Computer Vision for Viscosity Measuring by Falling Sphere Method

KARIN SITTINANTAKORN
THANATHAS CHOTISILP
TIRATH PATTARASIRIPONG

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF ENGINEERING IN MECHANICAL ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2021

Year 2021
Department of Mechanical Engineering Program Mechanical Engineering
Faculty of Engineering King Mongkut's Institute of Technology Ladkrabang

Title: Computer Vision for Viscosity Measuring by Falling Sphere Method

Student

1.	Mr. Karin	Sittinantakorn	Student Number	61010026
2.	Mr. Thanathas	Chotisilp	Student Number	61010427
3.	Mr. Tirath	Pattarasiripong	Student Number	61010526

B.B.U Advisor
(Dr. Bumroong Puangkird)

Computer Vision for Viscosity Measuring by Falling Sphere Method

Mr. Karin	Sittinantakorn	61010026
Mr. Thanathas	Chotisilp	61010427
Mr. Tirath	Pattarasiripong	61010526
Dr. Bumroong	Puangkird	Advisor

Year 2021

ABSTRACT

Study the fluid properties through computer vision to understand the flow behavior of the fluid. In the fields of artificial intelligence and machine learning these days, image processing in computer vision has been used for several purposes, including experimental purposes in fluid mechanics to observe and extract the data from fluid flows. This thesis focuses on processing the given object's video that falls into a fluid to quantify the qualitative data, velocity, and viscosity, using Python programming language with a small single-board computer, Raspberry Pi4. There are two alternatives to detect the object in the image(each frame of the video), Canny edge detection and Deep learning. We decided to do the Canny edge detection as it is the faster option. After we can differentiate the object and the background, we can calculate the displacement of the object between two different frames with respect to the calculated pixels per metric ratio, with the known displacement and time interval (calculate from the frame per second of the video) we can find the velocity of the object.

Acknowledgement

Most of all, we would like to express our special thanks to our advisor Dr.Bumroong Puangkird, for providing us a golden opportunity to do this thesis with his supported knowledge and patience. He also has provided us with advice and a perspective to solve every struggling problem throughout this thesis.

We also thank the Department of Mechanical Engineering King Mongkut's Institute of Technology Ladkrabang for supporting us with the equipment we need to complete our thesis.

Finally, we would like to thank our families, Mr.Sompong Sittinantakorn, Mrs.Supara Pureemahawong, Mr.Porrawath Chotisilp, Ms.Tharabhorn Srinakornchai, Mr.Boonma pattarsiripong, Ms.Ubon Mansamrit, and Ms.Priyaporn Pinyawat for their supports, love, and encouragement. Without them, we would not be able to do this thesis by any chance.

Karin Sittinantakorn

Thanathas Chotisilp

Tirath Pattarasiripong

CONTENTS

LIST OF FIGURES

NUMENCLATURE

$\approx$	:	Almost equal to
∇	:	nabla
α	:	Alpha
π	:	pi
σ	:	sigma
δ	:	delta
β	:	beta
γ	:	Gamma
∂	:	Partial Different
ρ	:	density
τ	:	shear stress
μ	:	mu
A^T	:	Transpose matrix
ΔA	:	cross-sectional areas
F	:	Force
F_D	:	drag force or frictional force
MSE	:	Mean square error
MAD	:	Mean Absolute difference
m	:	mass
P_{abs}	:	Absolute pressure
P_{avg}	:	average pressure
P_g	:	gage pressure
r	:	radius of a sphere
Δt	:	during a time
U	:	Constant velocity
ΔV	:	volume of the element
v	:	speed of a given object that falls into fluid
η	:	viscosity
Δz	:	length

CHAPTER 1
INTRODUCTION

1.1 Background

The usage of fluid properties has been acknowledged in various fields of science. Some believe that analyzing the visualization flow has an advantage over mathematical modeling and typical measurement.

With the growth of computer technology these days, there are many digital image recording devices. Consequently, the improvement in computer technology makes image processing a vital tool in engineering. The way computer vision works is as same as human vision. However, computer vision uses artificial intelligence to train computers to understand complicated things in a visual world by using digital images obtained from cameras or videos and deep learning to identify, classify, and analyze objects. Then provides data that it has been trained to diagnose to gain a high-level understanding of a given process.

Raspberry pi is a single-boarded computer that has been used to study the flow of fluids through the computer vision process in this thesis because of its functionalities in supporting Python language for the image processing algorithm to detect the object and extract the necessary data.

1.2 Objective

To study and extract the properties of fluid using computer vision with raspberry pi.

1.3 Scope of Work

1. Study the properties of fluid from fluid mechanics.
2. Work on a Python algorithm to detect a given object.
3. Extract the velocity of a given object that falls into the fluid from a video through image processing.
4. Calculating the viscosity of the fluid by using data obtained from image processing.

1.4 Expected Benefits

1. Understanding the process of computer vision to apply it to our research or further usage.
2. Developing the algorithm for detecting the velocity of a given object by using raspberry pi4.
3. Developing a mathematical model with a detected velocity for providing viscosity.

CHAPTER 2

FUNDAMENTALS OF FLUID MECHANICS

2.1 Introduction

Fluid mechanics is a study of the behavior of fluids at rest or in motion. Fluid mechanics is based on the study of Newton's laws of motion, the conservation theorem, thermodynamics, and the physical mechanics of fluid. The subject is sectionalized into two parts, fluid statics, and fluid dynamics.

2.2 Properties of fluid

2.2.1 Density

The density of fluid ρ (rho) defines its mass per unit volume.

2.2.2 Specific Weight

The specific weight of fluid γ (gamma) defines its weight per unit volume

$$\gamma = \rho g$$

2.2.3 Newton's Law of Viscosity

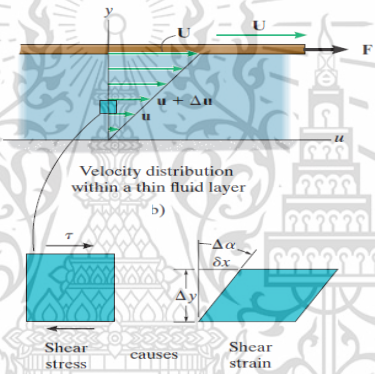


Figure 2.2.3 Newton's Law of Viscosity[1]

The element of fluid is deformed when a very small force **F** is applied to the plate. Consequently, after acceleration, the plate is brought into equilibrium by the viscous resistance of the fluid, so the plate begins to move with a constant velocity **U.** During this process, the plate creates a no-slip condition at the bottom on the plates so that the particles of fluids at the fixed surface remain at rest while those on the bottom side of the plate move with the same velocity as a plate. In between two surfaces, the layers of fluid move along with a velocity u.

2.2.4 Shear stress

Shear stress (τ) occurs during the motion that we have described in topic 2.2.3. It defines the force that elements of fluid affect on a unit area.

$$\tau = \lim_{\Delta A \to 0} \frac{\Delta F}{\Delta A} = \frac{dF}{dA}$$

2.2.5 Shear Strain

Due to the flow of fluid, the shear stress causes a fluid element to deform into the shape that is shown in fig 2.2.3 during a time Δt. The result of this deformation is called shear strain. Assuming the small angle during short time Δt is α (alpha)

$$\Delta\alpha \approx tan\,\Delta\alpha = \frac{\delta x}{\Delta y}$$

Since the element of fluid at the top part moves faster than the element at the bottom part. Therefore, the equation of this relationship can be expressed in an equation as

$$\frac{d\alpha}{dt} = \frac{du}{dy}$$

Or it can be written as

$$\tau = \mu \frac{du}{dy}$$

2.2.6 Reynolds number

The most common way to represent this number in fluid mechanics is based on the ratio of the inertia forces to viscous forces

2.2.7 Laminar flow

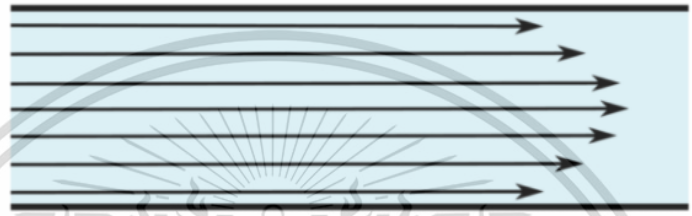


Figure 2.1.1.7 Laminar flow[4]

Laminar flow is the type of fluid flow that is characterized by a smooth path of fluid particles. The velocity of fluid flow in laminar flow is low compared to turbulent flow and there is no mixing between layers of flow. Reynolds number of laminar flows is less than 2000.

2.2.8 Turbulent flow

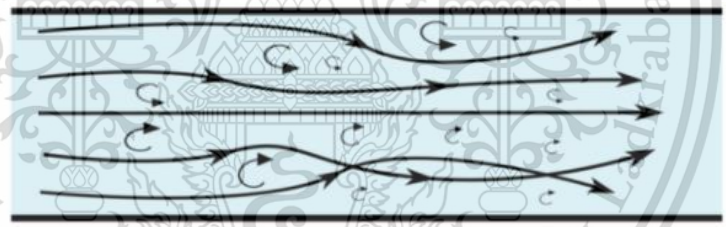


Figure 2.1.1.8 Turbulent flow[4]

Turbulent flow is the type of fluid flow that is characterized by an irregular and chaotic flow of fluid particles. The average velocity of fluid flow in turbulent flow is approximately equal to the velocity of the fluid at the center of the flow.

2.3 Fluid Statics

Fluid Statics is the study of fluids at rest or flows with constant velocity. It mainly dealt with the equilibrium of floating and submerged objects and the pressure exerted by the fluid.

If the fluid is at rest relative to the surface, there is only one force that fluid can exert on each other's molecule, which is the normal force. The effect of this force is called pressure. Pressure is defined as the force acting on an area divided by this area. If the surface has a finite area and the pressure is uniformly distributed over this area, then the average pressure is

$$P_{avg} = \frac{F}{A}$$

Pressure can have units of Pa (N/m^2), psi (lb/in^2).

If a fluid were removed from its container, a vacuum would be created and the pressure in the container would be zero. This is referred to as zero absolute pressure. The pressure higher than this value is referred to as absolute pressure, P_{abs}. standard atmospheric pressure is the average air pressure at sea level at a temperature of 15°C (59°F)

$$P_{atm} = 101.3\ kPa\ (14.70\ psi)$$

gage pressure(p_g) is the pressure measured relative to the atmospheric pressure, therefore

$$P_{abs} = P_{atm} + P_g$$

Static pressure variation Due to the weight of the fluid, the pressure would vary in a static fluid. Consider a small, slender vertical fluid element having cross-sectional areas ΔA, length Δz. For each element, the weight is included. It is the product of the fluid's specific weight γ and the volume of the element, $\Delta V = \Delta A \Delta z$.

The change in pressure from one side of each element to its opposite side is assumed to increase in the positive z direction and expressed as $(\frac{\partial p}{\partial Z})\Delta z$. If we apply the equation of force equilibrium, we obtain

$$\sum F_z = 0;\quad p(\Delta A) - \left(p + \frac{dp}{dz}\Delta z\right)\Delta A - \gamma(\Delta A \Delta z) = 0$$

$$dp = -ydz$$

2.3.1 Pressure variation for incompressible fluid

If the fluid is incompressible, its specific weight is constant. If we consider the liquid surface by the reference, assume that the pressure is p0 and z is measured downward positively as Fig. 2.3.1.

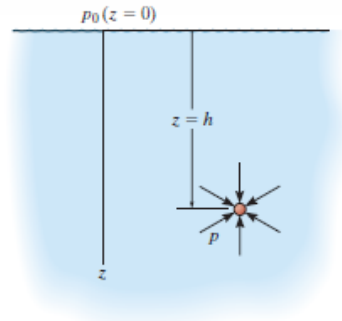


Figure 2.3.1 Pressure variation for incompressible fluid[1]

By integrating from the surface to the depth $z = h$, This process can be written as an equation as

$$\int_{P_0}^{P} dp = \gamma \int_0^h dz$$

$$P = P_0 + \gamma h$$

As we assume that the pressure $P_0 = P_{atm}$ then the pressure of incompressible fluid is

$$P = \gamma h$$

2.3.2 Measurement of static pressure

There are several ways to measure pressure in an engineering field. However, the most common way to measure pressure is using a measuring tool such as a barometer and manometer.

The most common way to measure atmospheric pressure is using a barometer which is a device invented by Evangelista Torricelli. It consists of a mercury inside for pressure measuring purpose.

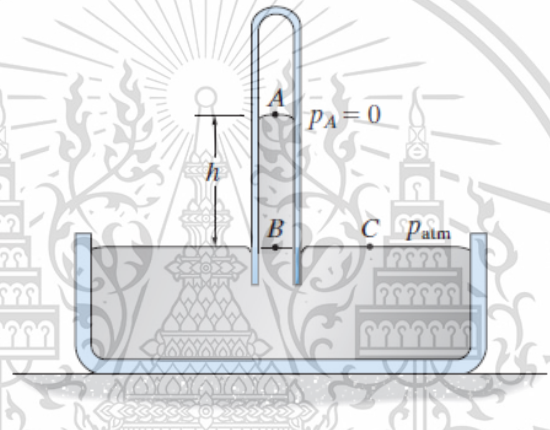


Figure 2.3.2a Barometer[1]

As the atmospheric pressure pushes down on the mercury surface, the pressure at point B and C becomes equal at the same horizontal level so the relationship of the pressure between among these three points becomes

$$P_B = P_A + \gamma H_g h$$

$$P_{atm} = 0 + \gamma H_g h = \gamma H_g h$$

The height h is stated in millimeters of mercury.

