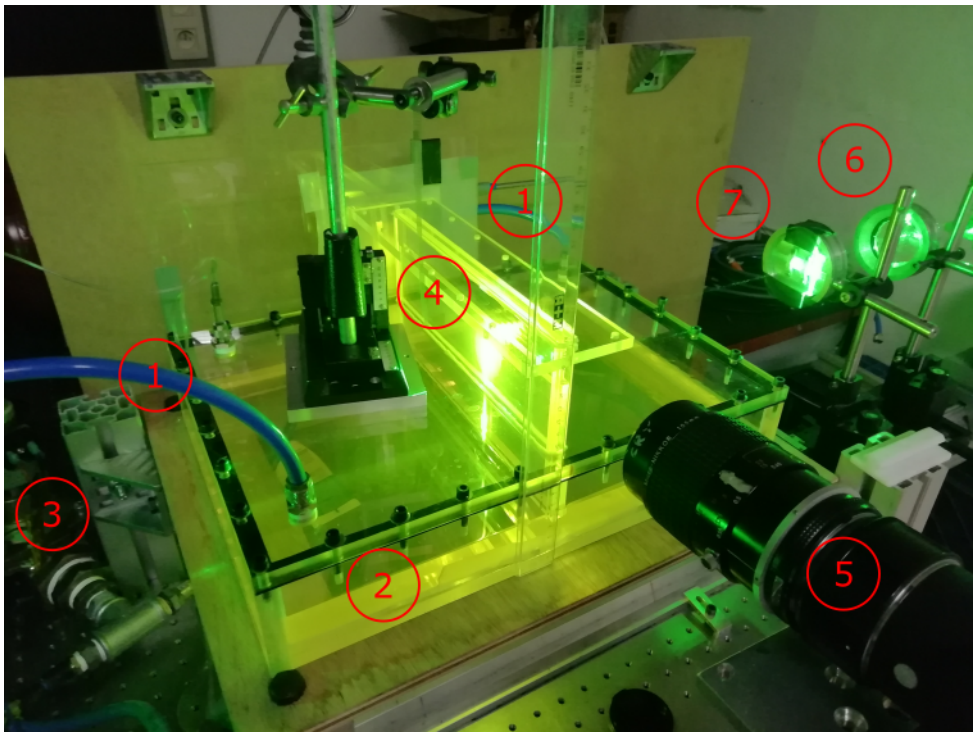


Figure 4.1: Descriptive photo of the quasi-capillary facility illuminated by the laser with each major component labelled

Figure 4.1 shows the two pressure lines from the pressurised tank, marked as 1. The tank was pressurised with the over-pressure of the experimental point under analysis. Four experimental points were considered for pressures of $1510Pa$, $1650Pa$, $1850Pa$ and $1900Pa$ with three test runs each. The pressure is then applied to the reservoir containing HFE 7200, marked as 2, once the electronic pressure valve marked as 3 is activated.

HFE7200 is the liquid that will penetrate the channel formed by the two parallel quartz plates, marked as 4. This liquid is seeded with particles of Pyrromethene 75820-500MG with a concentration of $6mg/L$. These are responsible for the fluorescent behaviour displayed.

Component number 5 is the high-speed camera and lens assembly. The camera model used was an SP-12000M-CXP4, which captured frames at 500 Hz. The lens mounted to this camera was a Nikon-Mirror 105 mm 1:2.8 with a Nikon PK-13 27.5 magnification ring.

The fluid is illuminated by a continuous green laser which first passes through a concave lens and then a cylindrical one. The concave lens, marked as 6 in figure 4.1, is responsible for focusing the laser beam, and the cylindrical lens turns it into a laser sheet, marked as 7 in figure 4.1.

These components are all employed to obtain the motion of the interface in the channel between the parallel plates. This is done by making use of the ”Level Detection and Recording” Technique. This technique involves seeding the fluid with particles that are lighter than its own particles and, thus, go to

its surface. The particles emit fluorescent light towards the camera upon being hit by green light from the laser sheet and show the shape of the meniscus in the laser plane. [28]

The laser is directed at the interface when the experiment is calibrated and is then recorded by the camera resulting in images similar to the one shown in figure 4.2.

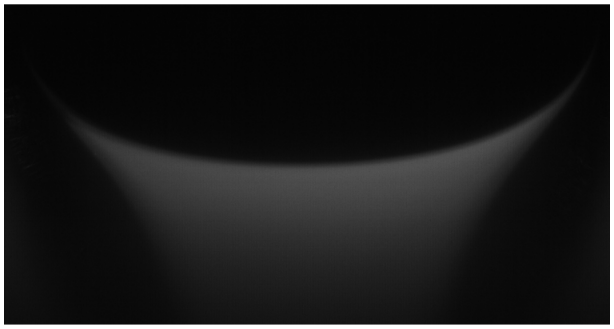


Figure 4.2: Image collected directly from the quasi-capillary plates campaign

4.1.2 Methodology

There were several procedures applied to process and obtain the results from the images recorded by the camera.

Firstly, the pressure sensor was calibrated, and its voltage was made to match the pressure that was being applied to the pressurised tank. This calibration resulted in a series of matching pressure and voltage points that were subsequently fitted using a linear fit. This set of points is represented in figure 4.3 and the fit equation is $pressure[Pa] = 210 \times voltage[V] - 9.89$.

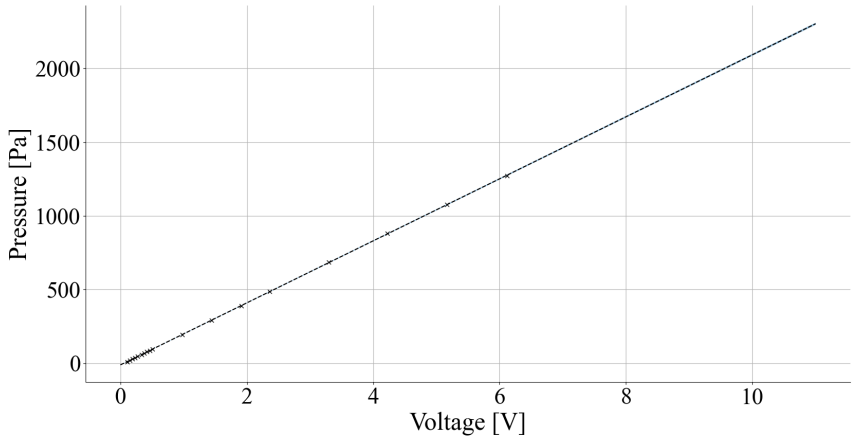


Figure 4.3: Pressure as a function of the voltage measured at the pressure sensor

The pressure calibration allowed the data from the pressure sensors to be synchronised with the data obtained from the recorded images. This match-up is done via a *Labview* algorithm that records the values from the sensors and matches them with a corresponding time stamp. The beginning of the

time scale is set when the program starts to run, and the subsequent time stamps are then recorded as shown in the time scale of figure 4.4.

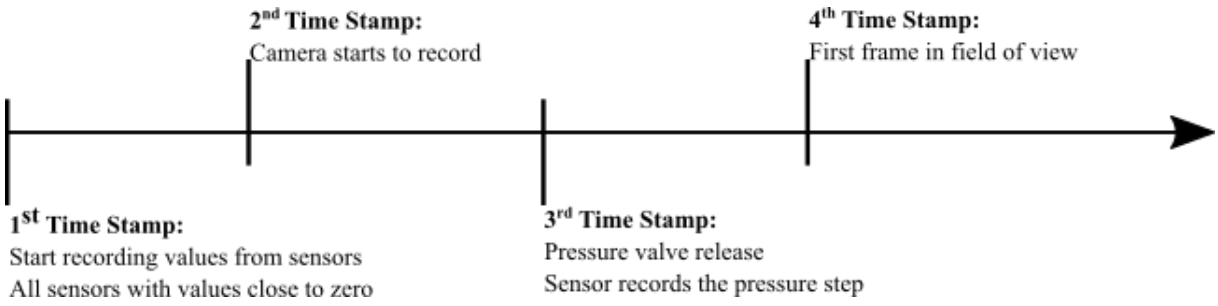


Figure 4.4: Time scale for the experimental procedure with a sequential representation of the time stamps

The fourth time stamp in figure 4.4 marks the beginning of the interface's observable movement since the camera's field of view does not cover the entire motion of the meniscus. This timestamp is not directly recorded in the *Labview* algorithm but is obtained by relating the frame number with the frame capture frequency, which in this case was $300Hz$. This timestamp highlights the need for a height offset for the meniscus since it started its rise on the third time stamp upon the release of the pressure valve.

All the images gathered after the fourth time stamp were treated and analysed using an algorithm developed during this thesis and based on previous work done by Domenico Fiorini and Miguel Mendez.

The first step is selecting the image which first captures the meniscus within the field of view. The frame number of this image will correspond to the fourth time stamp, and it is the first image where the meniscus will be detected. Then, the analysis interval is defined by choosing the last frame to be analysed. Thus, the total time that the height is being recorded and analysed is given by $time_{recording} = (n_{lastframe} - n_{initialframe})frequency_{capture}$.

The algorithm is run in parallel to analyse the previously selected images in several cores using a pre-existing function which accelerates the entire process. The function used for this is from the $concurrent.futures$ library in Python.

The algorithm fed into this function is responsible for all image processing and outputs stages. The image processing was subdivided into four different stages. The first one is, of course, to import the image. For this stage, the $OpenCV$ Python library provided the functions to read and rotate the images. By the end of this stage, the image being analysed was also cropped, making its width correspond to the channel width and, thus, obtaining the pixel-to-millimetre ratio for the image analysis process. Figure 4.2 shows an example of the resulting images from this stage.

In the second stage of this image processing algorithm, the image is denoised using a Non-Local Means Denoising filter from the $OpenCV$ library. This filter compares the intensity of pixels within the search area defined by its parameters and averages it with pixels of similar intensity. Essentially apply-

ing a low pass filter has the added benefit of not removing as much information from the image as a conventional one would. [29]

After this, the algorithm executes a function to detect the meniscus position. This function first applies a directional re-contrast filter to the image. This filter is helpful since it highlights the meniscus near the wall of the channel, where the meniscus tends to lack illumination. Figure 4.5 shows the impact this has on the image.

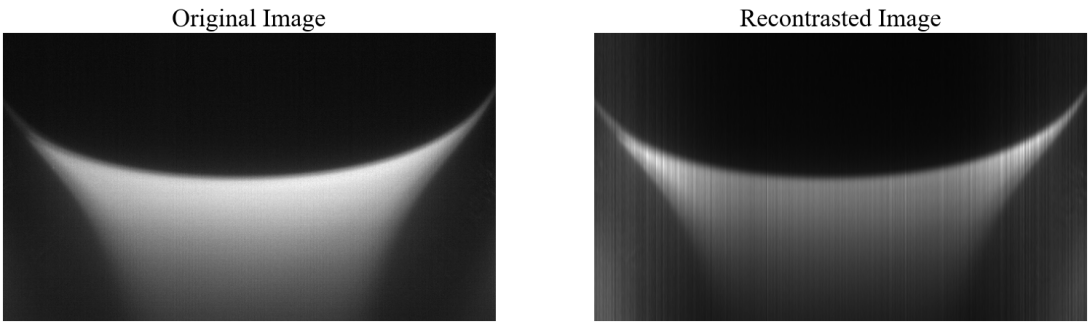


Figure 4.5: Directional re-contrasted image compared with an original image

This is achieved through equation 4.1, which describes a re-contrasting operation done column-wise. In this operation, each cell of a given column has its value changed by adding a parameter proportional to its column's intensity. This parameter is described in equation 4.2.

$$Recontrasted\ Image = Original\ Image + \beta \times Orginal\ Image \tag{4.1}$$

$$\beta_n = (\frac{\mu_n}{\mu_a})^{-k} \tag{4.2}$$

In equation 4.2, μ_n represents the average intensity of column n and μ_a the average intensity of the image. The ratio between the two is raised to the power of $-k$, which accounts for the intensity of this filter. Thus, it is possible to conclude that the columns with lower overall intensity will be multiplied by a higher factor. Thus the magnitude of re-contrasting in these will be higher. These sets of images result in the columns close to the image edge being brightened up.

After this, the image is convolved using a Sobel-like kernel suggested in reference Mendez et al. [30] and described in equations 4.3. This works as a high pass filter highlighting the horizontal edges as its peak values, as seen in image 4.6.

$$Kernel = \begin{bmatrix} -E \\ C \\ E \end{bmatrix} \tag{4.3a}$$

$$E = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix} \tag{4.3b}$$

$$C = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{4.3c}$$

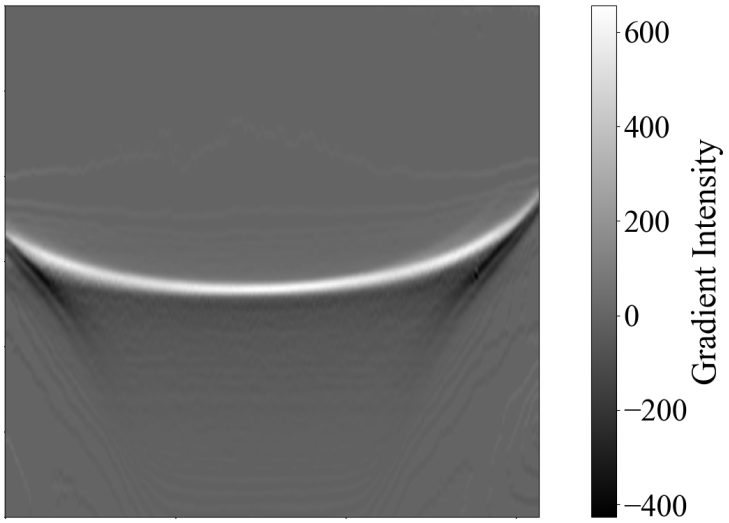


Figure 4.6: Resulting image from convolution with Sobel kernel which highlights the gradient intensity throughout the image

Figure 4.6 is the representation of the gradient of the original image, which is the result of *convolving* it with the previously mentioned kernel. As can be seen from the colour scale represents the intensity of the gradient, and the higher values of the gradient will correspond to the meniscus. [31]

To locate the position of these peaks in the image, the *find peaks* function of the *Scipy* library was used. Several considerations must be taken into account when using this function. Firstly, to avoid the detection of other points, either leftover noise or other, a height for the peaks must be defined. The threshold is the minimum value a cell must have to be considered during this search for peaks in the gradient, and it was adjusted before the analysis of each image set. However, it is worth pointing out that it did not vary much since the camera settings and illumination was very similar. Another parameter to this function is the distance between peaks. The distance between peaks is another easy way to exclude other points than those belonging to the meniscus since it considers only one point within a distance. This means that if the distance between peaks is defined as the length of each column being analysed, only one peak will be considered per column.

The only other operation left within the *detect edges function* is applying a median filter to the array

of points detected by the find peaks function. This filter prevents points that stray too far from the line detected for the meniscus from being considered.

With all these operations completed, the algorithm can now detect the meniscus positions throughout the image, as seen in figure 4.7.

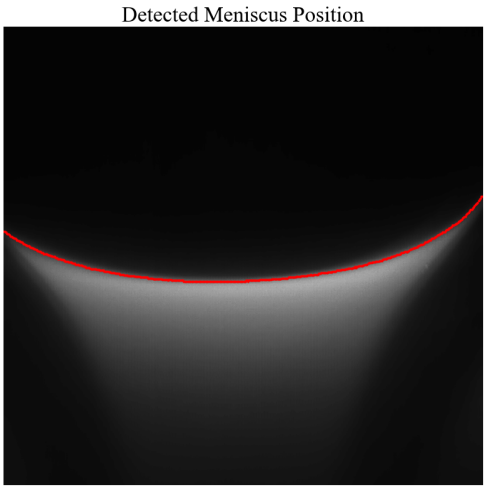


Figure 4.7: Detected meniscus position marked with a red line

After this, the algorithm runs a fit to the points represented in figure 4.7. The objective of this fit is to extract the contact angle, which is a boundary condition to this fit.

These points are fitted using the Young-Laplace equation, equation 2.1. This equation is used despite the dynamic nature of the phenomena because there is no adequate description for a dynamic meniscus.

Thus, the equation that is derived for this geometry is equation 2.7.

With the boundary conditions established in equation 2.8, this becomes a boundary value problem for which the contact angle will be optimised to obtain the best fit to the interface. This contact angle will be considered for the correlations tested against the model data. The fitting procedure and functions used are very similar to those used for the Sessile Droplet tests, and further explanation can be found in that experimental campaign's Methodology section.

After this, the data concerning the frame number of the image was analysed, the contact angle obtained for the frame, the average height of the meniscus in pixels for the frame and the height of the contact line (h for the $x = ChannelWidth$) is stored in *.txt* files to be interpreted by a different script responsible for adjusting the parameters to include height offsets and synchronise the data time-wise.

The time synchronisation is achieved taking into account the timeline presented in figure 4.4, and a measurement is taken each run of the final position of the meniscus once it has stabilised. This measurement is the distance between the final position of the meniscus once it has stopped moving and the bottom of the channel.

This method results in plots similar to those in figure 4.8, that have the height converted from pixels

to millimetres and the time synchronised properly with the pressure valve opening.

