ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

INVESTIGATION OF MICROSTRUCTURE MOVEMENT UNDER FLOW BY USING IMAGE PROCESSING AND DEEP LEARNING

M.Sc. THESIS

Saeed SARBAZZADEH KHOSROSHAHI

Department of Electronics and Communication Engineering

Electronics Engineering Programme

JULY 2023

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

INVESTIGATION OF MICROSTRUCTURE MOVEMENT UNDER FLOW BY USING IMAGE PROCESSING AND DEEP LEARNING

M.Sc. THESIS

Saeed SARBAZZADEH KHOSROSHAHI
(504191224)

Department of Electronics and Communication Engineering

Electronics Engineering Programme

Thesis Advisor: Asst. Prof. Ahmet Can ERTEN

JULY 2023

ISTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

AKIŞ ALTINDAKİ MİKROYAPI DEFORMASYONUNUN GÖRÜNTÜ İŞLEME VE DERİN ÖĞRENME KULLANILARAK İNCELENMESİ

YÜKSEK LİSANS TEZİ

Saeed SARBAZZADEH KHOSROSHAHI
(504191224)

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

Tez Danışmanı: Dr. Öğr. Üyesi Ahmet Can ERTEN

TEMMUZ 2023

Saeed SARBAZZADEH KHOSROSHAHI, a M.Sc. student of ITU **Graduate School** student ID 504191224, successfully defended the **thesis** entitled "**INVESTIGATION OF MICROSTRUCTURE MOVEMENT UNDER FLOW BY USING IMAGE PROCESSING AND DEEP LEARNING**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Asst. Prof. Ahmet Can ERTEN** Istanbul Technical University	.....
Jury Members :	**Asst. Prof. Tankut Akgül** Istanbul Technical University	.....
	Prof. Dr. Alper Kiraz Koç University	.....

Date of Submission : 25 JUNE 2023
Date of Defense : 6 JULY 2023

spouse and family, and all my good friends

FOREWORD

The main aim of this thesis is to enable the viscosity measurement using the micropillars method and enhance it using image processing and machine learning algorithms to speed up and simplify the measurement process for industrial and healthcare applications. This thesis is done upon completion of master's degree at the Department of Electronics Engineering of the Istanbul Technical University (ITU).

I would like to thank Dr. Tankut Akgul, for his great help on the image processing and machine learning topics and my adviser, Dr. Ahmet Can Erten, for his guidance and for giving me a chance to work in his lab. Also, I want to thank my friend and co-worker Reza Sadeghi for his efforts and his help on this project. This thesis was supported in part by the Istanbul Technical University Research Fund BAP under Project #42756.

MAY 2023 Saeed SARBAZZADEH KHOSROSHAHI

TABLE OF CONTENTS

Page

ABBREVIATIONS

KLT **:** Lucas-Kanade

CNN **:** Convolutional Neural Network

PDMS **:** Polydimethylsiloxane

USB3.0 **:** Universal Serial Bus3.0

CCD **:** Charge Coupled Device

CPU **:** Central Processing Unit

POC **:** Point of Care

SYMBOLS

cP	**:** Centipoise Unit
ml/h	**:** Volumetric flow Unit
S^{-1}	**:** Shear Rate Unit
γ	**:** Shear Rate
ml	**:** Volume Unit
H	**:** Height
D	**:** Dimension
$\vec{v}$	**:** Optical Flow Vector
u, v	**:** Optical Flow Vector's x and y Elements at a Location
$\vec{d}$	**:** Displacement Vector
d_x, d_y	**:** Represent the x and y Components of the Displacement Vector
I	**:** Brightness of the Image at Point (x, y) in the Picture Plane at Time t
$\delta_x, \delta_y, \delta_t$	**:** Stand for the Interframe Movement and Time Interval
R	**:** Harris Corner Detector's Scoring
λ_1, λ_2	**:** Eigenvalues of Intensity Change Error Function
fx, fy, ft	**:** Functions of KLT Patch Locations
$\mathbf{f_1}, \mathbf{f_2}$	**: C**hannel Feature Maps
w	**:** Width of Feature Maps
h	**:** Height Feature Maps
c	**:** Number of Channel of Feature Maps
K	**:** Patch of Squares
x1, x2	**:** Centers of Correlation of Feature Maps
s1, s2	**:** Strides of Correlation
x, y	**:** Values of Location in Circle Equation
a, b	**:** Location Parameters of Circle Center
r	**:** Radius

LIST OF TABLES

Page

LIST OF FIGURES

Page

INVESTIGATION OF MICROSTRUCTURE MOVEMENT UNDER FLOW BY USING IMAGE PROCESSING AND DEEP LEARNING

SUMMARY

In many industrial and biological applications, the viscosity of chemical and biological fluids is a crucial material property that needs to be precisely measured A variety of techniques has been developed to measure viscosity.

The micropillar-based microfluidic viscometer method uses viscosity and flow-induced micropillar displacement. Before making general viscosity measurements, calibration curves (viscosity vs. micropillar tip displacement) are created using solutions with known viscosities. Filming experiments with glycerol/water solutions with viscosities ranging from 2 to 100 cP at fixed flow (shear) rates are done to achieve this. The experiment is then repeated using a fluid sample whose viscosity is determined. In captured experiment videos with the sample fluid, the displacement of pillars is measured for this purpose using ImageJ, an image processing program. The measured displacements are then mapped to the calibration curves to determine the sample fluid's viscosity. Results obtained using this method are precise. The disadvantage is that using ImageJ to calculate displacement takes time and requires manual work. Therefore, in this study, we used two distinct image processing algorithms that yield results much more quickly. These are Lucas-Kanade (KLT), and Hough Circle, which are used in classical video processing, and the FlowNet2 neural network model.

The KLT algorithm is a popular differential technique for estimating optical flow. We tracked the four corners of the pillar tip using the KLT to determine the displacement of the pillars. The final displacement data was then determined by averaging these four corner displacements. Contrarily, the convolutional neural network (CNN) Flownet2 is employed in deep learning to interpret visual images. Huge displacement control and precise estimation of minute details in the optical flow field are two features of FlowNet2. We made use of the pre-trained FlowNet2 model on the THINGS dataset. We used the first frame of the video as the model's first entry and the frames that came after that as its second entry to find out about displacement.

We used ImageJ data as a reference to determine the methods' accuracy when determining the accuracy of the suggested methods for 10 videos. Regarding ImageJ, KLT, Hough Circle and FlowNet2 provided an average accuracy of 95.45%, 91.47%, and 95.62%, respectively. We saved a lot of time by using these techniques because we didn't need human assistance. With KLT, we were able to generate viscosity results 158 times faster with respect to ImageJ, with Hough Circle 396 times faster and with FlowNet2 10 times faster.

AKIŞ ALTINDAKİ MİKROYAPI DEFORMASYONUNUN GÖRÜNTÜ İŞLEME VE DERİN ÖĞRENME KULLANILARAK İNCELENMESİ

ÖZET

Kimyasal ve biyolojik sıvıların viskozitesi önemli bir malzeme özelliğidir. Polimerlerin, yağların, boyaların, gıdaların, ilaçların ve diğer malzemelerin viskozitesi, birçok endüstriyel işlemde üretim verimliliğini arttırmak için doğru bir şekilde ölçülmelidir. Bu nedenle, düşük numune hacimleri kullanarak viskoziteyi ölçmek için kolay, hızlı ve düşük maliyetli yöntemlerin geliştirilmesi, endüstriyel ve tıbbi alanda büyük önem taşımaktadır. Geleneksel viskozimetreler normalde ikiye ayrılır: yani, döner ve kılcal viskozimetreler, endüstriyel ve biyolojik uygulamalarda yaygın olarak kullanılmaktadır. Ancak, hantal ve pahalı enstrümantasyon ve operasyonel karmaşıklık genellikle rutin kullanımlarını sınırlar ve daha etkili alternatiflerin geliştirilmesini talep eder.

Viskoziteyi ölçmek amacıyla farklı yöntemler geliştirilmiştir. Bir yeni yöntemse, mikroakışkan çiplere yerleştirilen mikrosütunların yer değiştirmesi yoluyla viskoziteyi hesaplama yöntemidir. Bu yöntemde, viskozite ölçümleri yapmadan önce bilinen viskozitelere sahip solüsyonları kullanarak kalibrasyon eğrileri (viskoziteye karşı mikrosütun uç yer değiştirmesi) oluşturmuşlardır. Bunu, sabit akış (kesme) hızlarında 2 ila 100 cP arasında değişen viskozitelere sahip gliserol / su çözeltileri ile deneyler yaparak ve sıvının geçmesi sırasında mikrosütun uçlarının yer değiştirmesini filme alarak yapmışlardır. Çalışmalar, 15 ila 60 ml/saat arasında değişen dört farklı akış hızında gerçekleştirilmiştir. Ardından, viskozitesi bilinmeyen sıvının viskozitesini ölçmek için aynı deney düzeneği kullanılmıştır. Sıvının viskozitesi, videoda kayıt altına alınan mikrosütun uçlarının yer değiştirme miktarları ve daha önce çıkartılan kalibrasyon eğrileri vasıtasıyla tespit edilmiştir.

Deney düzeneğinde bir adet ters mikroskop (Nikon Eclipse TS100, Tokyo, Japonya), bir şırınga pompası (PHD Ultra 70-3007, Harvard Apparatus, Holliston, MA), bir CCD kamera (Point Gray Grasshopper3 USB3.0, Richmond, BC, Kanada) ve deneyi filme almak için bir bilgisayar kullanılmaktadır. Şırınga pompası kullanılarak sıvı numuneleri şırıngaya yerleştirilir ve mikroakışkan çipe enjekte edilir. Ters çevrilmiş mikroskop, 10X objektif merceğe ve görüntüleme bağlantı noktasına yerleştirilmiş 0,7 oranında küçültmeli merceğe sahiptir. Bu sayede videoda mikrosütunların 7X oranında büyütülmesi sağlanır. Mikroakışkan bir viskozimetre çipi oluşturmak için bir alüminyum (Al) kalıp kullanılmıştır. Mikrosütun (yükseklik H = 1500 μm, çap D = 300 μm) Al kalıbını işlemek için ise 1500 μm flüt uzunluğuna sahip 300 μm çapında bir matkap ucu (Corrodrill) kullanılmıştır.

CCD kamera mikrosütun uçlarının hareketini saniyede 22 adet yüksek çözünürlükte kare yakalamak suretiyle kayıt altına alır. Kameranın piksel boyutu 5,68 μm'dir. Kamera piksel boyutunu büyültme oranına (7X) böldüğümüzde videolarımızda 0,81

µm/piksel çözünürlük elde ettiğimiz görülmektedir. Mikroakışkan cihazdaki mikro sütunların fotoğrafları, 15 ila 105 ml/saat arasında değişen çeşitli akış hızlarında ve ayrıca statik durumda (akış yokken) yakalanır.

