

**PARAMETERIZATION OF RIVER WATER FLOW USING
IMAGES AND VIDEO-BASED TECHNIQUES**

**STUDENT NAME
NG JIE HAO**

**BACHELOR OF APPLIED SCIENCE IN DATA ANALYTICS
WITH HONOURS
UNIVERSITI MALAYSIA PAHANG**

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : NG JIE HAO
Date of Birth : 26 APRIL 1999
Title : PARAMETERIZATION OF RIVER WATER FLOW
USING IMAGES AND VIDEO-BASED TECHNIQUES
Academic Session : 2022/2023

I declare that this thesis is classified as:

- ☐ CONFIDENTIAL (Contains confidential information under the Official Secret Act 1997)*
☐ RESTRICTED (Contains restricted information as specified by the organization where research was done)*
☒ OPEN ACCESS I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:



(Student's Signature)

990426-01-5759
New IC/Passport Number
Date: 21st July 2023



(Academic Tutor's Signature)

DR. MOHD KHAIRUL BAZLI
BIN MOHD AZIZ
Name of Academic Tutor
Date: 21st July 2023



(Industry Coach's Signature)

SR. MOHD RADHIE BIN MOHD SALLEH
Name of Industry Coach
Date: 21st July 2023



SUPERVISOR'S DECLARATION

We hereby declare that we have checked this project report, and in our opinion, this final report of Data Science Project is adequate in terms of scope and quality for the award of the Bachelor of Applied Science in Data Analytics with Honours.

(Academic Tutor's Signature)

Full Name : DR. MOHD KHAIRUL BAZLI BIN MOHD AZIZ
Position : Seniors Lecturer
Date : 21st July 2023

(Industry Coach's Signature)

Full Name : SR. MOHD RADHIE BIN MOHD SALLEH
Position : SUPERVISOR
Date : 21st July 2023



STUDENT'S DECLARATION

I hereby declare that the work in this Data Science Project I Report is based on my original work.

(Student's Signature)

Full Name : NG JIE HAO

ID Number : SD20036

Date : 21st July 2023

PARAMETERIZATION OF RIVER WATER FLOW USING IMAGES AND
VIDEO-BASED TECHNIQUES

NG JIE HAO

Data Science Project Report submitted in fulfilment of the requirements
for the award of the degree of
Bachelor of Applied Science in Data Analytics with Honours

Centre for Mathematical Sciences
UNIVERSITI MALAYSIA PAHANG

JULY 2023

ACKNOWLEDGEMENT

First and foremost, I would like to thank my industry coach, Sr. Mohd Radhie Bin Mohd Salleh, of the I Net Spatial company, for trusting me with the responsibility of completing the project. Even though this project carries may be heavy, but rest assured that I will try my best to finish the project. The challenges that this project possesses will also ensure that our skills and knowledge will never ending learning.

I would also like to sincerely thank my academic tutor, Dr.Mohd Khairul Bazli Bin Mohd Aziz, for his support, guidance, and patience. Most importantly, he has always had the patience to hear my questions and give me suggestions. It has been a great pleasure and honour to have him as my academic tutor.

I must thank all of my friends who have collaborated with me on this project. Special thanks to my classmates for their assistance and support, which the project particularly requires.

Finally, I am grateful to everyone with whom I have had the pleasure of working on these projects. I am incredibly grateful to my family members, whose encouragement me in whatever I pursue.

ABSTRAK

Di Malaysia, salah satu bencana alam yang paling kerap melanda negara ini ialah banjir, terutamanya semasa musim monsun dari Oktober hingga Mac setiap tahun. Banjir sungai biasanya disebabkan oleh hujan lebat dan limpahan air ke daratan. Oleh itu, aliran air dan paras air dalam saluran air semula jadi telah menjadi penting disebabkan perubahan iklim kerana frekuensi hujan lebat meningkat, yang mengakibatkan banjir. Objektif kami adalah menggunakan sistem pengukuran halaju aliran air, yang merupakan alat berdasarkan imej untuk menganggarkan secara automatik halaju aliran. Sistem pengukuran halaju permukaan air menggunakan teknologi berdasarkan imej dengan kaedah pelacakan zarah (PTV) untuk menganggarkan halaju aliran air serta mengira aliran. Kaedah ini telah digunakan untuk mengenal pasti pemarameteran aliran air menggunakan imej dan video yang dikumpul dengan Pesawat Terbang Tanpa Awak (UAV). Dalam kajian ini, kami akan mengkaji penyelidikan yang menggunakan teknik penganggarkan halaju aliran air berdasarkan imej dan video. Tujuan utama kajian ini adalah untuk memberi tumpuan kepada pemarameteran aliran air sungai di sungai tertentu. Secara ringkasnya, kajian ini menyimpulkan dengan teknik berdasarkan imej dan video untuk menganggarkan halaju aliran air sungai.

Kata kunci: Teknik berdasarkan imej dan video, halaju aliran air, pelepasan, banjir.

ABSTRACT

In Malaysia, one of the most frequent natural disasters affecting the nation is flooding, particularly during the monsoon season from October to March every year. River floods are often caused by heavy rainfall and the overflowing of water onto land. Therefore, the water flow and water level in natural watercourses has become important due to climate change because the frequency of heavy rainfall has increased, resulting in floods. Our objective is to used water flow velocity measurement system, which are the image-based tool to automatically estimate flow velocities. The water surface velocity measurement system applies image based velocimetry with the particle tracking velocimetry (PTV) method to estimate water flow velocity in addition to calculating the discharge. This method was used to identify the parameterization of water flow with images and video collected with an Unmanned Aerial Vehicles (UAV). In this study, we will review the research that uses water flow velocity estimation techniques based on image and video based techniques. The main purpose of this study is to focuses on the parameterization of river water flow in certain river. In short, this study concludes with an image and video based technique to estimate the river's water flow velocity.

Keywords: Image and video based technique, water flow velocity, discharge, flood

TABLE OF CONTENT

TITLE PAGE	I
DECLARATION	II
ACKNOWLEDGEMENTS	VI
ABSTRAK	VII
ABSTRACT	VIII
TABLE OF CONTENT	IX
LIST OF TABLES	XIII
LIST OF FIGURES	XIV
LIST OF SYMBOLS	XVI
LIST OF ABBREVIATIONS	XVII
LIST OF APPENDICES	XVIII
CHAPTER 1 INTRODUCTION	1
1.1 Research Background	1
1.2 Problem Statement	2
1.3 Research Questions	2
1.4 Research Objectives	2

1.5	Research Scopes	3
1.6	Significance of study	3
1.7	Thesis Organization	3
CHAPTER 2 LITERATURE REVIEW		5
2.1	Introduction	5
2.2	Causes of Floods	5
2.3	Parameterization of river water flow	6
2.4	Unmanned Aerial Vehicles (UAV)	9
2.5	Image Processing	10
2.5.1	Particle Image Velocimetry (PIV)	10
2.5.2	Particle tracking velocimetry (PTV)	11
2.5.3	Optical Flow	12
2.5.4	Space Time Image Velocimetry (STIV)	12
2.5.5	Large Scale Particle Image Velocimetry (LSPIV)	13
2.6	Application of FlowVelo Tool	13
2.7	Summary	16
CHAPTER 3 METHODOLOGY		20
3.1	Introduction	20
3.2	Research Plan	20
3.3	Data Requirements	22
3.3.1	Area of Interest	23
3.4	Data Collection	24