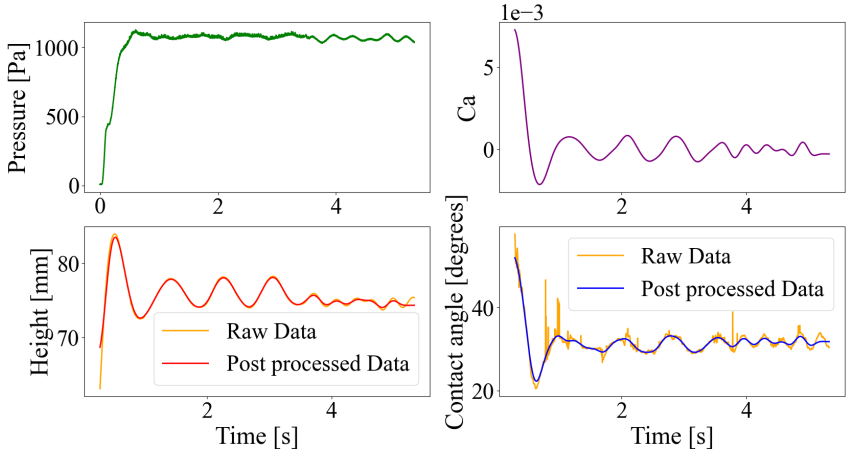


Figure 4.8: Sample of the data extracted from the image sets: the image on the top left side is a plot of the pressure on the reservoir vs time; the image on the top right side is a image of the capillary number vs time; the image on the bottom left side is a plot of the meniscus height vs time; the image on the bottom right side is a image of the contact angle obtained from the fitting process vs time

This data will be used to compare the behaviour of the fluid within the channel to the preexisting correlations. These results will be analysed in chapter 5.

4.2 Quasi Capillary U-tube Setup

The objective of this facility was to evaluate the accuracy of the corrections proposed in the literature [26]. A similar set-up to the one used during previous experimental campaigns in the U-Tube facility was employed. However, during this experiment, there was no interest in the dynamic nature of the meniscus movement, and thus, the pressure system was disconnected.

4.2.1 Facility Description

This facility is very similar to the one used for the dynamic experiments in the U-Tube. The most significant difference is the lack of pressure systems, i.e. the overpressure line was disconnected as well as the pressure sensors and pressure valve and all the systems used to record it all using the *LabView* algorithm.

Thus, the U-Tube was detached from the pressure system, and its leads were unfastened to allow for better access to place the calibration grid. The whole system can be seen in figure 4.9.

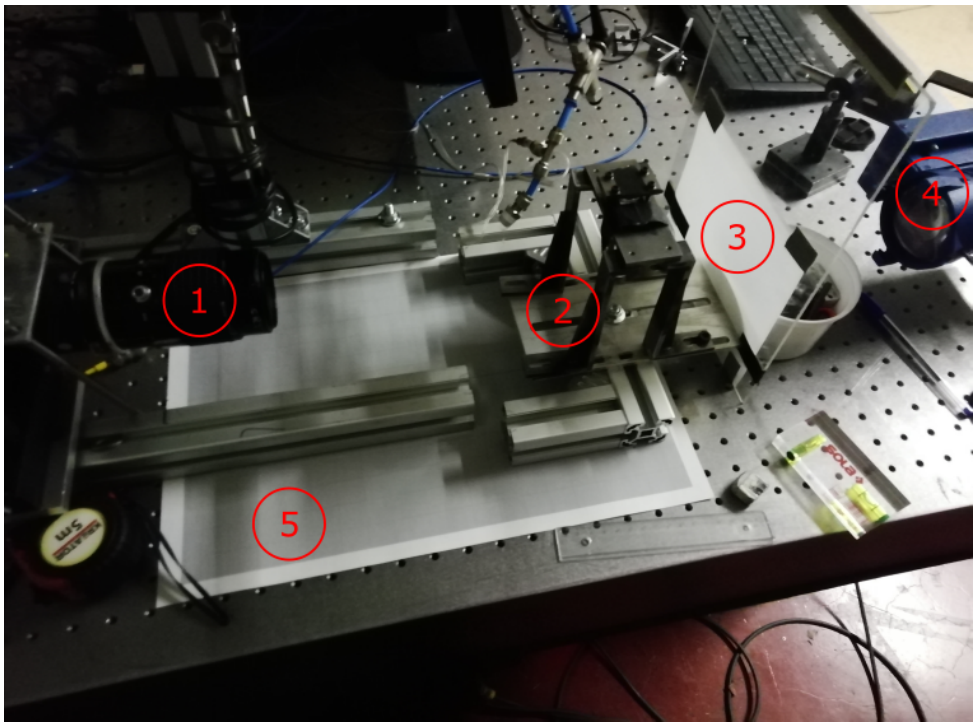


Figure 4.9: Experimental set-up for testing the correction for the radial distortion with each major component labelled

Upon closer analysis of image 4.9, the different components of this set-up can be identified. The feature identified as number 1, refers to the camera assembly, which in the case of the distortion experiments was made up of the camera itself, a Spark Series SP-1200M-CXP4 high-speed industrial camera, and the objective, a Nikon-Mirror 105 mm 1:2.8 objective. Component number 2 refers to the Utube and its support. This is where a grid to test the correction is inserted. This grid can be seen in figure 3.6, and there were several renditions of this grid. The grid that was ultimately chosen to perform most measurements was imprinted on a transparent plastic sheet. The space between the vertical lines was $0.5mm$ and between the horizontal lines $20mm$. Its use for testing the optical corrections will be discussed further ahead.

Component number 3 acted as a light diffuser for the light coming from the light source marked as number 4. Also important to note is the grid pattern marked as number 5. This was used to align the camera and get the distance between the camera sensor and the centre of the tube. This was also used to estimate the magnitude of the misalignment angle, which is marked in figure 3.3 as θ_0.

4.2.2 Methodology

To run this experiment, the first step was to place the grid carefully inside the U-Tube set-up using a pincer. Special care was taken to ensure the grid wrapped around the inner diameter of the tube as well

as it could.

After this, the backlight was turned on, and the images were ready to be captured. These images were the first ones to be processed in the context of this thesis and were the starting point for the image processing done for this thesis. For every experiment run, a video file was captured using the *GenICam* software. After this, the video file was broken down into frames using *VirtualDub* software. This software was used to collect data in all the experimental campaigns performed during this thesis.

This software applied two filters to the images: a re-contrasting filter to enhance the image's contrast and a brightness filter. These filters were applied to increase the contrast of the vertical black lines against the background. This image treatment will make detection much easier since the minimum values for the intensity of the image will be more pronounced.

After this, the images are imported to the script, where the data will be collected and the points corrected. Once the image is imported, it is denoised using the Fast Non Local Means Denoising Filter. This filter will look for pixels with similar intensities within a prescribed kernel and average those pixel intensities, just like it was described in the previous section regarding the quasi-capillary parallel plate set-up.

With this denoising process complete, the images are now ready for point detection. The approach used in this campaign differed from the one used for the meniscus detection in the previous section. In the case of the images of these experiments, the objects that will be detected are the vertical lines, which correspond to local intensity minima in the image. Thus, to detect them, a function from the *Scipy* library was employed called *argrelextrema* that searched for local minima within an interval of intensity from the absolute minimum intensity in the image. This interval had to be implemented to account for slight differences in intensity from the different vertical lines.

There are still two important parameters missing, the distance between the centre line of the tube and the aperture of the camera (L) and the camera misalignment angle (Θ_0), which takes into account if the camera is pointing towards the centre of the tube or not.

Figure 4.9 shows how the entire set-up was positioned on a grid pattern to measure the distance between the centre line of the U-Tube and the camera aperture as well as the misalignment angle. The position of the tube and the camera's aperture were then recorded on the grid, and the angle and distance were obtained.

However, these parameters are difficult to determine through a direct measurement method but can be determined through an inverse method. Thus, these parameters were determined by an optimisation algorithm which looks for values of Θ_0 and L that minimise the difference between the corrected points and calculated the grid positions that are not affected by distortion. This process is repeated 100 times with different randomly picked initial values for the optimisation to ensure that the absolute minimum of the error is picked. In addition, these values are compared with the measurements made experimentally

to verify their accuracy. The inverse method's results will be discussed in chapter 5.

4.3 Sessile Droplet Test Setup

The objective of these tests was to compare the static contact angle obtained for the sessile droplet test campaign, which is commonly used to get this parameter in the literature, with the static contact angle obtained for the other two configurations.

4.3.1 Facility Description

This facility used the set-up used during the optical distortion campaign. The most important difference was that the Utube assembly was replaced with an adjustable height platform, as in figure 4.10.

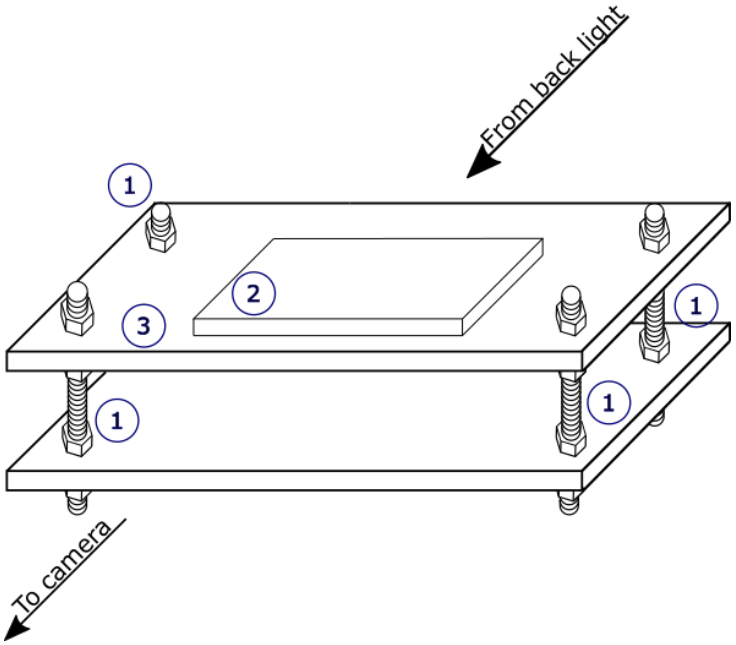


Figure 4.10: Detailed view of the height adjustable table set-up for the sessile droplet test

Figure 4.10 shows the adjustable struts, which allowed to perform fine height changes to better frame the pictures taken within the camera's field of view, marked as 1. It is also possible to see a representation of the quartz plate where the liquid was dropped, which is marked as 2.

This set-up was mounted in the same position as the set-up holding the U-Tube, and the back-light illuminated the droplet area in a similar fashion to the distortion campaign. The liquid was dropped through a syringe which was mounted on a bracket to keep its position steady relative to the quartz plate.

This is important because the droplets are very small, with average diameters between 5 and $8mm$, and to capture them properly, the camera needs to be focused on an appropriate spot. Thus, securing

the syringe with an optical support was the best solution found for consistent liquid placement. This was necessary since the time available to capture shots of the droplet on the quartz plate was limited by the high evaporative rate of HFE7200.

It is also important to note that the platform marked as 3 where the quartz plate was supported was levelled after every height adjustment to avoid droplet movement in one specific direction.

4.3.2 Methodology

The procedure for this experimental campaign has already been looked over in the previous section, but this section will explain it further. For each experiment run, the camera was focused on one spot of the plate using a calibration grid, as seen in figure 4.11.

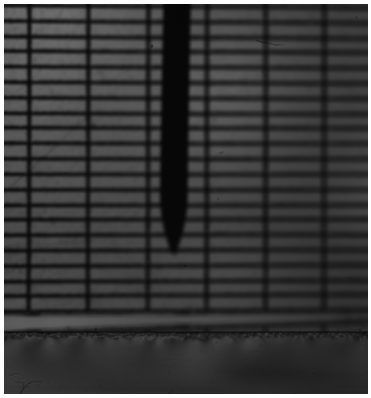


Figure 4.11: Calibration picture where the calibration grid is displayed along with the tip of the syringe used

This grid served a dual purpose. The first objective was to focus the camera on the spot in its field of view as close to the surface of the quartz plate as possible. The second objective was to guarantee that during the image processing phase, there was a way to obtain a pixel-to-millimetre ratio to convert the image measurements.

In addition, during this preparation phase, the syringe was also positioned near the calibration grid, as shown in figure 4.11. This placement is essential to ensure that the liquid is dropped in the vicinity of the area where the camera is focused and, thus, allows for quick capture of shots of the droplet.

After completing this procedure, the images were recorded using the same method as in the distortion set-up and quasi-capillary parallel plates. These images were then submitted to a similar process to the previous campaigns. The first operation was to set the coordinates corresponding to the droplet width and height. These coordinates mark out the area of the image which will be analysed. Following this, the images were denoised using the same filter used for the previously described experimental campaigns.

Then, the algorithm enters a function designed to detect the droplet edges similar to the one used

in the quasi-capillary parallel plates campaign. However, this function does away with the directional re-contrasting and changes the criteria to find the peaks. This change is done to accommodate the specifics of this image set, of which figure 4.12 is an example.

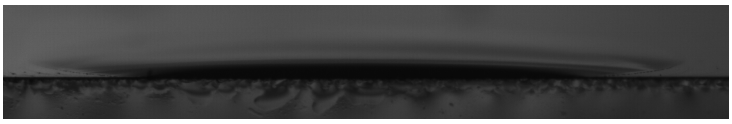


Figure 4.12: Captured image of a droplet of HFE7200

Image 4.12 is an example of an unfiltered image taken directly from the image set. It is quite hard to discern where the droplet's surface is and what should be considered to be the droplet's surface. This difficulty is typical of all the images captured, and as seen from the figure 4.13, the algorithm is capable of capturing several surfaces which seem to be adequate to consider as the droplet surface.

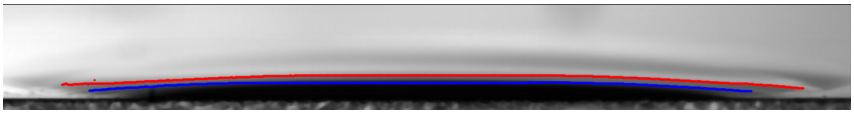


Figure 4.13: Surfaces droplet detected by the algorithm

Figure 4.13 shows why a different method to find the peaks of the gradient and, thus, the surface of the droplet was employed. Firstly, a restricted analysis is defined to consider only one of these surfaces separately. This area is marked out by the perceived width and height of the droplet according to the surface being considered. So, for example, to obtain the surface described by the red dots in figure 4.13, the area considered is restricted to the width and the height of the points detected, just as seen in image 4.14.

Figure 4.14: Example of the area analysed for a given surface

This area might include, in some instances, other surfaces. However, this issue is solved by removing the distance parameter from the *find peaks* function, which will result in both surfaces being detected. This change in the algorithm means that for each column of the image gradient analysed, more than one peak may appear, as illustrated in figure 4.15 if this is the case, then the appropriate peak is selected through considering the proper cell of the peaks array generated for each column. Alternatively, if the surface being analysed is the strongest in the analysis area, this operation might be unnecessary altogether if one considers an appropriate minimum value for the gradient.

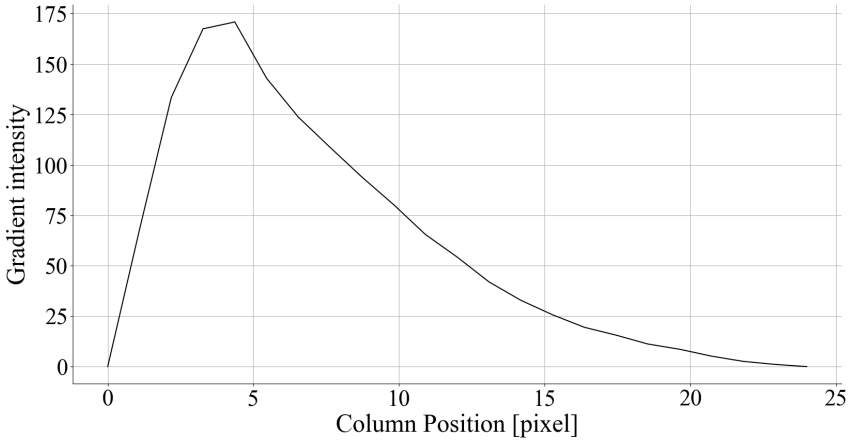


Figure 4.15: Peaks obtained for the middle column of figure 4.13

After the appropriate array of points is chosen as the droplet's surface, the algorithm can perform the fit. This fit is done through an optimisation process, where the *curve fit* function from the *Scipy Optimize* library optimises the parameters fed to a user-defined class, where the boundary value problem described in chapter 2 is solved. This boundary value is solved using the *solve bvp* function from the *Scipy Integrate* library. The final objective of this optimisation process is to minimise the difference between the computed solution for the boundary value problem and the points for the surface. The solution for this boundary value problem is extrapolated to match every point of the detected surface, and the contact angle and offset coordinates for the curve are then tweaked by the *curve fit* function until it can no longer reduce the difference between the two.

With this, a fit is produced for the droplet's surface, and the contact angle is obtained. Figure 4.16 is an example of the fit obtained.

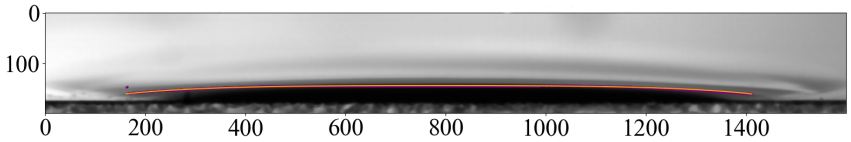


Figure 4.16: Fit for a given surface

The results of this experimental campaign all the others are discussed in chapter 5 and the code used for these experimental campaigns can be found in 9.