Bu yöntemde mikrosütun uçlarının yer değiştirme miktarı ImageJ isimli bir resim işleme yazılımı kullanarak insan yardımıyla hesaplanmaktadır. Bu sebepten, önerilen yöntem uzun vakit almakta ve hataya açık hale gelmektedir. Bu nedenle, bu tezde mikrosütun uçlarının yer değiştirmesini daha hızlı ve daha yüksek doğruluk payında hesaplamayı sağlayan üç farklı görüntü işleme stratejisi sunacağız. Önerdiğimiz yöntemlerden birincisi klasik video işlemede kullanılan Kanade-Lucas-Tomasi (KLT) algoritması, ikincisi ise derin öğrenme temelli FlowNet2 evrişimsel sinir ağı modelidir. Son yöntem ise Hough Circle görüntü işleme yöntemidir.

Bilgisayarlı görüde KLT yöntemi, optik akış tahmini için yaygın olarak kullanılan bir diferansiyel yöntemdir. Akışın, incelenen pikselin yerel bir komşuluğunda sabit olduğunu varsayar ve bu komşuluktaki tüm pikseller için temel optik akış denklemlerini en küçük kareler yöntemiyle çözer. Mikrosütunları takip etmek için önce üzerlerinde takip edilecek uygun özellikli noktalar tespit edilmelidir. Bu amaç doğrultusunda biz ilk önce Shi-Tomasi metodu ile mikrosütunlar üzerinde uygun 4 adet köşe noktası buluyoruz. Bu yöntem, yararlı olabilecek en uygun N köşeyi bulur ve istenmeyen köşeleri siler. Daha sonra KLT sayesinde tespit edilen köşe noktalarına KLT optik akış algoritmasını uyguluyoruz ve video da köşelerin hareketini takip ediyoruz. Gerçek zamanlı hareket algılamaya ulaşmak için KLT pencere boyutunu 500x200 piksel olarak seçtik. Mikrosütun üzerinde seçilen 4 adet nokta ve bu noktaların KLT algoritması kullanılarak hesaplanan optik akışı çıkartılır.

İkinci yöntemde, bir makine öğrenme algoritmasını kullanıyoruz, Bir evrişimsel sinir ağı (CNN), görsel veriyi derin öğrenmede yorumlamak için kullanılan bir tür yapay sinir ağıdır. CNN, görsel veride cisimlerin boyut veya konumdan bağımsız olarak tespit edilebilmesini sağlar. Optik akış tahmini alanında Dosovitskiy'nin CNN tabanlı FlowNet modeli oldukça başarılı sonuçlar vermiştir. Bu model basit bir CNN mimarisi kullanarak verilerden doğrudan optik akışı öğrenme fikrine dayanır. Daha sonra geliştirilen FlowNet2 modeli ise, yüksek miktarda optik akış olan veride bile son derece ince ayrıntıların doğru tahmini ve hızlı çalışma süreleri gibi yenilikler getirmiştir.

Mikrosütun uçlarının hareketinin tespitinde ikinci yöntem olarak FlowNet2 modelini kullanıyoruz. Bunun için ilk önce sütun arka planını resimde maskeliyoruz. Bu adım, FlowNet2'nin hassasiyetini artırmayı sağlamaktadır. Burada, THINGS veri seti tarafından önceden eğitilmiş FlowNet2 modelini kullanıyoruz. Daha sonra videonun ilk karesini modelin ilk girişi olarak, gelecek kareleri ise ikinci giriş olarak alıyoruz. Model, ilk kareye ilişkin videonun her karesi için akış dosyalarını oluşturuyor. Akış dosyaları bize sütun yer değiştirme verilerini veriyor.

Hough Circle yöntemini kullanarak deney videosunun her karesinde sütuna karşılık gelen çemberin merkezini buluyoruz. Ardından, çerçevelerin merkez konumlarını karşılaştırarak yer değiştirme sonuçlarını elde ederiz.

Son bölümde, üç yaklaşımın sonuçlarının ImageJ ile karşılaştırılması anlatılmıştır. Her deneme videosunda KLT yönteminin doğruluğunu hesaplamak için ImageJ verilerini

referans olarak kullandık ve her karede bulunan yer değiştirme miktarı için ImageJ doğruluğunun 1 olduğunu varsaydık. Ardından, her kare için KLT'nin yer değiştirme sonucunu ilgili ImageJ karesiyle karşılaştırdık. KLT sonuçlarının doğruluğunu daha sonra ImageJ ile ilişkili olarak hesapladık. Her karede bulunan mikrosütun uçlarının yer değiştirmesinin doğruluğunu hesapladıktan sonra, KLT yönteminin genel doğruluğunu elde etmek için sonuçların ortalamasını aldık. Aynı işlemleri FlowNet2 ve Hough Circle modelernin doğruluğunu belirlemek için de kullandık.

Sonuç olarak KLT, FlowNet2 ve Hough Circle yöntemlerin, ImageJ yöntemi ile ilgili olarak sırasıyla %95.45, %95.62 ve %91.47 ortalama doğruluk sağladı. İnsan yardımına ihtiyacımız olmadığı için bu teknikleri kullanarak çok zaman kazandık. KLT ile ImageJ'e göre 158 kat, Hough Circle ile 396 kat ve FlowNet2 ile 10 kat daha hızlı viskozite sonuçları üretebildik. Ayrıca, KLT yöntemi FlowNet2'den daha az hesaplama gücü gerektirir. Bu sebeple KLT gerçek zamanda çalışabilmektedir. Daha önce belirtildiği gibi ImageJ'deki görüntü işleme için insan yardımına ihtiyaç vardır, ancak önerdiğimiz yaklaşımlarda tüm işlemler bilgisayar programları tarafından otomatik olarak yapılabilmektedir. Bu da ImageJ'e göre büyük bir avantaj sağlamaktadır.

1. INTRODUCTION

Chemical and biological fluid viscosity is a crucial material characteristic. In many industrial processes, the viscosity of materials such as polymers, oils, paints, food, pharmaceuticals, and other substances must be precisely measured. to increase the efficiency and cost-effectiveness of manufacturing fermentation products during synthesis. Therefore, the creation of straightforward, rapid, and economical technologies to test viscosity with small sample volumes is essential in a variety of industrial and medical applications. Rotational [1] and capillary [2] viscometers are the two most common types of viscometers, and both are extensively used in industrial and biological applications. But due to the expensive and bulky equipment required for frequent use, as well as the complexity of the operations, more efficient alternatives had to be developed. Viscosity can be quickly and affordably measured in small volumes of material using microfluidic devices. Microfabrication and microfluidics have recently been used to make several tiny viscometers for this purpose.

Various approaches have been used to quantify viscosity. A novel approach to determining viscosity using the displacement of pillars that are mounted on microchips has been put forth in the paper [3]. Prior to performing viscosity measurements, the authors' novel method uses solutions with known viscosities to create calibration curves (viscosity vs. micropillar tip displacement). To achieve this, they conduct experiments with glycerol/water solutions with viscosities ranging from 2 to 100 cP at constant flow (shear) rates and document these experiments. There were four different flow rates used for the studies, ranging from 15 to 60 ml/h. Using this experiment and these standard curves, the viscosity of a fluid whose viscosity is unknown can then be found.

As a result, we can apply computer vision and machine learning techniques in a variety of other fields as a result of the advancements made in these fields. Object tracking is a sensitive topic because of the enormous variety of potential uses for computer vision, including video coding, medical diagnostics, industrial applications, and robotics. Despite the fact that there are numerous methods for tracking an object, including the

machine learning algorithms FlowNet [4], FlowNet2 [5], and Raft [6], as well as more traditional image processing algorithms like Lucas-Kanade [7] and Horn-Schunk [8], for the majority of industrial applications, we can optimize time and energy by using these algorithms. In this thesis, we attempted to use these algorithms to shorten the processing time for calculating the viscosity of liquids and to facilitate and improve the evaluation process.

1.1 Purpose of Thesis

In the viscosity measurement method using micropillars, they calculated the displacement of pillar tips with ImageJ software. ImageJ takes too much time and needs human assistance. Therefore, we will provide three unique image processing strategies that are faster. Here, we will use the Lucas-Kanade (KLT) algorithm, which is used in classical video processing, the Hough Circle image recognition method, and CNN, which is the FlowNet2 model. The main purpose of the thesis is to reach automatic measurement of viscosity without the need for human intervention. In addition to automating the measurement, these algorithms reduce the time required to set up and increase the measurement accuracy to an acceptable level.

1.2 Basics and Literature Review of Viscosity Measurement

Rotational and capillary viscometers are the two main types of conventional viscometers, and they are both widely used for industrial and biological purposes. Moreover, their routine usage is frequently restricted by bulky, costly instrumentation and technical complexity [9], which calls for the creation of better substitutes. A natural framework for quick, inexpensive viscosity tests utilizing tiny quantities of material is provided by microfluidic devices. To achieve this goal, various miniature viscometers have already been created by scientists employing microfabrication and microfluidics technologies [9]. Flow rate-sensing viscometers [10, 11], surface tension viscometers [12, 13], microfluidic comparator viscometers [13, 14, 15], and droplet-based viscometers [16, 17] are some of the several types of microfluidic viscometers that may be classified based on their principle of measurement. Pressure-sensing viscometers function by detecting the pressure drop across a microchannel at predetermined flow rates, frequently using two pressure sensors that are physically apart. The Hagen-Poiseuille equation may be used to determine the viscosities of both