3.5	Data Understanding	25
3.6	Data Preparation	26
3.7	Water Surface Flow Velocity of FlowVelo tool	26
3.7.1	Frame Preparation	28
3.7.2	Feature Search Area and Pose Estimation	29
3.7.3	Feature Detection and Filtering	29
3.7.4	Feature Tracking	30
3.7.5	Tracking Filtering	31
3.7.6	Velocity Measurement	32
3.8	Modelling	33
3.8.1	Autoregressive Integrated Moving Average (ARIMA)	33
3.8.2	Long Short-Term Memory (LSTM)	35
3.9	Evaluation	38
3.10	Deployment	39
CHAPTER 4 EXPECTED OUTCOMES AND CONCLUSION		40
4.1	Introduction	40
4.2	Data Preparation	40
4.3	Data Pre-processing	41
4.4	Data Analysis	43
4.4.1	GUI Interface	44
4.5	Result Water Flow Velocity	54
4.6	Modelling	59
4.6.1	Data Partitioning	60
4.6.2	Training	60

4.6.3	Result ARIMA and LSTM model	63
4.6.4	Evaluation	64
4.7	Discussion	65
CHAPTER 5 CONCLUSIONS AND RECOMMENDATIONS		67
5.1	Introduction	67
5.2	Summary and Conclusion	67
5.3	Recommendations	68
REFERENCES		69
LIST OF APPENDICES		74

LIST OF TABLES

Table 2.1	Methodology used in previous study	16
Table 4.1	Discharge estimated using flow velocities and cross sections retrieved from UAV data	57
Table 4.2	Errors of surface flow velocity between ARIMA and LSTM	64

LIST OF FIGURES

Figure 2.1	Discharge Measurement	8
Figure 2.2	Unmanned Aerial Vehicles (UAV) drones	9
Figure 3.1	Research Plan	22
Figure 3.2	Melana River	23
Figure 3.3	UAV camera captured the video from I Net Spatial Sdn Bhd	24
Figure 3.4	Data Processing Workflow	27
Figure 3.5	3D point cloud projected into image space	30
Figure 3.6	Tracking feature that have been discovered and filtered (blue circles)	30
Figure 3.7	Long Short-Term Memory (LSTM)	35
Figure 4.1	Image file of each frames	40
Figure 4.2	OpenCV installing	41
Figure 4.3	Import all related libraries	42
Figure 4.4	Main Frame	42
Figure 4.5	Original Image (PNG)	43
Figure 4.6	Gray Scaled Image (PNG)	43
Figure 4.7	Location GCPs image (PNG)	43
Figure 4.8	GUI Interface Co-Registration (FlowVelo tool)	44
Figure 4.9	GUI Main Interface Flow Velocity I (FlowVelo tool)	46
Figure 4.10	GUI Interface Flow Velocity II (FlowVelo tool)	48

Figure 4.11	GUI Interface Flow Velocity III (FlowVelo tool)	51
Figure 4.12	GUI Interface Flow Velocity IV (FlowVelo tool)	53
Figure 4.13	Result Flow Velocity in resultFlowVelo.file.	55
Figure 4.14	Result of tracked feature	55
Figure 4.15	Result of minimum count filter	56
Figure 4.16	Result of minimum distance filter	56
Figure 4.17	Result of maximum distance filter	56
Figure 4.18	Result of steadiness filter	56
Figure 4.19	Result of flow direction filter	57
Figure 4.20	Result of flow angle range filter	57
Figure 4.21	Results of flow velocity after application of statistical (threshold) velocity filtering	58
Figure 4.22	Velocity of tracked particle by each sequence	59
Figure 4.23	Best ARIMA model	60
Figure 4.24	Results of fitting the ARIMA(1, 0, 0) model	61
Figure 4.25	Parameters of LSTM model	62
Figure 4.26	LSTM model trained	62
Figure 4.27	Comparison Actual flow velocity between ARIMA predictions and LSTM predictions	63
Figure 4.28	Comparison actual flow velocity between ARIMA predictions and LSTM predictions	64

LIST OF SYMBOLS

Flow (Q)	Water Flow
Velocity (V)	Water surface velocity
Area (A)	Cross sectional area of the water flow
C	Particle concentration
N	Number of particles
V	Volume of fluid sample
V_x	Velocity in the x directions
V_y	Velocity in the y directions
V_z	Velocity in the z directions
t	Time interval between the frames.

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
2D	Two Dimension
3D	Three Dimension
ROI	Region of Interest
WSV	Water Surface Velocity
PIV	Particle Image Velocimetry
PTV	Particle Tracking Velocimetry
STIV	Space Time Image Velocimetry
LSPIV	Large Scale Particle Image Velocimetry
UAV	Unmanned Aerial Vehicles
SfM	Structure-from-Motion
GCPs	Ground Control Points
OpenCV	Open Source Computer Vision Library
RANSAC	Random Sample Consensus
ADCP	Acoustic Doppler Current Pro
ARIMA	Autoregressive Integrated Moving Average
LSTM	Long Short-term Memory
MAE	Mean Average Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error

LIST OF APPENDICES

Appendix A: Times Series model ARIMA and LSTM for Flow Velocity

CHAPTER 1

INTRODUCTION

1.1 Research Background

Disasters from nature are a global phenomenon and various communities cooperate to respond to these major natural disasters. One of the most common forms of natural disasters is flooding. This is the most common natural disaster in Malaysia, and it includes flash floods, seasonal floods, river floods, and coastal floods, particularly during the monsoon season from October to March. A river water flow monitoring system is essential for measuring water flow and reducing the risk of flood disasters. At the same time, water flow monitoring systems detect floods as early as possible, and governmental organisations and disaster response teams take timely measures to prevent significant damage to resource and save many lives. The primary goal of a river water flow monitoring system is to accurately predict increases in river water flow across time and space, as well as flood impacts.

In this age of technology, image and video based surveillance is becoming a standard gauging station configuration, creating favourable conditions for the development of image-based tool. To minimise costs, the image and video processing system with computer vision and python tool will replace these expensive applications for flow velocity calculating. This system will automatically measure the river water flow to calculate discharge from surface velocity and cross-sectional areas. This programme uses image and frame sequences to track flow velocities. It comprises determining the camera position, extracting the water automatically for feature searching, detecting and filtering particles, and scaling the tracks to flow rates. Therefore, water flow monitoring system is used to estimates the parameterization of river water flow by using particle tracking velocimetry technique. This method can be applied to aerial imagery as well as terrestrial imagery. The parameterization of river water flow using image and video based technique will be discussed in this project.

1.2 Problem Statement

Floods have occurred in certain areas of the world, especially in low-lying areas. These catastrophic events have resulted in the destruction of significant amounts of property and made the community live in discomfort, and caused death (Liu & Chan, 2003). Therefore, this study measures the actual parameterization of river water flow by using image and video processing to calculate discharge from surface velocity and cross-sectional areas on the river.

1.3 Research Questions

- i. What is the parameterization of river water flow?
- ii. Which method is the best for measuring the river water flow surface velocity based on image and video techniques?
- iii. What is the performance of proposed method of river water flow based on image and video techniques?

1.4 Research Objectives

- i. To identify the appropriate parameters to be used to parameterize river water flow based on image and video techniques.
- ii. To develop the best model for measuring the river water flow velocity based on image and video techniques.
- iii. To investigate the performance of the proposed method of river water flow velocity based on image and video techniques.

1.5 Research Scopes

This study will focus on river water in the River Johor. The scope of method studied focuses on image processing to measure the river water flow velocities and calculate discharge from surface velocity and cross-sectional areas. The study will utilize data collected and provided by I-Net Spatial Sdn Bhd. This collaboration with I-Net Spatial Sdn Bhd ensures access to reliable and high-quality data for the research.

1.6 Significance of study

This research will provide new knowledge into image processing to estimate the parameterization of river water flow on the river. Additionally, this study will provide important information for future research that will explore the impact river water flow. The information gathered, such as average surface flow velocity, discharge, and cross-section area, can be used as baseline data for subsequent studies exploring the impacts of river water flow on various aspects, such as ecological health, sediment transport, or flood risk assessment.