5

Results

Contents

5.1 Sessile Droplet test

The objective of these tests was to obtain the static contact angles for the combination of HFE7200 and quartz. Getting the contact angle was, however, more difficult than expected because the liquid droplet formed a thin film over the quartz plate characterised by a small contact angle.

Thus, capturing the geometry of the liquid droplets was a challenging procedure, involving many tries. This unusual measurement led us to develop a technique where the fluid needle was mounted in an optical bracket which held it in place to make the focusing easier with the grid displayed in chapter 3. Despite this, image quality was poor due to oscillations in dropping HFE7200 onto the quartz plate and the fast evaporation of the liquid droplets. Therefore, when determining the contact angle since more than one surface was identified per image, as seen in image 4.14.

To improve accuracy, many measurements were taken and the results analysed statistically. This analysis resulted in a static angle of $6.23°$ with a standard deviation of $3.53°$ on a sample of 20 independent images. This uncertainty is higher than the generally accepted $2°$ encountered in sessile droplet tests. However, this liquid has a significantly smaller contact angle than what is usually found and the evaporative rate is considerably higher. Given these constraints, the current approach represents a improvement in the accuracy of the results for HFE7200. For reference, image 5.1 is a comparison of a water droplet image against that of an HFE7200 droplet.[7] [2]

Figure 5.1: Captured image of a droplet of water and a droplet of HFE7200, where it is possible to see that the higher contact angle of water allows makes for a more clearly defined droplet

5.2 Radial Distortion

The method used to evaluate the radial distortion will be discussed in this section, as well as an evaluation of the uncertainty when determining the position of the points.

5.2.1 Experimental Method Used

The first method consisted in measuring directly the parameters L and Θ_0. However, this proved to be not accurate enough since both the distance and the angle were hard to measure. The uncertainty of the position of the aperture produced a corresponding uncertainty in the evaluation of the distance between the tube and camera. From outside of the camera, it is difficult to locate the position of the shutter, and thus, an estimate was made as to its position. This limitation extends to assessing the

camera misalignment angle, Θ_0, since not knowing the exact position of the centre of the shutter makes it much harder to calculate or measure this angle.

The evaluation of the uncertainty of these two input parameters had to be calculated to evaluate the uncertainty associated with the optical corrections. However, an uncertainty of 0.5 mm for the parameter L, which corresponds to half of the smallest division of the scale of the grid, was considered unrealistic due to the constraints described previously. Deriving the uncertainty considering only the grid resolution would yield optimistic results for the overall uncertainty of the correction and lead to misinterpretation of the results.

The issue with the uncertainty and determination of the parameters described above was addressed by implementing an inverse method, which also improved the expediency in applying an optical correction to the experimental campaigns performed in the U-Tube facility. The objective was, as described in chapter 4 to obtain the values for L and Θ_0 by comparing the corrected points with the positions of the grid lines and obtaining the value which minimised the difference between the two.

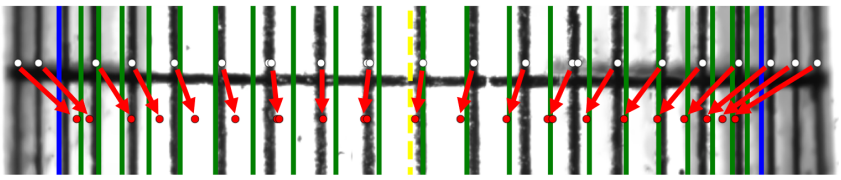


Figure 5.2: Distortion Results for Optimization of the L and Θ_0, where a section of the U-tube is displayed

Figure 5.2 represents a section of the U-Tube facility where the algorithm has been applied to correct the position of the grid lines. The white dots correspond to the algorithms' detection of the vertical grid lines, that are affected by distortion and must be corrected. The dashed yellow line marks the centre of the U-Tube, and the two blue lines mark the inner tube walls. The green vertical lines correspond to the vertical grid lines if no distortion took place. If the correction were to work as intended, it should shift the white dots onto the green lines once it was applied.

The red dots represent the corrected position of each of the white dots. The arrows connecting each white dot with a red one help identify the correction applied.

First, we may observe that, as expected, the magnitude of the correction increases towards the edges of the U-Tube, as can be seen on the graph displayed in figure 5.3.

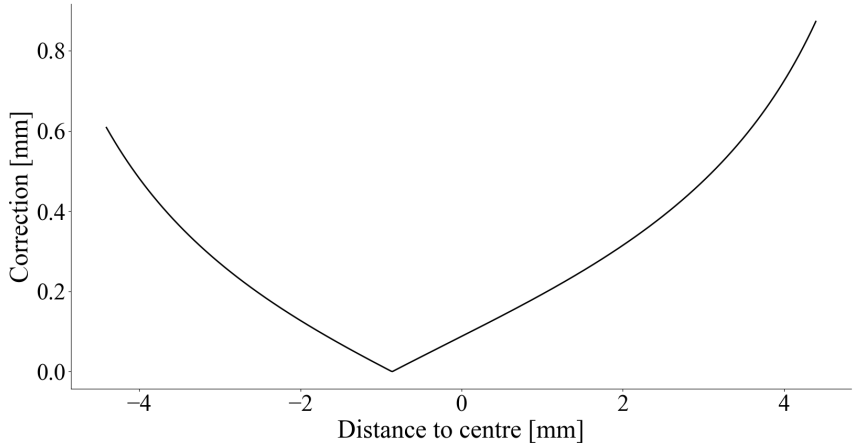


Figure 5.3: Correction for the Original Optimisation: magnitude of the correction as function of the radial coordinate

The results of figure 5.3 are to be expected since the closer to the radius of the tube, the higher the value of Ω and, thus, the value of Θ_A upon which the optical correction depends, as seen in equation 3.2.

This leads us to the next comment: the red points do not coincide with the green lines as they were supposed to. Even when considering the uncertainty related to the measurements of the parameters, as discussed previously, they do not coincide.

This difference is related with how the green lines are being generated. Errors impact the inverse method and the ability to evaluate its accuracy. To verify that they are being generated incorrectly, the approximation used to generate them must be evaluated.

The green grid was generated by considering that the centre of the tube would be unaffected by the distortion effects. As such, it would be reasonable to assume the closest point to the centre is not distorted. If this assumption was valid, then the position of the true grid lines could be easily obtained by considering the inner radius of the tube and the arc length between each of the vertical lines, which was imposed by the grid's design to be 0.2 mm long.

This approximation would be true if there was no camera misalignment, as can be seen in figure 5.4.

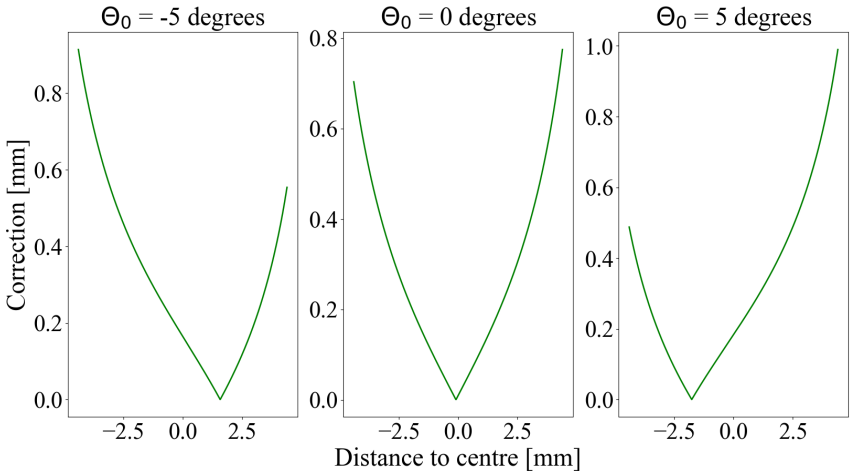


Figure 5.4: Influence of Θ_0 on the magnitude correction displayed for three different values of this variable

However, figure 5.4 shows that when Θ_0 is different from zero, the correction is not null in the centre of the U-Tube. Since the results from the direct measurements of the experimental campaigns yielded a Θ_0 different from zero, this method becomes inaccurate in estimating the experimental results. Thus, if an inverse method is to be implemented to improve the correction's parameters, a new approach must be implemented.

5.2.2 New Method Analysis

A new approach was developed to correct the assumptions of the previous method. This new approach was based upon a similar procedure described in Darzi and Park [26]. This paper suggested using a plug gauge with a diameter equal to the tube's inner diameter to test the accuracy of the correction.

This method could be applied to the images already available by considering the horizontal lines on the grid as seen in figure 3.1. Since the grid wraps around the inner diameter, the length of the visible part of a horizontal grid line corresponds to the inner diameter. Thus, if the correction is applied to the outer points of the horizontal grid line highlighted in red in figure 5.5, they should coincide with the vertical blue lines.

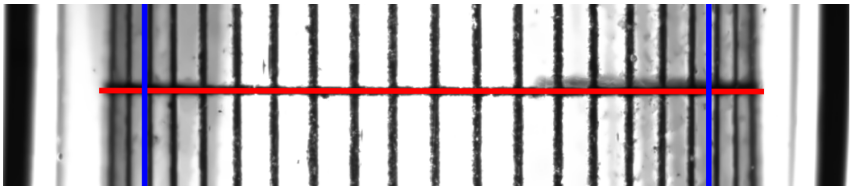


Figure 5.5: Detected red horizontal line and the inner tube walls represented in blue

The vertical blue lines in figure 5.5 represent the inner tube walls and are obtained using the pixel

to millimetre ratio which is the ratio between the tube width measured in millimetres and pixels. The calculation of the position of the walls is done in 5.1.

$$\begin{cases} x_{firstWall} = (R_{outer} - R_{inner}) * pix2mm \\ x_{secondWall} = (R_{outer} + R_{inner}) * pix2mm \end{cases} \tag{5.1}$$

This ensures that no approximation is introduced to obtain the points that will be used to optimise the L and Θ_0 parameters. However, since Θ_0 is the harder parameter to measure the optimisation was conducted only for this parameter, and L was considered to be $39cm$ as measured previously.

The results obtained for this optimisation are displayed in figure 5.6.

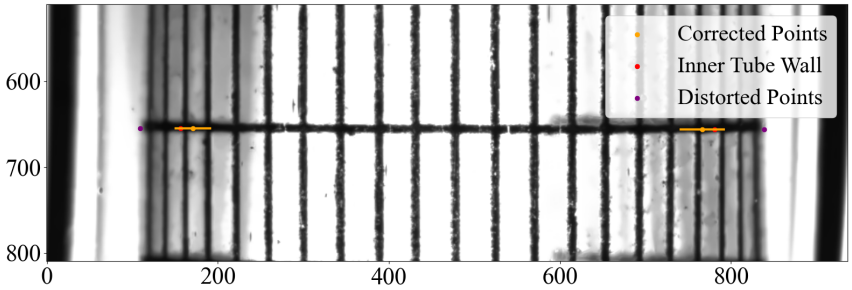


Figure 5.6: Detected points, corrected points, real position of the tube wall and uncertainty

Figure 5.6 displays similar levels of distortion on both sides of the image since the detected points for the position of the inner tube wall are at identical distances to the real positions of the tube walls. Accordingly, the angle obtained for this was $0.77°$. This result is to be expected since, for low values of Θ_0, the correction is symmetric relative to the central axis of the tube.

As for the error associated with this image and the calculation procedure, the difference between the diameter calculated using the corrected points and the real positions of the inner tube walls the difference obtained was 4.64%. When compared to the experimental error obtained by Darzi and Park [26], it is slightly higher, but it should be taken into account that despite the best efforts in placing the grid inside the tube as close as possible to the tube walls, the adherence of the grid to the tube walls can not be guaranteed and the distance to the tube could be measured more precisely with the aid of a laser distance measurer.

However, the issue of the adherence of the grid to the tube walls becomes irrelevant when considering that this procedure can be performed using the meniscus images directly. Since the meniscus adheres to the inner tube walls, there is no need to place a grid within the tube, and the calibration procedure for the camera misalignment angle becomes much simpler. In this case, only the outermost points of the meniscus have to be considered to perform the calibration. This calibration is achieved by comparing these points with the positions of the inner tube walls.

Finally, it is also worth noting that the results displayed in the image were obtained with an algorithm

that optimised the value of Θ_0 for 1000 random initial values of these parameters within an interval of $[-10, 10]\circ$. In addition, the uncertainty bars allocated to each corrected point were calculated using the derived uncertainties method, so an uncertainty of 5 mm was considered for the distance from the camera aperture to the centre of the tube and an uncertainty of $2\circ$ for the camera misalignment angle. [32]

The results seem to be satisfying considering that the real positions of the inner tube walls fall within the uncertainty associated with the corrected positions and a similar range of diameters to that of reference Darzi and Park [26]. It is also worth pointing out that the value obtained for Θ_0 is in line with the value obtained for using direct measurements of $2°$.

5.2.3 Meniscus Correction

The method previously described was applied t a set of images pertaining to a static interface. The results for an image belonging to that set are displayed in image 5.7.

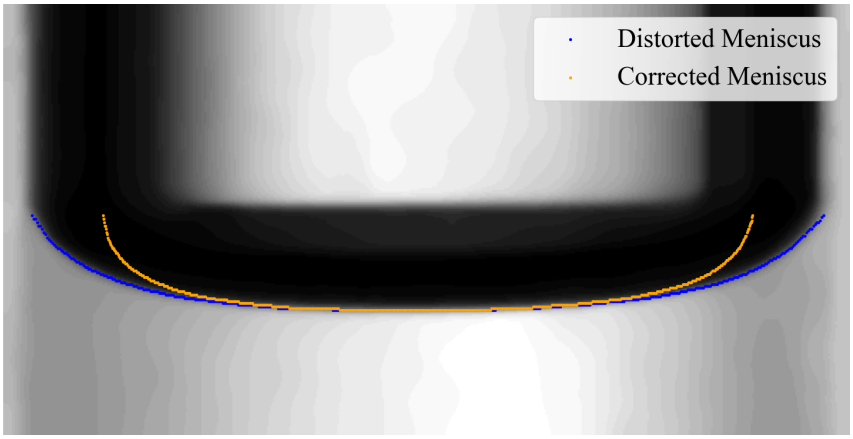


Figure 5.7: Detected meniscus interface in blue and the correction for the optical distortion of these points in orange

Figure 5.7 shows the detected points for the meniscus surface, represented in blue. These points correspond to the distorted position of the meniscus as captured directly by the camera and are to be corrected by the inverse method presented earlier. The corrected points that result from the inverse method are the ones in orange. As expected, a general shift towards the centre of the tube is observed in line with the predictions displayed in image 5.5. In addition, the correction's magnitude increases towards the edges as predicted in figure 5.4. This shift effectively decreased the contact angle obtained from the fit of the experimental points.

As in the sessile droplet campaign, the contact angle was obtained through an optimisation procedure to match the experimental points with the solution of equation 2.4, considering the boundary conditions imposed by 2.6. The contact angles were corrected from $34.05\circ$ to $8.55\circ$, emphasising the

importance of correcting the data of this experimental set-up.

5.2.4 Static Contact Angle

With the information gathered from the sessile droplet test campaign and the corrected quasi-capillary U-tube results, it is now possible to compare the static contact angle in axial-symmetric configurations with the results obtained from the quasi-capillary parallel plates. Static images from each experimental point in this set-up were collected and an average static contact angle of $26.75\circ$ with the standard deviation of the sample being $4.67\circ$. This differs from the results of the sessile droplet tests and quasi-capillary U-tube.

According to [33], the contact angle should be independent of geometry. However, this is not verified in the case of the 2D parallel plates. The difference between the axial-symmetric interfaces and the quasi-capillary parallel plates might be due to the detection of the interface. The resolution in the interface detection plays a relevant role in the measurement, given the uncertainty of the wall position and the influence of the meniscus illumination near it. That might be responsible for this difference since most likely, the steepest part of the meniscus curvature is not visible in the images collected utilizing the "Level Detection and Recording" technique as seen in figure 4.2.

5.3 Dynamic test in 2D Channel Facility

The objective of these tests was to compare the results obtained for the quasi-capillary U-tube with the set-up for the quasi-capillary parallel plates and adapt the correlations and models developed for the former facility to the new geometry.

The objective here is to show that similarly to what has been proven for the static contact angle in capillary set-ups, the dynamic contact angle should be independent of the geometry and, thus, the correlations developed for a tube with a circular cross-section should also describe what happens in the case of the quasi-capillary parallel plates.

In addition, these correlations aim to replace the need to extract the contact angle from a static approximation to the meniscus shape and, thus, substitute this parameter in the prevailing models for capillary flow as described by equation 2.30.

5.3.1 Quasi-Static Fit to the Meniscus

The contact angle data that will be analysed in the next section was obtained by a fit to the detected points of the meniscus. The fit and its equation have been discussed in chapter 2 and are based on an equation that does not consider the meniscus's motion. Despite this, using a bootstrapping method to calculate the variation of the contact angle for each frame, using a training set of 70% of the detected

points and the other 30% as the validation group, the variation uncertainty obtained for the contact angle extracted was very small, at an average of $0.0034\circ$. This uncertainty means that effectively the quasi-static fit performs very well in its role of assessing the meniscus shape. [32]

Another interesting comparison is between this new method of analysing the contact angle and the gradient method, which has been typically used to obtain the dynamic contact angle directly. The gradient method considers the value of the gradient of the meniscus shape near the wall is equal to the tangent of the dynamic contact angle. The difference between both methods is on average $0.708°$. This difference is represented over time in figure 5.8, where the light grey band represents the variation around the contact angle obtained by the fit.