Newtonian and non-Newtonian fluids when the size of the microchannel and the length between the pressure sensors are known. Commercially accessible viscometers based on this idea can measure a wide range of viscosities (0.2–100,000 cP) at high shear rates (up to 1.4 million s^{-1}) with little volume (20 ml). The sample volume needed to achieve correct pressure drops and shear rates for low viscosity (~1 cP) samples might be as high as 15 ml, despite the fact that pressure-sensing viscometers are ideal for point-of-care rheology operations [18]. Viscometers that measure flow rate over a microchannel when the pressure drops are known as flow rate sensing viscometers. Such a viscometer, for instance, was created by Hudson et al. [19], who used it to measure the viscosities of antibody solutions (0.7-10 cP) at a variety of shear speeds (10-10000 s^{-1}). Although flow rate monitoring viscometers reduce dead volume and data acquisition time, it might be difficult to collect high viscosity data at high shear rates. In order to passively pull fluids into microchannels, surface tension viscometers rely on capillary pressure and the moisture characteristics of surfaces. To calculate the viscosity and shear rate of an imbibing fluid inside a capillary, the advancing meniscus is time-tracked [12, 13]. The wettability of the sample fluid is determined by the surface qualities of the capillary, which can be a limiting issue even though the process is straightforward and does not include moving elements [20]. A sample fluid with an undetermined viscosity is co-flowing next to a benchmark fluid with a known viscosity within a microchannel in microfluidic comparator viscometers [11]. By measuring the position of the fluid-fluid contact over the channel width (perpendicular to the flow direction), the viscosity of the sample fluid is identified [15]. For example, Kang and colleagues [21] measured the viscosity of blood throughout a shear rate range of 10-1000 s^{-1} using a microfluidic comparator setup. Despite having a distinct advantage for detecting low-viscosity solutions, microfluidic comparator (co-flow) viscometers are prone to fluid boundary blurring (in the event of a miscible sample and reference fluid) or hydrodynamic instability and droplet formation (in the case of an immiscible sample and reference fluid). The creation of Nl-Pl-sized droplets is the basis of droplet microfluidic viscometers, which use a variety of transduction processes to measure viscosity. Li et al. [16] and Delamarre et al. [22] created microfluidic viscometers based on measuring the droplet length and droplet spacing, correspondingly, by producing aqueous droplets from a sample fluid into an oil phase. Parallel to this, Livak-Dahl, et al. [17] created a viscometer based on seeing an aqueous plug pass

through a microfluidic constriction and used the Hagen-Poiseuille equation to determine the viscosity. Finally, Schultz et al. [23] used particle-tracking microrheology to quantify the viscosity of all the 5 ml droplets simultaneously after employing droplet-based microfluidics to manufacture the various viscosities. Low sample usage is possible with droplet-based viscometers, and sample management techniques including separation, mixing, and dilution are also possible. Elastomeric cantilevers may now be used as force sensors to detect cell traction forces, cardiomyocyte contractility, and clot contractility thanks to developments in microfabrication and soft lithography [24]. In their latest work, Brücker et al. hypothesized that linear velocity profiles might be utilized to describe micropillar displacement in fluid flow and indicated that flexible PDMS micropillars may be employed as viscosity sensors [25]. To our understanding, only one micropillar-based microfluidic viscometer has been published in the literature by Judith et al. [26]. The viscosity of a fluid was measured using a micropillar array that was magnetically activated. With a sample volume of 20 ml, they obtained a viscosity measurement range of 5–5000 cP with a sensitivity of 1 cP. Although this technique offers a large dynamic range, making the devices is a time- and money-consuming operation. The micropillar viscosity measurement method proposes a unique technique for determining the viscosity of Newtonian and non-Newtonian liquids by quantifying the flow-induced deflection of elastic PDMS micropillars. A micromachined aluminum mold is used to create the micropillars by soft lithography. Micropillar deflection caused by fluid flow is measured and studied. Fluids of varied viscosities were injected into a microchannel holding a bendable micropillar array at varying flow rates. In particular, for metering viscosity, we calculate the net micropillar tip displacement along the direction of flow by measuring the location of the micropillar tip in the absence (static, no flow) and presence of fluid flow. Viscosity directly relates to micropillar tip displacement at a consistent flow rate. With a precision equivalent to 0.5 cP, viscosity evaluations have been made over a broad range of viscosity values between 2-100 cP. The tests used liquid sample volumes between 80-400 µl. and flow rates within 15-105 ml/h (shear rates γ = 60.5-398.4 s^{-1}) and 30 and 90 ml/h (shear rates γ = 120-345.1 s^{-1}), correspondingly, for liquids and non-Newtonian liquids (blood). It is possible to calculate viscosity via micropillars, by observing how much these microstructures deform when fluid samples of various viscosities flow through the microfluidic chip. The displacement of the micropillars along the flow direction

caused by interactions between the fluid flowing through the microfluidic device and the micropillars is generally equivalent to the fluid's viscosity. They first calculated the average micropillar deflection at a certain flow rate by using common glycerol/water solutions with different viscosities. So, the micropillars were continually recorded while the experimental fluid was delivered to them and interacted with the pillars. By employing data visualization tools, the movement of the micropillar tip was later calculated from these frames. All experiments were measured using a video at a frame rate of 22. Every video's first 25 frames were captured without any channel fluid flow. Then, a syringe pump was used to start the stream. The camera's pixels had a 5.68 mm dimension. They arrive at a quality of 0.81 mm/pixel by dividing the camera pixel size by the zoom (7x). The Nyquist theorem's prediction that the lowest length that can be distinguished is practically 2 pixels, or 1.56 mm, was utilized to evaluate the accuracy of viscosity measurements. A calibration graph mapping the viscosity values to the movement of the micropillar head is created using the deflection measurements at different flow (shear) velocities. Following that, the viscosity of unidentified liquids at a particular flow (shear) velocity is calculated using these calibration curves (Figure 1.1). The viscosity of glycerol/water samples varying from 2-100 cP at 25° $\boldsymbol{C}$ (28.3-86.75% (glycerol/water)) was measured in order to quantify the sensitivity and dynamic range of the microfluidic rheometer. In this method, a modified Matlab application was used to analyze the images captured in the recorded films. Two different recording scenarios were used for the videos: static (no flow in the channel) and dynamic (fluid flow in the channel). Each video's first 25 frames were captured with no fluid flowing through the channel. After that, a syringe pump was used to initiate the flow. Before the beginning of the flow, the micropillar's deflection is measured by imaging and monitoring its tip. The position of the micropillar tip in the initial static frame and each of the succeeding frames are compared to determine the micropillar tip movement. ImageJ software is used to

validate the Matlab displacement result. In this thesis, we will present two new, faster and more precise methods for measuring displacement.

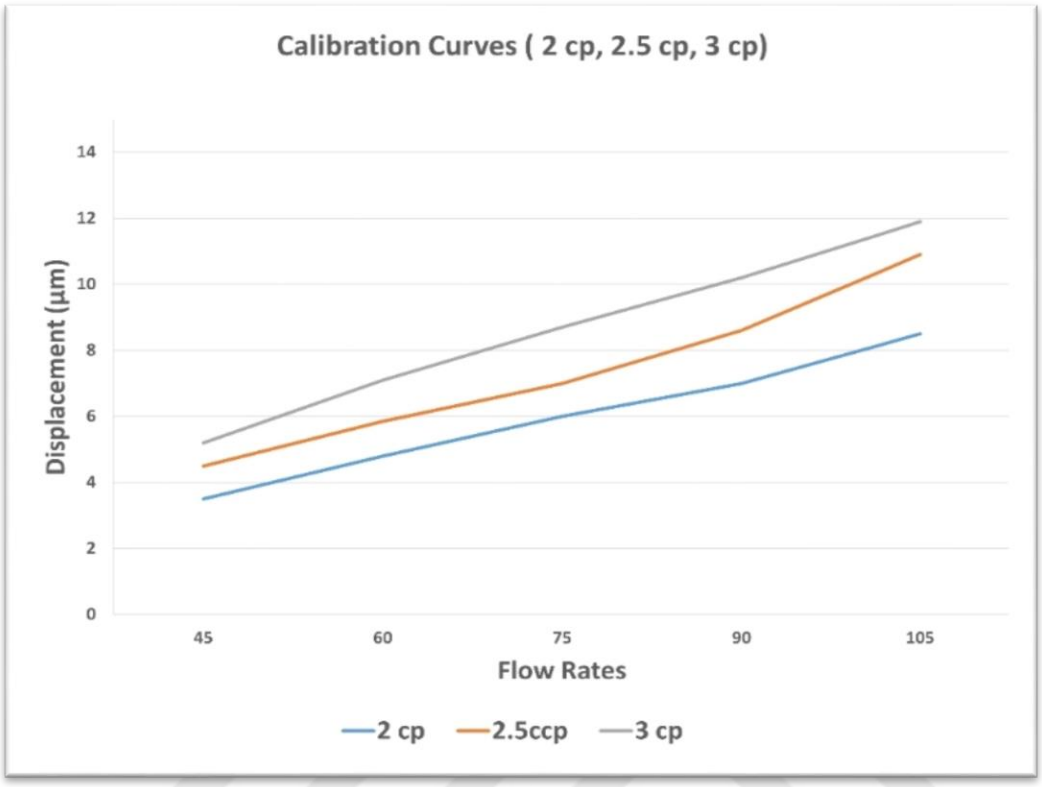

Figure 1.1: Calibration curves to measure the viscosity of an unknown [3].

1.3 Organization of This Study

The structure of this essay is as follows. In Chapter 2, we walk through the experimental setup for filming videos. Then, we talk about the experimental outcome of three proposed methods in Chapter 3. Chapter 4, concludes the study and offers suggestions for future research possibilities also, we talk about our side projects about measuring viscosity.

2. EXPERIMENTAL SETUP

This chapter discusses all the equipment needed to record the experiment video and viscosity measurement steps for this thesis.

2.1 Device Fabrication

A microfluidic viscometer chip was produced by implementing an aluminum (Al) mold, as shown in Figure 2.1. An aluminum mold was micromachined with microwells using a 300 μm diameter drilling tool (Corrodrill) with a 1500 μm flute length. The mold was cleansed and burnished after the micromachining procedure to clear it of any impurities. A 10:1 (base: crosslinker) combination of PDMS (Sylgard 184, Dow Corning, US) was created [27]. Prior to molding the combination onto the aluminum mold, the mixture was degassed for 20 minutes. A second degassing procedure was then carried out to be sure that the fresh raw PDMS fully covered the microwells. The Al mold with the PDMS combination was then heated for 1 hour at $75^{\circ}\boldsymbol{C}$ in the stove. The PDMS substrate was then removed from the aluminum mold by infusing ethanol in order to lessen the stickiness between both the PDMS and aluminum substrates and to maintain the integrity of the micropillars. The intake and outflow holes were created by punching two input holes. After that, oxygen plasma was used to attach the PDMS substrate to a glass plate (1" x 3") to create the fully functional microfluidic viscometer chip that can be seen in Figure 2.1. PDMS micropillar arrays with a ratio (H:D) of 5:1 (height H: 1500 μm, dimension D: 300 μm) were produced by the manufacturing procedure. The micropillar top, as well as the channel bottom, are separated by 100 μm due to the 1600 μm channel height.