1.7 Thesis Organization

This study is divided into five chapters, with Chapter 1 serving as the introduction to the project. It covers the research background, problem statement, research questions, research objectives, research scopes, significance of the study, and thesis organization. Research background provides a comprehensive overview of flooding and describes the technique that will be used to analyse the data. The problem statement describes the challenges faced. Research questions and objectives are the goals that need to be achieved throughout this study, while research scopes explain the data types and the technique will be used. The significance of the study explained the importance of understanding the impacts of river water flow on various aspects.

In the Chapter 2 literature review, causes of floods and parameterization of river water flow are explained in detail. Image processing techniques and the application of the FlowVelo tool have also been included in the contents. Chapter 3 of this project will explain more about the methodology used, data requirements, data collection, modelling, evaluation, and deployment. The data science cycle and workflow of the FlowVelo tool have also been visualised there, so readers can understand more about the process throughout this project. Chapter 4 is dedicated to data pre-processing, data preparation, data analysis, and discussion. This chapter presents the findings derived from the data analysis and facilitates meaningful discussions based on the results obtained. Finally, Chapter 5 presents the conclusion and recommendations of the project. This chapter offers valuable insights and recommendations for further research and potential actions to mitigate the issues identified.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

River is a key natural resource that significantly affects the global carbon and the climate catastrophe (Dobriyal et al., 2017). As a result, rivers are an essential component of the ecosystem and are essential to the life of people, plants and animals (Reyes-Avila et al., 2004). Numerous uses of river water include farming, drinking, and the creation of hydroelectric projects (Dobriyal et al., 2017). The abundance of water wreaks havoc and instills terror through the occurrence of floods (Reyes-Avila et al., 2004). Water management includes measuring water levels, flow, water quality, and other physical characteristics as part of environmental surveillance (De Oliveira Fleury et al., 2020). Water flow and water velocity are fundamental parameters used to characterise the properties of the hydrologic regime (Hauet et al., 2018). As a result, water flow velocity are necessary parameters for nature-based solutions, flood warning systems and plans for the management of sustainable water resources (Griesbaum et al. 2017).

2.2 Causes of Floods

Floods are the most deadly and devastating of all natural catastrophes, and they are the most prevalent and far-reaching natural disasters (Di et al., 2017). Floods have caused the highest number of casualties compared to other disasters (Mohanty et al., 2020). A river flood happens when a river fills up above its capacity, overflows the river bank, and runs into low-lying lands. Heavy rains and high tides are the two most frequent causes of river flooding (Mohanty et al., 2020). According to Hirschboeck et al. (2000), floods happen when a watershed receives an overwhelming volume of precipitation that surpasses the capacity of internal storage reservoirs within the basin and the drainage network to handle. In Malaysia, flooding may be divided into two main categories: seasonal monsoon floods, which are marked

by low intensity, extended rainfall, and unpredictable flash floods, which are marked by intense, brief rainfall (Lee et al., 2008) . Floods in rivers commonly take place within floodplains or wash areas because the water flows over the natural banks or constructed embankments when stream capacity is exceeded (Di et al., 2017). The rivers might gradually rise and stay high for several weeks (Lee et al., 2008). Multiple interconnected rivers may experience simultaneous flood peaks, which can cause significant flooding (Lee et al., 2008). Flooding has become more unpredictable and frequent over the last decade (Douglas et al., 2008). According to Liu & Chan (2003) , the public needs to be made more aware of warning systems and educated on how to respond appropriately because even the most advanced warning systems would be useless if people did not heed the warnings. The utilization of a flood monitoring system offers a fortunate advantage in effectively managing disaster risk assessment, issuing flood warnings, and facilitating water resource planning (Liu & Chan 2003).

2.3 Parameterization of river water flow

Parameterization of river water flow is known as a process of developing mathematical models that depict the flow of water in a river based on various factors (Zaimes et al., 2021). Water flow measurements are a crucial component of hydrologic regime for flood warning systems, adoption of natural-based solutions and protection structures (Jiwani et al., n.d., 2017). The hydrologic regime features are described using two essential parameters: water surface velocity and water discharge (Zaimes et al., 2021). Surface water flow is the constant flow of water in runoff or open channels (Mansour et al., 2015). The flow is frequently measured in terms of discharge, which is the volume of water or the flow rate across a channel's cross section in a certain amount of time (Sanders, 1998). According to (Dobriyal et al., 2017), river water flow monitoring may be useful in determining the optimum levels at which to manage water resources sustainably in the face of climate change.

Water Surface Velocity (WSV) estimation is one of the parameters of water flow in water management (Sirenden et al., 2022). It is helpful to carry out empirical rating curve calibration (Rozos et al., 2020). This curve can be used to measure the river discharge with

remote sensing. According to Gleason & Durand, (2020), using the Acoustic Doppler Current Profiler (ADCP) is the most sincerely way to estimate WFV. However, this method is highly risky since a human operator must be close to the water to do measurement for some rivers without bridges (Rozos et al., 2020). This method has inadequate spatial coverage in addition to safety reasons (Trieu et al., 2021a). Radars and cameras are both used in various non-intrusive techniques (Fulton et al., 2020). While camera or optic base methods were less expensive, more flexible, and could be installed either in a fixed, permanent place or on a UAV, radar technology was more expensive, time-consuming, and required skilled staff (Tauro et al., 2018). The depth-averaged velocity (DAV) is not the same as WSV, which is required to discharge (Gleason & Durand, 2020). According to Genc et al., (2015), the work shows that WSV and DAV has a liner relationship.

Calculating river water flow involves solve an equation that looks at the relationship between a number of different factors, such as the cross-sectional area, river length, and water velocity of the river. The ways to measure water flow is represented as following equation:

$$Q = A \times V \quad (2.1)$$

where:

(Q) = Water Flow (Discharge)

(V) = Water surface velocity

(A) = Cross sectional area of the water flow

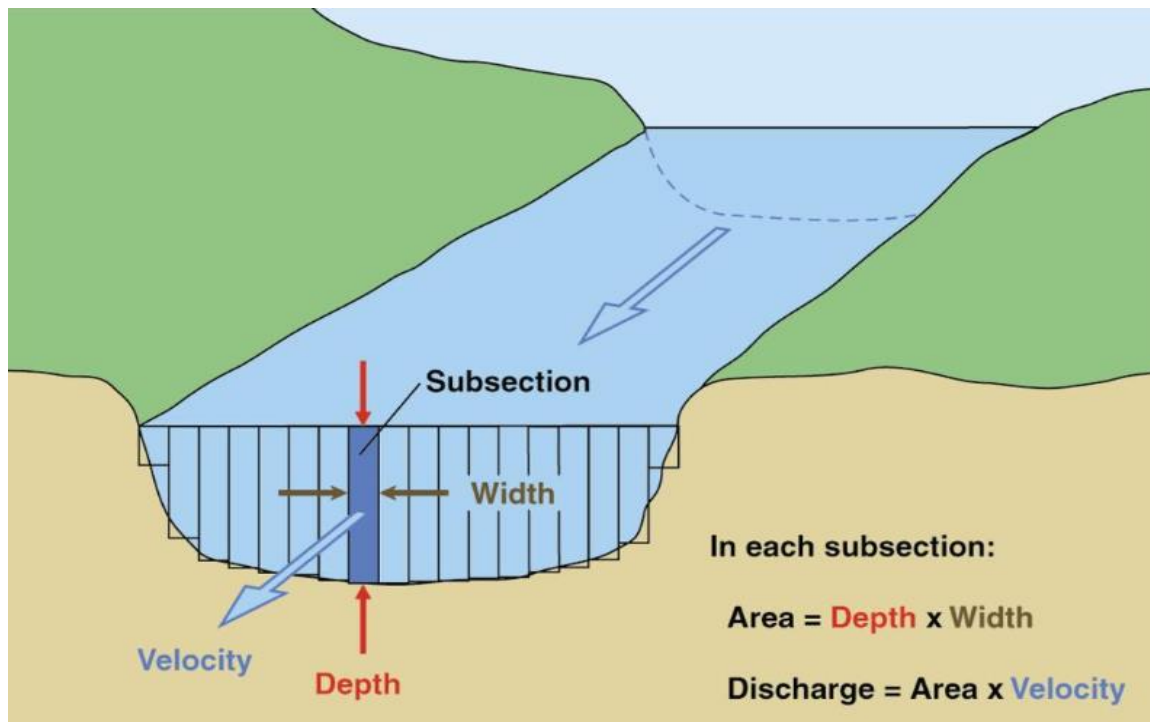


Figure 2.1: Discharge Measurement

Source: (Koutalakis & Zaimis, 2022a)