Moreover, the most common way to measure the gauge pressure is using a manometer

There are several types of manometers but the most 2 common way that have been used a lot are 1.Piezometer 2.U-tube manometer as shown in fig. 2.3.2b and fig 2.3.2c

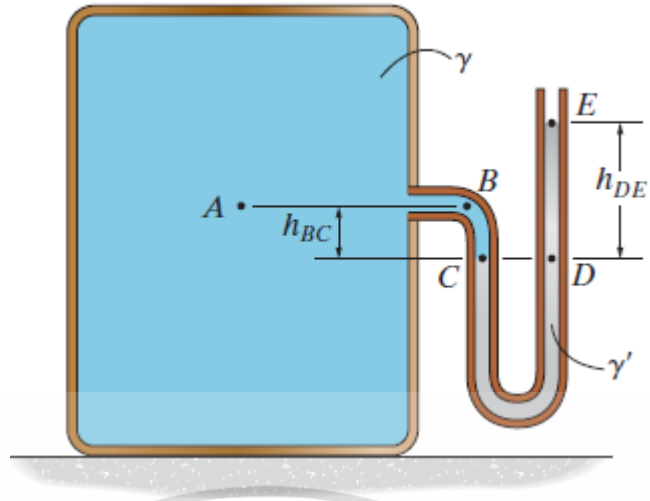


Figure 2.3.2b Piezometer[1]

The pressure at point A is $P_A = \gamma(h + d)$.

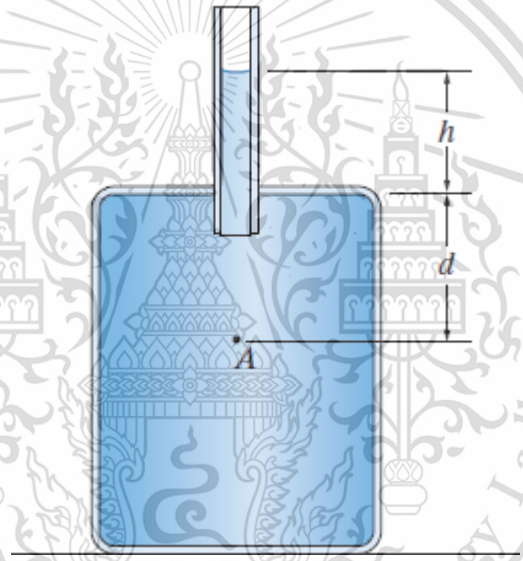


Figure 2.3.2c U-tube manometer[1]

Piezometers do not work well for measuring high gauge pressure, so a U-tube manometer is used instead. From fig. 2.3.2c, the pressure at point A is equal to the pressure at point B in the tube since both points are on the same horizontal level. Therefore, the pressure at point C is $P_C = P_A + \gamma h_{BC}$ and because of the pressure at point C and D are equal, then

$$\gamma' h_{DE} = P_A + \gamma h_{BC}$$

$$P_A = \gamma' h_{DE} - \gamma h_{BC}$$

2.3.3 Buoyancy

The principle of buoyancy states that when a body is placed in a static fluid, it is lift by a force that is equal to the weight of the fluid displaced by the body

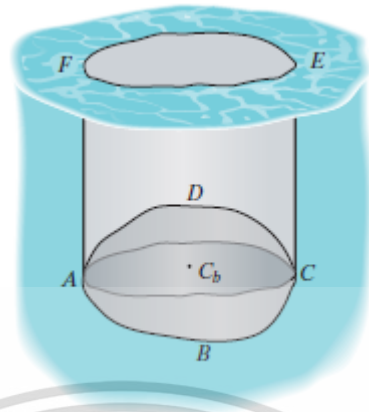


Figure 2.3.3a Submerged body[1]

The state of buoyancy can be explained by observing Fig. 2.3.3a. In Fig. 2.3.3a Because of fluid pressure, the resultant of vertical force acting upward on the bottom surface of the body, ABC, is equal to the weight of fluid contained above this surface, that is, within the volume ABCEFA. Furthermore, the resultant force due to pressure acting downward on the top surface of the body, ADC, is equal to the weight of fluid contained within the volume ADCEFA. The difference in these forces acts upward and is the buoyant force.

2.3.4 Stability

Due to the equilibrium, a body can float in a liquid in three ways, in stable, unstable, or neutral equilibrium.

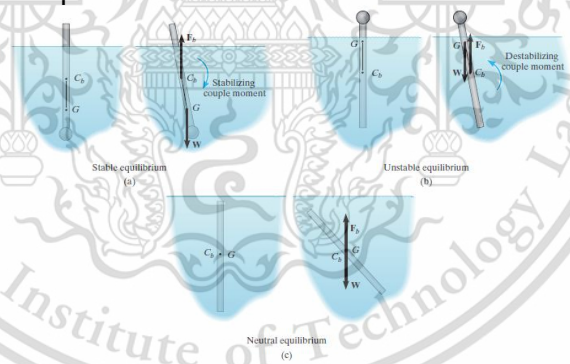


Figure 2.3.4 Stability

Stable equilibrium happens when the object is placed by its center of gravity is below its center of buoyancy, then a slight angular displacement of the bar, Fig. 2.3.4a, creates a couple of moments between the weight and buoyant force that causes the bar to return itself to the vertical position.

Unstable equilibrium happens when the object is placed in the liquid so that its center of gravity is above the center of buoyancy, Fig. 2.3.4b; then a slight angular displacement creates a couple of moment that causes the bar to rotate further from its equilibrium position.

Neutral equilibrium happens when the weight of the given object is removed so that the weight and buoyant force are balanced, then its center of gravity and center of buoyancy stay at the same point in Fig. 2.3.4c. Any rotation of a submerged bar causes it to remain in the newfound equilibrium position.

2.4 Fluid Dynamics

Fluid dynamics is the sub-discipline of fluid mechanics that studies the movement of fluids or flow of fluids. Fluid dynamics are required for the understanding of flow visualization.

For the study of fluid dynamics, three conservation laws are used for observing and analyzing fluid flow. The conservation laws, or what we called governing equations of fluid dynamics, are 1. Conservation of mass 2. Conservation of momentum 3. Conservation of energy.

2.4.1 Governing equations

2.4.1.1 Conservation of mass or Mass continuity

A continuity equation is an equation that describes the transport of some quantity. Especially with conserved quantities, Any continuity equation can be expressed in an "integral form," which applies to any finite region, or in a "differential form." the equation can be written as

$$\frac{\partial}{\partial t}\int_{CV} \rho d\,V + \int_{CS} \rho V_{\frac{f}{cs}} \,.dA = 0$$

2.4.1.2 Conservation of momentum

Conservation of momentum is the general law of physics according to which the quantity called momentum the characterizes motion never changes in an isolated collection of objects. Consequently, the total momentum of a system remains constant.

$$\sum F = \frac{\partial}{\partial t}\int_{CV} V_\rho dV + \int_{CS} V_\rho V_{\frac{f}{cs}} \,.dA$$

2.4.1.3 Conservation of energy

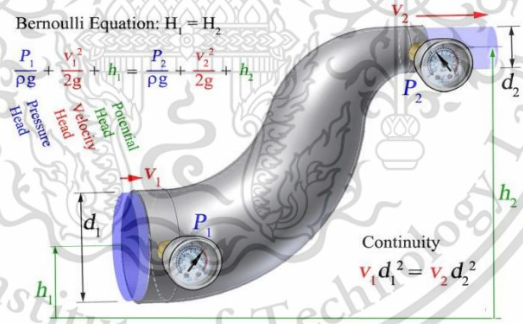


Figure 2.4.1.3 Conservation of energy[5]

Conservation of energy equation or Bernoulli's equation is the equation that states the relationship between pressure, speed, and height of any specific point on the steady streamline.

Bernoulli's equation can be viewed as a conservation of energy because it results from kinetic energy, potential energy, and the extra force gained by the system.

$$p_1 + \frac{1}{2}\rho v_1^2 + \rho g h_1 = p_2 + \frac{1}{2}\rho v_2^2 + \rho g h_2$$

2.4.2 Stokes' Law

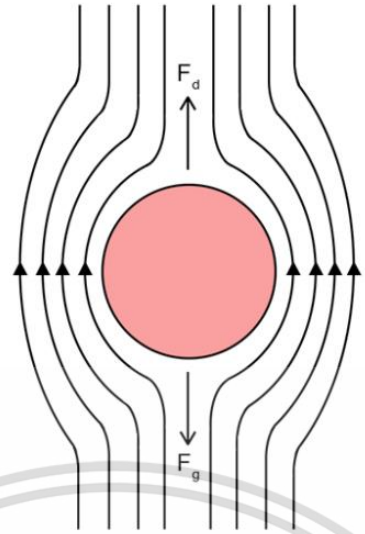


Figure 2.4.2 Stokes' Law[8]

Stokes' Law represents the relationship of a frictional force or drag force of a sphere moving in a Newtonian fluid and other quantities shown in the Stoke's equation.

$$F_D = 6\pi\eta r v$$

F_D	-	drag force or frictional force
η	-	viscosity
r	-	radius of a sphere
v	-	speed of a given object that falls into fluid

2.4.3 Navier-Stokes equation

Velocity profile can be calculated in many ways including Navier-Stokes equation.

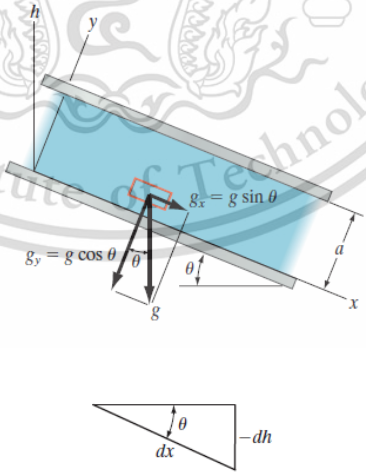


Figure 2.4.3 Navier-Stokes equation

From Fig 2.4.3 x, y, z axes are given. Assuming that the fluid is steady incompressible flow in x-direction, so that v = w =0. So, the continuity equation becomes

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} + \frac{\partial (\rho w)}{\partial z} = 0$$

$$0 + \rho\frac{\partial u}{\partial x} + 0 + 0 = 0$$

The result shows that $\frac{\partial u}{\partial x} = 0$

Assuming the symmetry happens in z-direction and the flow is steady simply say that u is not the function of z but a function of y. Therefore, the result of the Navier-Stokes equation becomes

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z}\right) = \rho g_x - \frac{\partial P}{\partial x} + \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right)$$

$$0 = \rho g\left(-\frac{dh}{dx}\right) - \frac{\partial P}{\partial x} + \frac{d^2 u}{\partial y^2}$$

$$\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z}\right) = \rho g_y - \frac{\partial P}{\partial y} + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right)$$

$$0 = -\rho g\cos\theta - \frac{\partial P}{\partial y} + 0$$

$$\rho\left(\frac{\partial w}{\partial t} + u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z}\right) = \rho g_z - \frac{\partial P}{\partial z} + \mu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right)$$

$$0 = 0 - \frac{\partial P}{\partial z} + 0$$

The result shows that P is constant in z-direction so that

$$P = -\rho(g\cos\theta)y + f(x)$$

Navier-Stokes equation can be rearranged by using the relationship $\gamma = \rho g$ with an integration

$$\frac{d^2 u}{\partial y^2} = \frac{1}{\mu}\frac{d}{dx}(P + \gamma h)$$

$$\frac{d^2 u}{\partial y^2} = \frac{1}{\mu}\frac{d}{dx}(P + \gamma h)y + C_1$$

$$u = \frac{1}{\mu}\left[\frac{d}{dx}(P + \gamma h)\right]\frac{y^2}{2} + C_1 y + C_2$$

2.4.4 Steady laminar flow between parallel plates

Assume that the fluid is Newtonian and its flow is laminar that flows between two parallel plates

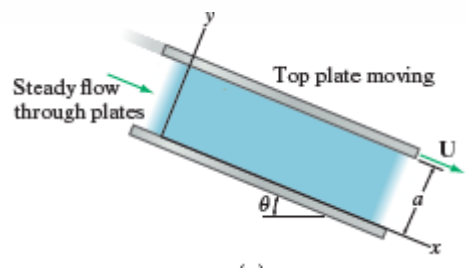


Figure 2.4.4a Flow between two plates

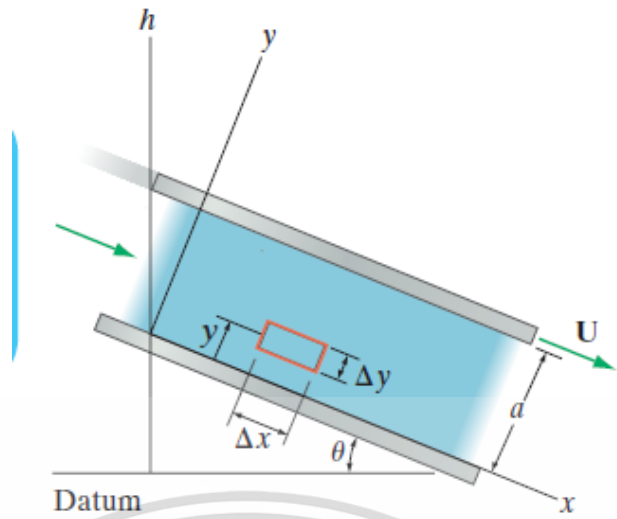


Figure 2.4.4b Fluid element between two plates

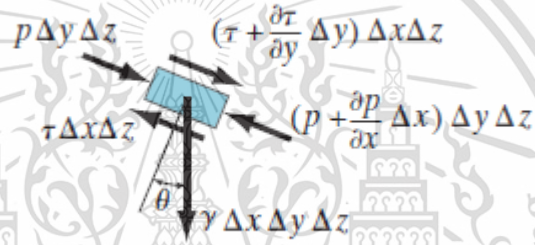


Figure 2.4.4c Free body diagram of fluid element

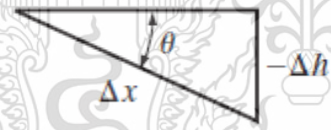


Figure 2.4.4d The angle from free body diagram of fluid element

The plates are separated by the distance a. Assume that the top plate is moving with the velocity U and it is relative to the bottom plate. As in Fig. 2.4.4a the velocity is constant in the x direction but not in the y direction. With the assumption that the fluid is a steady laminar flow and it is incompressible in order to obtain shear-stress the momentum equation is being applied. The free body diagram is shown in fig. 2.4.4c and the equation of this process is

$$\sum F_x = \frac{\partial}{\partial t}\int_{cv} V_\rho \, d\forall + \int_{cs} V_\rho V_{f/cs} \cdot dA$$

$$p\Delta y\Delta z - \left(p + \frac{\partial p}{\partial x}\Delta x\right)\Delta y\Delta z + \left(\tau + \frac{\partial \tau}{\partial y}\Delta y\right)\Delta x\Delta z - \tau\Delta x\Delta z$$