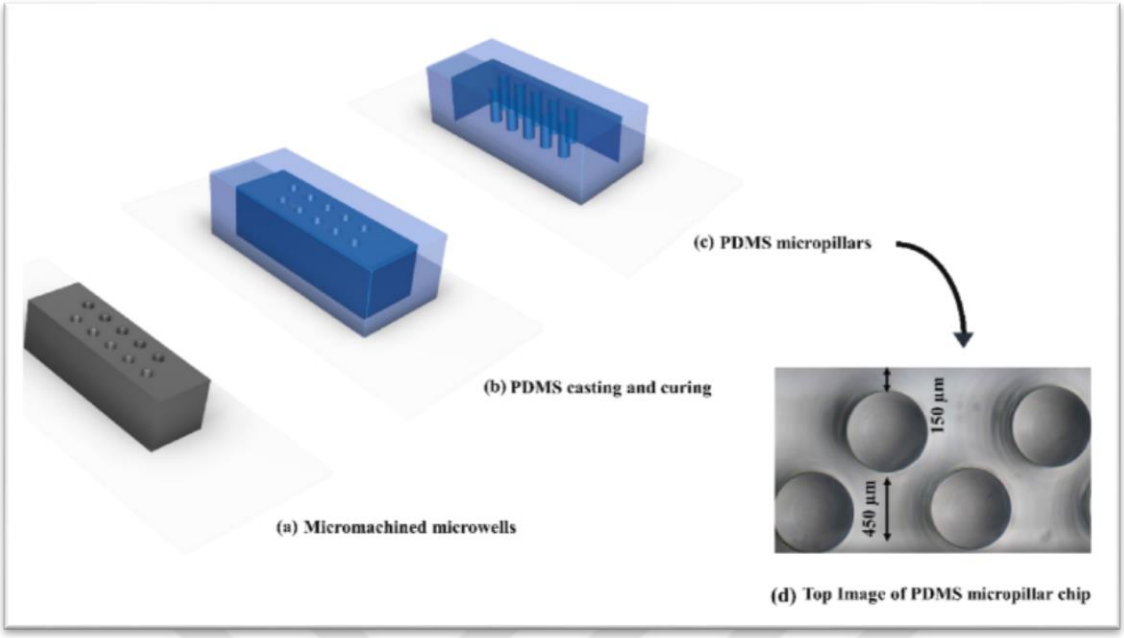

Figure 2.1: (a) Aluminum mold with micromachined 10 microwells each with an aspect ratio (H:D) of 5:1 (height H:1500 µm and diameter D:300 µm). (b) PDMS casting and curing (c) Final PDMS chip having micropillars with height H :1500 µm and diameter D:300 µm. Channel length L:6000 µm, channel width W:900µm. (d) Top view image of PDMS chip [3].

2.2 Recording Setup

An inverted microscope (Nikon Eclipse TS100, Tokyo, Japan), a syringe pump (Ultra 70-3007, Harvard Apparatus, Holliston), a CCD camera (Point Grey Grasshopper3 USB3.0, Richmond, Canada), and a computer were used to film the experiment. Using a syringe pump, the fluid samples were placed into a syringe and injected into the microfluidic chip (Figure 2.2). The inverted microscope was used to scan micropillars, which had a 10-objective lens, and a 0.7 demagnification lens positioned at the imaging port, giving an effective magnification of 7. The CCD camera was used to capture high-quality video images of micropillar deflection at 22 frames per second. The pixel size of the camera was 5.68 µm and by dividing the camera pixel size by magnification (7x) We get a resolution of 0.81 µm/pixel in our videos. The photos of micropillars in the microfluidic device were taken at varied flow rates ranging from 15 to 105 ml/h, as well as at a static (no flow) condition.

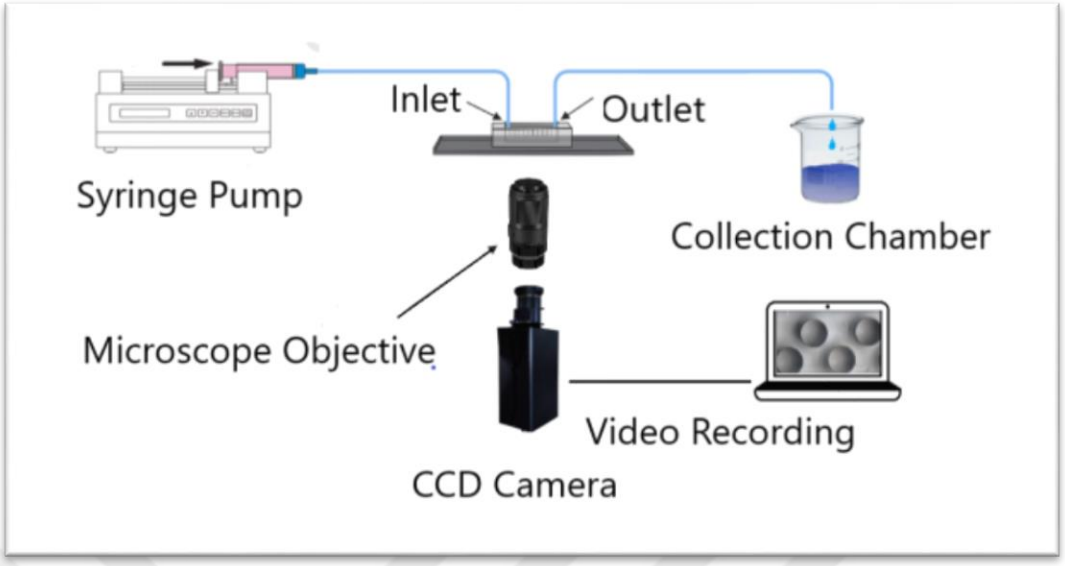

Figure 2.2: The experimental set-up for the micropillar-based microfluidic viscometer is shown schematically. The microfluidic device's micropillars are imaged using a microscope with a 10X objective lens, a syringe pump to inject the test fluid, a CCD camera, and a laptop.

3. INVESTIGATION OF MICROSTRUCTURE DISPLACEMENT

In this chapter, we will discuss generating displacement data from recorded experiment videos using the Lucas-Kanade, FlowNet2 and Hough Circle algorithms. We also go over the proposed method's literature and primers. Then we will discuss and compare the results of the three methods.

3.1 Basics and Literature of Lucas-Kanade Image Processing Method

After checking articles on viscosity measurement research, it's time to move on to the Computer Vision part. Digital image processing is when a digital computer uses an algorithm to process digital images. In the last 30 years, its use has grown at a very fast rate. Digital image processing is a very broad field that includes both digital signal-processing techniques and image-specific ones. There are several ways to process digital images: image enhancement, video processing, image restoration, image analysis, and image compression. Several programming languages, such as C++, Java, Python, and others, can be used to process digital images. In this project, the OpenCV library is used to write the code for processing digital images. OpenCV is an open-source C++ library for image processing and computer vision. It was created by Intel and is now supported by Willow Garage. It's free for both business and non-business use. It is a library that is mostly used for processing in real-time. Now, it has hundreds of built-in functions that run image processing and computer vision algorithms. This makes it easier to make advanced computer vision applications. Motion estimation has been the focus of various studies in computer vision. This extensive study has resulted in a large variety of estimating methods; too many to cover here. However, it is fascinating to quickly review some fundamentally well-known findings. The local window-based technique of Lucas-Kanade [7] which we use in this thesis and the global regularized framework of Horn and Schunck [8] are seminal works that were initially presented for stereovision and made popular for estimate by Barron et al. [28]. Insofar as they rely on a first-order Taylor expansion of the spatio-temporal intensity function, Horn and Schunck and Lucas-Kanade are gradient-based approaches. Large

motion scenarios have been included in the realm of application of gradient-based techniques since the mid-1990s thanks to iterative and multiresolution implementations. The computer vision community has put a lot of work into making objective comparisons using benchmark pictures and performance metrics [28, 29, 30]. While calculation time is mentioned but not taken into account in the evaluation process, these evaluations concentrate on the precision of optical flow. The LucasKanade method was the first picture alignment method (Lucas and Kanade,1981). The motion field in a picture is the actual motion of the item in the 3-D environment, represented on the picture plane. In motion pictures, the "flow" grayscale at the picture plane is referred to as the optical flow. Optical flow, also known as picture flow, is the perceived motion at the picture plane that has aspects of velocity. We use v to represent the optical flow $\vec{v} = (u, v)$. where, accordingly, u and v stand for the optical flow vector's x and y elements at a location. Whereas, if the optical flow is identified using two sequential photos, it manifests as a displacement vector $\vec{d}$ out from characteristics in the initial image to those in the next. Assume, $\vec{d} = (d_x, d_y)$ where d_x and d_y represent the x and y components of the displacement vector at a location [4]. Gradient based optical flow calculation is based on a straightforward formula for conserving picture intensity. where I (x, y, t) denote the brightness of the image at point (x, y) in the picture plane at time t. We believe that the brightness of a certain pattern point will stay unchanged while the pixel moves. As a result,

$$I\ (\ x\ ,\ y\ ,\ t) = I\ (\ x + \delta x\ ,\ y + \delta y\ ,\ t + \delta t) \tag{3.1}$$

where δ_x, δ_y, and δ_t, correspondingly, stand for the interframe movement and time interval. One way to express the first-order differential equation of (3.1) is as follows:

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{3.2}$$

where:

$$\frac{dx}{dt} = u \qquad \frac{dy}{dt} = v \tag{3.3}$$

Can be employed to produce an approximation of the velocity component of the picture intensity spatial gradient. We set just another restriction that assesses the velocity flow's deviation from cleanliness in order to also locate the second component. This new measurement is expressed as follows:

$$\left\{\frac{\partial u}{\partial x}\right\}^2+\left\{\frac{\partial u}{\partial y}\right\}^2+\left\{\frac{\partial v}{\partial x}\right\}^2+\left\{\frac{\partial v}{\partial y}\right\}^2 \tag{3.4}$$

We can develop an adaptive approach for optical flow by merging formulas (3.4) and (3.3) with appropriate weights and utilizing calculus of variation to reduce the overall errors [31].