Cross sectional area is the river width multiplied by average water depth. The calculated velocity coefficients are multiplied by the averaged surface flow velocity values to account for depth-averaged velocities. The depth-averaged velocity is multiplied by the cross-section area to estimate discharge. Research should compute the average cross-sectional area for aggregate the findings, and then divide by two to get the average cross-sectional area for the study river reach.

2.4 Unmanned Aerial Vehicles

Unmanned Aerial Vehicles (UAV), also referred to as drones, are flying machines that can be operated by a person or by autonomous onboard workstations (Mittal et al., 2020). Based on Figure 2.2, it is an image of Unmanned Aerial Vehicles.



Figure 2.2: Unmanned Aerial Vehicles (UAV) drones

Source: (A.Savin et al., 2017)

Drones can gather on-demand images from low-altitude airspace for a variety of uses, including emergency item delivery, border patrols, emergency rescue and visual crowd surveillance (Mittal et al., 2020). The commercial and recreational use of drones is gaining popularity nowadays (Chen et al., 2018). More and more, drones are being utilized as robotics platforms. One advantage of employing a drone instead of another robot, like a wheeled robot able to go closer to the items in the area in order to detect anything there. (Mittal et al., 2020). UAV and drone images are very dissimilar from images obtained with a conventional camera (Sun et al., 2020).

2.5 Image Processing

Image processing is one of the technologies that is expanding quickly today (Pumo et al., 2021). It is a technique of applying operations on the image to produce an enhanced image and extract useful information from it (Tauro et al., 2018). It is a form of processing where an image is used as the input and the output may be an image feature related to that image (Tauro et al., 2018). These techniques provide quick, inexpensive, real-time observations under any flow conditions, with excellent spatial resolution (Pumo et al., 2021). According to Tauro et al., (2016), an optical technique called image-based velocimetry construct maps of the surface water velocity by using video that captured by a camera. Surface tracers are detected and monitored to determine surface velocity vectors and distances (Tauro, 2016). Large-scale particle image velocimetry (LSPIV) has been extensively used for measuring flow velocity (Admiraal et al., 2004). According to these methods were initially created for controlled laboratory research under particle image velocimetry (PIV) and particle tracking velocimetry (PTV) (Huang et al., 2018).

2.5.1 Particle Image Velocimetry (PIV)

Particle Image Velocimetry (PIV) known as an imaging method is used to identify the velocity distribution (Fujita et al. 1998). According to Fujita et al. (1998), the particle image velocimetry (PIV) method is high accurate water flow surface velocity. The most widely used approach is PIV to measure surface velocity, which measures the velocity of particles in a stream using an image analysis method (Naves et al., 2020). There are some complete image-based remote sensing system for the study of river flow measurement (Fujita et al., 1998). According to Eltner et al. (2021), particle tracking velocimetry (PTV) and PIV are two methods frequently used to determine WSV by using camera. Moving particles are used in both of these methods and their variations to estimate WSV Eltner et al. (2021). PIV tracks a collection of particles, whereas PTV tracks a single particle or item on the surface of the river Eltner et al. (2021). Optical Flow (OF) based algorithms also estimate WSV using moving particle or object, which employ correlation-based algorithms (H. Wu et al., 2019). According to (Eltner et al., 2021b), replace the requirement for particles by using PIV and PTV to measure WSV

from the thermal signature of river flow. The method has been used in diverse areas, including river discharge measurements (Coz et al., 2014).

2.5.2 Particle Tracking Velocimetry (PTV)

Particle tracking velocimetry (PTV) is a non-invasive technique that uses the identification and determination of specific particle tracers to produce velocity vectors that are randomly localised (Lloyd et al., 1995). Application of PTV is recorded in numerous environmental studies and is intended for low seeding density flows (Kimura et al., 2011). PTV has been extensively used to study unsteady flows because velocities are generally only approximated for individual particles and no inferences are usually made about the relative positions of neighbouring particles (LI et al., 2013). Many algorithms have been used to identify and monitor individual particle detection and tracking, including cross-correlation, heuristics and relaxation bases on previous understanding of the flow (Aleixo et al., 2011). In the beginning, machine learning algorithms were already used to enhance PTV performance (Grant & Pan, 1995). However, previous attempts to use shallow neural networks to look for particle correspondence are insufficient and have flaws (Pereira et al., 2006). In recent years, contemporary deep learning techniques have been effectively applied to diverse subjects, such as fluid mechanics problems, model reduction issues, deep learning-based PTV, fluid flow simulations, and particle reconstruction of tomographic PTV (Brunton et al., 2020). Additionally, a predictive model is suggested to resolve the particles linking using recurrent neural networks, which significantly enhances the PTV's capability to resolve velocity (Mallery et al., 2020). However, the predictive model in has limited generalizability because each trained model is only tailored for a single flow case (Mallery et al., 2020). There's still a lot that can be done to adjust PTV models so they take full use of deep networks.

2.5.3 Optical Flow

Optical flow is used in a variety of computer vision applications with motion data, including data compression and robotics (Teed & Deng, 2020). According to (Horn & Schunck, 1981) the difference between the 2D projections of the 3D scene onto the picture plane, optical flow is the perceived movement between two images. It is feasible to train a model to forecast the optical flow directly using deep neural networks, where a network contains several hidden levels (Teed & Deng, 2020). Optical flow analysis is non-intrusive, enables immediate determination of the whole flow field, and is not always dependent on the employment of expensive equipment (Detert et al., 2017). The primary goal of optical flow analysis applied to rivers is to estimate surface velocity. The depth-averaged flow velocities are in turn reflective of the surface velocities (Hauet et al., 2018). In short, surface velocity fields computed using optical analysis techniques to generalisations about flow patterns in general (Tu et al., 2019).

Sparse optical flow is a processing technique that measures the apparent motion of a single portion of the image's pixel when a flow field estimate is produced (Tu et al. 2019). According to Tu et al. (2019), there is a method to getting a sparse flow estimate by using the differential technique, which calculates a pixel displacement in a local region based on pixel intensity.

Dense optical flow is a video processing technique that measures object motion without the use of a tracer (Zach et al., 2007). It is a technique for determining the direction and velocity of moving things by examining their displacement in two subsequent video images (Weinzaepfel et al., 2013).