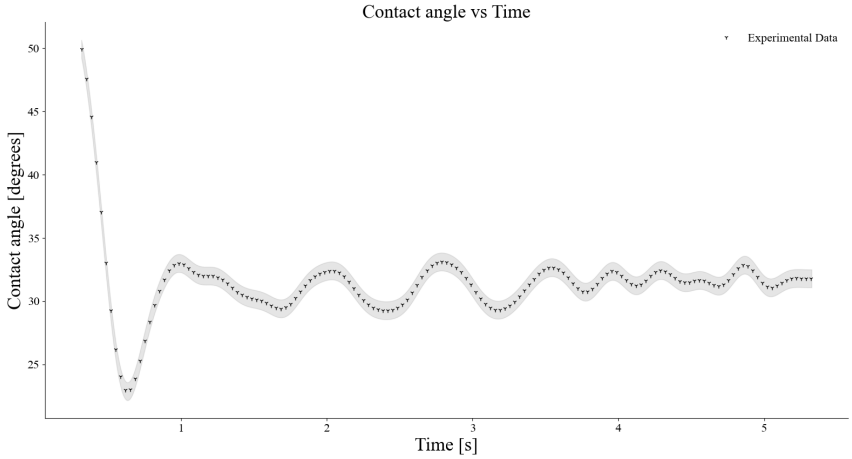


Figure 5.8: Results from the static fit to the quasi-capillary parallel plates meniscus and its uncertainty

To analyse the contact angle correlation we will use this difference as a proxy of the uncertainty of the measured contact angle.

It should also stressed that the wall position and the way it was measured may also account for some added uncertainty in the contact angle extracted, as explained previously.

5.3.2 Correlations

To analyse the correlations with the experimental data, we will first consider the Hoffman-Tanner correlation, which is one of the most common basis for developing several correlations [34].

This correlation, proposed by Hoffman [19], accounts for the relation of the dynamic contact angle with the capillary number, Ca. That relates the viscous forces with surface tension in the liquid-gas interface, as defined in 5.2. [33]

$$\text{Ca} = \frac{\mu}{\sigma} V \tag{5.2}$$

Where μ represents the dynamic viscosity, σ represents the surface tension and V is the velocity of

the three-phase interface. Hoffman used it in his correlation as expressed in 5.3.

$$\theta_d^3 - \theta_s^3 = a\text{Ca} \tag{5.3}$$

The correlation displayed in 5.3 assumes that a linear relationship between the meniscus three-phase interface and the contact angle. The contact angle for static conditions is also added to this correlation as an offset to the data. Figure 5.9 shows an optimisation of the correlation for the experimental data.

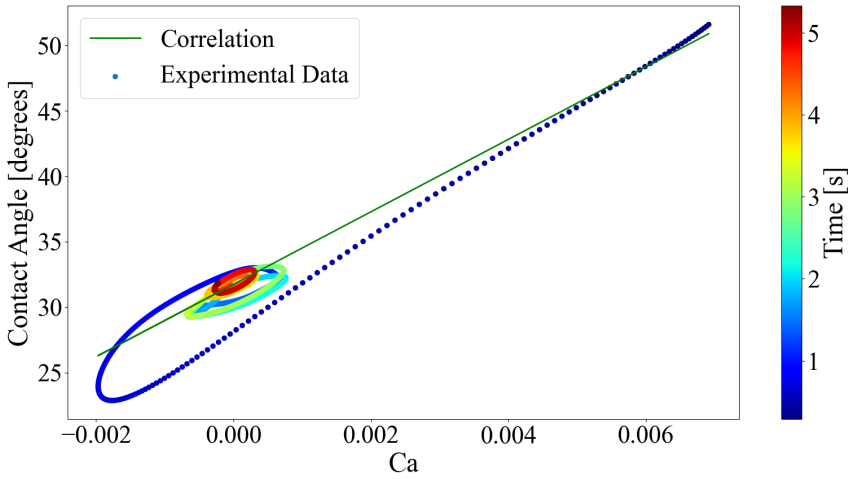


Figure 5.9: Experimental data for the contact angle plotted against time and the prediction by Hoffman-Tanner's correlation

Figure 5.9 displays the contact angle as a function of the capillary number extracted for the experimental run with the lowest pressure value applied to the reservoir 4. As indicated in the colour bar, the meniscus appears in the field of view with a high velocity and decelerates as time passes. So, towards the end of the observation time, the relation between the capillary number and the dynamic contact angle stops being linear as the contact angle oscillates around its static value. This is incompatible with Hoffman-Tanner's correlation, as can be seen from the green line in figure 5.9. The optimisation of the correlation's coefficients leads to a average the initial linear portion of the motion with the oscillations in the later stages. However, there is still some merit to this correlation since for for capillary numbers bigger than 0.002 the behaviour of the interface is linear.

Thus, the need for a correlation with extra terms that account for the effect of acceleration changes in the latter stages of the liquid motion. At this stage, the meniscus is oscillating around its equilibrium position. So to consider the effect of gravity and the acceleration of the contact line on the correlation, a dimensionless unit A is introduced where $A = \frac{a}{g}$ with a representing the contact line acceleration and g the gravitational acceleration. This new correlation can be seen in 5.4. [35]

$$\Theta_d = a\text{Ca}^b + c\text{A} + F(\Theta_s) \tag{5.4}$$

This correlation considers the acceleration, and it is possible to see in figure 5.10 that it describes better the variation of the contact angle with time and the capillary number.

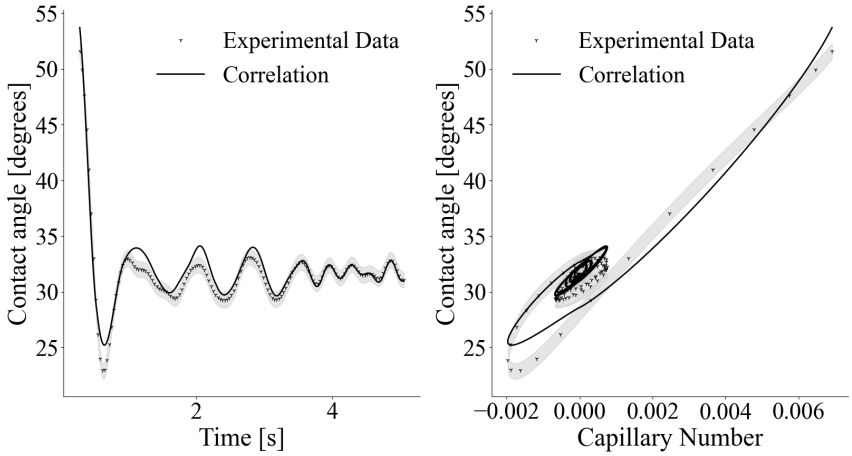


Figure 5.10: Results for the correlation proposed by Manuel Ratz. On the left side the correlation predictions and experimental data for the contact angle are plotted against time and on the right side they are plotted against the capillary number

Figure 5.10 shows that correlation 5.4 can follow the contact angle trend with time quite accurately, especially near the equilibrium position. The correlation has a cyclic elliptical behaviour around its equilibrium angle equal to that of the experimental data. The similarities between the experimental data and the correlation are also supported by the plot on the left, where the oscillations around the equilibrium contact angle are aligned. However, there are still some noticeable differences in the peaks in the later stages of the motion as seen in the plot on the left. Nonetheless, these differences are within the range of the uncertainty associated with the experimental measurements.

The problem with this correlation lies in the initial stages of the meniscus movement, where the correlation does not follow the experimental data with the same precision. As a result, the first oscillation produces lower contact angles than those predicted by the correlation, and the three following fluctuations are misaligned and overshoot the contact angle recorded by the experimental data.

To solve this issue, Domenico Fiorini suggested introducing an unsteady term proportional to the rate of change of the dynamic contact angle with time. [3] This term enables the correlation to adapt better to the rate of change of the dynamic contact angle and resolve the issues noted with correlation 5.4 since it performed very well when compared to the quasi-capillary U-tube experimental data. This correlation can be seen in 5.5.

$$\alpha\dot{\Theta}' + \Theta' = a\mathrm{Ca} + bA \tag{5.5}$$

In this equation, Θ' does not represent the dynamic contact angle but rather a parameter which also accounts for the static contact angle that offsets this correlation, similar to Hoffman-Tanner's correlation.

This parameter is given by 5.6.

$$\Theta' = \Theta_D^3 - \Theta_S^3 \tag{5.6}$$

The analytical solution of the correlation is represented in 5.7.

$$\Theta'(t) = \frac{1}{\alpha} e^{-\frac{t}{\alpha}} (a \int_0^t Ca(t') e^{\frac{t'}{\alpha}} dt' + b \int_0^t A(t') e^{\frac{t'}{\alpha}} dt') \tag{5.7}$$

In this equation, the entire history of the capillary number and acceleration plays a part in determining the dynamic contact angle. In addition, it is possible to see a decay factor responsible for the exponential decay of any disturbance in the meniscus shape.

To optimise the coefficients in equation 5.5 a set of initial conditions for the motion were provided to solve this first-order linear differential equation. These correspond to the contact angle observed within the first captured frame and the average meniscus velocity and acceleration up to that point which is calculated using the meniscus height and the time since the pressure valve opened. In addition to this, the boundaries for the optimisation of the coefficients were progressively adjusted to get a more accurate solution. The results for this procedure are displayed in figure 5.11.

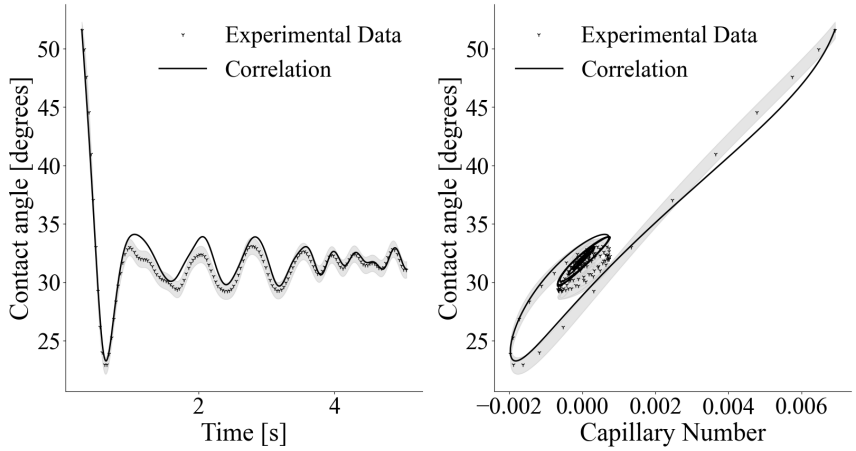


Figure 5.11: Results for the correlation proposed by Domenico Fiorini. On the left side the correlation predictions and experimental data for the contact angle are plotted against time and on the right side they are plotted against the capillary number

Figure 5.11 shows that the positions around the static contact angle are well described. However, this correlation performs much better than correlation 5.4 in the initial oscillation since it can follow the dip in the contact angle value very closely, just like it does in the following oscillation. In addition, it can replicate the harmonic behaviour of the dynamic contact angle quite accurately with the frequency in the oscillations matching that of the experimental data.

These results were achieved for the coefficients displayed in the following table.

α	a	b
0.15	26.42	3.92

Table 5.1: Table with the optimised coefficients of equation 5.5

These coefficients were compared to the experimental data for the remaining points, with each point consisting of a different over-pressure and several test runs as seen in chapter 4. In this regard, the model performed similarly with an offset to the data, which corresponds only to the difference in the static contact angle offset as seen in image 5.12.

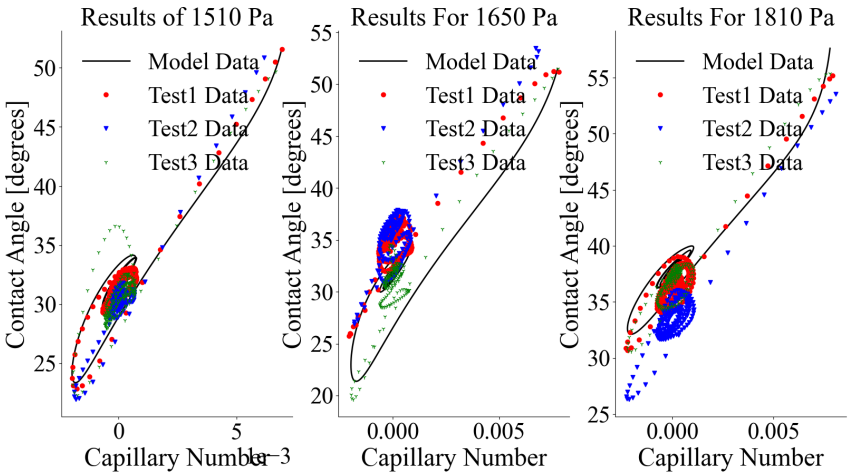


Figure 5.12: Model data against the experimental data obtained from several experimental points and their respective tests

5.4 Model Implementation

Considering the previous results, it is now possible to address the main objective of this campaign, which was to test the existing formulation for the capillary motion for of a liquid interface in the quasi-capillary parallel plates and use it to optimise the coefficients for dynamic contact angle correlation using the height variation of the meniscus.

To implement this formulation both equations 2.30 and 5.5 were coupled into a system of ordinary differential equations described in the set of equations 5.8.

$$\begin{cases} \dot{h} = v \\ \dot{v} = f(h, v, \theta') \\ \dot{\theta}' = f(\theta', v, \dot{v}) \end{cases} \tag{5.8}$$

This means that equation 5.5, the dynamic contact angle correlation, will substitute θ in the surface tension term of equation 2.30, the model for capillary motion. By doing this, it is now possible to optimise

the contact angle correlation coefficients using the experimental height data, which eliminates the need to measure the contact angle directly or through a fit of the meniscus.

This approach is different from the approaches described in the previous sections. In those sections, the contact angle has been measured directly and those measurements were used to assess the correlations and evaluating their ability to describe the phenomena. The approach in these previous section depends on several approximations were made, mainly using the Young-Laplace equation to measure the dynamic contact angle from each frame. In addition to this, measuring the contact angle is very difficult due to the uncertainty related with the wall positioning and meniscus brightness near it.

The new approach uses correlation 5.5 developed with the aid of the data measured during the experimental campaign and do away with the need for this approximation by eliminating the need for measuring the contact angle altogether and optimise the description of the motion of the interface by comparing it against the meniscus positions. The success of this new approach depends, firstly, on the validity of equation 2.30 and its initial conditions, and, secondly, on the dynamic contact angle correlation. This correlation has already been validated in the previous section. The initial conditions of equation 2.30 are the initial height and velocity of the liquid column when the meniscus first appears in the image. The initial height is approximately the height of the lower boundary of the camera's field of view, and the initial velocity is approximately the average velocity up to that point. We experimented different initial values and the best overall fit of the experimental data was obtained with the referred initial values.

Figure 5.13 exhibits the solution of equation set 5.8 with the most common parameters found in the literature for the capillary model 2.30. The numerical simulation shows both higher frequency and lower damping than the experimental data. This is an indication that the present test rig requires different parameters, that we are going to identify in the next sections. This second order equation is similar to spring-mass-damper system. [24] [25]

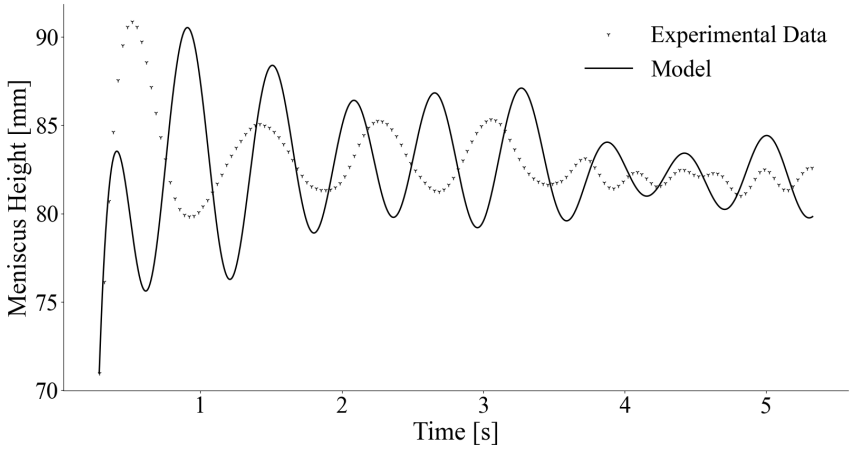


Figure 5.13: Plot of the model equation using the coefficients from the literature against the experimental data

5.4.1 Added Mass

The mass of fluid inside the channel is known, from the geometry, but the overall moving mass is greater, including some fluid in the reservoir. This contribution to the inertia term is commonly known as the added mass. For a tube of radius R, with its lower end submerged in an unbounded reservoir, [12] calculated that the added mass is equivalent to increase the liquid column height by $73/60\,R$. According to figure 5.13, the added mass is far greater for a channel between two parallel plates, because the experiments show a lower natural frequency of the liquid oscillations. Since the natural frequency of a dynamic system is roughly inversely proportional to the square root of the mass, and the experimental data exhibits a frequency 1.5 times greater than the simulations, the effective moving mass must increase by a factor of about 1.2. That means that the added mass corresponds approximately to an added height of $0.2\bar{h} \simeq 28\,\frac{\delta}{2}$. Figure 5.14 was obtained with this estimate of the added mass, that improves significantly the quality of the simulations.