Net pressure force *Net shear force*

$$+\gamma\Delta x\Delta y\Delta z sin\,\theta = 0 + 0$$

Weight

If we rearrange the equation with the relationship of the angle shown in Fig 2.4.4d. Then the equation becomes

$$\frac{\partial \tau}{\partial y} = \frac{\partial}{\partial x}(p + \gamma h)$$

Since the pressure is the function of x, we use the total derivation and integrate the equation above with respect to y.

$$\tau = \left[\frac{d}{dx}(p + \gamma h)\right] y + C_1$$

This equation can be used in both laminar and turbulent flow. Then after apply the concept that the fluid is Newtonian and laminar flow with the Newton's law of viscosity, the equation becomes

$$\mu \frac{du}{dy} = \left[\frac{d}{dx}(p + \gamma h)\right] y + C_1$$

Then, integrating respect to y

$$u = \frac{1}{\mu}\left[\frac{d}{dx}(p + \gamma h)\right]\frac{y^2}{2} + \frac{C_1}{\mu} y + C_2$$

These constant can be eliminated by obtaining the no slip condition

$$\tau = \frac{U\mu}{a} + \left[\frac{d}{dx}(p + \gamma h)\right]\left(y - \frac{a}{2}\right)$$

Shear-stress distribution Laminar and turbulent flow

$$u = \frac{U}{a} y + \frac{1}{2\mu}\left[\frac{d}{dx}(p + \gamma h)\right](y^2 - ay)$$

Velocity profile Laminar flow

Considering the pressure at two point shown in fig. 2.4.4e

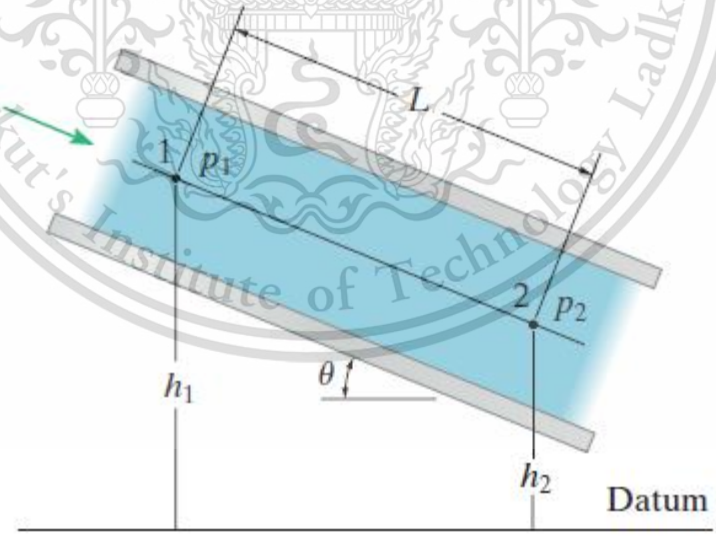


Figure 2.4.4e Pressure between two point

The equation can be written as

$$\frac{d}{dx}(p + \gamma h) = \frac{p_2 p_1}{L} + \gamma \frac{h_2 h_1}{L}$$

2.4.5 Steady laminar flow within a smooth pipe

This section is focused on the determination of velocity and shear stress that make the flow steady and incompressible within a smooth pipe

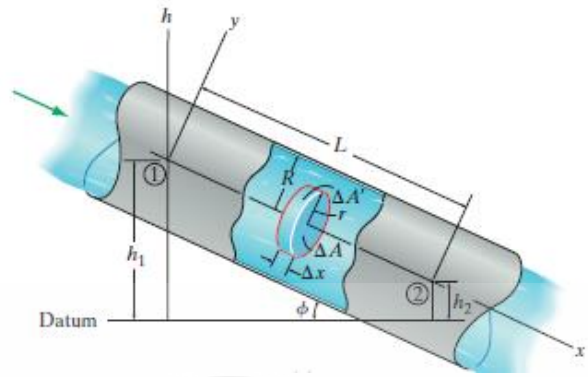


Figure 2.4.5a Laminar flow within a smooth pipe

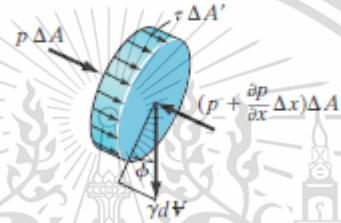


Figure 2.4.5b Free body diagram of fluid element in a smooth pipe

When applying momentum equation to the control volume in Fig. 2.4.5a, Therefore , the free body diagram is drawn in fig 2.4.5b the equation can be written as

$$\sum F_x = \frac{\partial}{\partial t}\int_{cv} V_\rho dV + \int_{cs} V_\rho V_{f/cs} \cdot dA$$

$$p\Delta A - \left(p + \frac{\partial p}{\partial x}\Delta x\right)\Delta A + \tau\Delta A' + \gamma\Delta A' + \gamma\Delta V \sin\emptyset = 0 + 0$$

Applying the cross-sectional area from Fig. 2.4.5a then the shear stress for both laminar and turbulent flow can be calculated by using equation

$$\tau = \frac{r}{2}\frac{\partial}{\partial x}(p + \gamma h)$$

Assuming that the flow is laminar and the fluid is Newtonian, with the Newton's law of viscosity.

$$\frac{du}{dr}\frac{r}{2\mu}\frac{\partial}{\partial x}(p + \gamma h)$$

Then use the total derivative of x and integrating respect to r.

$$u + \frac{r^2}{a\mu}\frac{d}{dx}(p + \gamma h) + C$$

In order to eliminate these constants , we apply the no slip condition so the velocity can be calculated by

$$u = -\frac{(R^2 - r^2)}{4\mu}\frac{d}{dx}(p + \gamma h)$$

2.4.6 Fully developed flow from an entrance

For laminar flow, when a fluid that first stays at rest begins to flow with acceleration to enter the pipe's entrance. Its velocity is nearly uniform at first then, as the fluid travels further, due to its viscosity, the velocity of fluid element near the wall begins to decrease. Consequently, the velocity of the fluid at the wall becomes zero. After that, the viscous layers form a boundary layer begins to spread outward from the wall towards the pipe's centerline until the central core of fluid, which initially had uniform velocity, narrows down and disappears at a distance L. So the result of this, the flow becomes fully developed and the parabolic velocity profile for laminar flow becomes constant as shown in Fig. 2.4.6a

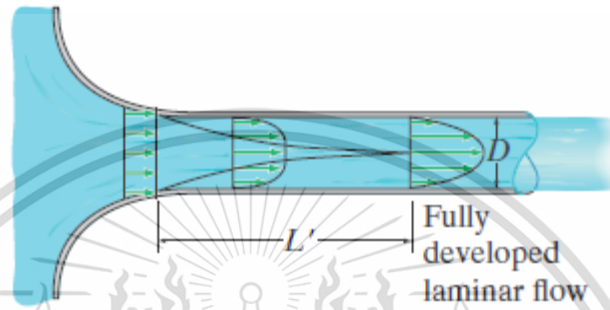


Figure 2.4.6a Fully developed laminar flow

The entrance L' for laminar flow can be calculated by using equation

$$L' = 0.06(Re)D$$

for $Re \leq 2300$

For turbulent flow, the entrance length does not all depend on the Reynolds number. It depends more on the shape and the roughness of the pipe. Through experiments and computer analysis, it has been found that fully developed turbulent flow can occur within a shorter distance compared to laminar flow.

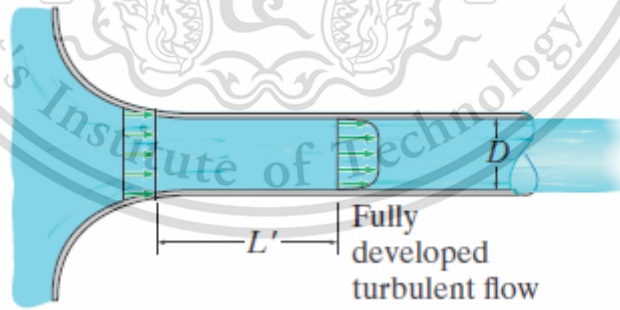


Figure 2.4.6b Fully developed turbulent flow

CHAPTER 3

Computer vision

3.1 Introduction to Computer Vision and Raspberry pi

Nowadays, Computer vision is used in many kinds of fields such as mathematics and engineering. It is used to analyze images and videos that have been captured to study and extract information from captured images and videos.

3.1.1 OpenCV

Open-Source Computer Vision (defined as OpenCV) is an important library source for computer vision. It plays an important role in the machine learning and artificial intelligence field because of its functionalities for image processing and computer vision. It also works with many programming languages, including Python, and it contains lots of necessary functions for machine learning and computer vision.

3.1.2 Raspberry Pi

Raspberry Pi is a small single-board computer developed to work on computer skills and programming languages such as Python. OpenCV allows Raspberry pi to observe, analyze, and extract specific data from images and videos.

3.2 Image processing algorithms

Algorithms are required for processing images in computer vision. The two general algorithms that play an essential role in computer vision are 1. Edge detection 2. Region detection and labeling.

3.2.1 Edge detection

Figure 3.2.1 Canny edge map for our pill image [6]

Edge detection is used for labeling and identifying the boundaries of a given object. The edge detection process is about detecting a discontinuity in the intensity of images or the change in brightness of images. There are several kinds of edge detection algorithms. However, the most commonly used is Canny Edge Detection which is an algorithm used to identify a wide range of edges. Its functionalities are simply converting the image into grayscale for an accurate calculation. Consequently, this method provides a detailed result for further use.

3.2.1.1 Types of edges

1. Step edge

A step edge forms when there is an abrupt change in pixel intensity from one side of the discontinuity to the other, Fig. 3.2.1.1a. These types of edges tend to be easy to detect.

2.Ramp edge

Similar to the step edge, the difference is that the change in pixel intensity is not instantaneous. Instead, the change in pixel value occurs at a short but finite distance, Fig. 3.2.1.1b.

3.Ridge edge

A ridge edge is similar to combining two ramp edges bumped right against another. A ramp edge occurs when image intensity changes abruptly but returns to the initial value after a short distance, Fig. 3.2.1.1c.

4.Roof edge

Unlike the ridge edge, it slowly ramps up on either side of the edge, but it falls back down at the peak quite quickly, Fig. 3.2.1.1d.

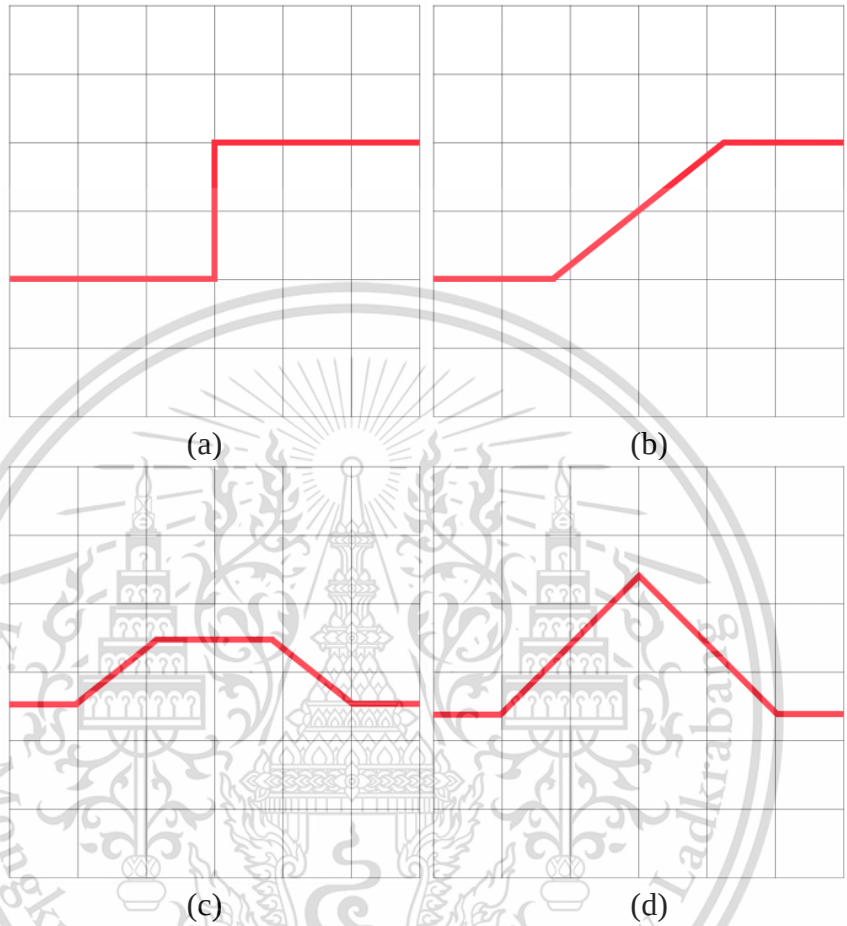


Figure 3.2.1.1 Types of edges [6]

3.2.1.2 Canny edge detection

Canny edge detection algorithm is a multi-step process consisting of

1. Gaussian smoothing

Smoothing an image allows us to ignore the detail and instead focus on the actual structure and outline of the images' objects to processing them further.

Gaussian blurring is similar to average blurring. We use a weighted mean instead of a simple mean, where neighborhood pixels closer to the central pixel contribute more "weight" to the average. Gaussian smoothing is used to remove noise that approximately follows a Gaussian distribution. Based on this weighting, we will preserve more of our image's edges than average smoothing.

Gaussian smoothing uses a kernel of M x N, where both M and N are odd integers. Since the weighting pixels are based on their distance from the central pixel, we need an equation to construct our kernel. The equation for a Gaussian function in one direction is:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}}$$

extend this equation to two directions, one for the x-axis and the other for the y-axis, respectively:

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where x and y are the respective distances to the horizontal and vertical center of the kernel and σ is the standard deviation of the Gaussian kernel. The result is shown in Fig. 3.2.1.2.