2.5.4 Space Time Image Velocimetry (STIV)

One of the methods established by (Fujita et al., 2007) is the space-time image velocimetry (STIV). STIV does tracking in one dimension rather than two dimension, which speeds up performance (Q. Wang et al., 2021). STIV generates the imaginary line to analyse water ripple motion and estimate WSV (Fujita et al., 2007). These lines of pixel brightness fluctuation are used to build a space-time frame. According to Fujita et al. (2007) assessed the

precision of STIV, they discovered that it is a method for measuring streamwise velocity distributions more effectively and precisely than LSPIV. It has been successful to apply the standard STIV to a several of flood flow measurements, but it is challenging to assess the STI's quality using the optical gradient tensor approach (Aberle et al., 2017). According to Fujita et al. (2007), STI is a suggested new method that uses an auto-correlation function instead of the original STI's 2-D Fourier Transform. Due of the poor quality of STI, (Q. Wang et al., 2021) suggested a novel denoising approach and integrated it with Fast Fourier Transform (FFT) based STIV to increase accuracy. Convolution Neural Network (CNN) is used to overcome poor STI (Watanabe et al., 2021).

2.5.5 Large Scale Particle Image Velocimetry (LSPIV)

LSPIV was a technique that originally created by (Fujita et al., 1998). This technique monitors features on the water's surface that are formed either by naturally occurring floating particles or by the distortions caused by waves, ripples, breeze, or turbulence (Muste et al., 2008). In shallow-flow circumstances, (Muste et al., 2014) demonstrate that it is possible to extract flow depth using the velocity pattern obtained using LSPIV. The mobile application of LSPIV (large-scale PIV) was created by (Kim et al., 2008) and may be swiftly deployed. In five months, real-time LSPIV system has been developed for river water flow observation (Hauet et al., 2008).

2.6 Application of FlowVelo tool

The water flow velocity measurement created for the application based on the PTV method and feature-based tracking (Chetverikov, 2001). PTV is one of the most widely used method for a variety of flow monitoring in recent years (Chetverikov, 2001). This application used Python and the OpenCV library to run an image processing technique in this investigation to measure water surface velocity as well. This study will be applied PTV for the modelling part to estimate the water flow velocity. The water flow velocity measurement model for computer vision tasks that is more accurate is PTV. This study presents an automated process

utilizing a Python tool to evaluate surface flow velocities in river through particle tracking velocimetry (PTV).

Particle Tracking Velocimetry (PTV) is an essential technique utilized in fluid dynamics research and engineering applications to measure and analyze fluid flow velocities (Tauro et al., 2019). PTV provides valuable insights into the dynamics and behavior of fluid flows, aiding in the design and optimization of various processes and systems (Aleixo et al., 2011). It involves the introduction of tracer particles into a fluid flow and subsequent tracking of their motion to extract velocity information. The particles, typically small solid particles or droplets, are seeded into the flow and illuminated by laser light or other light sources (Tauro et al., 2019). The basic principle of PTV involves the following steps:

Particle seeding is a process that tracer the particles often small solid particles or droplets, are introduced into the fluid flow (Pereira et al., 2006). These particles should have a density close to that of the fluid and be small enough to follow the flow faithfully. The particle concentration can be estimated using equation:

$$C = N/V \quad (2.2)$$

Where C is the particle concentration, N is the number of particles, and V is the volume of fluid sample.

Estimating particle concentration is crucial because it provides valuable information for various aspects of PTV analysis. It helps in understanding the spatial distribution of particles within the flow and influences the accuracy of velocity calculations.

Image acquisition is a crucial aspect of the Particle Tracking Velocimetry (PTV) process. It involves illuminating the fluid flow using laser light or other light sources and capturing a series of high-speed images to track the particle motion (Khalid et al., 2019). Selecting the appropriate camera with suitable specifications is essential to ensure accurate and detailed velocity measurements (Vieira Do Nascimento et al., n.d.). In each image frame, the positions of the tracer particles are identified and tracked over time. There are different

algorithms and techniques for particle tracking, and the choice depends on the characteristics of the flow and the particle concentration.

Particle matching is a crucial step in Particle Tracking Velocimetry (PTV) where the positions of particles in subsequent image frames are compared to establish correspondences and track their displacements (Chetverikov, 2001). However, this process can be particularly challenging in scenarios where particles are closely spaced or when occlusions occur due to the complex fluid flow (Mallery et al., 2020). When particles are close together, it becomes difficult to differentiate and accurately associate each particle's position in one frame with its corresponding position in the subsequent frame (Mallery et al., 2020). This proximity can lead to potential errors in particle identification and tracking.

The first step of particle displacement in velocity calculation is to determine the displacement of each particle between consecutive image frames (Tauro, 2016). This is achieved by comparing the particle's position in the initial frame with its position in the subsequent frame (Pereira et al., 2006). The displacement can be calculated as:

$$x = x(t_2) - x(t_1), y = y(t_2) - y(t_1), z = z(t_2) - z(t_1) \quad (2.3)$$

where x , y , and z represent the displacement in the x , y , and z directions, respectively. The $x(t_1)$, $y(t_1)$ and $z(t_1)$ denote the initial position coordinates of the particle, while $x(t_2)$, $y(t_2)$ and $z(t_2)$ represent the corresponding position coordinates in the subsequent frame.

Time Interval: To calculate the velocity, it is necessary to determine the time interval between consecutive frames. This can be achieved using the frame rate of the camera or by synchronizing the image acquisition system with a known time reference.

Velocity Calculation:

$$V_x = \frac{x}{t}, V_y = \frac{y}{t}, V_z = \frac{z}{t} \quad (2.4)$$

Where V_x , V_y , and V_z indicate the velocity in the x , y , and z directions, respectively, and t represents the time interval between the frames.

By calculating the velocities of multiple particles using this formula, researchers can obtain information about the flow dynamics, such as the velocity field, turbulence, and vorticity (Genc et al., 2015).

Overall, the velocity calculation in PTV involves dividing the particle's displacement by the time interval to determine its velocity in each direction (Hauet et al., 2018). These velocity measurements contribute to a comprehensive analysis of fluid flow behavior and play a significant role in numerous scientific and engineering applications.

2.7 Summary

Table 2.1 Methodology used in previous study

No	Title of Research	Author	Result / Implementation
1	Method for Measuring the Surface Velocity Field of a River Using Image Acquired by a Moving Drone	(Yu K & Lee J, 2023)	<ul style="list-style-type: none"> They using GPS data, the moving directions of the drone and of the image were calculated, and each image point was converted into a physical UTM (Universal Transverse Mercator) coordinate system
2	River Flow Measurements Utilizing UAV-Based Surface Velocimetry and Bathymetry Coupled with Sonar	(Koutalakis & Zaimes, 2022b)	<ul style="list-style-type: none"> This study presents a non-contact methodology to estimate streamflow based on data collected from Unoccupied Aerial Systems (UAS). Surface velocity is estimated by using image-based velocimetry software while river bathymetry is measured with a floating sonar,

No	Title of Research	Author	Result / Implementation
			tethered like a pendulum to the UAV
3	Two-Dimensional Space-Time Image Velocimetry for Surface Flow Field of Mountain Rivers Based on UAV video	(Han et al., 2021)	<ul style="list-style-type: none"> • Two-dimensional STIV is proposed to obtain the magnitude and direction of the velocity automatically. • The method determines the velocity direction by rotating the search line to find the space time image is reliable.
4	Photogrammetry for Free Surface Flow Velocity Measurement: From Laboratory to Field Measurements	(Trieu et al., 2021b)	<ul style="list-style-type: none"> • There are two photogrammetry-based PTV algorithms presented in this study regarding different types of surface particles employed on the water flow.
5	Flow velocity and discharge measurement in rivers using terrestrial and unmanned-aerial-vehicle imagery	(Eltner et al., 2020)	<ul style="list-style-type: none"> • The method is based on particle-tracking velocimetry (PTV) and comprises an automatic definition of the search area for particles to track. • The method can be applied to different perspectives, including terrestrial and aerial imagery.
6	Measuring Surface Velocity of Water Flow by Dense Optical Flow Method	(Wu et al., 2019)	<ul style="list-style-type: none"> • Dense optical flow method was used to process the water flow video to get estimated surface velocity of water flow. • The results indicate that the method had a good accuracy.