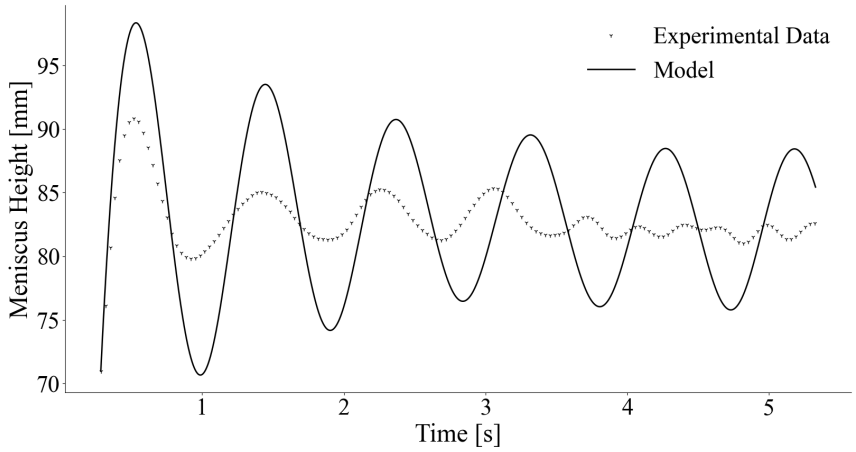


Figure 5.14: Plot of the model equation using optimised added mass factor against the experimental data

The value obtained for the added mass parameter is very different from the one indicated in the literature. In [12], creeping flow is considered and, thus, the range of Reynolds number considered is much smaller than the one obtained for the experimental data of Re $\simeq 200$, which would imply a significant difference in the phenomena description. Furthermore, the geometry described in [12] is a axial-symmetric which is not the case for the experimental set-up used. All in all, this results in the increase of the influence of the flow within the channel on the surrounding reservoir which leads to a higher added mass coefficient to make up for that difference.

5.4.2 Dissipation

Figure 5.14 shows that the dissipation parameters are too small to reproduce the experimental evolution. There are two kinds of dissipation terms, the viscous force on the wall and the Hagenbach-Couette correction 2.28. The viscous force on the wall was initially evaluated as if the flow was a developed Couette flow under a given pressure gradient. The whole dissipative term is

$$\text{Dissipative Terms} = \left[\frac{12\mu}{\rho\delta^2}h + \frac{K_H\dot{h}}{2} + \frac{\mu K_C}{2\delta\rho}\right]\dot{h}. \tag{5.9}$$

The average height is $\bar{h} = 50\,\text{mm}$, the channel thickness is $\delta = 5\,\text{mm}$ and an average vertical velocity of $\dot{h} = 2\text{mm/s}$. For the present fluid properties and coefficients given in the literature, $K_H = 7/3$ and $K_C = 4$, the Hagenbach-Couette correction represents about 20% of the total magnitude of the dissipative terms. However, if we consider the first oscillation of the liquid column where the vertical velocities are on average $\dot{h} = 200\text{mm/s}$, the Hagenbach-Couette correction assumes 95% of the total magnitude of the dissipative terms. So, we conducted an optimisation procedure for the Hagenbach-Couette coefficients based on the varying impact this correction has throughout the motion. This led us to $K_H = 1000$ and $K_C = 17.6$ which were used to generate the plot in figure 5.15.

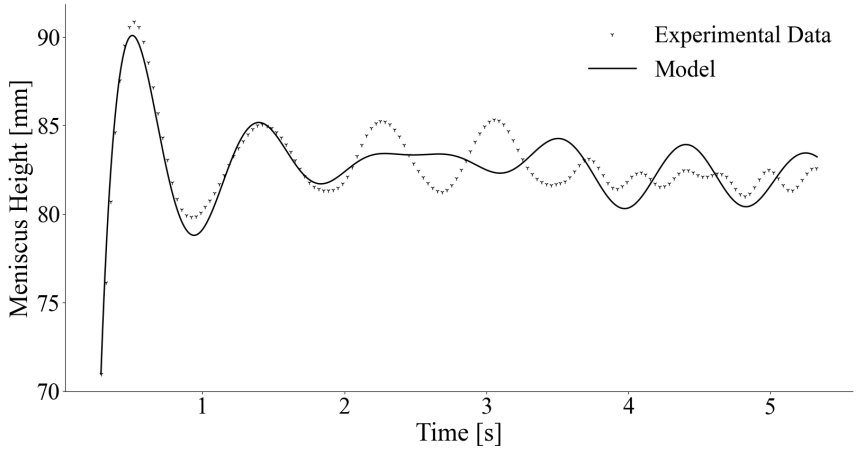


Figure 5.15: Plot of the optimised model equation predictions and the experimental data against time

Figure 5.15 shows that there are still significant differences between the model and the experimental data. The flow in this oscillating liquid column is not developed and the cross-sectional average velocity changes over time. These two features affect both the wall shear stress and the vertical component of the momentum flow rate. The curvature of the shape of the free surface also changes in time and, therefore, the pressure jump due to the surface tension. The PIV velocity profiles measured by Manuel Ratz [35] demonstrated a departure from the parabolic velocity profile towards a flatter velocity profile, see figure 5.16 of that report. As stated before, the wall shear is not important and, as we will see later, the change in the surface tension and the momentum flow rate also have a negligible impact on the

dynamic of the liquid column.

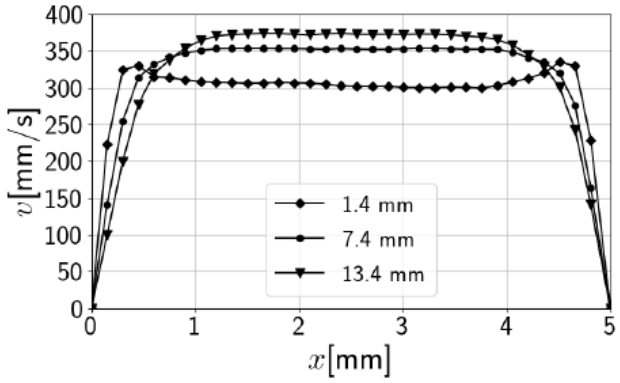


Figure 5.16: Velocity profile at several distances from the interface

5.4.3 Impact of the Dynamic Contact Angle

Figure 5.17 takes into consideration the previous optimisation procedure and tests the model sensitivity to each parameter. The forces considered in the model are plotted as a function of the height change they would produce on the liquid column if they were applied by themselves.

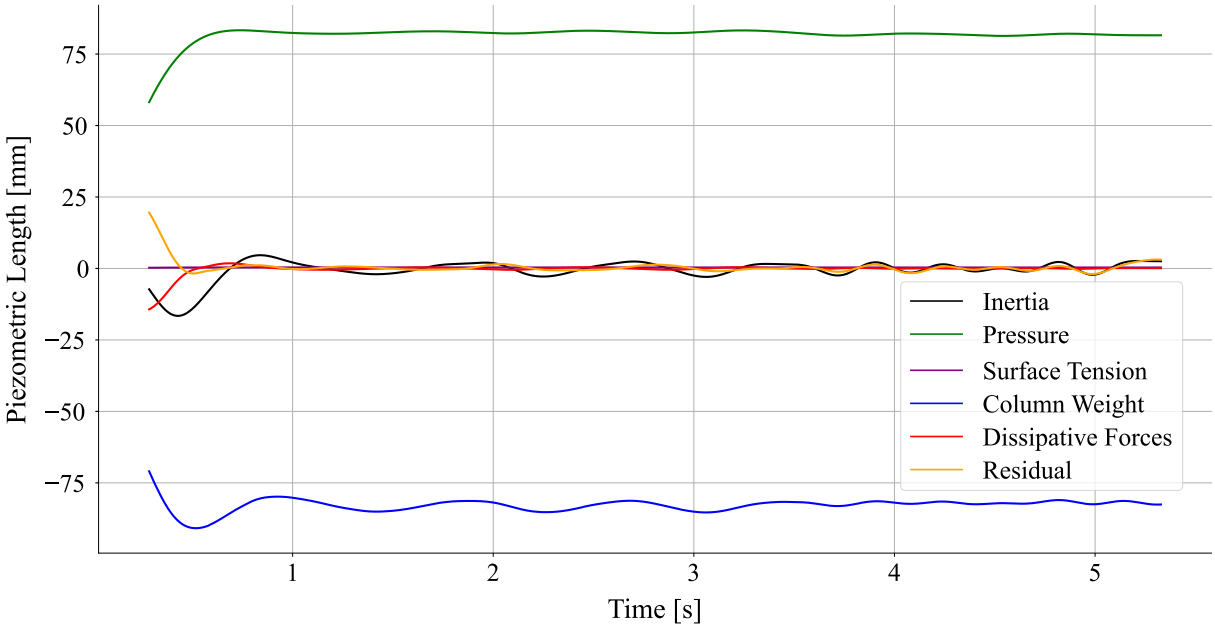


Figure 5.17: Plot of the forces produced by each term in the model equation as the height change they would produce in the liquid column

Figure 5.17 represents forces per unit length divided by $\rho g \delta$. This result is the height of a hydrostatic liquid column subject to that force. Figure 5.17 shows that the major forces at play are pressure and gravity, which somehow cancel out.

Up until the latter stages of the motion, the dissipative forces seem to be closely linked to the pressure as seen in figure 5.18. This suggests that the velocity is linked with the pressure applied to the reservoir and not influenced by surface tension.

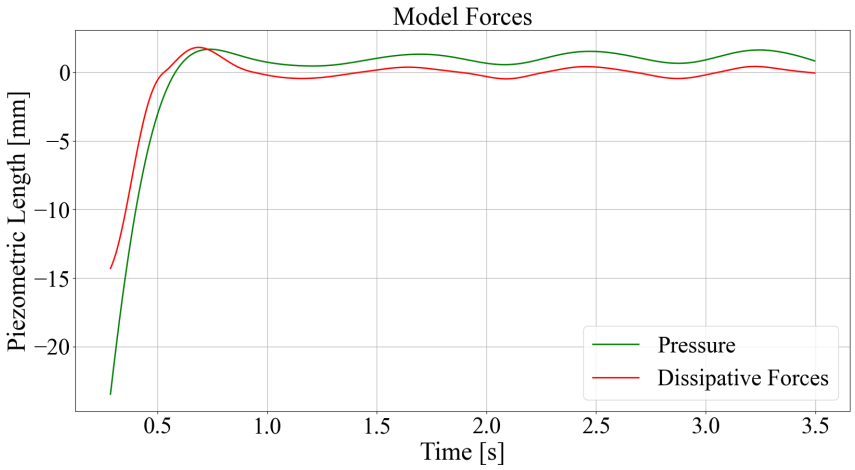


Figure 5.18: Plot of the pressure evolution in the reservoir against and dissipative term against time

Figure 5.17 shows that the modulus of the surface tension is small compared to the other forces at play, and its variation is also very small despite the variation in the contact angle, as seen in figure 5.19.

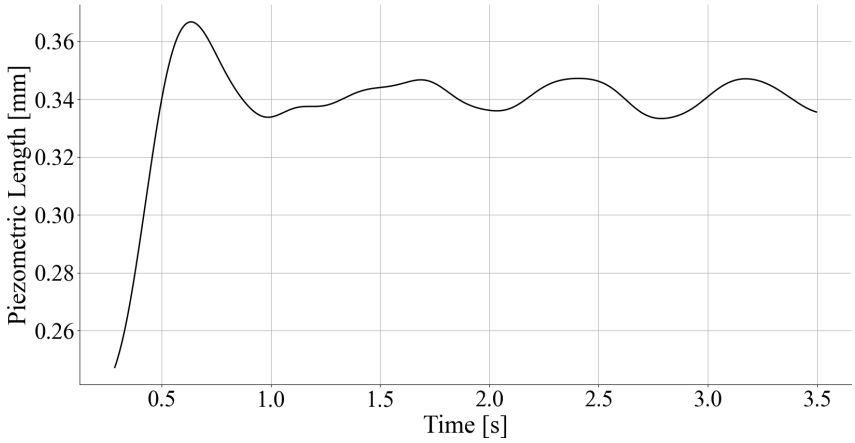


Figure 5.19: Plot of the surface tension forces against time

From this plot, it is possible to assume that the model is not yet sensitive enough to require adjustments of this order of magnitude, especially considering the difference in predicted and experimental heights displayed in figure 5.15. To test the sensitivity of the results to the contact angle correlation, the model was run using the coefficients for the contact angle correlation previously obtained and using coefficients one order of magnitude higher. Figure 5.20 shows that both solutions overlap within plotting precision.

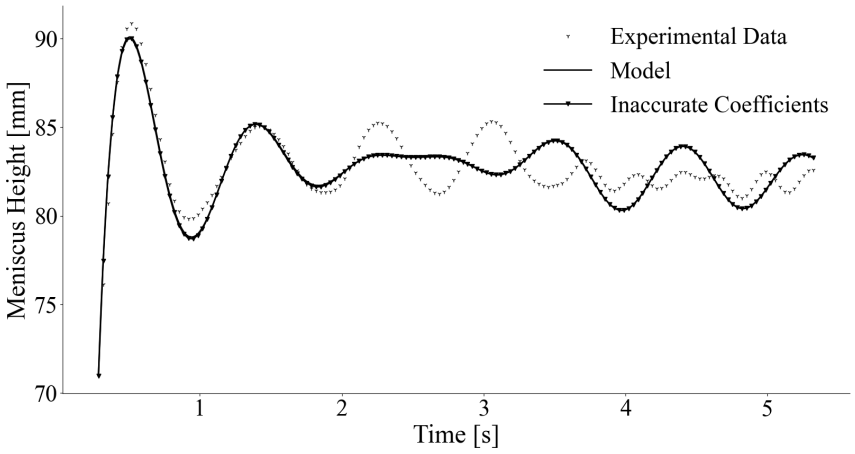


Figure 5.20: Plot of the experimental data against model predictions using the dynamic contact angle correlation coefficients obtained in the optimisation process and a random set of coefficients for this correlation

5.4.4 Additional modelling

To improve the model sensitivity to the contact angle variations, modelling of the motion of the meniscus should be expanded to account for three dimensional phenomena. The vibrations the meniscus interface presents in the latter stages of the motion could be better modelled by accounting for the vibration modes along the channel length.

If we were to compare it with a string constrained at each end, the first vibration mode would impact the measurements of the meniscus height significantly. As seen in figure 5.21, an anti-node would be present at the middle of the channel, which would impact the height measurements taken at this point. For the second vibration mode this would not be an issue, since the a node would appear in the middle of the channel.

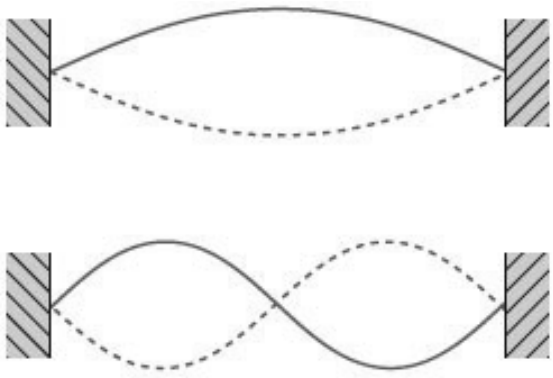


Figure 5.21: Vibration modes for a string constrained at its edges

6

Conclusion

Contents

Chapter 5 enables us to answer the research questions formulated in the introduction.

Did the prevailing model for optical correction yield the expected results?

The results obtained with the method employed for obtaining the Θ_0 parameter seem to be very encouraging since the error associated with the correction of the inner diameter was similar to that found in the literature that conducted identical experiments. [26] [27]

This method allows for a much simpler correction of the results without requiring more expensive instrumentation. Furthermore, the correction performed on the images from the quasi-capillary u-tube yielded similar contact angles to that obtained using the data from the quasi-capillary parallel plates.

Is the static contact angle similar across different geometries?

According to Jiang et al. [33], the static contact angle for the same fluid and material should be independent of geometry. Yet, there was a noticeable difference between the axial-symmetric menisci and the quasi-capillary parallel plates. This difference was attributed to the image cropping procedure for the parallel plates facility, which might have omitted parts of the image near the wall where the meniscus was not visible.

Are the prevailing integral models adequate to describe the motion of the liquid column between Parallel Plates?

As seen in chapter 5, the common models used for the capillary tubes, the geometry usually studied in the literature, required many modifications to adapt them to the parallel plates geometry. Surface tension is less significant for the column between parallel plates of a given width than inside a tube of similar diameter. Furthermore, our test rig had a larger width than it is common in capillary tubes. On the other hand, the introduction of a term that accounted for the acceleration of the contact line and the history of the meniscus shape led to a good description of the contact angle when compared to the more traditional correlations in reference [19]. In addition, there is a need to better describe the unsteady phenomena, mainly with the velocity profile with the channel and the changing velocity field at the tube entrance.

Are the capillary models sensitive to an optimisation based on the correlation's coefficients?

With the current model, an optimisation of the correlation's coefficients inside the surface tension term does not seem to be feasible. However, further work on better modelling of the unsteady velocity profile and improving of the coefficients can achieve enough accuracy. Nevertheless, the present correlation for the contact angle seems to capture its variation, having the same harmonic behaviour and being able to significantly predict its values. So, this might be a worthy topic of research for future work.