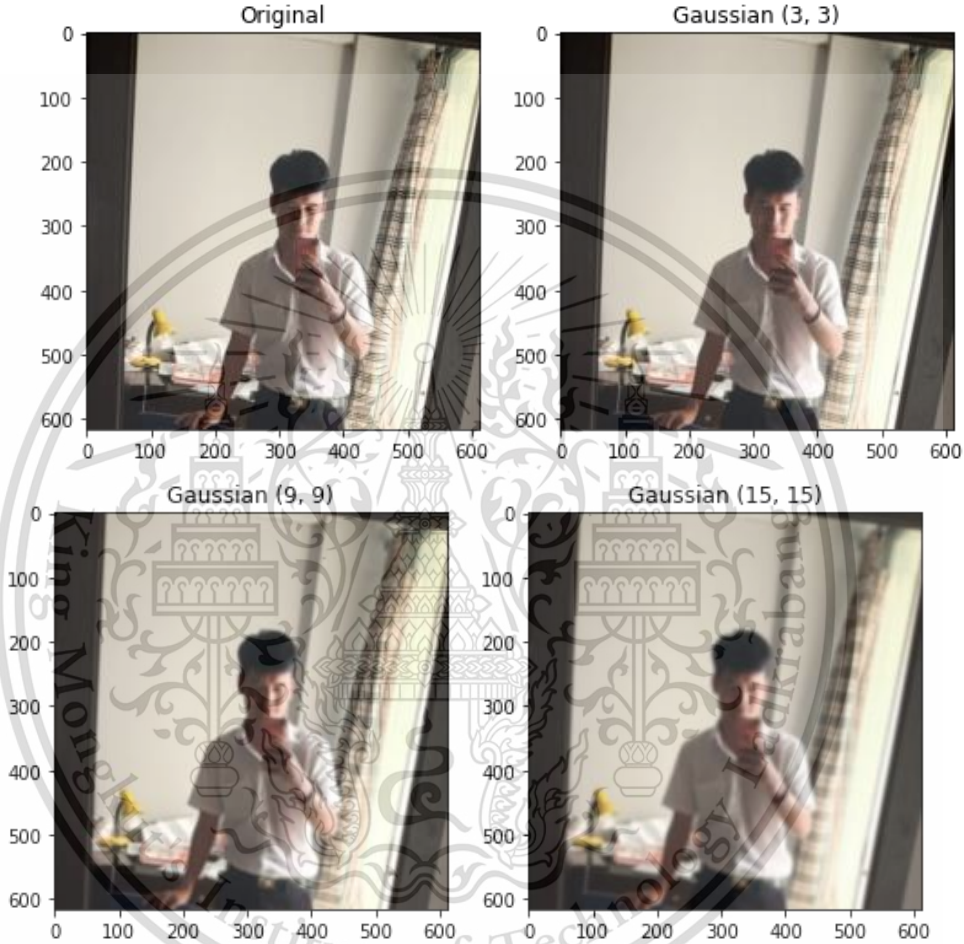


Figure 3.2.1.2. Gaussian blur with different kernel sizes.

3.2.2 Region detection and labeling

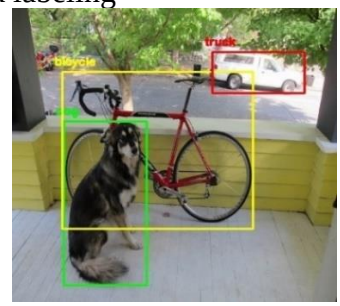


Figure 3.2.2 Region detection and labeling[7]

Region detection identifies the region of binary images and then labels and analyzes the boundary by numbers. Analyzing the boundary of images help in studying their properties, such as parameter. This method is widely used in several kinds of detection, such as object detection, face detection, and face recognition. So, this function is further used in detecting an object to extract its properties in a process.

3.3 Velocity measurement algorithms

3.3.1 Block matching algorithms

Block matching is one of the most common ways to detect and estimate the motion of fluid particles by detecting the movement of an object between two frames of pictures. The process sets the first frame as a reference with a divided sub-block and then observes the block's displacement between two frames to calculate the velocity.

3.3.2 The best match between the block calculation

This method is used in the decision-making in considering what blocks should be compared to each other.

The matching of two blocks is based on mean square error and mean absolute difference to get the extract blocks for comparison.

Mean square error (MSE)

$$MSE = \frac{1}{N^2}\sum_{i=0}^{n-1}\sum_{i=0}^{n-1}(C_{ij} - R_{ij})^2$$

Mean Absolute difference (MAD)

$$MAD = \frac{1}{N^2}\sum_{i=0}^{n-1}\sum_{i=0}^{n-1}(C_{ij} - R_{ij})$$

3.3.3 Optical flow algorithms

Because of the growth of technology in Computer vision, the optical flow method is an important method for computer vision due to its functionality in extracting a relative motion of object and viewer. This method has been more popular than the best match between block method because it is difficult and time-consuming to calculate MSE and MAD.

The optical flow equation is known as the conservation of brightness equation. This equation states that if the selected object in the picture frame is followed by the object, the brightness of the object is not changed as there is no change of brightness caused by the environment.

$$\frac{DI}{Dt} = \frac{\partial I}{\partial t} + U_i E_i$$

This equation can be written in vector form as

$$(I_x, I_y).(u, v) = -\frac{\partial I}{\partial t}$$

Therefore, the magnitude of the vector can be calculated by :

$$-\frac{\frac{\partial I}{\partial t}}{\sqrt{I_x^2 + I_y^2}}$$

The conservation of brightness equation is difficult to use due to the effect of brightness from the environment. However, this method is a suitable method for estimating a tiny movement between frames.

3.3.4 Horn-Schunck algorithm

The Horn-schnuck algorithm is one of the general fundamental algorithms in optical flow measurement. It is based on what is known as the smoothness constraint. This method assumes that if a fluid moves slowly, the square of the velocity gradient is minimum.

$$(\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 \; and \; (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2$$

Therefore, this method is used for calculating the error such that the error of the brightness equation is

$$E_I = I_x U + I_y V + I_t$$

So that the error of calculating the smoothness of velocity flow can be written as

$$\mathrm{E_S^2} = (\frac{\partial \mathrm{u}}{\partial \mathrm{x}})^2 + (\frac{\partial \mathrm{u}}{\partial \mathrm{y}})^2 + (\frac{\partial \mathrm{v}}{\partial \mathrm{x}})^2 + (\frac{\partial \mathrm{v}}{\partial \mathrm{y}})^2$$

Summing these two equations and adding a smoothness factor (alpha) so the equation can be written as

$$E^2 = \int\int E_I^2 + \alpha^2 E_S^2 dxdy$$

The following equation can cooperate with Euler Lagrange equations and be written as

$$I_x\left(I_x U + I_y V + I_t\right) - \alpha^2 \nabla^2 u = 0$$

$$I_y\left(I_x U + I_y V + I_t\right) - \alpha^2 \nabla^2 v = 0$$

Using Laplace velocity approximation

$$\nabla^2 u \approx k(\bar{u} - u)\nabla^2 v \approx k(\bar{u} - u)\nabla^2 v \approx$$

So, the equation of the system can simply be written as

$$(\alpha^2 + I_x^2)u + I_x I_y = \alpha^2 \bar{u} - I_x I_t$$

$$I_x I_y u + (I_y^2 \alpha^2) = \alpha^2 \bar{v} - I_y I_t$$

solving the equation for velocity components from these linear equations

$$\left(\alpha^2 + I_x^2 + I_y^2\right)u = +\left(\alpha^2 + I_y^2\right)\bar{u} - I_x I_y \bar{v} - I_x I_t$$

$$\left(\alpha^2 + I_x^2 + I_y^2\right)v = +(\alpha^2 + I_x^2)\bar{v} - I_x I_y \bar{v} - I_y I_t$$

The Horn-Schunck algorithm can be done as the assumption is settled but finding the suitable smoothness coefficient (alpha) is very difficult to obtain.

3.3.5 Lucas-Kanade method

There is another general algorithm that helps in solving optical flow equations, it is called Lucas-Kanade's method. This method assumes that the motion between two frames is slow, and the optical velocity is constant in each small block of the image frame.

The equation is written in matrix form as

$$\begin{bmatrix} I_x(P_1) & I_y(P_1) \\ \vdots & \vdots \\ I_x(P_n) & I_y(P_n) \end{bmatrix} \cdot \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -I_t(P_1) \\ \vdots \\ -I_t(P_n) \end{bmatrix}$$

Pn is the pixel inside the block window

The equation could be written as v = b

Therefore, using the principle of "Least squares", multiplying both sides by transpose matrix A^T then the equation can be written as

$$A^T A v = A^T b$$

So that the velocity equation in matrix form is

$$v = (A^T A)^{-1} A^T b$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i I_x(p_i)^2 & \sum_i I_y(p_i)\, I_x(p_i) \\ \sum_i I_y(p_i)\, I_x(p_i) & \sum_i I_x(p_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(p_i)\, I_t(q_i) \\ -\sum_i I_y(p_i)\, I_t(q_i) \end{bmatrix}$$

3.3.6 Visualization of Velocity

When observing and calculating the velocity through computer vision, the best and common way is to use a vector. The vector can represent the fluid flow as the mathematical complex number form $u + vi$ and combine with the block matching method or Lucas-Kanade method.

3.4 Deep Learning

Deep learning is a subfield of machine learning. Moreover, it is said to be the subfield of artificial intelligence(AI). Deep Learning is a simulation of the human brain's processing patterns. Using a neuron-like network to process, Deep Learning separates the data and details once the data is received and then analyzes the strengths and differences of the data in depth. It is similar to filtering the data in layers. Then sum the data to output and check how the result is, wrong or right.

For example, with a single animal that does not know what it is, the Deep Learning process checks and predicts 'maybe' this animal, not necessarily saying it has wings or a tail. Deep Learning 'predicts.' first. If deep learning makes a mistake, it learns and adjusts the processing to make the output more accurate. Furthermore, the more it learns, the more Deep Learning understands and goes deeper into the finer details until able to notice even the slightest difference in the data without the need for human guidance.

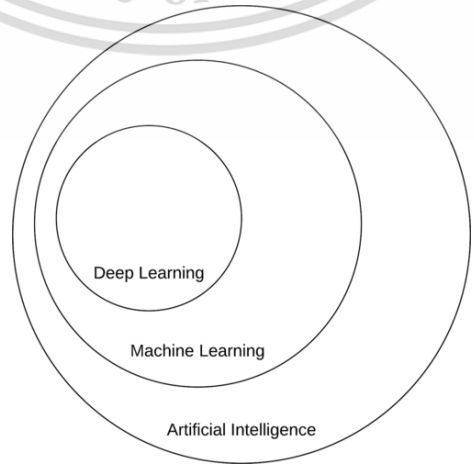


Figure 3.4 Deep learning Chart[9]

In this project, there are two methods of training deep learning models to detect the given object. The first method is trained by using Tensorflow & Keras and the second method is trained by Yolov4.

3.4.1 History of Deep Learning and Neural Network

Deep learning has existed since the 1940s under different names such as cybernetics, connectionism, and even Artificial Neural Networks (ANNs). ANNS was Inspired by the neuron cells of the human brain that interact with each other to form the nervous system. Therefore, ANNs were not meant to be computer brains. They are just a model used for connecting models and processing information through artificial neural networks.

The first neural network model was invented by McCulloch and Pitts in 1943, it was made to become a binary classifier. It could recognize two categories. There was the problem that the weights file that is used for determining class labels could not be done automatically but instead manually by humans.

In the 1950s the seminal Perceptron algorithm was invented by Rosenblatt. This model could do the process that it could not do in 1943 which is automatically performance of weights file. The concept of Perceptron algorithm is shown in Figure 3.4.1. It was formed by the concept of Stochastic Gradient Descent(SGD) which is still valid for training neural networks nowadays.

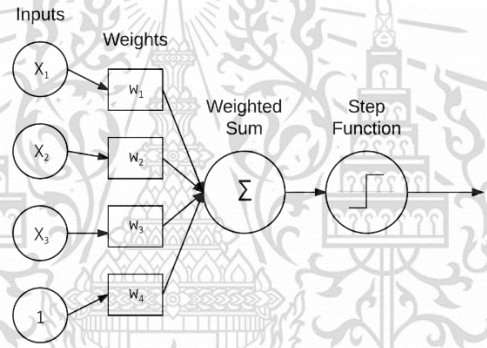


Figure 3.4.1a Perceptron algorithm diagram[9]

In 1969 Minsky and Papert published neural network research that demonstrated the error in the linear solver function of Perceptron. This research indicated that Perceptron was not able to solve nonlinear problems so it was not a good linear classifier.

Furthermore, the backpropagation algorithm along with the research by Werbos, Rmelhart et al and Lecun et al was able to fix the neural networks from its error of solving linear function. Their research showed the backpropagation algorithm could enable multi-layer feedforward neural networks for training.

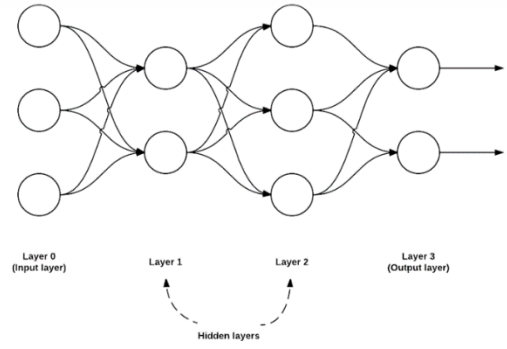


Figure 3.4.1b Backpropagation algorithm diagram[9]

Nowadays, the latest version of neural networks is called deep learning. It has developed so far since the first version itself, its processing is much faster and it has become specialized in training data. The usage of trained neural networks with many more hidden layers has become well-known that are capable of hierarchical learning which sets the basic concept of process in lower layers and complex process within higher layers in order to follow the pattern efficiently.

When the application of deep learning is applied to any type of learning the typical example of the usage of this process is called Convolutional Neural Network(CNNs). Convolutional Neural Network was invented by LeCun et al in 1998. It was first used to recognize handwritten characters automatically by learning the discriminating patterns which are called filters. CNNs learn from images by sequentially putting the stacked layers on top of each other and differentiating them, filters in lower levels usually represent the edges and corners while the higher layers represent other finer details.