No	Title of Research	Author	Result / Implementation
			<ul style="list-style-type: none"> • Is a feasible and promising approach to analyzing the surface velocity distribution of water flow.
7	Optical Flow For Image-Based River Velocity Estimation	(Khalid et al., 2019)	<ul style="list-style-type: none"> • Novel motion estimation technique for image-based river velocimetry called optical flow. • Optical flow provided good results without any additional terms.
8	Efficient and accurate estimation of water surface velocity in STIV	(Fujita et al., 2019)	<ul style="list-style-type: none"> • Improvement in determining accurate surface velocity from space time image (STI) used in STIV. • Novel technique was developed that utilizes the two dimensional auto-correlation function of the image intensity.
9	PTV-Stream: A simplified particle tracking velocimetry framework for stream surface flow monitoring	(Tauro et al., 2019)	<ul style="list-style-type: none"> • PTV-Stream offers a versatile alternative to cross-correlation-based PTV by affording the identification and tracking of features of any shape transiting in the field of view.
10	Video Processing Based Water Surface Velocity Measurement Using Spatial Cross Correlation Technique	(di Cristo, 2011)	<ul style="list-style-type: none"> • Noncontact velocity measurement system based on LSPIV. • Image based method uses cross-correlation as the similarity index to calculated for patterns enclosed in a Region of interest.

No	Title of Research	Author	Result / Implementation
			<ul style="list-style-type: none"> The calculation of the surface water velocity using the image based technique is a reliable, accurate and instantaneous measurement approach.
11	Applications of Image Recognition for Real-Time Water level and Surface Velocity	(Lin et al., 2013)	<ul style="list-style-type: none"> Real-time water monitoring system using the image processing techniques of image binarization, character recognition and water line detection. Surface velocity recognition by proposed system apply the (Particle Image Velocimetry) PIV to obtain the water surface velocity.

According to Table 2.1, different methodologies were used in the previous study to river water flow measurement. To conclude the literature study, the river water flow should be adopted in an image-based velocimetry system in order to enhance the efficiency and accuracy of estimating the water surface velocity. Image-based velocimetry systems apply Particle Image Velocimetry (PTV) to obtain water surface velocity.

This study will focus on the parameterization of river water flow by using images and video-based techniques. In addition, river water flow is measured by using image-based velocimetry for estimation and measurement as its basis.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter covers the methodologies used and our research plan. The chapter demonstrates image-based water velocity measurement process to estimate the water flow velocity. Based on the water flow velocity and cross-sectional areas, we can estimate discharge on the river.

3.2 Research Plan

In this subtopic, the Data Science research technique will be applied to meet the objective of this study. Business understanding will be the initial phase of the research plan. The business understanding will help clarify the problem and define how the proposed architecture will achieve its goal. In this initial phase, the recommended architecture together with a problem description and objectives will be defined.

Next, data requirement is the phase with identifying the data required to measure the water flow velocity. The collected data must be public access and meet the data criteria in order to measure the water flow velocity.

Data collection was captured the river by using drones. Each video and image was prepared by following steps: image selection and spilt the image from video. Data understanding involves exploring and examining the collected data to identify patterns and relationships in the data.

After that, the next stage is data preparation. This involves cleaning the data by handling missing values, outliers, or inconsistencies.

The modelling phase is where data science techniques are applied to build models that address the research objectives. Modelling will be created for our image based tool on the PTV method.

Evaluation of the models plays a critical role in assessing their performance and their alignment with the research objectives. It involves the use of various metrics and validation techniques to measure accuracy, precision, recall, and other relevant indicators. This evaluation process helps identify the strengths and weaknesses of the models, enabling necessary refinements.

Once the models have undergone satisfactory evaluation, they are ready to be deployed in real-world scenarios to generate actionable insights and implement solutions. This deployment phase may encompass integrating the models into existing systems, developing dedicated software applications, or creating intuitive dashboards that empower decision-makers to effectively leverage the derived findings.

Lastly, the result will be presented which are water surface velocity measurement. Along with the evaluation methodology of achievement of measurement will be used to determine the success of the proposed of the systems.

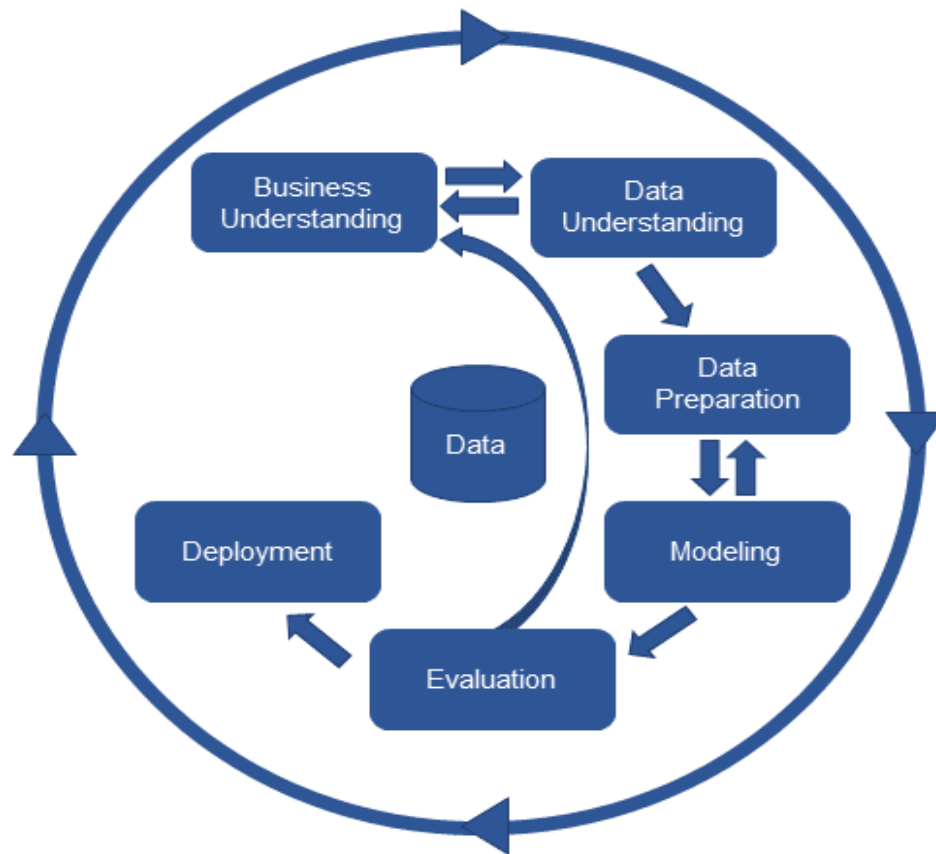


Figure 3.1: Research Plan
Source: (Niakšu, 2015)

3.3 Data Requirements

The goal of this module is to collect video and image dataset and evaluate a system to estimate the water flow velocity. The video dataset will be used are public data. The secondary data is information that has already been gathered and documented by another party. This study will use the secondary data that are provided from I Net Spatial.

3.3.1 Area of Interest

One of the specific river is the Melana River. Melana River is sub-catchments of Skudai River. It stretches approximately 15.9 kilometers, making it one of the notable contributors to the overall length of the river system. Based on Figure 3.2, it shows the image of the Melana River.