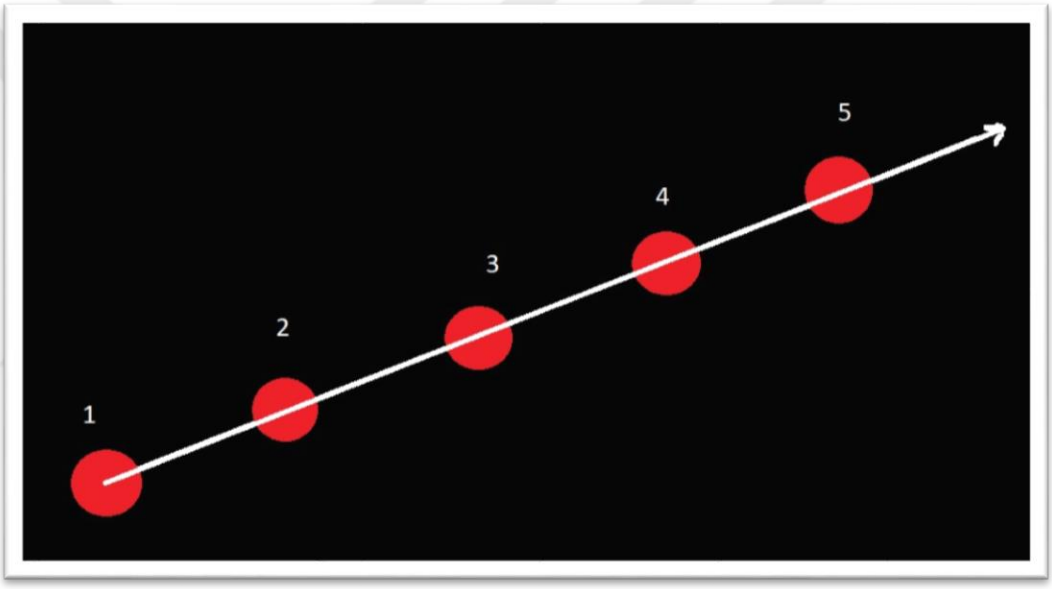

Figure 3.1: Optical flow vector for pixel movement

3.2 Video Processing Using Lucas–Kanade Algorithm

In computer vision, the KLT method is a widely used differential method for optical flow estimation. It assumes that the flow is essentially constant in a local neighborhood of the pixel under consideration, and solves the basic optical flow equations for all the pixels in that neighborhood, by the least-squares criterion. To track the pillars, we must first locate the corner points on them. The built-in OpenCV function, cv2.goodFeaturesToTrack() is used to find the corners. This function is built on the Shi-Tomasi Corner Detector algorithm [32]. In their work Good Features to Track, Tomasi performed a little tweak to it, and the findings are superior to those of the

Harris Corner Detector [33]. The following provided the Harris Corner Detector's scoring function:

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \tag{3.5}$$

Which λ_1, λ_2 are eigenvalues of intensity change Error Function and R is the score function to identify corners in the image. Instead of this, Shi-Tomasi proposed the equation If it is greater than a threshold value, it is considered as a corner.

$$R = \min(\lambda_1, \lambda_2) \tag{3.6}$$

By using cv2.goodFeaturesToTrack() the corner points of the first frame are stored. This method finds the top corners, which might be useful and we don't want to detect each and every corner. Then, by using the KLT Algorithm offered by OpenCV, cv2.calcOpticalFlowPyrLK(); [34] we track these corners along the video. For reaching real-time movement detection, we choose cv2.calcOpticalFlowPyrLK(), window size (500,200) and we select 4 corners to track (Figure 3.2). For calculating optical flow, the presumption that all of the nearby pixels will move similarly has been seen. The 3x3 patch used in the Lucas Kanade method surrounds the spot. The motion is the same at each of the nine points. For these nine positions, we can locate (fx, fy, ft). Thus, our current challenge is to solve nine equations involving two over determined unknowns. The least-square fit method yields the result. The optical flow vectors from these corners are received by the optical flow function. It employs pyramids to deal with large motions. Small motions are eliminated as displacement rises up the pyramid, while large motions are transformed into small motions. Hence, by using KLT, function obtains optical flow with the scale. For each frame, the KLT algorithm finds four corners. Then, in the X-axis, it finds displacement for subsequent frames in relation to the first frame. KLT repeats the process for each of the 4 corners. Then take the average of these four corner displacements to get the pillar's final displacement data.

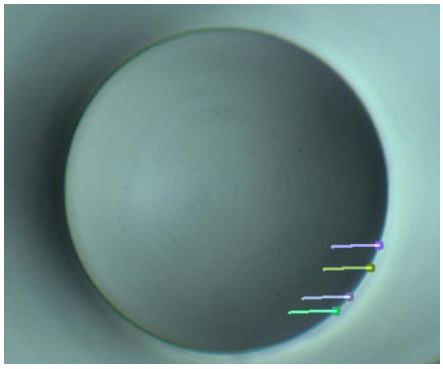

Figure 3.2: Optical flow tracking point following pillars motion

For 10 experiment videos, Table 3.1 compares the precision and processing speed (process time per frame) of KLT to ImageJ. The table shows that KLT achieves 95.45% average precision in terms of ImageJ displacement accuracy with a 158 times faster processing time. Additionally, the procedure is fully automatic and completed without human assistance. We can easily measure unknown fluid viscosity by generating displacement values with KLT.

Table 3.1: KLT method accuracy with respect to ImageJ and speed of process.

Experiment Video	KLT Accuracy	Video Frame Count	KLT Process Speed (Second per frame)	ImageJ Process Speed (second per frame)
75 ml/hr 25 cp	0.984	48	0.05	7.79
90 ml/hr 25 cp	0.978	47	0.06	7.38
105 ml/hr 25 cp	0.945	46	0.06	8.17
120 ml/hr 25 cp	0.941	40	0.06	9.42
75 ml/hr 50 cp	0.95	64	0.06	9.34
105 ml/hr 50 cp	0.93	69	0.06	6.10
135 ml/hr 50 cp	0.944	44	0.06	8.63
60 ml/hr 75 cp	0.966	47	0.06	7.70
90 ml/hr 75 cp	0.95	50	0.06	7.72
120 ml/hr 75 cp	0.957	35	0.058	6.97

Also, we use KLT to determine pillar displacement on Raspberry Pi models. Raspberry Pi models are small calculation units that could be used in portable, low-cost viscometers. Raspberry Pi 3 and 4 speed comparisons for displacement measurements from 5 videos are presented in Table 3.2. In this comparison, we compare overclocked and normal clock of the Raspberry Pi CPU, and we compare one-point displacement detection with four-point detection. Here, Pi3 is overclocked from 0.6 GHz to 1.2 GHz, and Pi4 is overclocked from 1.5 GHz to 2.2 GHz. In addition, heat sinks are used to

keep CPUs cool. Based on our tests, accuracy does not appear to have changed between these devices and the notebook. According to Table 3.2 and Table 3.3, the Raspberry Pi4 is five times slower than the Notebook, while the Raspberry Pi3 is twelve times slower. Based on the table results, the Pi4 has better speed than its older version, so we can use it to build an affordable viscometer for many industrial and medical applications.

Table 3.2: KLT method speed (time/frame) on overclocked Pi3 and Pi4 boards.

Experiment Video	Notebook 4 Points (Second per frame)	Pi4(OC) 1 Point (Second per frame)	Pi4(OC) 4 Points (Second per frame)	Pi3(OC) 1 Point (Second per frame)	Pi3(OC) 4 Points (Second per frame)
75 ml/hr 25 cp	0.055	0.277	0.313	0.685	0.715
90 ml/hr 25 cp	0.057	0.279	0.312	0.693	0.721
105 ml/hr 25 cp	0.057	0.282	0.316	0.688	0.725
75 ml/hr 50 cp	0.061	0.299	0.323	0.709	0.742
105 ml/hr 50 cp	0.063	0.301	0.326	0.713	0.744

Table 3.3: KLT method speed (time/frame) on Pi3 and Pi4 boards.

Experiment Video	Notebook 4 Points (Second per frame)	Pi4 1 Point (Second per frame)	Pi4 4 Points (Second per frame)	Pi3 1 Point (Second per frame)	Pi3 4 Points (Second per frame)
75 ml/hr 25 cp	0.055	0.302	0.346	0.712	0.748
90 ml/hr 25 cp	0.057	0.304	0.352	0.722	0.760
105 ml/hr 25 cp	0.057	0.298	0.356	0.707	0.736
75 ml/hr 50 cp	0.061	0.312	0.363	0.736	0.772
105 ml/hr 50 cp	0.063	0.321	0.358	0.748	0.779

3.3 Basics and Literature of FlowNet2

In several computer vision domains, convolutional neural networks have replaced other techniques as the go-to alternative. They are typically used for classification [35], but nowadays introduced architectures also permit per-pixel processes like semantic segmentation [36] or depth estimation from an image [37]. Numerous publications have already used optical flow to apply machine-learning approaches. Gaussian scale mixtures are used by Sun et al. [38] to analyze optical flow stats and develop regularizes, and by Rosenbaum et al. [39] to describe regional optical flow stats using Gaussian mixture methods. A training set of flow fields' critical elements are

calculated by Black et al. [40]. They next calculate the parameters of a linear combination of these "base flows" in order to anticipate optical flow. Other approaches instruct classifiers to choose from several inertial estimations [41] or to determine occlusion probability [42]. There was also research about the use of neural network models for unsupervised learning of disparity or movement within video frames. The relationships among a pair of images are often modeled using multiplicative dynamics in these techniques. From the latent variables, disparities and optical flow may then be deduced. According to Krizhevsky et al. [43], convolutional neural networks trained with backpropagation [35] exhibit good performance on large photo categorization. This sparked a wave of research on using CNNs for diverse computer vision applications. Despite the fact that there hasn't been much study on using neural networks for flow estimation, there has been research on optical flow estimation with CNNs. With CNNs trained in a supervised or unsupervised way, Fischer et al. [44] generate visual features and compare these attributes using Euclidean distance. In order to forecast how semantically similar image patches are, Zbontar and LeCun [45] train a CNN using a Siamese layout. These approaches differ significantly from FlowNet in that they are patch-based and rely on pre-processing for spatial aggregation. CNNs have recently been used for edge detection [46] and key point estimation [47]. These challenges include per-pixel estimations, which make them comparable to optical flow estimates. They quickly examine several methods because our designs are heavily influenced by the latest events in these per-pixel prediction problems. Given sufficient labeled data, convolutional neural networks are believed to be particularly effective at learning input-output relations. So, using ground truth flows and picture pairs as our inputs, they train a network to predict the x-y flow fields straight from the images. This is known as an end-to-end learning algorithm. Refinement CNNs Combining convolutional layers with pooling, or spatially decreasing the feature maps, is effective at extracting high-level abstract characteristics from pictures. In order to make network training computationally possible and, more importantly, to enable the consolidation of data over significant portions of the input pictures, pooling is required. To generate dense per-pixel estimations, they must find a technique to modify the coarse pooled depiction because pooling reduces resolution. A straightforward option is to combine the two input photos and pass them through a very general network, letting the network select how to analyze the image pair and extract the motion information. This is seen in Figure

3.3 (top). They refer to this architecture as "FlowNetSimple," as it just contains convolutional layers. Making two distinct yet similar processing streams for the two photos and combining them afterward is a simple procedure, as illustrated in Figure 3.3 (bottom). The network is limited by this design to first create meaningful representations of the two pictures individually before combining them at a later stage. Figure 3.3 depicts the network topology "FlowNetCorr," which includes this layer (bottom). The correlation layer enables the network to compare each patch from f1 with each route from f2, given two multi-channel feature maps f1, f2: $\square^2 \rightarrow \square^c$, with w, h, and c being their width, height, and number of channels, respectively. Consider just one patch evaluation at this time. The following is the definition of the "correlation" of two patches with centers at x1 in the first map and x2 in the second map:

$$c(x_1 + x_2) = \sum_{0 \in [-k,k] \times [-k,k]} (f_1(x_1 + o), f_2(x_2 + o)) \tag{3.7}$$

For a patch of squares K: 2k + 1. It should be noted that Eq. 3.7 is identical to one stage of a convolution in neural networks, except that it convolves data with additional data rather than using a filter. It does not have trainable weights because of this. Calculating c (x1, x2) requires $c.K^2$ multiplications. A huge outcome from comparing all patch combinations would $w^2.h^2$ such calculations, rendering effective forward and backward passes impractical. As a result, they incorporate striding in both feature maps and restrict the maximum displacement for evaluations due to computational concerns. They limit the range of x2, and for each point x1, we calculate correlations c (x1, x2) only within a D: = 2d + 1 region. Strides s1 and s2 are used to calculate x1 generally and x2 regionally in the area surrounding x1. According to theory, the correlation produces a four-dimensional outcome: for any set of two 2D locations, we may calculate the correlation value, which corresponds to the scalar product of both of the vectors containing the values of the trimmed patches. In practice, they channel the proportional dispersion. As a result, they get an outcome for the dimensions $(w \times h \times D^2)$. They applied the derivatives with regard to every bottom blob appropriately for the backward pass. In addition to the artificial Flying Chairs dataset, they present the results of FlowNet training on the Sintel, KITTI, and Middlebury datasets. In addition, they experiment with variational refining of the projected flow fields and network fine-tuning on Sintel data.