6.1 Future Work

With this said, future work should focus on better modelling for the quasi-capillary parallel plate set-up. The model used is based on equations developed for the rise of liquid in a capillary tube, and a

departure from these equations might be needed, especially regarding their description of the dissipative terms. The term for the pressure loss at the entrance to the channel should be changed to account for the different flow conditions when the liquid is rising and receding. In Levine et al. [12], the fluid is constantly rising, which is not the case for this quasi-capillary flow. In addition, a departure from the steady flow approximations might be required to account for the velocity profile. As seen in figure 5.16, the velocity profile is not parabolic throughout the motion, and this needs to be accounted for to model the term concerning the viscous forces on the wall. Finally, better detection of the meniscus interface in the vicinity of the walls of the quasi-capillary parallel plates is needed to establish a fairer comparison between the different static contact angles.

Bibliography

[1] Tian X. Korhonen J.T. Huhtamäki, T. Surface-wetting characterization using contact-angle measurements. *Nature Protocols*, 13:17, 2018. doi: 10.1021/i160025a005.

[2] S. R. Friedman, M. Khalil, and P. Taborek. Wetting transition in water. *Phys. Rev. Lett.*, 111:5, Nov 2013. doi: 10.1103/PhysRevLett.111.226101.

[3] Fiorini D. Dynamic contact angle analysis using inverse methods. 12th Symposium of VKI PhD Research. von Karman Institute, 2021. Under Revision.

[4] P. Muthu Vijay and Amar Prakash. Analysis of sloshing impact on overhead liquid storage structures. *Impact Journal*, 2:16, Aug 2014.

[5] Philipp Behruzi, Francesco De Rose, and Francesca Cirillo. *Coupling sloshing, GNC and rigid body motions during ballistic flight phases*. doi: 10.2514/6.2016-4586.

[6] Abdullah Alghunaim, Suchata Kirdponpattara, and Bi min Zhang Newby. Techniques for determining contact angle and wettability of powders. *Powder Technology*, 287:14, 2016. doi: https://doi.org/10.1016/j.powtec.2015.10.002.

[7] Yuehua Yuan and T. Randall Lee. Contact angle and wetting properties. In *Surface Science Techniques*, Springer Series in Surface Sciences. Springer Berlin, Heidelberg, 2013.

[8] Anna Rudawska and Elzbieta Jacniacka. Analysis for determining surface free energy uncertainty by the owen–wendt method. *International Journal of Adhesion Adhesives*, (29):6, 2009.

[9] Jordan G. Petrov and Rossen V. Sedev. Quasi-static shape of the steady moving meniscus — comparison of the apparent and extrapolated dynamic contact angles. *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 74(2):8, 1993. doi: https://doi.org/10.1016/0927-7757(93)80266-H.

[10] Markus Deserno. The shape of a straight fluid meniscus, 2005. Lecture notes Max-Planck Institute For Polymer Research, Mainz, Germany.

[11] Adam Eales, Nick Dartnell, Simon Goddard, and Alexander Routh. Evaporation of pinned droplets containing polymer - an examination of the important groups controlling final shape. *AIChE Journal*, 61, 02 2015. doi: 10.1002/aic.14777.

[12] Samuel Levine, James Lowndes, Eric J Watson, and Graham Neale. A theory of capillary rise of a liquid in a vertical cylindrical tube and in a parallel-plate channel: Washburn equation modified to account for the meniscus with slippage at the contact line. *Journal of Colloid and Interface Science*, 73(1):15, 1980. doi: https://doi.org/10.1016/0021-9797(80)90131-9.

[13] J Szekely, A.W Neumann, and Y.K Chuang. The rate of capillary penetration and the applicability of the washburn equation. *Journal of Colloid and Interface Science*, 35(2):5, 1971. doi: https://doi.org/10.1016/0021-9797(71)90120-2.

[14] Michael Stange, Michael Dreyer, and Hans Rath. Capillary driven flow in circular cylindrical tubes. *Physics of Fluids*, 15, 08 2003. doi: 10.1063/1.1596913.

[15] Edward W. Washburn. The dynamics of capillary flow. *Phys. Rev.*, 17:10, Mar 1921. doi: 10.1103/PhysRev.17.273.

[16] Steven Deutsch. A preliminary study of the fluid mechanics of liquid penetrant testing. *Journal of research of the National Bureau of Standards*, 84:6, 1979.

[17] D. C. Bogue. Entrance effects and prediction of turbulence in non-newtonian flow. *Industrial & Engineering Chemistry*, 51(7):4, 1959. doi: 10.1021/ie50595a044.

[18] F. A. Seyer and P. J. Catania. Laminar and turbulent entry flow of polymer solutions. *The Canadian Journal of Chemical Engineering*, 50(1):5, 1972. doi: https://doi.org/10.1002/cjce.5450500106.

[19] Richard L Hoffman. A study of the advancing interface. i. interface shape in liquid—gas systems. *Journal of Colloid and Interface Science*, 50(2):13, 1975. doi: https://doi.org/10.1016/0021-9797(75)90225-8.

[20] O. V. Voinov. Hydrodynamics of wetting. *Fluid Dynamics*, (11):7, 1976. doi: https://doi.org/10.1007/BF01012963.

[21] Michael H. Rausch, Lorenz Kretschmer, Stefan Will, Alfred Leipertz, and Andreas P. Fröba. Density, surface tension, and kinematic viscosity of hydrofluoroethers hfe-7000, hfe-7100, hfe-7200, hfe-7300, and hfe-7500. *Journal of Chemical & Engineering Data*, 60(12):6, 2015. doi: 10.1021/acs.jced.5b00691.

[22] E. M. Sparrow, S. H. Lin, and T. S. Lundgren. Flow development in the hydrodynamic entrance region of tubes and ducts. *The Physics of Fluids*, 7(3):338–347, 1964. doi: 10.1063/1.1711204.

[23] N. V. K. Dutt. Determination of hagenbach and coutte correction factors for the flow of power law fluids. *Physics and Chemistry of Liquids*, 35(4):17, 1998. doi: 10.1080/00319109808030591.

[24] G Franklin, J.D. Powell, and Abbas Emami-Naeini. *Feedback Control Of Dynamic Systems*. 01 1994.

[25] B. Yang. Theory of vibration — fundamentals. In S. Braun, editor, *Encyclopedia of Vibration*, pages 1290–1299. Elsevier, Oxford, 2001. ISBN 978-0-12-227085-7. doi: https://doi.org/10.1006/rwvb.2001.0112. URL https://www.sciencedirect.com/science/article/pii/B0122270851001120.

[26] Milad Darzi and Chanwoo Park. Optical distortion correction of a liquid-gas interface and contact angle in cylindrical tubes. *Physics of Fluids*, 29(5), 2017. doi: 10.1063/1.4982902.

[27] M.L. Lowe and P.H. Kutt. Refraction through cylindrical tubes. *Experiments in Fluids*, 13(5):315–320, 1992. doi: 10.1007/BF00209503.

[28] Balazs Toth, Jerome Anthoine, and Johan Steelant. Cryogenic sloshing investigation by means of non-intrusive measurement techniques. page 14, 2016.

[29] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-Local Means Denoising. *Image Processing On Line*, 1:4, 2011. doi: 10.5201/ipol.2011.bcm_nlm.

[30] M. A. Mendez, L. Németh, and J. M. Buchlin. Measurement of Liquid Film Thickness via Light Absorption and Laser Tomography. In *European Physical Journal Web of Conferences*, volume 114 of *European Physical Journal Web of Conferences*, 2016. doi: 10.1051/epjconf/201611402072.

[31] Abdulbasit Alazzawi, Husam Alsaadi, Abidaoun Shallal, and Saad Albawi. Edge detection-application of (first and second) order derivative in image processing. 12 2015.

[32] M.A. Mendez and F. Fontaneto. Principles of measurement systems, 2020. Lecture notes VKI.

[33] Tsung-Shann Jiang, O.H Soo-Gun, and John C Slattery. Correlation for dynamic contact angle. *Journal of Colloid and Interface Science*, 69(1):4, 1979. ISSN 0021-9797. doi: 10.1016/0021-9797(79)90081-X.

[34] Stephan F. Kistler. The hydrodynamics of wetting. *Wettability*, 1993.

[35] Manuel Ratz. Experimental analysis of contact line dynamics using optical techniques and inverse methods. Master's thesis, Technische Fakultät Praktikumsarbeit Universität, 2021.

7

Curvature Calculations

In this appendix, a description of the curvature for both the droplet and both menisci of the quasi-capillary u-tube and parallel plates.

7.1 Polar coordinates

Both the droplet and quasi-capillary u-tube meniscus can be described in polar coordinates, $z = h(r, \theta)$. A functional that is zero on both surfaces can be defined as such $g(r, \theta, z) = z - h(r, \theta)$. This functional can be used to compute the normal to both surfaces using equation 7.1.

$$n = \frac{\nabla g}{\mid \nabla g \mid} = \frac{\hat{z} - h_r \hat{r} - \frac{1}{r} h_\theta \hat{\theta}}{(1 + h_r^2 + \frac{1}{r^2} h_\theta^2)^{1/2}} \tag{7.1}$$

The local curvature can then be calculated using the gradient of the normal in equation 7.2.

$$\nabla \cdot n = \frac{-h_{\theta\theta} - h_r^2 h_{\theta\theta} + h_r h_\theta - r h_r - \frac{2}{r} h_r h_\theta^2 - r^2 h_{rr} - h_{rr} h_\theta^2 + h_r h_\theta h_{r\theta}}{r^2 (1 + h_r^2 + \frac{1}{r^2} h_\theta^2)^{3/2}} \tag{7.2}$$

If the axisymmetry of both interfaces is taken into account then equation 7.3 is obtained.

$$\nabla \cdot n = \frac{-r h_r - r^2 h_{rr}}{r^2 (1 + h_r^2)^{3/2}} \tag{7.3}$$

7.2 Cartesian coordinates

In the case of the quasi-capillary parallel plates the coordinate system used is Cartesian so a new formulation is necessary: $z = h(x, y)$. Once again, a functional is defined as $f(x, y, z) = z - h(x, y)$ which is zero in the meniscus surface. With this in mind the normal vector is given by equation 7.4.

$$n = \frac{\nabla f}{\mid \nabla f \mid} = \frac{\hat{z} - h_x \hat{x} - h_y \hat{y}}{(1 + h_x^2 + h_y^2)^{1/2}} \tag{7.4}$$

This leads to the curvature expressed in equation 7.5.

$$\nabla \cdot n = \frac{-(h_{xx} + h_{yy}) - (h_{xx} h_y^2 + h_{yy} h_x^2) + 2 h_x h_y h_{xy}}{(1 + h_x^2 + h_y^2)^{3/2}} \tag{7.5}$$

If the two dimensional nature of the phenomena is taken into consideration, then the curvature equation reduces to equation 7.6.

$$\nabla \cdot n = \frac{-h_{xx}}{(1 + h_x^2)^{3/2}} \tag{7.6}$$

Equations 7.6 and 7.3 are the ones used to deduce the equation for the static fit of the interface.

8

Distortion Angles

In this appendix, a description for all the refraction contact angles and the angles with horizontal axis is made.

8.1 Refraction angles

The refraction angles as described in figure 3.3 can be related with Ω, through the following equations:

$$sin\alpha = \frac{n_o}{n_f}\frac{L}{R_i}sin\Omega \tag{8.1}$$

$$sin\beta = \frac{n_o}{n_w}\frac{L}{R_i}sin\Omega \tag{8.2}$$

$$sin\gamma = \frac{n_o}{n_w}\frac{L}{R_o}sin\Omega \tag{8.3}$$

$$sin\delta = \frac{L}{R_o}sin\Omega \tag{8.4}$$

8.2 Angles with Horizontal Axis

With the refraction angles calculated, it is now possible to calculate the angles with the horizontal axis as represented in figure 3.3 and described in the following equations.

$$\theta_A = \delta + \beta - \gamma - \Omega + \theta_o \tag{8.5}$$

$$\theta_B = \delta - \Omega + \theta_o \tag{8.6}$$

$$\theta_C = \pi + \delta - 2\alpha + \beta - \gamma - \Omega + \theta_o \tag{8.7}$$