The usage of CNNs has become widely known in a short of time. CNNs are now commended for their ability to classify images and pushing the state-of-the-art forward in the computer vision field for the advantage of machine learning.

3.4.2 Hierarchical Feature Learning

Machine learning algorithms are generally divided into three parts, supervised learning, unsupervised learning, and semi-supervised learning.

For supervised learning, an algorithm is set in both given inputs and given outputs for the point of learning patterns so that the computer can mind map the input data automatically for the correct output. Therefore, if the output is incorrect, the supervised learning model corrects the false data to become more accurate next time.

About unsupervised learning, unsupervised learning algorithms always try to automatically discriminate between features without any suggested information to discover what the inputs are. This type of learning is clearly a more challenging problem than supervised learning because it is easier to define the discriminated patterns than the unknown clues.

Semi-supervised learning is a combination of supervised learning and unsupervised learning. This type of learning is more common than supervised learning, the training algorithm on a labeled dataset in each record to process the outcome information. The semi-supervised algorithm allows computers to conclude patterns and identify the connection between a given variable and the rest of the dataset based on the given information. The goal of machine learning for image classification is to take sets of images and identify their patterns to identify their classes or objects of them.

The Deep Learning model increases the performance of computers in machine learning models. In the past, when there was no deep learning, the method to quantify the contents of an image was hand-engineered features. The use of raw pixels as inputs to machine learning models was rare compared to now which is common for deep learning. For each dataset, deep learning performs feature extraction which is the process of collecting the detail of an input image then quantifying it to the settled algorithm called feature extractor, and returning the detail of each dataset in vector form to quantify the contents of a given dataset.

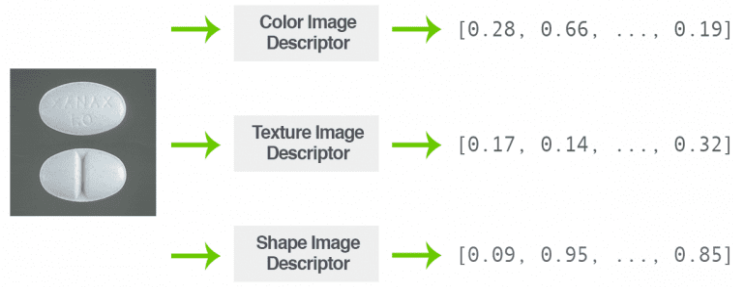


Figure 3.4.2 An example of quantifying the image.[9]

3.4.3 TensorFlow & Keras

Keras is a deep learning Application Learning Interface(API) written in Python, running on the machine learning platform TensorFlow. It focuses on enabling the quick process of experimental purpose. The key of Keras is to be able to get the result from an idea as rapidly as possible for doing research.

In the development of machine learning, it may not be accessible to code because of the complexity of libraries that are used for development. Google has developed open-source software libraries for programming dataflows for different tasks. It is called TensorFlow.

Tensorflow is one of the best deep learning developed by Google. It has been used to develop many products such as search engines, automatic dictionary translation, image captioning, and recommendations. For visualization development, Google has bought AI to help develop user experience in terms of speed to bring results and accuracy. For example, the suggestion of a word from just its second letter.

3.4.4 YOLO

YOLO stands for 'You Only Look Once' an algorithm that plays an important role in detecting and recognizing objects in a picture or video in both recording and real-time.YOLO object detection provides the probabilities of the images by differentiating their classification along with employing convolutional neural networks(CNN) to detect objects. The algorithm works as its name. It requires only a single propagation to get a neural network to work. The prediction of the set of images is done by a single algorithm if their classifications are already trained and saved. The YOLO algorithm consists of various versions of itself. Nevertheless, the most significant version these days is YOLOv4.

In this project, there are two methods of training deep learning models to detect the given object. The first method is trained by using Tensorflow & Keras and the second method is trained by Yolov4.

CHAPTER 4
RESEARCH METHODOLOGY

4.1 Introduction

Developing software to measure the velocity of an object based on visual information such as video clips or camera feed requires an algorithm that can track a given object and find the displacement in that motion by obtaining the correlation between the pixel in the video and the actual displacement.

4.2 Detection

4.2.1 Canny edge detection

We use a Canny edge detector to differentiate between the object and the background as it is a simple and yet effective algorithm to find the edge of an object in the image.

The workflow Canny edge detector.

Step 1: Smooth the image using a Gaussian filter to remove high-frequency noise.

Step 2: Compute the gradient intensity representations of the image.

Step 3: Apply non-maximum suppression to remove “false” responses to edge detection.

Step 4: Apply thresholding using a lower and upper boundary on the gradient values.

Step 5: Track edges using hysteresis by suppressing weak edges that are not connected to strong edges.

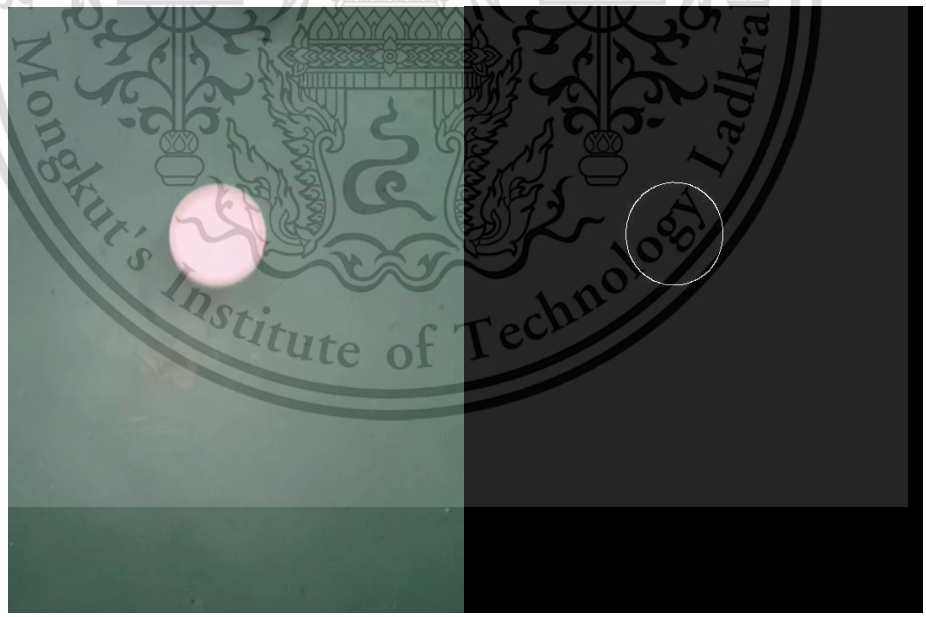


Figure 4.2.1 Detection: (left)original image (right) Image after using Canny edge detector.

4.2.2 Deep-learning object detection

Deep-learning object detection is a powerful tool to detect an object, but we need to train our own object detection framework to use this tool.

4.3 Tracking

After differentiating the object and the background, we find the centroid of an object and keep the position to compare it with the next frame.

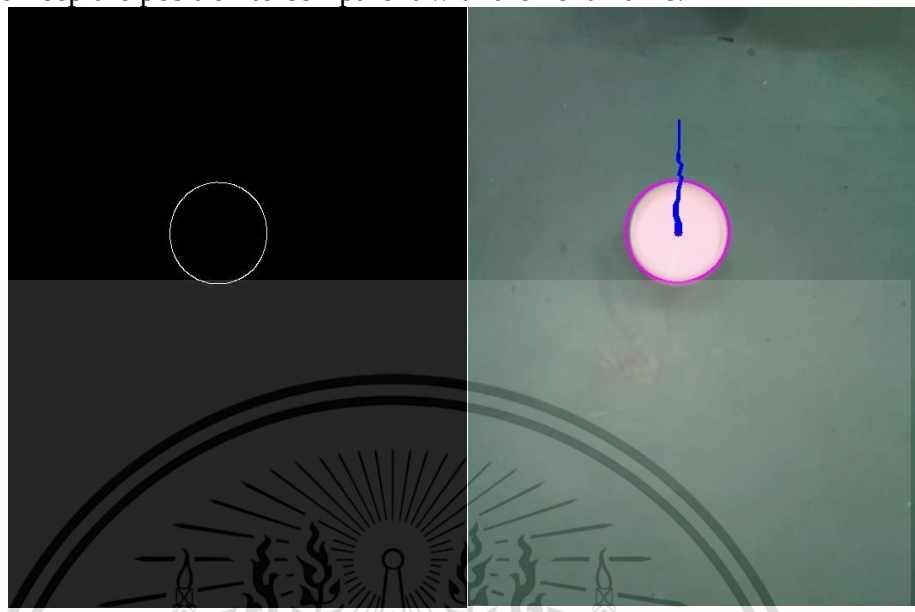


Figure 4.3 Tracking: (left) Image after using Canny edge detector. (right) Draw the outline and the tracking point on the centroid to the original image.

4.4 Extract the data

First, we need to calibrate the pixel in the image to the actual distance. Then find the distance between two frames from the fixed time interval diameter of 7.4 cm. Then find the distance between two frames from the fixed time interval.

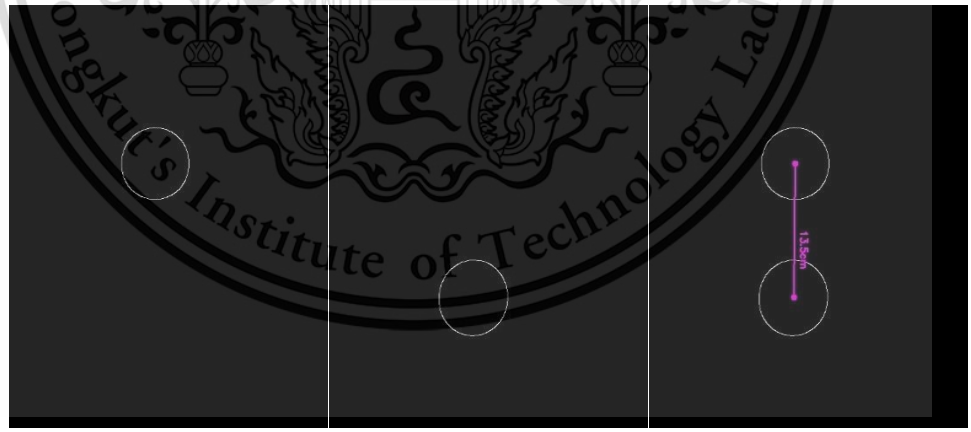


Figure 4.4 Extract the data: (left) Frame 138. (middle) Frame 157. (right) Calculated displacement.

4.5 Process the data

Now we know the distance and time interval, we can easily calculate the velocity of an object. The camera is filming at 30 frames per second.

$$157 - 138 = 19$$

$$\left(\frac{30\ frame}{1\ sec}\right)\left(\frac{13.5\ cm}{19\ frame}\right) = 21.3 cm/sec$$

Substitute the velocity and other known properties into the math model and calculate the viscosity of the fluid by using Stoke's Law and Newton's law of motion for force balancing.

$$\sum F_y = F_d + F_b - mg = 0$$

$$F_d = 6\pi\eta r v$$

$$mg - F_b = \frac{4}{3}\pi r^3(\rho_{ball} - \rho_{fluid})g$$

$$\eta = \frac{\frac{4}{3}\pi r^3(\rho_b - \rho_f)g}{\pi 6 r v} = \frac{2}{9}\frac{r^2(\rho_b - \rho_f)g}{v}$$

4.6 Model Training

4.6.1 Training by Tensorflow&Keras Method

We use the Tensorflow&Keras algorithm for training our custom sphere model.

Step1: We get the sphere picture from the internet for 176 pictures as a data set.

Figure 4.6.1.a example of sphere's picture

Step2: We locate the boundary of a sphere in each picture by using excel to locate the coordinate points and then make the boundary by using python programming.

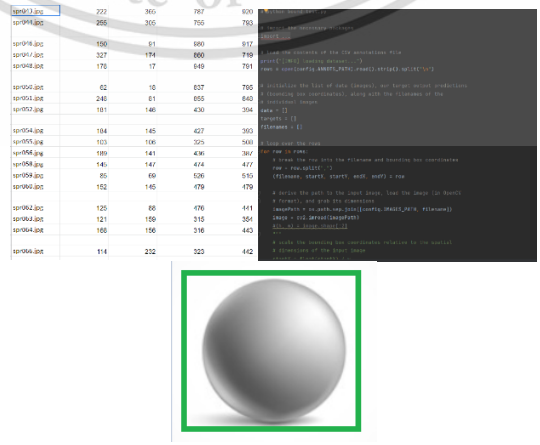


Figure 4.6.1b Process of step2

Step3: We use the Tensorflow&Keras library to train the sphere model.

```
# import the necessary packages
from pyimagesearch import config
from tensorflow.keras.applications import VGG16
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
import cv2
import os

# load the contents of the CSV annotations file
print("[INFO] loading dataset...")
rows = open(config.ANNOTS_PATH).read().strip().split("\n")

# initialize the list of data (images), our target output predictions
# (bounding box coordinates), along with the filenames of the
# individual images
data = []
targets = []
filenames = []

# loop over the rows
for row in rows:
```

Figure 4.6.1c Process of training by Tensorflow & Keras

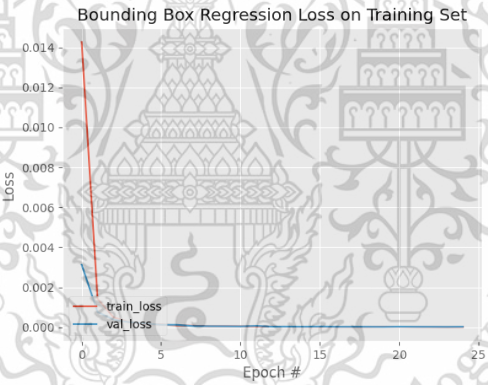
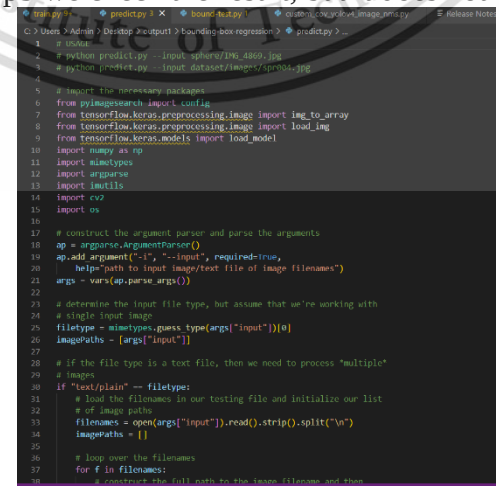