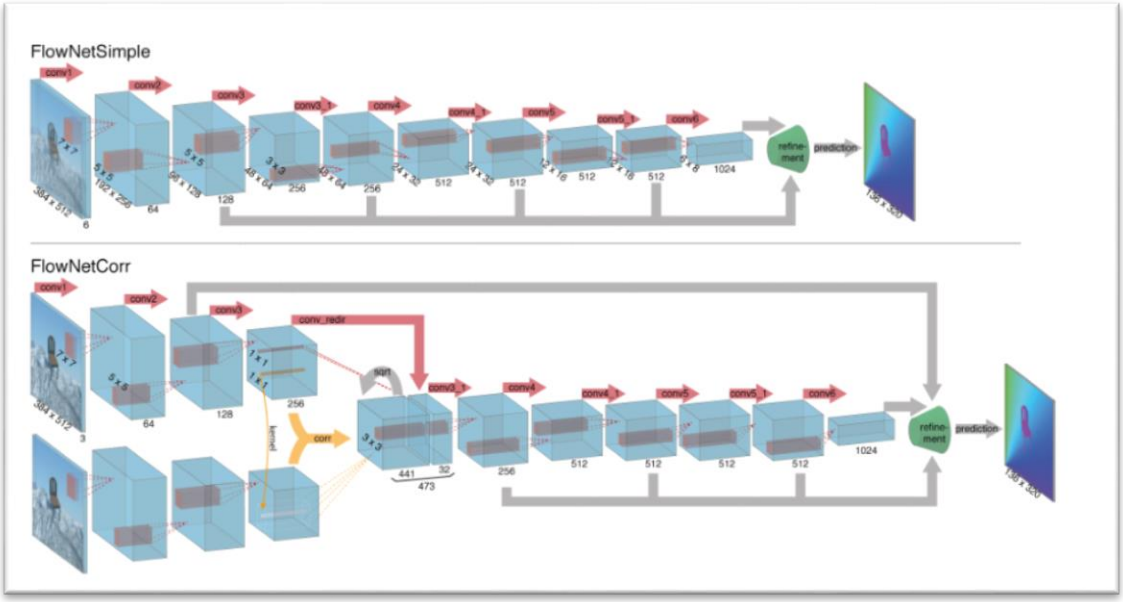

Figure 3.3: FlowNetSimple and FlowNetCorr layer schematic [4].

The benefits of the original FlowNet are carried over into FlowNet2, including the ability to handle huge dispersion, accurate vertical flow field estimates, the capacity to learn priorities for particular situations, and quick runtimes. In addition, it eliminates issues with predicted flow fields' noisy artifacts and minor displacements. As a result, implementations like action detection and movement segmentation dramatically score higher, elevating FlowNet2 to a useful level. The path to FlowNet 2.0 involves a number of progressive yet significant changes that are not tangentially related to the reported issues. They assess the impact of dataset scheduling. Surprisingly, using the relatively complex training data from Mayer et al [48]. A learning schedule made up of many datasets, though, considerably enhances performance. In this area, they also discovered that the implementation of FlowNet with an express correlation layer works better than just the version without one. Furthermore, they describe a warping technique and demonstrate how overlaying many networks with this procedure may dramatically enhance the outcomes. They create a variety of networks with varied sizes and runtimes by changing the depth of the layer and the size of individual elements. By doing so, they can manage the exchange between precision and processing power. They also create networks with frame rates ranging from 8 to 140 frames per second. Lastly, they concentrate on actual data and tiny, subpixel movements. In order to do this, they developed a specific network and a unique training dataset. They demonstrate that the design developed using this dataset functions admirably with the kind of subtle movements found in actual recordings. They add a network that trains to optimally fuse the previous layered network with the tiny dispersion network in order to obtain the best efficiency on unbounded displacements. FlowNet2 outperforms the previous FlowNet on the Sintel and KITTI benchmarks. It could provide interactive frame rates and predict minor and major dispersion with an incredibly high level of accuracy.

3.4 Video Processing Using FlowNet2 Model

A convolutional neural network (CNN) is a type of artificial neural network used to interpret visual images in deep learning. CNN is based on the shared-weight design of the convolution kernels or filters that slide along input features and given translation equivariant responses known as feature maps, they are also known as shift invariant or space invariant artificial neural networks. In the area of optical flow estimation,

Dosovitskiy's FlowNet was a game-changer. The idea of learning optical flow directly from data using a simple CNN architecture was completely new. The resultant FlowNet2 inherits the original FlowNet's features, such as huge displacement mastery, accurate estimate of extremely fine details in the optical flow field, the ability to learn priors for specific circumstances, and quick runtimes. For using the FlowNet2 model first, we apply OpenCV built-in function, cv2.HoughCircles() [49] to remove the pillar background. (Figure 3.4.a) Hough transform can be used to detect circular objects in an image. For applying Hough transform First, the Canny edge detector is used to compute the edge image. Then, for each edge pixel, the gradient data is calculated by using the Sobel operator. Next, it increases the accumulator cells that are located on both sides of the gradient for every edge pixel. Then, choose every accumulator cell that is a local maximum and is higher than a predetermined threshold. These are all solid options for circle centers among the accumulator cells. Finding the ideal radius for every contender is the next step after obtaining the center possibilities. It is clear that the parameter space is only 1D. For each potential center, simply determine how far away it is from all edge pixels in order to fill this 1D accumulator array, which is exactly what the radius is. After detecting the pillar, we will mask its background to fully remove unwanted pixels. This step is essential to increase the precision of FlowNet2. Here, we use the pre-trained FlowNet2 model, which is trained by the THINGS dataset. Then, we take the first frame of the video as the first entry of the model and the upcoming frames as the second entry. The model generates flow files for each frame of video concerning the first frame. (Figure 3.4.b) Flow files give us the pillar displacement data. Table 3.4 demonstrates that, with a processing time that is ten times faster, FlowNet2 achieves an average precision of 95.62% in terms of ImageJ displacement accuracy. The process is also entirely automated and can be completed without the assistance of a human. Due to heavy computational power requirements, FlowNet didn't show promising results on Raspberry Pi boards. Also, by trading between precision and timing, we can get faster results with FlowNet2 at a cost of precision. For this reason, instead of giving full video resolution to the network, we give cropped 200x200 video to the network. Table 3.5 shows the precision and timing results for this method. The overall accuracy dropped to 81% relative to ImageJ, but the operation is 25 times faster than ImageJ, and we have a 2.5-time speed increase relative to general FlowNet2.

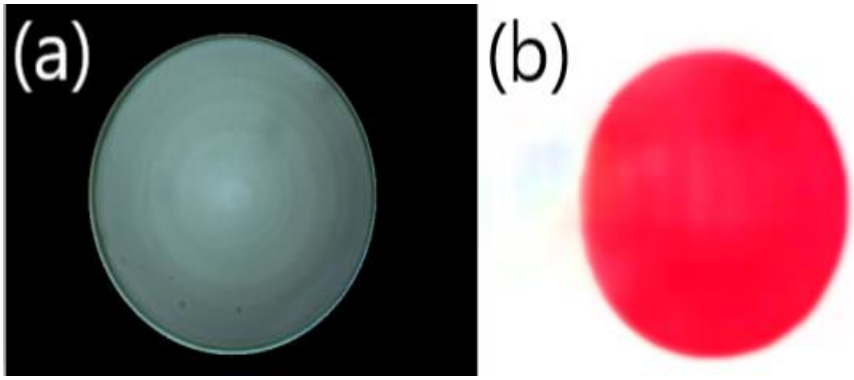

Figure 3.4: (a) pillar frame after removing background. (b) FlowNet2 color map which contains displacement data.

Table 3.4: FlowNet2 method accuracy with respect to ImageJ and speed of process (taken time per frame).

Experiment Video	FlowNet2 Accuracy	Video Frame Count	FlowNet2 Frame Process Speed (second per frame)	ImageJ Frame Process Speed (Second per frame)
75 ml/hr 25 cp	0.911	48	0.75	7.791
90 ml/hr 25 cp	0.961	47	0.74	7.382
105 ml/hr 25 cp	0.934	46	0.75	8.173
120 ml/hr 25 cp	0.950	40	0.77	9.425
75 ml/hr 50 cp	0.942	64	0.7	9.343
105 ml/hr 50 cp	0.939	69	0.72	6.101
135 ml/hr 50 cp	0.914	44	0.74	8.636
60 ml/hr 75 cp	0.932	47	0.77	7.702
90 ml/hr 75 cp	0.914	50	0.72	7.720
120 ml/hr 75 cp	0.858	35	0.74	6.971

Table 3.5: FlowNet2 cropped video method accuracy with respect to ImageJ and speed of process (taken time per frame).