9

Distortion Angles

9.1 Optical Distortion Code

```
#%%NECESSARY LIBRARIES

import sys
sys.path.append('../libraries/')
#import lib_ImgProcessing as ImPro
#import lib_CapillaryFuncs as capillary
import numpy as np
import cv2
from matplotlib import pyplot as plt
from scipy.signal import savgol_filter
import os
import sys
import time
from scipy.optimize import minimize
from random import randrange
from scipy.signal import find_peaks

plt.close('all')
#%%CUSTOM FUNCTIONS

#function to import image
def import_images(image_path, image_name, image_number, image_format, rotation_angle, crop_coordinates):

    toavg_img = np.zeros((crop_coordinates[1]-crop_coordinates[0],crop_coordinates[3]-crop_coordinates[2],image_number[1]-image_number[0]))

    for k in range(image_number[0],image_number[1]):

        file_path = image_path + os.sep + image_name + '%02d' %k + image_format

        print('Read image ' + str(k))

        img   = cv2.imread(file_path,0)

        #image rotation (if proper cropping hasn't been done yet, this might make the script not work)
        image_center = tuple(np.array(img.shape[1::-1]) / 2)
        rot_mat = cv2.getRotationMatrix2D(image_center, rotation_angle, 1.0)
        img_rotated = cv2.warpAffine(img, rot_mat, img.shape[1::-1], flags=cv2.INTER_LINEAR)

        dst   = cv2.fastNlMeansDenoising(img_rotated,10,10,7,21)

        crop_img =dst[crop_coordinates[0]:crop_coordinates[1],crop_coordinates[2]:crop_coordinates[3]]

        toavg_img[:,:,k] = crop_img


    image = np.mean(toavg_img,axis=2)

    return image

#function that sorts the vectors
def sortingvectors(x_index, y_index):

    """
    Sorts vector x_index and matches this with y_index

    ---
    Parameters:
        x_index: vector to be sorted
```

```
        y_index: vector to be matched
    """
    #sort points according to their position along the x axis

    #creating the vectors that will be sorted
    x_sorted = []
    y_sorted = []


    x_sorted, y_sorted = (list(t) for t in zip(*sorted(zip(x_index, y_index))))

    x_sorted = np.array(x_sorted)

    y_sorted = np.array(y_sorted)

    return y_sorted, x_sorted

#function to detect horizontal lines
def detect_horizontal_line(image, search_interval):

    #defining image area to be searched
    image_band = image[search_interval[0]:search_interval[1], :]

    #assembling kernel to find horizontal line
    E = np.array([[1,2,1],
                  [1,2,1],
                  [1,2,1]])

    C = np.array([[0,-1,0],
              [0,0,0],
              [0,1,0]])

    kernel_MM = np.vstack((-E, C, E))

    #high pass filtering the recontrasted image
    filtered_image = cv2.filter2D(image_band,-1, kernel_MM)

    #define height and distance of the peaks to be considered
    height_peaks = 250
    distance_peaks = 25
    peaks_positions = np.zeros(image_band.shape)

    #obtain line positions
    for column in range(filtered_image.shape[1]):

        peaks, _ = find_peaks(filtered_image[:,column], height = height_peaks,
    distance= distance_peaks)

        if(len(peaks) != 0):
            peaks_positions[peaks, column] = 1

    #storing line positions in two separate arrays
    y_line, x_line = np.where(peaks_positions == 1)

    #offsetting line position
    y_line = y_line + search_interval[0]

    y_line, x_line = sortingvectors(x_line, y_line)

    return x_line, y_line

#get the points in extremities
def points_extremities(line_x, line_y):
    x_1 = line_x[0]
    x_2 = line_x[len(line_x)-1]

    y_1 = line_y[0]
```

```
    y_2 = line_y[len(line_y)-1]

    return np.array([x_1, x_2]), np.array([y_1, y_2])

#function to obtain real position line
def real_positions(R_O, R_I, pix2mm):

    #inner tube wall for the left side
    x_real1= (R_O - R_I)/pix2mm

    #inner tube wall for the right side
    x_real2 = (R_O + R_I)/pix2mm

    return x_real1, x_real2

#function to correct points
def corrected_position(x_coordinate, pix2mm, R0, Ri, L_cm, T_degrees):

    """
    This function corrects the position of the detected points

    ---
    Parameters:
        x_coordinate: detected x positions
        pix2mm: pixel to mm ratio
        R0: outer utube diameter [mm]
        Ri: inner utube diameter [mm]
        vertical_dividers: number of strips in the calibration grid
        L_cm: distance between the center of the utube and camera [cm]
        T_degrees: angle between camera and utube [degrees]
    """

    #converting inputs to correct dimensions
    L = L_cm*10
    T0 = np.radians(T_degrees)
    x_raw = x_coordinate*pix2mm

    # Refractive indexes
    no = 1     # refraction index air
    nf = 1.282  # refraction index HFE
    nw = 1.458   # refraction index quartz

    X = (x_raw-R0)  # shift the coordinate sistem to go from [0:2R] to [-R:R]
    h1p    = np.sqrt(R0**2-X**2)
    h1     = L-h1p
    omega = np.arctan(X/h1)   # angle of view of the camera

    # refraction angles calculation
    alpha = np.arcsin((no/nf)*(L/Ri)*np.sin(omega))
    beta  = np.arcsin((no/nw)*(L/Ri)*np.sin(omega))
    gamma = np.arcsin((no/nw)*(L/R0)*np.sin(omega))
    delta = np.arcsin((L/R0)*np.sin(omega))

    TA = delta+beta-gamma-omega+T0
    TB = delta-omega+T0

    # compute correction
    correction = R0*(np.sin(TA-TB)/np.sin(np.radians(180)-beta))*np.sin(TA-beta)

    index_nan = np.argwhere(np.isnan(correction))

    if len(index_nan) != 0:
        print('Unable to calculate the correction!')

    #this is to prevent the cases in which the algorithim is not able to calculate
    alpha or beta
    correction = np.delete(correction, index_nan)
```

```
    x_raw = np.delete(x_raw, index_nan)
    X = np.delete(X, index_nan)

    #compute the corrected positions
    x_corrected     = x_raw - correction

    return x_corrected

#function to correct points
def corrected_position_uncertainty(x_coordinate, pix2mm, R0, Ri, L_cm, T_degrees):

    """
    This function calculates the uncertainty of the corrected points

    ---
    Parameters:
        x_coordinate: detected x positions
        pix2mm: pixel to mm ratio
        R0: outer utube diameter [mm]
        Ri: inner utube diameter [mm]
        vertical_dividers: number of strips in the calibration grid
        L_cm: distance between the center of the utube and camera [cm]
        T_degrees: angle between camera and utube [degrees]
    """

    #converting inputs to correct dimensions
    L = L_cm*10
    T0 = np.radians(T_degrees)
    x_raw = x_coordinate*pix2mm

    # Refractive indexes
    no = 1     # refraction index air
    nf = 1.282  # refraction index HFE
    nw = 1.458   # refraction index quartz

    X = (x_raw-R0)  # shift the coordinate sistem to go from [0:2R] to [-R:R]
    h1p    = np.sqrt(R0**2-X**2)
    h1     = L-h1p
    omega = np.arctan(X/h1)   # angle of view of the camera

    # refraction angles calculation
    alpha = np.arcsin((no/nf)*(L/Ri)*np.sin(omega))
    beta  = np.arcsin((no/nw)*(L/Ri)*np.sin(omega))
    gamma = np.arcsin((no/nw)*(L/R0)*np.sin(omega))
    delta = np.arcsin((L/R0)*np.sin(omega))

    TA = delta+beta-gamma-omega+T0
    TB = delta-omega+T0

    # compute correction
    correction = R0*(np.sin(TA-TB)/np.sin(np.radians(180)-beta))*np.sin(TA-beta)

    index_nan = np.argwhere(np.isnan(correction))

    if len(index_nan) != 0:
        print('Unable to calculate the correction!')

    #this is to prevent the cases in which the algorithim is not able to calculate
    alpha or beta
    correction = np.delete(correction, index_nan)
    x_raw = np.delete(x_raw, index_nan)
    X = np.delete(X, index_nan)

    #compute the corrected positions
    x_corrected     = x_raw - correction

    #measured variables
```

```
    uR0 = 0.001 #mm

    uRi = 0.001 #mm

    uL = 5 #mm

    uT0 = 0. #rad

    ucrop_index = 1 #pix

    ux_coordinate = 1 #pix

    #derived variables

    #pix2mm uncertainty

    upix2mm = np.sqrt(((2/(crop_indx2-crop_indx1))*uR0)**2+((2*R0/(crop_indx2-
    crop_indx1)**2)*ucrop_index)**2)

    #x_raw uncertainty

    ux_raw = np.sqrt((x_coordinate*upix2mm)**2+(pix2mm*ux_coordinate)**2)

    #omega uncertainty

    uX = np.sqrt((ux_raw)**2+(uR0)**2)

    uh1p = np.sqrt(((R0/np.sqrt(R0**2-X**2))*uR0)**2+((X/np.sqrt(R0**2-X**2))*uX)**2)

    uh1 = np.sqrt(uL**2+uh1p**2)

    uomega = np.sqrt(((1/(1+(X/h1)**2))*(1/h1)*uX)**2+((1/(1+(X/h1)**2))*(X/(h1**2))*
    uh1)**2)

    #beta uncertainty

    w1 = no*L*np.sin(omega)/(nf*Ri)

    ubeta = np.sqrt(((1/np.sqrt(1-w1**2))*(no*L*np.cos(omega)*uomega)/(nf*Ri))**2+((1/
    np.sqrt(1-w1**2))*(no*np.sin(omega)*uL)/(nf*Ri))**2+((1/np.sqrt(1-w1**2))*(no*L*np.
    sin(omega)*uRi)/(nf*Ri**2))**2)

    #gamma uncertainty

    w2 = no*L*np.sin(omega)/(nw*R0)

    ugamma = np.sqrt(((1/np.sqrt(1-w2**2))*(no*L*np.cos(omega)*uomega)/(nw*R0))**2+((1/
    np.sqrt(1-w2**2))*(no*np.sin(omega)*uL)/(nw*R0))**2+((1/np.sqrt(1-w2**2))*(no*L*np.
    sin(omega)*uR0)/(nw*R0**2))**2)

    #delta uncertainty

    w3 = L*np.sin(omega)/R0

    udelta = np.sqrt(((1/np.sqrt(1-w3**2))*(L*np.cos(omega)*uomega)/R0)**2+((1/np.sqrt
    (1-w3**2))*(np.sin(omega)*uL)/R0)**2+((1/np.sqrt(1-w3**2))*(L*np.sin(omega)*uR0)/(
    R0**2))**2)

    #TA and TB uncertainty

    uTA = np.sqrt(udelta**2+ubeta**2+ugamma**2+uomega**2+uT0**2)

    uTB = np.sqrt(udelta**2+uomega**2+uT0**2)

    #correction uncertainty

    aux1 = (np.sin(TA-TB)*np.sin(TA-beta)/np.sin(np.pi-beta))*uR0
```

```
317
318     aux2 = (R0*(np.cos(TA-TB)*np.sin(TA-beta)+np.sin(TA-TB)*np.cos(TA-beta))/np.sin(np.
        pi-beta))*uTA
319
320     aux3 = (R0*np.cos(TA-TB)*np.sin(TA-beta)/np.sin(np.pi-beta))*uTB
321
322     aux4 = R0*((np.cos(np.pi-beta)*np.sin(TA-TB)*np.sin(TA-TB)/(np.sin(np.pi-beta)**2))
        -(np.sin(TA-TB)*np.cos(TA-beta)/np.sin(np.pi-beta)))*ubeta
323
324     ucorrection = np.sqrt((aux1)**2+(aux2)**2+(aux3)**2+(aux4)**2)
325
326     #x_corr uncertainty
327
328     ux_corr = np.sqrt(ux_raw**2+ucorrection**2)
329
330     return x_corrected/pix2mm, ux_corr/pix2mm
331
332 #cost function to be minimized
333 def cost_function(coef, x_coordinate, pix2mm, R0, Ri, L, inner_tube_walls):
334
335     #quantities to be optimized
336     T0 = coef
337
338     #obtaining the corrected position
339     x_corrected = corrected_position(x_coordinate, pix2mm, R0, Ri, L, T0)
340
341     #to prevent times where the correction is correctly calculated
342     try:
343         error = np.linalg.norm((x_corrected[:]-inner_tube_walls[:]*pix2mm))
344     except:
345         print('Correction disregarded!')
346         error = 100000
347
348     return error
349
350 #%%IMAGE INPUTS AND IMPORT
351
352 #image path
353 image_path = 'trialT20_1'
354 image_name = 'test'
355 image_format = '.png'
356
357 #image numbers
358 n_start = 0
359 n_finish = 1
360
361 #rotation angle (in case image is not upright)
362 rotation_angle = np.radians(0)
363
364 #crop coordinates
365 crop_indx1 = 1915
366 crop_indx2 = 2852
367
368 crop_indy1 = 1000
369 crop_indy2 = 2000
370
371 #obtaining the image
372 image = import_images(image_path, image_name, [n_start, n_finish], image_format,
        rotation_angle, [crop_indy1, crop_indy2, crop_indx1, crop_indx2])
373
374 #%%HORIZONTAL LINE INPUTS
375
376 #LINE 1
377 #search interval for the horizontal line
378 y_upper = 640
379 y_lower = 670
380
```

```
#search for line
line_x, line_y = detect_horizontal_line(image, [y_upper, y_lower])

x1, y1 = points_extremities(line_x, line_y)

#LINE2
#search interval for the horizontal line
y_upper = 480
y_lower = 510

#search for line
line_x, line_y = detect_horizontal_line(image, [y_upper, y_lower])

x2, y2 = points_extremities(line_x, line_y)

#LINE3
#search interval for the horizontal line
y_upper = 320
y_lower = 350

#search for line
line_x, line_y = detect_horizontal_line(image, [y_upper, y_lower])

x3, y3 = points_extremities(line_x, line_y)

#%%OBTAINING REAL POSITIONS FOR THE CORRESPONDING LINE POINTS

#tube dimensions
R_o = 6
R_i = 4

#obtaining the pixel to mm ratio
pix2mm = 2*R_o/(crop_indx2-crop_indx1)

#obtain positions of inner tube walls
x_real1, x_real2 = real_positions(R_o, R_i, pix2mm)


#%%PLOT ANALYSIS AREA

plt.figure()
plt.imshow(image, cmap = 'gray')
plt.plot(x1, y1-5, color = 'red', linestyle = 'solid', linewidth = 5)
plt.plot([x_real1, x_real1], [0, crop_indy2-crop_indy1], color = 'blue', linestyle = '
    solid', linewidth = 2)
plt.plot([x_real2, x_real2], [0, crop_indy2-crop_indy1], color = 'blue', linestyle = '
    solid', linewidth = 2)
plt.ylim((y1[0]+100, y1[0]-100))
plt.title('Inner Diameter vs Distorted Inner Diameter')

#%%INVERSE METHOD TO OBTAIN L AND THETA_0

n_trials = 10

T = np.zeros(n_trials)

k = 0

while k < n_trials:

    print('Trial number: ' + str(k+1))

    #boundaries for L and Theta_0
    upper_L = 41
    lower_L = 37
    upper_T0 = 10
    lower_T0 = -10
```

```
446
447     #random starting value within the range
448     L = ((upper_L-lower_L)/100)*randrange(100) + lower_L
449     T0 = ((upper_T0-lower_T0)/100)*randrange(100) + lower_T0
450
451     x0 = (T0,)
452
453     #boundaries for optimization
454     bnds = ((lower_T0, upper_T0),)
455
456     coordinates_points = np.array([x1])
457     tube_walls = np.array([x_real1, x_real2])
458
459     solution = minimize(cost_function, x0, args = (coordinates_points, pix2mm, R_o, R_i
        , L, tube_walls,), method = 'Powell', options={'maxiter' : 100000}, bounds = bnds)
460
461     T[k] = solution.x
462
463     k += 1
464
465 T_final = np.mean(T)
466 std = np.std(T)
467
468 print('Theta will be: ' + str(T_final) + ' degrees with an uncertainty of ' + str(std)
        + ' degrees.')
469
470 #plot camera misalignment angle
471 plt.figure()
472 plt.hist(T, bins= 'auto')
473 plt.title('Histogram Camera Misalignment Angle')
474
475 #%%APLLY CORRECTION AND CALCULATE UNCERTAINTY
476
477 x_corrected, ux_corr = corrected_position_uncertainty(x1, pix2mm, R_o, R_i, 39, T_final
        )
478
479 #%%CALCULTATE ERROR IN DIAMETER
480
481 #true inner diameter
482 Di = 2*R_i
483
484 #corrected inner diameter
485 Dic = (x_corrected[1] - x_corrected[0])*pix2mm
486
487 #error for diameter
488 error_diameter = abs(((Di - Dic)/Di)*100)
489
490 print('The error for the inner diameter is: ' + str(error_diameter) + '%.')
491 #%%PLOT RESULTS
492
493 plt.figure()
494 plt.imshow(image, cmap = 'gray')
495 plt.scatter(x_corrected, y1 - 5, c='orange',edgecolor ='orange',linewidth=0.5, s =30,
        label = 'Corrected Points')
496 plt.scatter([x_real1, x_real2], y1 - 5, c='red',edgecolor ='red',linewidth=0.5, s =30,
        label = 'Inner Tube Wall')
497 plt.scatter(x1, y1 - 5, c='purple',edgecolor ='purple',linewidth=0.5, s =30, label = '
        Distorted Points')
498 plt.ylim((y1[0]+150, y1[0]-150))
499 plt.legend()
500
501 #plot uncertainty
502 k = 0
503
504 while k < len(ux_corr):
505
506     interval1 = ((x_corrected[k] - ux_corr[k]))
```

```
507
508     interval2 = ((x_corrected[k] + ux_corr[k]))
509
510     plt.plot([interval1, interval2], np.array([y1[k],y1[k]]) - 5, color = 'orange',
        linestyle = 'solid', linewidth = 1)
511
512     k += 1
```

9.2 Sessile Droplet Test Code

```
1
2  #%%NECESSARY LIBRARIES
3
4  import sys
5  sys.path.append('../libraries/')
6  import numpy as np
7  import cv2
8  from matplotlib import pyplot as plt
9  from scipy.signal import savgol_filter
10 from scipy import ndimage, misc
11 import os
12 import sys
13 import time
14 from scipy.optimize import curve_fit
15 from scipy.interpolate import interp1d
16 from scipy.integrate import solve_bvp
17 from scipy.signal import find_peaks
18 from scipy.ndimage import median_filter
19
20 plt.close('all')
21
22 #%%STARTING RUN TIME
23 start = time.time()
24
25 #%% FIT CLASSES
26
27 class fitting_model():
28     def __init__(self, model_name, algorithm, x, R):
29         self.x       = x
30         self.opt_algorithm = algorithm
31         self.model_name = model_name
32         self.lc = 0.00095 #[m]
33         self.R = R
34         if model_name == "quasi-static-full":
35             R     = self.R*0.001 #[m]
36             def func(x,yo,xo,theta):
37                 def DMS(x,h):
38                     #diferential equation
39                     return np.vstack((h[1], h[0]/(self.lc**2)*(1+h[1]**2)**(3/2)+h[1]/x
    ))
40                 def bc(ymR, ypR):
41                     return np.array([ymR[1],ypR[1]-np.tan(theta)])
42                 xx = np.linspace(1.e-8,R,10000)
43                 xx2 = np.linspace(-R,R,1000000)
44                 y_init = np.zeros((2,xx.size))
45                 res = solve_bvp(DMS, bc, xx, y_init, max_nodes=1000000)
46                 if res.success == True: self.success = res.success
47                 else: print(res.message)
48                 result = interp1d(xx2*10**3+xo,yo+res.sol(abs(xx2))[0]*10**3, kind = '
    cubic',fill_value ='extrapolate')
49                 return result(x)
50             self.BBm   = [0, self.R - 0.01, np.radians(2)]
51             self.BBM   = [0.03, self.R + 0.01, np.radians(90)]
```

```
            self.func = func

def fitting_qsf(x_index, y_index, pix2mm, x, R):

    """
    This function fits the quasi-static model to the interface.

    ---
    Parameters:
        x_index: x coordinates for the fit
        y_index: y coordinates for the fit
        pix2mm: pixel to mm ratio
        x: vector spawning the full width of the image

    """
    #conversion of coordinates from pixel to mm
    x_converted = x_index*pix2mm

    y_converted = y_index*pix2mm

    #fit using quasi-static curve
    model_qsf = fitting_model('quasi-static-full', 'trf', x_index, R)

    coef_qsf, pcov_qsf = curve_fit(model_qsf.func, x_converted, y_converted, bounds = (
    model_qsf.BBm, model_qsf.BBM))

    # y_fit_qsf = model_qsf.func(x*pix2mm, coef_qsf[0], coef_qsf[1], coef_qsf[2],
    coef_qsf[3])/pix2mm
    y_fit_qsf = model_qsf.func(x*pix2mm, *coef_qsf)/pix2mm

    theta = np.degrees(coef_qsf[2])

    y0 = coef_qsf[0]/pix2mm

    x0 = coef_qsf[1]/pix2mm

    return y_fit_qsf, theta, y0, x0

#########-----##########
def sortingvectors(x_index, y_index):

    """
    Sorts vector x_index and matches this with y_index

    ---
    Parameters:
        x_index: vector to be sorted
        y_index: vector to be matched
    """
    #sort points according to their position along the x axis

    #creating the vectors that will be sorted
    x_sorted = []
    y_sorted = []


    x_sorted, y_sorted = (list(t) for t in zip(*sorted(zip(x_index, y_index))))

    x_sorted = np.array(x_sorted)

    y_sorted = np.array(y_sorted)

    return y_sorted, x_sorted

#########-----##########
def detect_edges_upper(image_ed, threshold):