Figure 4.6.1d Graphical representation of Bounding Box Regression Loss

After completing Step3 we check the result, but it does not work as we expect.



```
# USAGE
# python predict.py --input sphere0IMG_4869.jpg
# python predict.py --input dataset/images/spr004.jpg

# import the necessary packages
from pyimagesearch import config
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.models import load_model
import numpy as np
import mimetypes
import argparse
import imutils
import cv2
import os

# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--input", required=True,
    help="path to input image/text file of image filenames")
args = vars(ap.parse_args())

# determine the input file type, but assume that we're working with
# single input image
filetype = mimetypes.guess_type(args["input"])[0]
imagePaths = [args["input"]]

# if the file type is a text file, then we need to process *multiple*
# images
if "text/plain" == filetype:
    # load the filenames in our testing file and initialize our list
    # of image paths
    filenames = open(args["input"]).read().strip().split("\n")
    imagePaths = []

    # loop over the filenames
    for f in filenames:
        # construct the full path to the image filename and then
```

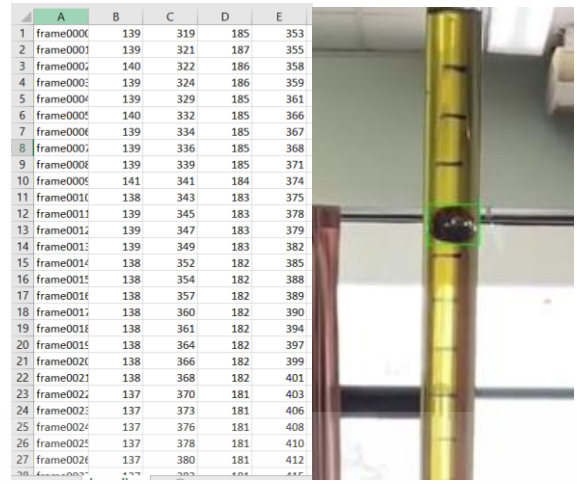
Figure 4.6.1e Process of checking result

Figure 4.6.1f Result of object detection test

We make the hypothesis that the data set that we use for training does not match the sphere in the testing video due to the distorted frame of the sphere that falls into the fluid.

We repeat Step1 with the pictures we get from obtaining the video frame by frame for 829 pictures and repeat Step2 to Step3 again.

Figure 4.6.1g Picture from video



	A	B	C	D	E
1	frame0000	139	319	185	353
2	frame0001	139	321	187	355
3	frame0002	140	322	186	358
4	frame0003	139	324	186	359
5	frame0004	139	329	185	361
6	frame0005	140	332	185	366
7	frame0006	139	334	185	367
8	frame0007	139	336	185	368
9	frame0008	139	339	185	371
10	frame0009	141	341	184	374
11	frame0010	138	343	183	375
12	frame0011	139	345	183	378
13	frame0012	139	347	183	379
14	frame0013	139	349	183	382
15	frame0014	138	352	182	385
16	frame0015	138	354	182	388
17	frame0016	138	357	182	389
18	frame0017	138	360	182	390
19	frame0018	138	361	182	394
20	frame0019	138	364	182	397
21	frame0020	138	366	182	399
22	frame0021	138	368	182	401
23	frame0022	137	370	181	403
24	frame0023	137	373	181	406
25	frame0024	137	376	181	408
26	frame0025	137	378	181	410
27	frame0026	137	380	181	412

Figure 4.6.1h Repeating step2 and 3

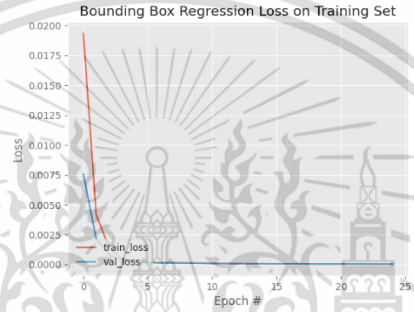


Figure 4.6.1i Graphical representation of bounding box regression loss

As a result of this method, it does not detect the sphere properly. We assume that the Tensorflow&Keras method is not working with a small and unique object due to the distortion of the sphere by the fluid.

Figure 4.6.1j Result of Tensorflow&Keras method

4.6.2 Training by YOLO Method

We use the YOLO method for the training sphere model.

Step1: Preparing YOLO algorithm on “Darknet” which is an open-source neural network framework for YOLO.

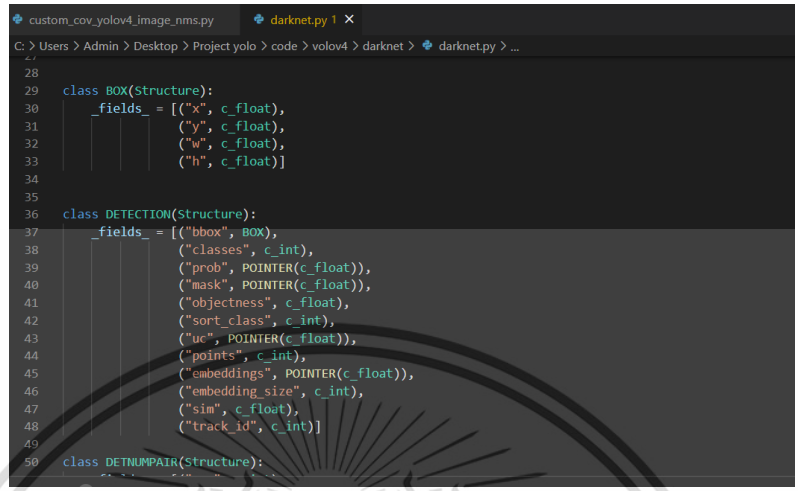


Figure 4.6.2a YOLO algorithm

Step2: Preparing the dataset for 829 pictures and setting the boundary of each picture by using Labellmg program.

Figure 4.6.2b Labelling boundary

Step3: After preparing the data set, we train the YOLO to detect our pictures on Google Collab because the Google GPU has high performance on coding, especially in training a model. Training on Google Collab takes 6.5 hours meanwhile, the training on Raspberry Pi4 takes 17.6 hours regarding the difference between CPU on Raspberry Pi4 and Google GPU.

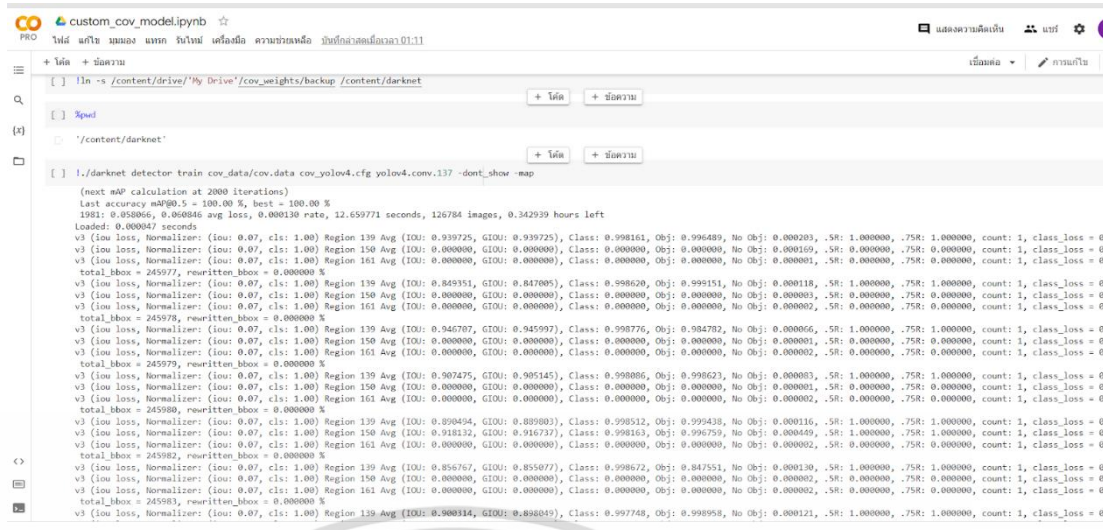


Figure 4.6.2c Processing on Google Collab

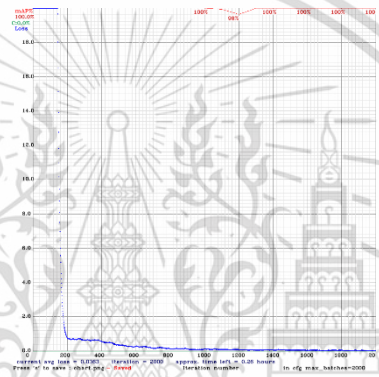


Figure 4.6.2d Graphical representation of training loss on YOLO method

Step4: After getting a weight file which is the file that contains our sphere model, we test it on Raspberry Pi4.

chart_cov_yolov4.png	4/24/2022 4:31 PM	PNG File	76 KB
cov_yolov4.cfg	4/24/2022 4:29 PM	Configuration Sou...	12 KB
cov_yolov4_best.weights	4/28/2022 1:27 PM	WEIGHTS File	250,016 KB
yolov4.cfg	8/5/2020 5:12 PM	Configuration Sou...	14 KB
yolov4.weights	8/5/2020 5:24 PM	WEIGHTS File	251,678 KB

Figure 4.6.2e Result of training

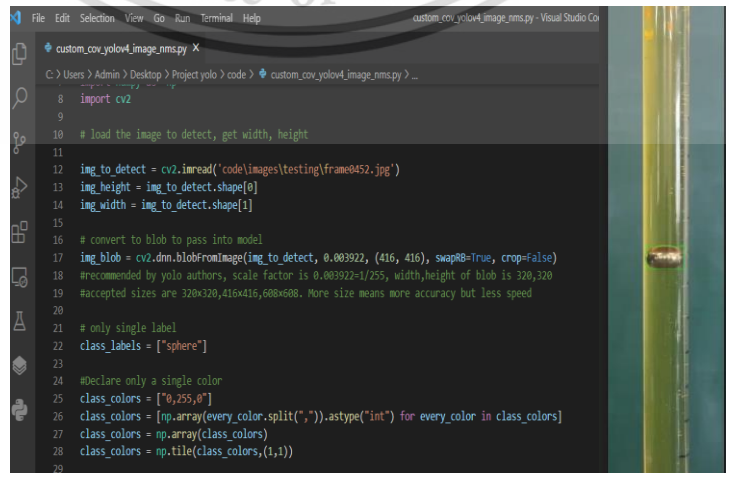


Figure 4.6.2f Result of YOLO method

The result is that our Raspberry Pi4 is able to detect the sphere. Comparing the result of Tensorflow&Keras method, the YOLO method is more accurate than Tensorflow because we are enabled to detect our sphere with the same data set using YOLO but it does not work with Tensorflow&Keras.

Figure 4.6.2g Result of object detection test

4.6.3 Tracking Distance & Speed

After getting the trained sphere model, we run the testing video that the sphere is falling along the tube to obtain the distance that the sphere has moved along the video then calculate the obtained distance into speed and do the python programming to track the speed of the sphere.

Figure 4.6.3 speed tracking

Where inv is an instant speed(cm/s) measured frame by frame and avv is an average speed(cm/s) calculated from the first and last frames that the sphere moves.

Then we export the result of this tracking into excel

Test 1

frame	cen y	inst vel	avg vel
0	74.0	0.000	0.000
1	78.5	6.660	6.660
2	85.0	9.620	8.140
3	91.0	8.880	8.387
4	99.0	11.840	9.250
5	103.5	6.660	8.732
6	110.0	9.620	8.880
7	114.0	5.920	8.457
8	120.5	9.620	8.603
9	130.5	14.800	9.291
10	135.0	6.660	9.028
11	139.5	6.660	8.813
12	145.5	8.880	8.818
13	150.5	7.400	8.709
14	160.0	14.060	9.091
15	165.0	7.400	8.979
16	171.5	9.620	9.019
17	176.5	7.400	8.924
18	181.5	7.400	8.839
19	194.5	19.240	9.386
20	199.5	7.400	9.287
21	204.0	6.660	9.162
22	208.5	6.660	9.048
23	215.0	9.620	9.073
24	227.0	17.760	9.435
25	231.0	5.920	9.294
26	236.5	8.140	9.250
27	244.0	11.100	9.319
28	248.0	5.920	9.197
29	256.0	11.840	9.288
30	261.0	7.400	9.225
31	268.0	10.360	9.262
32	275.5	11.100	9.319
33	275.5	0.000	9.037
34	287.5	17.760	9.294
35	293.5	8.880	9.282
36	299.0	8.140	9.250
37	305.5	9.620	9.260
38	309.5	5.920	9.172
39	317.5	11.840	9.241
40	323.0	8.140	9.213
41	331.0	11.840	9.277
42	336.0	7.400	9.232
43	339.0	4.440	9.121
44	348.5	14.060	9.233
45	353.5	7.400	9.192
46	362.0	12.580	9.266
47	368.0	8.880	9.258
48	370.0	2.960	9.127
49	380.5	15.540	9.258
50	385.5	7.400	9.220
51	393.0	11.100	9.257
52	400.0	10.360	9.278
53	403.5	5.180	9.201
54	412.5	13.320	9.277
55	418.0	8.140	9.257
56	423.0	7.400	9.224
57	430.0	10.360	9.244
58	434.0	5.920	9.186
59	443.0	13.320	9.256
60	448.5	8.140	9.238
61	453.5	7.400	9.208
62	461.0	11.100	9.238
63	466.0	7.400	9.209
64	474.0	11.840	9.250
65	479.5	8.140	9.233
66	483.5	5.920	9.183
67	491.0	11.100	9.211
68	494.5	5.180	9.152
69	504.5	14.800	9.234
70	510.0	8.140	9.218
71	518.0	11.840	9.255
72	524.0	8.880	9.250
73	529.0	7.400	9.225
74	536.5	11.100	9.250
75	542.0	8.140	9.235
76	545.5	5.180	9.182
77	552.5	10.360	9.197
78	557.5	7.400	9.174
79	568.0	15.540	9.255
80	573.5	8.140	9.241
81	580.5	10.360	9.255
82	588.0	11.100	9.277
83	589.5	2.220	9.192
84	598.5	13.320	9.241