Experiment Video	FlowNet2 Accuracy	Video Frame Count	FlowNet2 Frame Process Speed (second per frame)	ImageJ Frame Process Speed (Second per frame)
75 ml/hr 25 cp	0.881	48	0.3	7.791
90 ml/hr 25 cp	0.911	47	0.31	7.382
105 ml/hr 25 cp	0.895	46	0.31	8.173
120 ml/hr 25 cp	0.866	40	0.36	9.425
75 ml/hr 50 cp	0.89	64	0.26	9.343
105 ml/hr 50 cp	0.808	69	0.2	6.101
135 ml/hr 50 cp	0.592	44	0.35	8.636
60 ml/hr 75 cp	0.915	47	0.3	7.702
90 ml/hr 75 cp	0.78	50	0.3	7.720
120 ml/hr 75 cp	0.662	35	0.36	6.971

3.5 Video Processing Using Hough Circle Method

A Hough circle transform is an image transformation that enables the extraction of circular objects from images, regardless of how full the circle is. Additionally, the transform is only selective for circles and typically ignores extended ellipses. High radial symmetry objects are effectively sought after by the transform, which gives each degree of symmetry one "vote" in the search space. The transform can calculate the centroid and radius of any circular object in an image by exploring a 3D Hough search space. A circle can be represented as $(x-a)^2 + (y-b)^2 = r^2$ where a, b represents the circle center and r is the radius. So, we require 3 parameters (a, b, r) to completely describe the circle. From the equation we need a 3D accumulator for the Hough transform, which would be highly ineffective. So OpenCV uses a trickier method, the Hough Gradient Method which uses the gradient information of edges. We require a 3D accumulator from the equation, which would be incredibly inefficient. Therefore, OpenCV employs a tougher technique called the Hough Gradient Method, which makes use of edge gradient information. A slightly more complex technique used by OpenCV is called the Hough gradient method. So, let's examine how those functions. The name of the method makes it apparent that it considers gradient data. In the previous method, the matching circles were drawn for each edge point in the parameter space, which increased the number of accumulator cells. Now, however, we merely increase the accumulator cells in the gradient direction of each edge pixel instead of

painting the entire circle. There are two steps in this algorithm. First, find all the logical candidates for the circle centers. Second, the optimal radius is then determined for each potential center. OpenCV provides a built-in cv2.HoughCircles() function that finds circles in a grayscale image using the Hough transform. In this function, the Canny edge detector is used to calculate the edge image. Then, for each edge pixel, the gradient information is calculated using the Sobel operator. Now, we increase the accumulator cells that are located in both directions of the gradient for each edge pixel. Then it chooses every accumulator cell that has a local maximum and is higher than a predetermined threshold. These are all solid options for circle centers among the accumulator cells. Finding the ideal radius for each contender is the next step after obtaining the center candidates. It is obvious that the parameter space is now only 1D. This 1D accumulator array needs to be filled; therefore, for each possible center, simply compute its distance from all edge pixels (exactly what the radius is) and increase the accumulator cell in that direction. The radius with the most edge pixel votes will be considered the best radius. For each candidate center, repeat this. Delete the centers that are near the previously chosen center but don't have enough support. The Hough Gradient technique roughly performs this. Table 3.6 demonstrates that, with a processing time that is 396 times faster, FlowNet2 achieves an average precision of 91.47% in terms of ImageJ displacement accuracy.

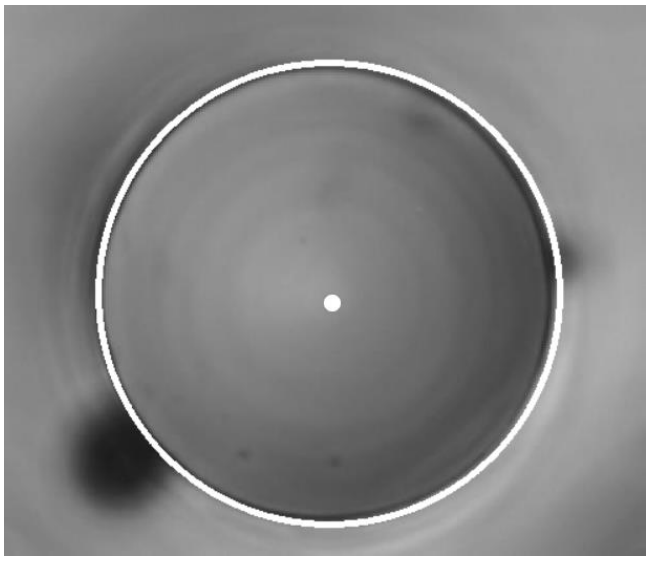

Figure 3.5: Detecting pillar center using the Hough Circle method.

Table 3.6: Hough Circle video method accuracy with respect to ImageJ and speed of process (taken time per frame).

Experiment Video	Hough Circle Accuracy	Video Frame Count	Hough Circle Frame Process Speed (second per frame)	ImageJ Frame Process Speed (Second per frame)
75 ml/hr 25 cp	0.971	48	0.02	7.791
90 ml/hr 25 cp	0.944	47	0.03	7.382
105 ml/hr 25 cp	0.94	46	0.03	8.173
120 ml/hr 25 cp	0.876	40	0.03	9.425
75 ml/hr 50 cp	0.907	64	0.02	9.343
105 ml/hr 50 cp	0.912	69	0.03	6.101
135 ml/hr 50 cp	0.935	44	0.03	8.636
60 ml/hr 75 cp	0.951	47	0.02	7.702
90 ml/hr 75 cp	0.911	50	0.02	7.720
120 ml/hr 75 cp	0.80	35	0.02	6.971

3.6 Experimental Results

In this study, we propose three methods: the Lucas-Kanade algorithm, FlowNet2 and Hough Circle. In this part, we compare the outcomes of the three approaches with ImageJ. Figure (Figure 3.6) shows the displacement of each frame for three experiment videos with different flow rates and viscosities. For calculating the accuracy of the KLT method in each experiment video, we use ImageJ data as a reference and assume that ImageJ accuracy for the displacement of each frame is 1. Then, for each frame, we compare the displacement result of KLT with the corresponding frame of ImageJ. The accuracy of KLT frames is then calculated in relation to ImageJ. After computing the accuracy for each frame's displacement, we average the results to get the overall accuracy for the KLT method. The same operations are used to determine the correctness of the FlowNet2 and Hough Circle models. We provided a table that displays the errors and accuracy of the diverse manners. We examined the accuracy and errors of 10 distinct videos with varying flow rates (Figure 3.7). Table 3.6 shows the average accuracy and error for 10 videos. According to the table, FlowNet2 has higher accuracy and a lower error rate than KLT and Hough Circle. Furthermore, the KLT and Hough Circle approach are real-time procedures and require less computational power than FlowNet2. Also, FlowNet2 requires significantly more time to process data. With KLT we achieved 158 times faster calculation speed with respect to ImageJ, and with FlowNet2 we decreased the processing time about 10 times

compared to ImageJ. In terms of time savings, the Hough Circle method is so promising and is 396 times faster than ImageJ. For the processing in ImageJ, we require human assistance, but in the proposed approaches, everything is done by computer programs, which is a big advantage over ImageJ. Also, the results highly depend on the quality and features of the video recording. Air bubbles and inappropriate lighting during video recording may reduce the accuracy of the viscosity calculation in the presented methods.

Table 3.7: Accuracy and speed comparison results of proposed methods.

Method	Accuracy	Speed (second per frame)
KLT	95.45	0.05
FlowNet2	95.62	0.74
Hough Circle	91.47	0.02
ImageJ	100	7.92

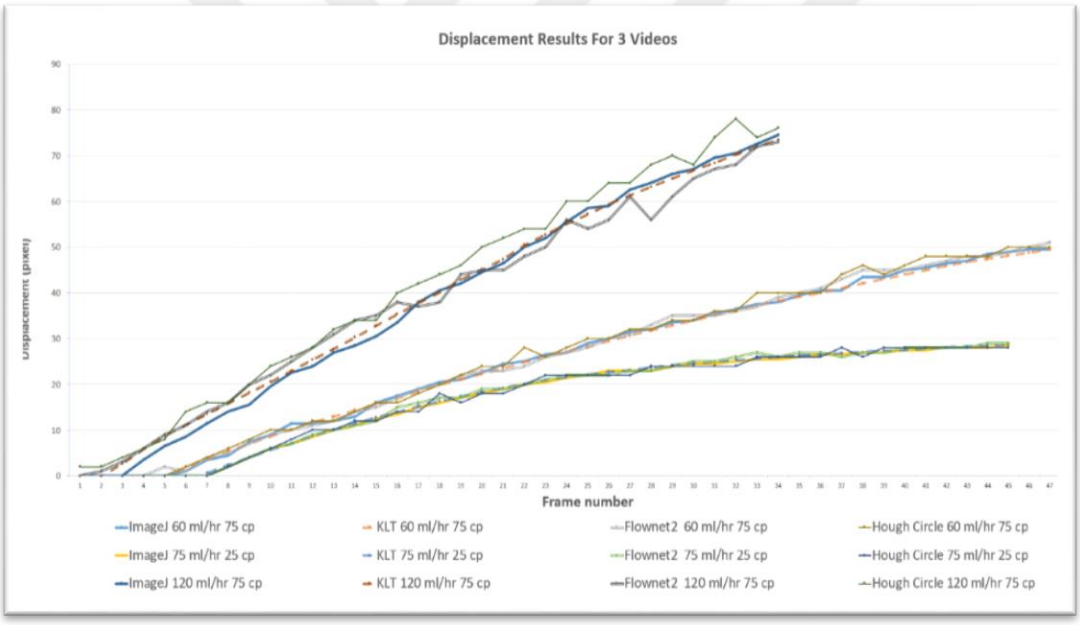

Figure 3.6: Displacement comparison of three proposed methods with ImageJ for three videos: 60mlhr 75cp, 75mlhr 25cp and 120mlhr 75cp. (displacement scale is the pixel difference between the first frame and the next frames.)

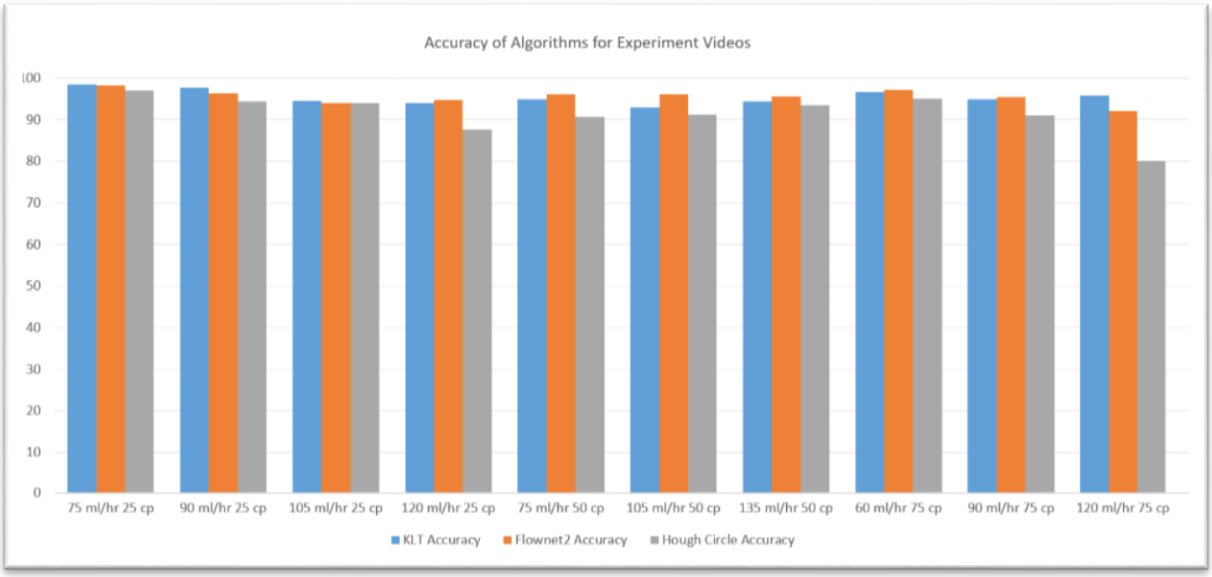

Figure 3.7: Calculated accuracy for three methods with respect to ImageJ.