Figure 3.2: Melana River

The Skudai River, also known as ‘Sungai Skudai’, is located in Johor, Malaysia. Its primary tributary originates from a small creek located within an oil palm plantation in Kg. Sedenak, Kulai, and flows southwards towards the city of Johor Bahru. Eventually, it empties into Danga Bay, Tampoi. The main tributary extended for approximately 43 km, whereas the total length of all tributaries around 308 km. The Skudai River Basin (SRB) covers a land area of 270 km and is divided into 22 sub-catchments. Notable tributaries and sub-catchments include the Danga River (15.3 km), Melana River (15.9 km), Senai River (10.4 km), Anak

Sungai Melana (5.8 km), UTM River (5.3 km), Kempas River (4.8 km), and Sri Sengkang River (8.4 km). This intricate network of streams flows through various towns such as Kulai, Senai, Skudai, and Tampoi.

The locations chosen for the experimental study are Melana River close to the Jalan Tenang in Taman Damai Jaya, Johor Bahru. The approximated river width was 13 m.

3.4 Data Collection

The video dataset was captured the Melana River by using UAV (unmanned-aerial-vehicle) DJI Phantom 4 Pro with a gimbal camera, a remote control and a drone body. Based on Figure 3.3, it shows UAV camera captured the video from I Net Spatial Sdn Bhd.



Figure 3.3: UAV camera captured the video from I Net Spatial Sdn Bhd

In this study, the drones hovered in a medium-high selected location 15 to 20 meter and captured video where they can clearly monitor the river without being obstructed. It is essential to secure the position of the drones when captured video to the target to ensure accurate measurements. The river flow was clean, and the field measurements were done in daylight. The video was captured with a resolution of 1920 x 1080 pixels in using the Phantom 4 Pro camera. There has to be a 3D model of the region of interest (ROI), and ground control points (GCPs) are needed to reference the image data. Both must be covered by the camera's field of view. Geographically referenced video data will be aligned and calibrated using Ground Control Points (GCPs). The video will be divided into images. The UAV imagery was collected to estimate image-based flow velocity estimation.

3.5 Data Understanding

In this subtopic, we will discuss a video dataset provided by I Net Spatial in MOV (QuickTime) file format. The dataset consists of video files, and in order to work with the data effectively, it may be necessary to split the videos into individual frames, which can be treated as images. The MOV file format is commonly used for storing video and audio data. It is a container format that can hold multiple tracks of different types, such as video, audio, and text. In the context of this dataset, the video track contains the visual information that we are interested in analyzing.

To process the video dataset, we can use software that can extract frames from the MOV files. These frames represent individual images captured at specific time intervals within the video. Splitting the videos into frames allows us to work with each image independently and perform various image processing or computer vision tasks to estimate the water flow velocities.

3.6 Data Preparation

For this study, Jupyter Notebook will be utilised for data preparation process. Firstly, the provided data from I Net Spatial will be loaded into Jupyter Notebook. This involves cleaning the data by handling missing values, outliers, or inconsistencies. Data may need to be transformed, normalized, or aggregated to make it suitable for modeling.

3.7 Water Surface Flow Velocity Workflow of FlowVelo tool

The general method for measuring surface flow velocities from UAV and terrestrial video sequences is introduced in this section. In this processing step, the information about FlowVelo tool is explained in detail. The FlowVelo tool is implemented in Python and uses the OpenCV library (Bradski, 2000). The workflow begins with frame preparation, which involves converting the video into individual images. Then, pose estimation comes into play to ensure camera stability by accurately determining its position. Next, the feature search area is used to identify and isolate water areas based on the water level. After that, feature detection comes into action, which helps identify relevant features for tracking. Subsequently, feature filtering is applied to eliminate false feature results. With feature tracking, selected features are continuously tracked across consecutive frames. Track filtering is implemented to refine the resulting velocity tracks. Finally, the velocity retrieval step involves projecting image tracks into object space and filtering out any outliers.

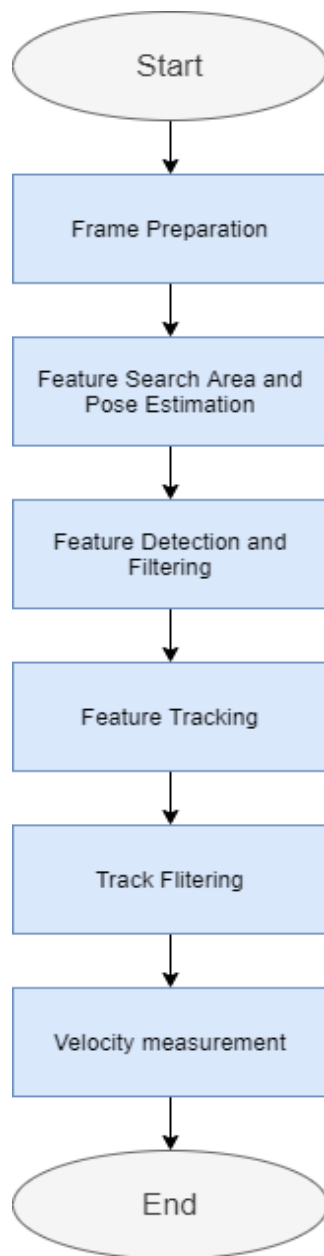


Figure 3.4: Data Processing Workflow

3.7.1 Frame Preparation

Before undergoing data processing, videos are converted into individual frames. Image co-registration is needed, particularly in cases where the camera used for video capture is not stable, such as UAV data. This step involves automatically aligning every frame in the video sequence with the first frame of the same sequence. This step enables consistent analysis and comparison across the frames.

The water region Harris features are considered outliers in this study as their appearance changes across frames (Harris & Stephens, 1988). Any moving features that can still be matched in the water area are filtered during homography parameter estimation. This is due to the points are classified as outliers when fitting a model using RANSAC (random sample consensus) (Foley et al., 1981). Only consistent and reliable connected image points outside the river are kept to compute the homography parameters.

There are five parameters that can be adjusted to control the co-registration of each image in the FlowVelo tool. The first parameter is the maximum number of key points, which determines the number of features obtained in each frame. Increasing this number can enhance matching accuracy and robustness, but it may also result in longer processing times. The second parameter is the number of good matches required between two frames to establish the homography. A higher value improves robustness, but it can lead to processing failures if the expected number of feature matches is not achieved. Additionally, the choice of feature descriptor can be specified for matching purposes. The option to perform repeated feature matching can be selected, which can improve accuracy but may also increase processing time. Lastly, the co-registration process can be applied either to the initial frame or to each subsequent frame in the sequence. By adjusting these parameters, users can fine-tune the co-registration process in the FlowVelo tool to suit their specific requirements.

3.7.2 Feature search area and pose estimation

The search area must be defined to identify particles since feature detectors search for areas with a lot of contrast in the river region. Since the contrast is higher than the water surface, the points of interest would be discovered there.

The region of interest known as the feature search is determined according to the water level to obscure the image. The camera observed the water level and 3D surface model to automatically define water area of the river. The 3D surface model is clipped based on the water level value, ensuring that only the points below the water surface are projected onto the image. As a result, knowledge of pose estimation and the internal geometry of the camera is required. The estimation of the camera's position is determined through spatial resection, which takes into account the coordinates of ground control points and the interior camera parameters. The three-dimensional point cloud of the river reach under observation is projected into two-dimensional image and any gaps are filled using morphological closing. Next, the contour of the underwater area is extracted, forming the basis for the search mask. In the absence of a 3D surface model, a tracking area of interest may be provided through a mask file.

3.7.3 Feature detection and filtering

The Shi-Tomasi feature (Good Feature to Track; GFTT) detector is used to find particles (Shi & Tomasi, 1994). This method detects particles in a similar way with Harris corner detector, but with a different scoring mechanism to determine the validity of a feature (Fig. 3.7). While there are several other feature detectors available, Tauro et al. (2018) have tested several techniques and found that the GFTT detector works effectively, even in areas with low contrast.