85	602.5	5.920	9.202
86	608.5	8.880	9.198
87	616.0	11.100	9.220
88	620.5	6.660	9.191
89	626.0	8.140	9.179
90	634.0	11.840	9.209
91	640.0	8.880	9.205
92	645.0	7.400	9.186
93	649.0	5.920	9.151
94	657.0	11.840	9.179
95	664.5	11.100	9.199
96	670.5	8.880	9.196
97	675.5	7.400	9.178
98	679.0	5.180	9.137
99	685.5	9.620	9.142
100	694.5	13.320	9.183
101	700.5	8.880	9.180
102	704.5	5.920	9.148
103	712.0	11.100	9.167
104	719.0	10.360	9.179
105	727.5	12.580	9.211
106	731.0	5.180	9.173
107	737.5	9.620	9.177
108	740.5	4.440	9.134
109	745.5	7.400	9.118
110	756.0	15.540	9.176
111	760.5	6.660	9.153
112	765.5	7.400	9.138
113	773.0	11.100	9.155
114	779.0	8.880	9.153
115	783.5	6.660	9.131
116	789.5	8.880	9.129
117	796.5	10.360	9.139
118	802.0	8.140	9.131
119	805.5	5.180	9.098
120	812.0	9.620	9.102
121	821.5	14.060	9.143
122	827.0	8.140	9.135
123	831.0	5.920	9.109
124	836.5	8.140	9.101
125	843.0	9.620	9.105
126	851.0	11.840	9.127
127	854.0	4.440	9.090
128	859.5	8.140	9.082
129	865.5	8.880	9.081
130	871.0	8.140	9.074
131	877.0	8.880	9.072
132	885.0	11.840	9.093
133	889.0	5.920	9.069
134	897.0	11.840	9.090
135	903.0	8.880	9.088
136	908.0	7.400	9.076
137	914.5	9.620	9.080
138	920.0	8.140	9.073
139	925.5	8.140	9.066
140	931.0	8.140	9.060
141	935.5	6.660	9.043
142	945.5	14.800	9.083
143	950.0	6.660	9.066
144	955.0	7.400	9.055
145	959.5	6.660	9.038
146	965.5	8.880	9.037
147	972.0	9.620	9.041
148	979.5	11.100	9.055
149	985.0	8.140	9.049
150	990.0	7.400	9.038
151	995.0	7.400	9.027
152	1000.0	7.400	9.016
153	1009.0	13.320	9.044
154	1013.5	6.660	9.029
155	1019.0	8.140	9.023
156	1024.5	8.140	9.018
157	1029.5	7.400	9.007
158	1035.5	8.880	9.006
159	1043.0	11.100	9.020
160	1048.0	7.400	9.010
161	1053.0	7.400	9.000
162	1059.0	8.880	8.999
163	1063.0	5.920	8.980
164	1073.0	14.800	9.015
165	1077.5	6.660	9.001
166	1082.5	7.400	8.991
167	1088.0	8.140	8.986
168	1093.0	7.400	8.977
169	1100.5	11.100	8.989
170	1108.5	11.840	9.006
171	1113.5	7.400	8.997
172	1118.0	6.660	8.983
173	1122.0	5.920	8.966
174	1127.5	8.140	8.961
175	1136.0	12.580	8.981
176	1143.0	10.360	8.989
177	1147.0	5.920	8.972
178	1151.5	6.660	8.959

179	1157.5	8.880	8.959
180	1167.0	14.060	8.987
181	1171.5	6.660	8.974
182	1178.0	9.620	8.978
183	1183.0	7.400	8.969
184	1185.5	3.700	8.940
185	1193.0	11.100	8.952
186	1199.5	9.620	8.956
187	1205.5	8.880	8.955
188	1210.5	7.400	8.947
189	1214.5	5.920	8.931
190	1222.0	11.100	8.942
191	1230.0	11.840	8.957
192	1234.5	6.660	8.946
193	1239.5	7.400	8.938
194	1245.0	8.140	8.933
195	1249.0	5.920	8.918
196	1257.0	11.840	8.933
197	1264.0	10.360	8.940
198	1270.0	8.880	8.940
199	1273.5	5.180	8.921
200	1278.5	7.400	8.913
201	1286.0	11.100	8.924
202	1293.5	11.100	8.935
203	1298.5	7.400	8.927
204	1303.5	7.400	8.920
205	1307.5	5.920	8.905
206	1314.0	9.620	8.909
207	1323.0	13.320	8.930
208	1328.0	7.400	8.923
209	1332.5	6.660	8.912
210	1337.5	7.400	8.905
211	1342.0	6.660	8.894
212	1347.5	8.140	8.890
213	1357.5	14.800	8.918
214	1362.0	6.660	8.908
215	1367.5	8.140	8.904
216	1371.0	5.180	8.887
217	1377.0	8.880	8.887
218	1387.0	14.800	8.914
219	1391.0	5.920	8.900
220	1396.0	7.400	8.893
221	1400.5	6.660	8.883
222	1406.0	8.140	8.880
223	1414.0	11.840	8.893
224	1420.0	8.880	8.893
225	1425.5	8.140	8.890
226	1430.5	7.400	8.883
227	1434.5	5.920	8.870
228	1439.0	6.660	8.861
229	1449.0	14.800	8.886
230	1454.0	7.400	8.880
231	1457.5	5.180	8.864
232	1462.0	6.660	8.854
233	1468.5	9.620	8.858
234	1477.0	12.580	8.874
235	1485.0	11.840	8.886
236	1488.5	5.180	8.871
237	1494.0	8.140	8.868
238	1496.5	3.700	8.846
239	1502.0	8.140	8.843
240	1513.0	16.280	8.874
241	1518.0	7.400	8.868
242	1521.5	5.180	8.852

Test 2

frame	cen y	inst vel	avg vel
0	84.5	0.000	0.000
1	95.0	15.540	15.540
2	101.0	8.880	12.210
3	106.0	7.400	10.607
4	111.5	8.140	9.990
5	117.5	8.880	9.768
6	125.5	11.840	10.113
7	133.0	11.100	10.254
8	138.0	7.400	9.898
9	144.0	8.880	9.784
10	148.0	5.920	9.398
11	159.0	16.280	10.024
12	161.0	2.960	9.435
13	167.5	9.620	9.449
14	173.5	8.880	9.409
15	179.5	8.880	9.373
16	192.0	18.500	9.944
17	198.0	8.880	9.881
18	202.5	6.660	9.702
19	208.5	8.880	9.659
20	216.5	11.840	9.768
21	224.0	11.100	9.831
22	230.5	9.620	9.822

23	235.5	7.400	9.717
24	240.5	7.400	9.620
25	249.0	12.580	9.738
26	256.5	11.100	9.791
27	264.0	11.100	9.839
28	268.0	5.920	9.699
29	274.0	8.880	9.671
30	281.5	11.100	9.719
31	287.5	8.880	9.692
32	294.0	9.620	9.689
33	299.5	8.140	9.642
34	303.5	5.920	9.533
35	311.5	11.840	9.599
36	316.5	7.400	9.538
37	324.0	11.100	9.580
38	329.0	7.400	9.523
39	333.0	5.920	9.430
40	343.5	15.540	9.583
41	350.5	10.360	9.602
42	355.5	7.400	9.550
43	362.0	9.620	9.551
44	366.0	5.920	9.469
45	374.5	12.580	9.538
46	382.5	11.840	9.588
47	387.5	7.400	9.541
48	395.0	11.100	9.574
49	401.5	9.620	9.575
50	407.5	8.880	9.561
51	411.5	5.920	9.489
52	417.0	8.140	9.463
53	422.0	7.400	9.425
54	427.5	8.140	9.401
55	440.0	18.500	9.566
56	445.0	7.400	9.528
57	450.5	8.140	9.503
58	456.5	8.880	9.492
59	467.0	15.540	9.595
60	471.5	6.660	9.546
61	476.0	6.660	9.499
62	482.0	8.880	9.489
63	486.5	6.660	9.444
64	496.5	14.800	9.528
65	503.0	9.620	9.529
66	507.5	6.660	9.485
67	515.0	11.100	9.510
68	519.0	5.920	9.457
69	527.5	12.580	9.502
70	533.0	8.140	9.483
71	538.0	7.400	9.453
72	542.5	6.660	9.414
73	549.0	9.620	9.417
74	559.5	15.540	9.500
75	565.0	8.140	9.482
76	569.5	6.660	9.445
77	575.5	8.880	9.437
78	582.5	10.360	9.449
79	590.5	11.840	9.480
80	594.5	5.920	9.435
81	600.5	8.880	9.428
82	608.0	11.100	9.449
83	615.0	10.360	9.460
84	621.5	9.620	9.461
85	627.0	8.140	9.446
86	633.0	8.880	9.439
87	639.0	8.880	9.433
88	644.0	7.400	9.410
89	652.5	12.580	9.445
90	656.5	5.920	9.406
91	662.5	8.880	9.400
92	667.5	7.400	9.379
93	673.5	8.880	9.373
94	682.5	13.320	9.415
95	688.0	8.140	9.402
96	692.5	6.660	9.373
97	700.5	11.840	9.399
98	706.0	8.140	9.386
99	714.5	12.580	9.418
100	719.5	7.400	9.398
101	726.0	9.620	9.400
102	732.0	8.880	9.395
103	736.0	5.920	9.361
104	745.0	13.320	9.399
105	749.0	5.920	9.366
106	753.5	6.660	9.341
107	761.0	11.100	9.357
108	766.5	8.140	9.346
109	772.0	8.140	9.335
110	779.0	10.360	9.344
111	785.0	8.880	9.340
112	789.5	6.660	9.316
113	796.0	9.620	9.319
114	802.5	9.620	9.321
115	810.0	11.100	9.337
116	815.0	7.400	9.320

117	821.0	8.880	9.316
118	827.5	9.620	9.319
119	834.0	9.620	9.322
120	840.0	8.880	9.318
121	844.5	6.660	9.296
122	851.0	9.620	9.299
123	856.5	8.140	9.289
124	862.0	8.140	9.280
125	870.5	12.580	9.306
126	876.0	8.140	9.297
127	883.0	10.360	9.305
128	888.0	7.400	9.290
129	894.0	8.880	9.287
130	900.5	9.620	9.290
131	906.5	8.880	9.287
132	910.5	5.920	9.261
133	916.5	8.880	9.258
134	923.0	9.620	9.261
135	930.5	11.100	9.275
136	937.0	9.620	9.277
137	943.0	8.880	9.274
138	948.0	7.400	9.261
139	953.5	8.140	9.253
140	962.0	12.580	9.276
141	967.0	7.400	9.263
142	973.0	8.880	9.260
143	978.0	7.400	9.247
144	982.5	6.660	9.229
145	990.0	11.100	9.242
146	996.0	8.880	9.240
147	1002.0	8.880	9.237
148	1007.5	8.140	9.230
149	1014.0	9.620	9.233
150	1018.0	5.920	9.211
151	1026.0	11.840	9.228
152	1032.0	8.880	9.226
153	1037.0	7.400	9.214
154	1042.5	8.140	9.207
155	1047.5	7.400	9.195
156	1054.5	10.360	9.203
157	1060.5	8.880	9.201
158	1066.5	8.880	9.198
159	1071.5	7.400	9.187
160	1077.5	8.880	9.185
161	1084.5	10.360	9.193
162	1092.0	11.100	9.204
163	1096.5	6.660	9.189
164	1103.0	9.620	9.191
165	1106.5	5.180	9.167
166	1115.5	13.320	9.192
167	1121.5	8.880	9.190
168	1127.0	8.140	9.184
169	1131.5	6.660	9.169
170	1137.0	8.140	9.163
171	1142.5	8.140	9.157
172	1150.5	11.840	9.173
173	1156.5	8.880	9.171
174	1161.5	7.400	9.161
175	1166.5	7.400	9.151
176	1171.5	7.400	9.141
177	1179.5	11.840	9.156
178	1186.0	9.620	9.159
179	1191.5	8.140	9.153
180	1196.5	7.400	9.143
181	1202.0	8.140	9.138
182	1211.0	13.320	9.161
183	1216.0	7.400	9.151
184	1220.5	6.660	9.137
185	1226.0	8.140	9.132
186	1230.5	6.660	9.119
187	1240.0	14.060	9.145
188	1245.0	7.400	9.136
189	1251.0	8.880	9.135
190	1255.5	6.660	9.121
191	1260.5	7.400	9.112
192	1264.5	5.920	9.096
193	1275.0	15.540	9.129
194	1280.5	8.140	9.124
195	1285.5	7.400	9.115
196	1291.0	8.140	9.110
197	1298.0	10.360	9.117
198	1303.5	8.140	9.112
199	1310.0	9.620	9.114
200	1315.0	7.400	9.106
201	1319.5	6.660	9.094
202	1326.5	10.360	9.100
203	1335.0	12.580	9.117
204	1339.5	6.660	9.105
205	1344.5	7.400	9.097
206	1349.0	6.660	9.085
207	1353.5	6.660	9.073
208	1364.0	15.540	9.104
209	1369.0	7.400	9.096
210	1374.0	7.400	9.088

211	1379.0	7.400	9.080
212	1383.5	6.660	9.068
213	1391.5	11.840	9.082
214	1398.5	10.360	9.087
215	1404.0	8.140	9.083
216	1408.5	6.660	9.072
217	1413.0	6.660	9.061
218	1418.5	8.140	9.057
219	1426.5	11.840	9.069
220	1433.0	9.620	9.072
221	1438.5	8.140	9.068

222	1443.0	6.660	9.057
223	1448.5	8.140	9.053
224	1458.0	14.060	9.075
225	1461.5	5.180	9.058
226	1467.5	8.880	9.057
227	1472.0	6.660	9.046
228	1477.0	7.400	9.039
229	1488.0	16.280	9.071
230	1492.0	5.920	9.057
231	1497.0	7.400	9.050
232	1502.5	8.140	9.046