4. FUTURE WORKS

Along with working on developing methods for metering the viscosity of Newtonian liquids we are working on developing automated methods for measuring the viscosity of non-Newtonian liquids, especially blood. We are trying to develop a point-of-care device for measuring blood viscosity for healthcare and medical applications. Understanding the causes of hemorrhage, directing hemostatic medications, and anticipating the risk of bleeding during subsequent anesthetic or surgical operations all depend on the perioperative monitoring of blood coagulation. In addition, we are building data set using synthesis images for training FlowNet2 and Raft. Also, we are working on recording more videos to increase our data sets.

REFERENCES

[1] **McKennell, R.** (1956). Cone-Plate Viscometer. *Analytical Chemistry*, *28*(11), 1710–1714.

[2] **Galvin, G., Hutton, J., & Jones, B.** (1981). Development of a high-pressure, high-shear-rate capillary viscometer. *Journal of Non-Newtonian Fluid Mechanics*, *8*(1–2), 11–28.

[3] **Mustafa, A., Eser, A., Aksu, A. C., Kiraz, A., Tanyeri, M., Erten, A., & Yalcin, O.** (2020). A micropillar-based microfluidic viscometer for Newtonian and non-Newtonian fluids. *Analytica Chimica Acta*, *1135*, 107–115.

[4] **Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., ... & Brox, T.** (2015). FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2758-2766).

[5] **Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., & Brox, T**. (2017). FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2462-2470).

[6] **Chiantia, S., & London, E**. 1507-Pos Board B277 Investigation of the Martini Force Field for Lipid Raft Membranes Ryan S. Davis1, Mohamed Laradji2. 1Princeton University, Princeton, TN, USA, 2University of Memphis, Memphis, TN, USA.

[7] **Lucas, B. D., & Kanade, T.** (1981). *An iterative image registration technique with an application to stereo vision* (Vol. 81, pp. 674-679).

[8] **Lei, Y., Jinzong, L., & Dongdong, L.** (2007). Discontinuity-preserving optical flow algorithm. *Journal of Systems Engineering and Electronics*, *18*(2), 347-354.

[9] **Espulgar, W., Yamaguchi, Y., Aoki, W., Mita, D., Saito, M., Lee, J. K., & Tamiya, E.** (2015). Single cell trapping and cell–cell interaction monitoring of cardiomyocytes in a designed microfluidic chip. *Sensors and Actuators B: Chemical*, *207*, 43-50.

[10] **Hudson, S. D., Sarangapani, P., Pathak, J. A., & Migler, K. B.** (2015). A microliter capillary rheometer for characterization of protein solutions. *Journal of pharmaceutical sciences*, *104*(2), 678-685.

[11] **Solomon, D. E., Abdel-Raziq, A., & Vanapalli, S. A.** (2016). A stress-controlled microfluidic shear viscometer based on smartphone imaging. *Rheologica Acta*, *55*(9), 727-738.

[12] **Srivastava, N., & Burns, M. A.** (2006). Analysis of non-Newtonian liquids using a microfluidic capillary viscometer. *Analytical chemistry*, *78*(5), 1690-1696.

[13] **Srivastava, N., & Burns, M. A.** (2006). Electronic drop sensing in microfluidic devices: automated operation of a nanoliter viscometer. *Lab on a Chip*, *6*(6), 744-751.

[14] **Lee, J., & Tripathi, A.** (2005). Intrinsic viscosity of polymers and biopolymers measured by microchip. *Analytical chemistry*, *77*(22), 7137-7147.

[15] **Bruneau, C. H., Guillot, P., Panizza, P., Salmon, J. B., Joanicot, M., Colin, A., & Colin, T.** (2006). A viscosimeter on a microfluidic chip. *Langmuir*, *22*(14), pp.

[16] **Li, Y., Ward, K. R., & Burns, M. A.** (2017). Viscosity measurements using microfluidic droplet length. *Analytical chemistry*, *89*(7), 3996-4006.

[17] **Livak-Dahl, E., Lee, J., & Burns, M. A.** (2013). Nanoliter droplet viscometer with additive-free operation. *Lab on a Chip*, *13*(2), 297-301.

[18] **Pipe, C. J., Majmudar, T. S., & McKinley, G. H.** (2008). High shear rate viscometry. *Rheologica Acta*, *47*(5), 621-642.

[19] **Al-Roubaie, S., Jahnsen, E. D., Mohammed, M., Henderson-Toth, C., & Jones, E. A.** (2011). Rheology of embryonic avian blood. *American Journal of Physiology-Heart and Circulatory Physiology*, *301*(6), H2473-H2481.

[20] **Pan, L., & Arratia, P. E.** (2013). A high-shear, low Reynolds number microfluidic rheometer. *Microfluidics and nanofluidics*, *14*(5), 885-894.

[21] **Kang, Y. J., & Yang, S.** (2013). Integrated microfluidic viscometer equipped with fluid temperature controller for measurement of viscosity in complex fluids. *Microfluidics and nanofluidics*, *14*(3), 657-668.

[22] **DeLaMarre, M. F., Keyzer, A., & Shippy, S. A.** (2015). Development of a simple droplet-based microfluidic capillary viscometer for low-viscosity Newtonian fluids. *Analytical chemistry*, *87*(9), 4649-4657.

[23] **Schultz, K. M., & Furst, E. M.** (2011). High-throughput rheology in a microfluidic device. *Lab on a Chip*, *11*(22), 3802-3809.

[24] **Beussman, K. M., Rodriguez, M. L., Leonard, A., Taparia, N., Thompson, C. R., & Sniadecki, N. J.** (2016). Micropost arrays for measuring stem cell-derived cardiomyocyte contractility. *Methods*, *94*, 43-50.

[25] **Brücker, C., Spatz, J., & Schröder, W.** (2005). Feasability study of wall shear stress imaging using microstructured surfaces with flexible micropillars. *Experiments in fluids*, *39*(2), 464-474.

[26] **Judith, R. M., Lanham, B., Falvo, M. R., & Superfine, R.** (2018). Microfluidic viscometry using magnetically actuated micropost arrays. *Plos one*, *13*(7), e0200345.

[27] **Wang, Z., & Qin, X.** (2012, March). The manufacture of micropillars with high depth-to-width ratio, and the comparison between two typical materials. In *2012 7th IEEE International Conference on Nano/Micro Engineered and Molecular Systems (NEMS)* (pp. 677-680). IEEE.

[28] **Barron, J. L., Fleet, D. J., & Beauchemin, S. S.** (1994). Performance of optical flow techniques. *International journal of computer vision*, *12*(1), 43-77.

[29] **Baker, S., Scharstein, D., Lewis, J. P., Roth, S., Black, M. J., & Szeliski, R.** (2011). A database and evaluation methodology for optical flow. *International journal of computer vision*, *92*(1), 1-31.

[30] **Gwosdek, P., Zimmer, H., Grewenig, S., Bruhn, A., & Weickert, J.** (2010, September). A highly efficient GPU implementation for variational optic flow based on the Euler-Lagrange framework. In *European Conference on Computer Vision* (pp. 372-383). Springer, Berlin, Heidelberg.

[31] **Horn, B. K., & Schunck, B. G.** (1981). Determining optical flow. *Artificial intelligence*, *17*(1-3), 185-203.

[32] **Url-1** <*https://docs.opencv.org/3.4/d4/d8c/tutorial_py_shi_tomasi.html*>, date retrieved 26.12.2022.

[33] **Url-2** <*https://docs.opencv.org/3.4/dc/d0d/tutorial_py_features_harris.html*>, date retrieved 26.12.2022.

[34] **Url-3** <*https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html*>, date retrieved 26.12.2022

[35] **LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D.** (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, *1*(4), 541-551.

[36] **Mémin, E., & Pérez, P.** (1998). Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, *7*(5), 703-719.

[37] **Eigen, D., Puhrsch, C., & Fergus, R.** (2014). Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, *27*.

[38] **Sun, D., Roth, S., Lewis, J. P., & Black, M. J.** (2008, October). Learning optical flow. In *European Conference on Computer Vision* (pp. 83-97). Springer, Berlin, Heidelberg.

[39] **Rosenbaum, D., Zoran, D., & Weiss, Y.** (2013). Learning the local statistics of optical flow. *Advances in Neural Information Processing Systems*, *26*.

[40] **Black, M. J., & Yacoob, Y.** (1997). Recognizing facial expressions in image sequences using local parameterized models of image motion. *International Journal of Computer Vision*, *25*(1), 23-48.

[41] **Kennedy, R., & Taylor, C. J.** (2015, January). Optical flow with geometric occlusion estimation and fusion of multiple frames. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition* (pp. 364-377). Springer, Cham.

[42] **Leordeanu, M., Zanfir, A., & Sminchisescu, C.** (2013). Locally affine sparse-to-dense matching for motion and occlusion estimation. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1721-1728).

[43] **Krizhevsky, A., Sutskever, I., & Hinton, G. E.** (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90.

[44] **Fischer, P., Dosovitskiy, A., & Brox, T.** (2014). Descriptor matching with convolutional neural networks: a comparison to sift. *arXiv preprint arXiv:1405.5769*.

[45] **Bontar, J., & Lecun, Y.** (2014). Computing the Stereo Matching Cost with a Convolutional Neural Network [J]. *IEEE*.

[46] **Ganin, Y., & Lempitsky, V.** (2014, November). Neural Network Nearest Neighbor Fields for Image Transforms. In *Asian conference on computer vision* (pp. 536-551). Springer, Cham.

[47] **Hariharan, B., Arbeláez, P., Girshick, R., & Malik, J.** (2015). Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 447-456).

[48] **Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., & Brox, T.** (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4040-4048).

[49] **Url-4 <** *https://docs.opencv.org/4.x/da/d53/tutorial_py_houghcircles.html>*, date retrieved 26.12.2022.

CURRICULUM VITAE

Name Surname **:** Saeed Sarbazzadeh Khosroshahi

EDUCATION **:**

- **B.Sc.** **:** 2017, University of Urmia, Electrical Engineering, Electronic Engineering

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **S. Sarbazzadeh Khosroshahi**, R. Sadeghi, T. Akgul and A. Can Erten "Investigation of Microstructure Movement Under Flow by Using Image Processing and Deep Learning" IGRS'22