```

```
    """
    This function detects the edges on the image

    ---
    Parameters:
        image_ed: matrix containing the image whose edges are to be detected
        threshold: value for the high pass filter inferior to this are discarded
    """

    #initializing gradient matrix
    gradient_peaks = np.zeros(image_ed.shape)

    #assembling convolution kernel
    E = np.array([[1,2,1],
                      [1,2,1],
                      [1,2,1]])

    C = np.array([[0,-1,0],
                [0,0,0],
                [0,1,0]])

    kernel_MM = np.vstack((-E, C, E))

    #high pass filtering the recontrasted image
    filtered_image = cv2.filter2D(image_ed,-1, kernel_MM)

    #finding the position of the edges
    for column in range(image_ed.shape[1]):

        peaks = []

        peaks, _ = find_peaks(-1*filtered_image[:,column], height = threshold)

        if len(peaks) > 1:
            #so we always capture the top peaks
            gradient_peaks[peaks[0], column] = 1
        else:
            gradient_peaks[peaks, column] = 1

    y_peaks, x_peaks = np.where(gradient_peaks == 1)

    #median filtering the points
    y_median = median_filter(y_peaks, size = 10)

    #sorting the vectors
    y_final, x_final = sortingvectors(x_peaks, y_median)

    return y_final, x_final

#########-----##########
def detect_edges_lower(image_ed, threshold):

    """
    This function detects the edges on the image

    ---
    Parameters:
        image_ed: matrix containing the image whose edges are to be detected
        threshold: value for the high pass filter inferior to this are discarded
    """

    #initializing gradient matrix
    gradient_peaks = np.zeros(image_ed.shape)

    #assembling convolution kernel
    E = np.array([[1,2,1],
                      [1,2,1],
```

```
                        [1,2,1]])

    C = np.array([[0,-1,0],
                  [0,0,0],
                  [0,1,0]])

    kernel_MM = np.vstack((-E, C, E))

    #high pass filtering the recontrasted image
    filtered_image = cv2.filter2D(image_ed,-1, kernel_MM)

    #finding the position of the edges
    for column in range(image_ed.shape[1]):

        peaks = []

        peaks, _ = find_peaks(-1*filtered_image[:,column], height = threshold)

        if len(peaks) > 1:
            #so we always capture the top peaks
            gradient_peaks[peaks[len(peaks) - 1], column] = 1
        else:
            gradient_peaks[peaks, column] = 1

    y_peaks, x_peaks = np.where(gradient_peaks == 1)

    #median filtering the points
    y_median = median_filter(y_peaks, size = 10)

    #sorting the vectors
    y_final, x_final = sortingvectors(x_peaks, y_median)

    return y_final, x_final
#%% IMAGE IMPORT

#path name
folder = "Trial_15"

testname = "test"

img_format = ".png"

name_path = folder + os.sep + testname

#crop coordinates (lower surface)
lower_indx1 = 1262
lower_indx2 = 2512
lower_indy1 = 1991
lower_indy2 = 2014

#crop coordinates (upper surface)
upper_indx1 = 1200
upper_indx2 = 2628
upper_indy1 = 1978
upper_indy2 = 2009

#image mask (image to be shown on plots)
image_cut_left = 1100
image_cut_right = 2700
image_cut_top = 1850
image_cut_bottom = 2050

#defining rotation angle (if necessary)

angle = 0

#number of images to be imported
```

```
251
252 n_start = 480
253 n_finish = 490
254
255 pix2mm = 0.005
256 #check if the path to the image exists
257
258 if os.path.isfile(name_path + str(n_start) + img_format) is False: sys.exit('The
        selected experiment is not at the path indicated or either the path/name is wrong')
259
260 #creating 3D matrix to store the images
261
262
263 lib_selected = np.zeros((image_cut_bottom-image_cut_top, image_cut_right-image_cut_left
        , n_finish-n_start))
264 lib_lower = np.zeros((lower_indy2-lower_indy1, lower_indx2-lower_indx1, n_finish-
        n_start))
265 lib_upper = np.zeros((upper_indy2-upper_indy1, upper_indx2-upper_indx1, n_finish-
        n_start))
266
267 #importing all the images + cropping + denoising + rotation
268 i = 0
269
270 for k in range(n_start, n_finish):
271
272     image_path = name_path + str(k) #image file
273
274     image = cv2.imread(image_path + img_format,0)  # read the image
275
276     #image rotation
277     image_center = tuple(np.array(image.shape[1::-1]) / 2)
278     rot_mat = cv2.getRotationMatrix2D(image_center, angle, 1.0)
279     img_rotated = cv2.warpAffine(image, rot_mat, image.shape[1::-1], flags=cv2.
        INTER_LINEAR)
280
281     dst = cv2.medianBlur(img_rotated, 7)
282     selected_part = dst[image_cut_top:image_cut_bottom, image_cut_left:image_cut_right]
283     lower_surface = dst[lower_indy1:lower_indy2,lower_indx1:lower_indx2]
284     upper_surface = dst[upper_indy1:upper_indy2,upper_indx1:upper_indx2]
285
286     lib_selected[:, :, i] = selected_part #storing images to be displayed
287     lib_lower[:, :, i] = lower_surface
288     lib_upper[:, :, i] = upper_surface
289
290     print(str(i+1) + " image(s) processed")
291
292     i += 1
293
294
295 #averaging image to be displayed
296 display_image = np.mean(lib_selected, axis=2)
297
298 #lower surface image
299 lower_image = np.mean(lib_lower, axis=2)
300
301 #upper surface image
302 upper_image = np.mean(lib_upper, axis=2)
303 #%%DETECT EDGES LOWER SURFACE
304
305 ##defining parameters for detection
306 threshold = 50
307
308 #detecting the edges of the droplet
309 y_edge_lower, x_edge_lower = detect_edges_lower(lower_image, threshold)
310
311 #%%DETECT EDGES UPPER SURFACE
312
```

```
##defining parameters for detection
threshold = 80

#detecting the edges of the droplet
y_edge_upper, x_edge_upper = detect_edges_upper(upper_image, threshold)

#%%PLOT AREA DETECTED SURFACES

#display offset (lower edges)
offset_horizontal_lower = lower_indx1 - image_cut_left
offset_vertical_lower = lower_indy1 - image_cut_top

#display offset (upper edges)
offset_horizontal_upper = upper_indx1 - image_cut_left
offset_vertical_upper = upper_indy1 - image_cut_top

#plot
plt.figure()
#image displayed (!= image analysed, smaller area)
plt.imshow(display_image, cmap = 'gray')
#detected points
plt.scatter((x_edge_lower + offset_horizontal_lower), (y_edge_lower +
    offset_vertical_lower), c='blue',edgecolor = 'blue',linewidth=0.5, s =5)
plt.scatter((x_edge_upper + offset_horizontal_upper), (y_edge_upper +
    offset_vertical_upper), c='red',edgecolor = 'red',linewidth=0.5, s =5)
plt.title('Surfaces detected for the surface of the droplet')

#%%FIT FOR THE LOWER SURFACE

#calculate parameters
R_lower = 0.5*(lower_indx2 - lower_indx1)*pix2mm #[mm]
x_lower = np.linspace(0, lower_image.shape[1]-1, lower_image.shape[1], dtype=int)

#Static fit
y_fit_qsf_lower, theta_lower, y0, x0  = fitting_qsf(x_edge_lower, y_edge_lower, pix2mm,
     x_lower, R_lower)

print('The static contact angle for the lower surface: ' + str(theta_lower) + ' degrees
    .')

#%%PLOT FIT FOR THE LOWER SURFACE

plt.figure()
plt.imshow(display_image, cmap = 'gray')
plt.plot((x_lower + offset_horizontal_lower), (y_fit_qsf_lower + offset_vertical_lower)
    , color = 'orange', linestyle = 'solid', linewidth = 1.5)
plt.title('Fit for the lower surface')

#%%FIT FOR UPPER SURFACE

#calculate parameters
R_upper = 0.5*(upper_indx2 - upper_indx1)*pix2mm #[mm]
x_upper = np.linspace(0, upper_image.shape[1]-1, upper_image.shape[1], dtype=int)

#Static fit
y_fit_qsf_upper, theta_upper, y0, x0  = fitting_qsf(x_edge_upper, y_edge_upper, pix2mm,
     x_upper, R_upper)

print('The static contact angle for the upper surface: ' + str(theta_upper) + ' degrees
    .')

#%%PLOT FIT FOR THE UPPER SURFACE

plt.figure()
plt.imshow(display_image, cmap = 'gray')
plt.plot((x_upper + offset_horizontal_upper), (y_fit_qsf_upper + offset_vertical_upper)
    , color = 'purple', linestyle = 'solid', linewidth = 1.5)
```

```
plt.title('Fit for the upper surface')
```

9.3 Inverse Method Code

```

#%%IMPORTED LIBRARIES

import sys #for clearing variables
sys.path.append('../libraries/')
import numpy as np
import cv2 #for image processing
from matplotlib import pyplot as plt #for ploting
import os #for getting data paths
import time #for setting run time
from scipy.interpolate import interp1d #for interpolating
from scipy import integrate #for solving ode
from scipy.optimize import minimize #for optimizing
import scipy.signal as sci #for filtering pressure
from collections.abc import Iterable
from scipy import interpolate as interp

plt.close('all')

#%%DEFINE FUNCTIONS

#########-----##########
#function that finds the closest value index
def find_closest_value_index(vector, value):

    """
    Finds index of the cell with the closest value to "value".

    ---
    Parameters:
        vector: array to be searched
        value: value to look for

    """

    difference = abs(np.array(vector) - value)

    index = int(np.argmin(difference))

    return index

#########-----##########
#low pass filter
def filter_signal(signal, cutoff_frequency):

    windows = sci.firwin(numtaps = signal.shape[0]//10, cutoff = cutoff_frequency,\
                         window='hamming', fs = 500)
    # filter the signal
    example_filtered = sci.filtfilt(b = windows, a = [1], x = signal,
                                    padlen = 10, padtype = 'even')
    # get the index at which to clip again
    clip_index = signal.shape[0]
    clipped_signal = example_filtered[:clip_index]
    return clipped_signal

#########-----##########
def calculate_eq_height(t, pressure, theta_S, parameter_array):

    delta = parameter_array[1]
```

```
    rho = parameter_array[2]
    sigma = parameter_array[4]
    g = parameter_array[5]
    epsilon = parameter_array[6]

    pressure_term = pressure(t[len(t)-1]) + rho*g*epsilon

    surface_tension_term = (2/delta)*sigma*np.cos(theta_S)

    h_eq = (1/(rho*g))*(pressure_term+surface_tension_term) #[m]

    return h_eq*1000 #[mm]

#########-----##########
#model equation with coefficients ready to be optimized
def model_equation(y, t, coeff_dca, pressure, parameter_array,):

    #break down parameter array
    k = parameter_array[0]
    delta = parameter_array[1]
    rho = parameter_array[2]
    mu = parameter_array[3]
    sigma = parameter_array[4]
    g = parameter_array[5]
    epsilon = parameter_array[6]
    K_h = parameter_array[7]
    K_c = parameter_array[8]

    #break down coefficients
    alpha = coeff_dca[0]
    a = coeff_dca[1]
    b = coeff_dca[2]

    #define a static contact angle
    theta_s = np.radians(31.75)

    #break down y vector
    h = y[0] #interface position
    v = y[1] #interface velocity
    theta = y[2] #custom dynamic contact angle (theta = theta_d**3-theta_s**3)

    #BREAKING DOWN THE SYSTEM OF EQUATIONS INTO TERMS

    #FIRST TERM TO INTEGRATE (velocity)
    dh_dt = v

    #SECOND TERM TO INTEGRATE (acceleration)
    inertial_term = (h + k*(delta/2))

    pressure_term = (pressure(t) + rho*g*epsilon)/rho

    #extracting the dynamic contact angle
    theta_d = np.clip(theta+theta_s**3, 1e-10, (2*np.pi)**3)**(1/3)

    surface_tension_term = (2*sigma*np.cos(theta_d))/(rho*delta)

    weight_term = h*g

    viscous_term = (12*mu*h*v)/(rho*delta**2)

    dissipative_term = 0.5*(K_h*v*abs(v) + (mu*K_c*v)/(delta*rho))

    dv_dt = (1/inertial_term) * (pressure_term + surface_tension_term - weight_term -
    viscous_term - dissipative_term)

    #THIRD TERM TO INTEGRATE (rate of change of contact angle)

```

```
126     dtheta_dt = (1/alpha)*(-theta+a*(v*Ca_factor)+b*dv_dt/g)
127 
128     return (dh_dt, dv_dt, dtheta_dt)
129 
130 #########-----##########
131 #function that assembles both the model and the correlation for the dynamic contact
        angle
132 def solve_model(t, pressure, parameter_array, coeff, y0):
133 
134     #solving model equation
135     y_solution = integrate.odeint(model_equation, y0, t, args=(coeff, pressure,
        parameter_array,), mxstep=500000)
136 
137     #getting h
138     h_model = y_solution[:, 0]*1000 #[mm]
139 
140     #getting dynamic contact angle
141     theta = y_solution[:, 2]
142     theta_s = np.radians(31.75)
143     theta_d = np.degrees(np.clip(theta+theta_s**3, 1e-10, (2*np.pi)**3)**(1/3)) # [
        degrees]
144 
145     return h_model, theta_d
146 
147 #########-----##########
148 #cost function for optimization of the coefficients
149 def cost_function(coeff, h_exp, t, pressure, parameter_array, y0):
150 
151     #solving model
152     h_ode, theta_d = solve_model(t, pressure, parameter_array, coeff, y0)
153 
154     #calculate error
155     ERROR = np.linalg.norm((h_ode[:] - h_exp[:]))
156 
157     print('Difference between experimental and model data: ' + str(ERROR))
158 
159     return ERROR
160 
161 
162 #%%IMPORT DATA AND TREAT DATA
163 
164 #path to files
165 path = 'data.npz'
166 
167 #loading npz file
168 data = np.load(path)
169 
170 #assigning variables
171 t = data['time']
172 h_average = data['height']
173 pressure_temp = data['pressure']
174 
175 #making a time dependant pressure
176 time_pressure = np.linspace(0, t[len(t)-1], len(pressure_temp))
177 pressure = interp1d(time_pressure, pressure_temp, axis=0, fill_value="extrapolate")
178 
179 #%%DEFINING IMPORTANT PARAMETERS
180 
181 #inertia factor
182 k = 73/60
183 
184 #Hagenbach coefficient
185 K_h = 7/3
186 
187 #Couette coefficient
188 K_c = 4
189 
```

```
#channel width
delta = 0.005 #[m]

#HFE density
rho = 1431.6644463636364 #[kg/m3]

#HFE friction coefficient
mu = 0.0006889890816381588 #[m2/s]

#HFE surface tension
sigma = 0.013974441375291385 #[N/m]

#gravitational acceleration
g = 9.81 #[m/s2]

#hydrostatic pressure offset
epsilon = 0.006 #[m]

#factor to turn velocity into capillary number
Ca_factor = mu/sigma


#defining parameter array
parameter_array = [k, delta, rho, mu, sigma, g, epsilon, Ca_factor, K_h, K_c]

#%%CORRECTING EXPERIMENTAL HEIGHT

pix2mm = 0.007299

aux = np.linspace(0, len(h_average) - 1, len(h_average))
index = find_closest_value_index(aux, 0.9*len(h_average))

h_eq = calculate_eq_height(t, pressure, np.radians(31.75), parameter_array)

h_exp = -1*(h_average - np.mean(h_average[index:len(h_average)-1]))*pix2mm + h_eq

#%%DERIVING OTHER VARIABLES

velocity_temp = np.gradient(h_exp, t)*0.001
acceleration_temp = np.gradient(velocity_temp, t)

#adapting making variables time dependant
velocity = interp.interp1d(t, velocity_temp, axis=0, fill_value="extrapolate")
acceleration = interp.interp1d(t, acceleration_temp, axis=0, fill_value="extrapolate")

#%%SOLVE EQUATION WITHOUT OPTIMIZING THE COEFFICIENTS

print('Solving equation for predetermined coefficients')

#coefficients for the correlation
coeff = np.array([0.149695, 26.4747, 3.92117])

#defining initial conditions
y0 = [h_exp[0]*0.001, velocity(t[0]), (np.radians(52)**3-np.radians(31.75)**3)]

h_model, theta_d = solve_model(t, pressure, parameter_array, coeff, y0) #[mm]  and [
    degrees]

#%%DISPLAY RESULTS

plt.figure()
plt.plot(t, h_exp, color = 'purple', linestyle = 'solid', linewidth = 2, label = '
    Experimental Data')
plt.plot(t, h_model, color = 'orange', linestyle = 'solid', linewidth = 2, label = '
    Model')
plt.legend()
plt.title('Experimental Data vs Model (non-optimized coefficients)')
```

```
254
255 plt.figure()
256 plt.plot(t, theta_d, color = 'purple', linestyle = 'solid', linewidth = 2)
257 plt.title('Solution for theta (non-optimized coefficients)')
258
259 print('Finished solving equation for predetermined coefficients')
260
261 #%%SOLVE EQUATION WITH COEFFICIENT OPTIMIZATION
262
263 print('Optimizing coefficients for the model')
264
265 #boundaries for the coefficients in the optimization procedure
266 b0 = (1e-02, 1)
267 b1 = (0, 30)
268 b2 = (0, 5)
269
270 bnds = (b0, b1, b2)
271
272 #starting values for the optimization (values from theta optimization)
273 x0 = [0.149695, 26.4747, 3.92117]
274
275 #run the optimization
276 solution = minimize(cost_function, x0, args = (h_exp, t, pressure, parameter_array, y0
        ,), method = 'Powell', options={'disp': True, 'ftol':0.0000001, 'maxiter':10},
        bounds = bnds)
277
278 #storing the coefficients
279 alpha = solution.x[0]
280 a = solution.x[1]
281 b = solution.x[2]
282 optimized_coeff = [alpha, a, b]
283
284 #getting the model data
285 h_optimized, theta_d_optimized = solve_model(t, pressure, parameter_array,
        optimized_coeff, y0)
286
287 #%%DISPLAY RESULTS
288
289 plt.figure()
290 plt.plot(t, h_exp, color = 'yellow', linestyle = 'solid', linewidth = 2, label = '
        Experimental Data')
291 plt.plot(t, h_optimized, color = 'blue', linestyle = 'solid', linewidth = 2, label = '
        Model')
292 plt.legend()
293 plt.title('Experimental Data vs Model')
294
295 plt.figure()
296 plt.plot(t, theta_d_optimized, color = 'purple', linestyle = 'solid', linewidth = 2)
297 plt.title('Solution for theta')
298
299 print('Finished optimizing coefficients for the model')
```