Test 3

frame	cen y	inst vel	avg vel
0	101.5	0.000	0.000
1	106.5	7.400	7.400
2	111.5	7.400	7.400
3	118.0	9.620	8.140
4	126.5	12.580	9.250
5	131.0	6.660	8.732
6	137.5	9.620	8.880
7	144.0	9.620	8.986
8	148.0	5.920	8.603
9	157.0	13.320	9.127
10	162.0	7.400	8.954
11	167.0	7.400	8.813
12	172.0	7.400	8.695
13	178.5	9.620	8.766
14	192.5	20.720	9.620
15	198.0	8.140	9.521
16	202.0	5.920	9.296
17	206.0	5.920	9.098
18	213.0	10.360	9.168
19	221.5	12.580	9.347
20	229.0	11.100	9.435
21	233.5	6.660	9.303
22	240.5	10.360	9.351
23	248.5	11.840	9.459
24	255.5	10.360	9.497
25	260.5	7.400	9.413
26	266.5	8.880	9.392
27	269.0	3.700	9.181
28	282.0	19.240	9.541
29	287.0	7.400	9.467
30	291.0	5.920	9.349
31	298.0	10.360	9.381
32	301.5	5.180	9.250

33	310.0	12.580	9.351
34	315.0	7.400	9.294
35	319.5	6.660	9.218
36	325.5	8.880	9.209
37	333.5	11.840	9.280
38	344.0	15.540	9.445
39	349.0	7.400	9.392
40	355.0	8.880	9.380
41	360.5	8.140	9.349
42	369.0	12.580	9.426
43	375.5	9.620	9.431
44	379.0	5.180	9.334
45	386.0	10.360	9.357
46	391.5	8.140	9.330
47	400.5	13.320	9.415
48	406.0	8.140	9.389
49	410.5	6.660	9.333
50	418.0	11.100	9.368
51	423.0	7.400	9.330
52	432.0	13.320	9.407
53	436.5	6.660	9.355
54	440.5	5.920	9.291
55	448.0	11.100	9.324
56	454.0	8.880	9.316
57	464.0	14.800	9.412
58	470.0	8.880	9.403
59	474.0	5.920	9.344
60	479.5	8.140	9.324
61	485.5	8.880	9.317
62	494.5	13.320	9.381
63	500.5	8.880	9.373
64	505.5	7.400	9.343
65	512.0	9.620	9.347
66	522.0	14.800	9.429

67	527.0	7.400	9.399
68	532.0	7.400	9.370
69	535.5	5.180	9.309
70	544.0	12.580	9.356
71	551.5	11.100	9.380
72	557.5	8.880	9.373
73	563.0	8.140	9.356
74	570.0	10.360	9.370
75	575.5	8.140	9.354
76	583.0	11.100	9.377
77	587.5	6.660	9.341
78	594.0	9.620	9.345
79	599.5	8.140	9.330
80	607.0	11.100	9.352
81	614.5	11.100	9.373
82	620.0	8.140	9.358
83	628.0	11.840	9.388
84	632.0	5.920	9.347
85	638.0	8.880	9.341
86	646.0	11.840	9.370
87	649.5	5.180	9.322
88	656.0	9.620	9.326
89	660.5	6.660	9.296
90	665.5	7.400	9.275
91	678.0	18.500	9.376
92	681.0	4.440	9.322
93	686.0	7.400	9.302
94	693.5	11.100	9.321
95	699.0	8.140	9.308
96	707.0	11.840	9.335
97	713.0	8.880	9.330
98	719.0	8.880	9.326
99	725.0	8.880	9.321
100	729.0	5.920	9.287
101	738.0	13.320	9.327
102	744.0	8.880	9.323
103	748.5	6.660	9.297
104	753.5	7.400	9.278
105	760.5	10.360	9.289
106	769.0	12.580	9.320
107	773.0	5.920	9.288
108	779.0	8.880	9.284
109	783.5	6.660	9.260
110	790.0	9.620	9.263
111	797.5	11.100	9.280
112	803.5	8.880	9.276
113	808.5	7.400	9.260
114	816.0	11.100	9.276
115	818.5	3.700	9.227
116	829.0	15.540	9.282
117	832.5	5.180	9.247
118	839.0	9.620	9.250
119	844.5	8.140	9.241
120	849.5	7.400	9.225
121	861.0	17.020	9.290
122	864.5	5.180	9.256
123	870.5	8.880	9.253
124	877.5	10.360	9.262
125	881.0	5.180	9.229
126	891.0	14.800	9.274
127	896.0	7.400	9.259
128	899.5	5.180	9.227
129	903.5	5.920	9.201
130	911.0	11.100	9.216
131	920.0	13.320	9.247
132	925.0	7.400	9.233
133	930.0	7.400	9.219
134	937.0	10.360	9.228
135	942.5	8.140	9.220
136	950.5	11.840	9.239
137	955.5	7.400	9.226
138	959.5	5.920	9.202
139	965.0	8.140	9.194
140	970.5	8.140	9.187
141	975.5	7.400	9.174
142	984.0	12.580	9.198
143	991.0	10.360	9.206
144	997.5	9.620	9.209
145	1000.5	4.440	9.176
146	1007.0	9.620	9.179
147	1013.5	9.620	9.182
148	1020.0	9.620	9.185
149	1024.5	6.660	9.168
150	1028.0	5.180	9.141
151	1034.0	8.880	9.140
152	1043.0	13.320	9.167
153	1048.5	8.140	9.161
154	1054.0	8.140	9.154
155	1058.0	5.920	9.133
156	1064.0	8.880	9.131
157	1073.5	14.060	9.163
158	1079.0	8.140	9.156
159	1085.0	8.880	9.155
160	1088.0	4.440	9.125

161	1093.0	7.400	9.114
162	1103.0	14.800	9.150
163	1108.5	8.140	9.143
164	1113.5	7.400	9.133
165	1118.0	6.660	9.118
166	1123.5	8.140	9.112
167	1131.5	11.840	9.128
168	1138.0	9.620	9.131
169	1142.5	6.660	9.116
170	1147.5	7.400	9.106
171	1153.0	8.140	9.101
172	1158.5	8.140	9.095
173	1167.0	12.580	9.115
174	1173.5	9.620	9.118
175	1176.5	4.440	9.091
176	1182.0	8.140	9.086
177	1189.0	10.360	9.093
178	1196.5	11.100	9.105
179	1203.0	9.620	9.107
180	1208.0	7.400	9.098
181	1212.0	5.920	9.080
182	1217.5	8.140	9.075
183	1226.0	12.580	9.094
184	1232.5	9.620	9.097
185	1236.5	5.920	9.080
186	1240.0	5.180	9.059
187	1248.0	11.840	9.074
188	1256.5	12.580	9.093
189	1262.0	8.140	9.088
190	1267.0	7.400	9.079
191	1270.5	5.180	9.058
192	1276.0	8.140	9.053
193	1286.5	15.540	9.087
194	1292.0	8.140	9.082
195	1297.5	8.140	9.077
196	1301.0	5.180	9.057
197	1305.5	6.660	9.045
198	1314.0	12.580	9.063
199	1320.5	9.620	9.066
200	1326.0	8.140	9.061
201	1330.0	5.920	9.046
202	1335.0	7.400	9.038
203	1342.5	11.100	9.048
204	1351.0	12.580	9.065
205	1354.5	5.180	9.046
206	1360.0	8.140	9.042
207	1364.5	6.660	9.030
208	1372.5	11.840	9.044
209	1380.0	11.100	9.054
210	1385.0	7.400	9.046
211	1389.5	6.660	9.034
212	1393.5	5.920	9.020
213	1399.5	8.880	9.019
214	1409.5	14.800	9.046
215	1413.5	5.920	9.031
216	1419.5	8.880	9.031
217	1422.5	4.440	9.010
218	1429.0	9.620	9.012
219	1440.5	17.020	9.049
220	1443.0	3.700	9.025
221	1449.0	8.880	9.024
222	1454.0	7.400	9.017
223	1458.0	5.920	9.003
224	1469.5	17.020	9.039
225	1473.0	5.180	9.021
226	1478.5	8.140	9.018
227	1483.0	6.660	9.007
228	1487.5	6.660	8.997
229	1495.0	11.100	9.006
230	1503.0	11.840	9.018
231	1509.0	8.880	9.018
232	1513.0	5.920	9.004
233	1516.5	5.180	8.988

Number of test	Average speed of sphere of each test (cm/s)
1	8.852
2	9.046
3	8.988
Average	8.962

We test the speed by tracking 3 times to get the average speed that we need for further calculations.

Chapter 5

Result

According to the experiment, the properties that we use are :

Speed of sphere	0.08962 m/s
Mass of sphere	16.3 g
Diameter of sphere	15 mm.
Density of oil engine	0.895 g/cm^3
Expected viscosity of oil engine (at 40 degrees celsius)	155.6 cSt

Figure 5.1 sphere weight 16.3g

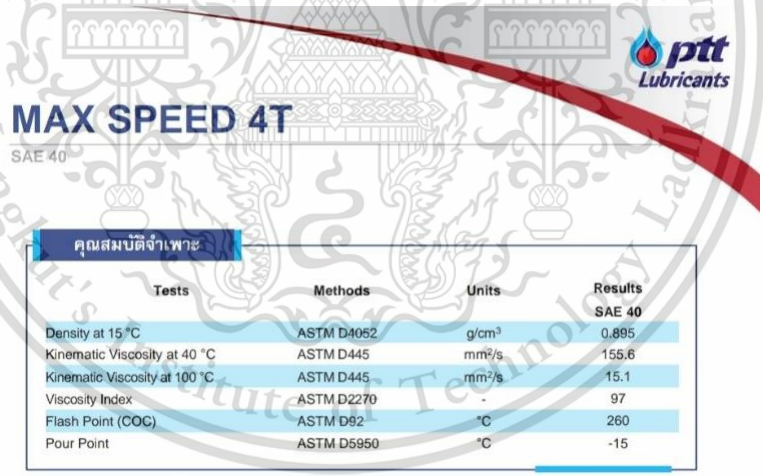


Figure 5.2 specification catalog

Convert cm/s to (m/s)

$$\frac{8.962}{100} = 0.08962$$

Sphere density (kg/m^3)

$$\frac{0.0163}{\frac{4}{3}\pi(\frac{0.015}{2})3} = 9224$$

Dynamic viscosity (Pa*s)

$$\frac{9.81 \cdot 0.015^2 \cdot (9224 - 895)}{18 \cdot 0.08962} = 11.38$$

Kinematic viscosity (m^2/s)

$$\frac{11.385}{895} = 0.01272$$

Convert to (cSt)

$$0.01272 \cdot 10^6 = 12{,}720$$

Figure 5.3 calculate kinematic viscosity in the program

Chapter6
Conclusion

In this thesis, the development of OpenCV and Deep learning is successfully done with the usage in mechanical engineering calculation by training the deep learning model to track the falling sphere in order to calculate the viscosity of the fluid.

In the training model, the Tensorflow&Keras method is not successful, this happens because the dataset might not be good enough for Tensorflow&Keras, and the amount of the given dataset is relatively few compared to the other model training for object detection. Therefore, YOLO method is successfully done because its library was made specifically for object detection so that it could be optimized effectively with our dataset.

The result of the experiment which is speed from the tracking is reasonable due to the scales of the experimental tube. Therefore, the viscosity from the calculation is not as expected.

The error might occur due to the size of the testing tube should be larger than the typical experimental tube because with this size assumption the fluid has been pressed and the shear stress would be irregular. And there are small bubble along the tube, which could affect the density of the fluid.

Bibliography

[1] Hibbler, RC 2017, *Fluid Mechanics*, 2nd edn, Pearson, London.

[2] Gerhart, PM, Gerhart, AL & Hochstein, JI 2015, *Munson, Young and Okiishi's Fundamentals of Fluid Mechanics,* 8th edn, Wiley, Hoboken, NJ.

[3] Pritchard, PJ 2011, *Fox and McDonald's Introduction to Fluid Mechanics,* 8th edn, Wiley, Hoboken, NJ.

[4] IsscoThai Technologies 2019, *Laminar Flow Cabinet,* IsscoThai Technologies, viewed 18 November 2021, <https://www.isscothai.com/learning-center/laminar-flow-cabinet/>.

[5] Khan Academy 2021, *What is Bernoulli's equation?*, Khan Academy, viewed 18 November 2021, <https://www.khanacademy.org/science/physics/fluids/fluid-dynamics/a/what-is-bernoullis-equation>.

[6] Rosebrock, A 2021, *OpenCV Edge Detection (cv2.Canny)*, PyImageSearch, viewed 19 November 2021, < https://www.pyimagesearch.com/2021/05/12/opencv-edge-detection-cv2-canny/>.

[7] Ponnusamy, A 2018, *YOLO Object Detection with OpenCV and Python,* Medium, viewed 19 November 2021, < https://towardsdatascience.com/yolo-object-detection-with-opencv-and-python-21e50ac599e9>.

[8] Science Facts 2021, *Stokes' Law,* Science Facts, viewed 19 November 2021, <

https://www.sciencefacts.net/stokes-law.html>.

[9] Rosebrock, A 2016, *Practical Python and OpenCV: An Introductory, Example Driven Guide to Image Processing and Computer Vision,* 3rd edn, https://www.pyimagesearch.com/practical-python-opencv/.

[10] Hosmer, C 2018, *Defending IoT Infrastructures with the Raspberry Pi: Monitoring and Detecting Nefarious Behavior in Real Time,* Apress, New York, NY.

[11] Pajankar, A 2020, *Raspberry Pi Computer Vision Programming: Design and implement computer vision applications with Raspberry Pi, OpenCV, and Python 3,* 2nd edn, Packt, Birmingham.

[12] Aminfar, A 2015, *Computer Vision in Fluid Mechanics,* Master's Thesis, University of California, Riverside.

[13] Forsyth, DA & Ponce, J 2002, *Computer Vision: A Modern Approach,* 2nd edn, Pearson, London.