The undesirable particles require removal for accurate tracking. For example, reflections of sunlight on waves can create high-contrast areas on the water surface, which must be eliminated to avoid tracking of erroneous particles (Lewis & Rhoads, 2018). A nearest neighbour search is conducted to identify areas with dense particle clusters. The particle is not subjected to additional examination if there are a large number of characteristics within a search

radius. Additionally, particles with levels of brightness below a threshold are removed to avoid including wave shadows as features (Lewis & Rhoads, 2018).

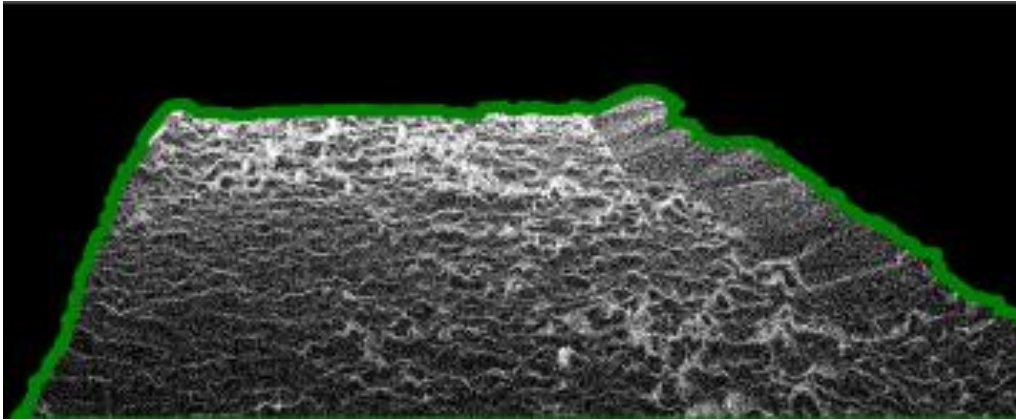


Figure 3.5: 3D point cloud projected into image space

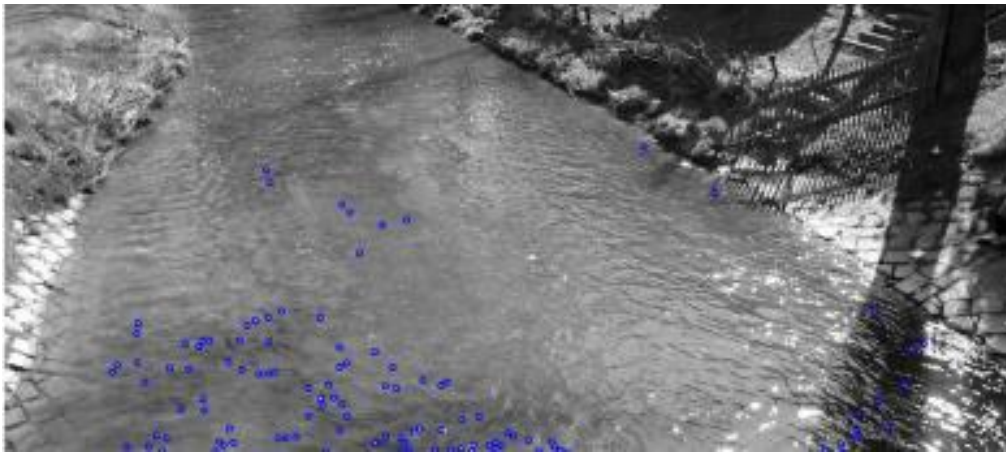


Figure 3.6: Tracking feature that have been discovered and filtered (blue circles)

3.7.4 Feature Tracking

Once features have been found, they are monitored across succeeding frames. Tracking is accomplished using normalised cross-correlation (NCC), which accounts for variations in brightness and light between frames. NCC is applied within a defined search area usually 15 pixels to identify which spots for each feature have the highest correlation scores. These locations probably match the new places where the migrated particles on the water surface.

Sub-pixel is used to enhance the matching accuracy. To calculate the phase shift between two identically sized templates and matched search regions, the data is translated to the frequency domain. The last matching locations serve as the new templates for the following frame. The features were monitored for 40 frames while new features were discovered every 15th frame. It is advantageous that features should be detected more often than how many frames there are tracked over.

3.7.5 Tracking Filtering

The result tracking is possible to have inaccurate tracking findings such as tracks that diverge substantially from the main flow direction. A speckle that is tracked as a feature but has a low contrast, will cause confusing matching scores. Hence, it is necessary to filter the resulting velocity tracks. Tauro et al., (2018) addressed this issue by removing false trajectories based on criteria like minimum track dimensions and track direction.

In this process, the flow parameters of the river are assumed. Six parameters are considered: minimum frame count for a tracked feature, minimum and maximum tracking distances, flow steadiness, range of track directions, and deviation from the average flow direction. For a track to be regarded as having accurate velocity information, it must satisfy these requirements.

The first initial requirement specifies the minimum percentage of frames across which the features should be traceable. The essential premise is that if a feature can only be traced for a few frames, it is more likely to become a speckle occurrence rather than a well-defined flowing particle. However, in order to circumvent any limitations on flow velocities and camera frame rates, the minimum value specified is 0.

The second and third filtering requirement are based on features are tracked, including thresholds for minimum and maximum distances. These thresholds may be roughly estimated if the image scale and the expected range of river flow are specify. The minimum and maximum distance parameters in this investigation are specify to 0.1 and 10 pixels.

The feature's directional flow behaviour as demonstrated by a steadiness measure is the fourth criteria. For each track, the sub-track directions are examined. The idea is to conduct observations of rivers under practically constant flow conditions, the frequent variations in flow direction within a track are a sign of measurement error and should be filtered. The range of each sub-track direction is also taken into account when determining the flow behaviour. The track will be excluded if the range is greater than a specified threshold (120°).

The last requirement looks at the river's primary flow direction. All track average direction is determined and if it is more than or lower than the buffer threshold, the individual tracks are excluded from next step. The buffer value should be determined considering the overall unpredictability of the river flow pattern. A low value assumes a more uniform flow. It should be emphasised that the directed filter only works to a limited effectiveness in complicated flow situations such as turbulent or irregular rivers. In these situations, local filters ought to be used over these global options

3.7.6 Velocity Measurement

In this final stage of processing, the distances measured in pixels are converted to metric units, allowing for the calculation of flow velocities in metres per second. It may perform the projection of image measurement into object space using the established camera posture and internal geometry and generating a three-dimension image of the light ray that leaves the image plane and travels through the camera's projection centre. This ray is then intersected with a three-dimension surface model, specifically the planar water surface at the water level, to determine the 3D object coordinates of each image measurement. The intersection points with the water surface are used to obtain the real-world coordinates of the beginning and ending points of each track. Metric flow velocities are determined by taking into account the separation between these places, the number of captured frames and the camera's frame rate.

Following the metric velocity tracks through an additional filtering step using an outlier filter to eliminate other outliers. The threshold for filtering is determined by the average velocity multiplied by a certain multiple of its standard deviation. By adjusting the multiple value, more or fewer features can be filtered, with a lower value retaining only tracks that

closely align with the average velocity. In the present study, the value of the parameter is 1.5. This filtering process plays a crucial role in handling challenging tracking situations.

In terms of tracking reliability, it should be noted that when using cameras that provide a distorted image of river flow, measurements near the sensor are preferred. This preference stems from the fact that particles cover a great amount of pixels in closer to the camera compared to those further away. Consequently, a small measurement error of 1 pixel near the camera could correspond to 1 cm, while the same error at a greater distance might equate to 1 m. Additionally, tracking accuracy diminishes as the distance increases due to an increased occurrence of glancing ray intersections with the water surface.

3.8 Modelling

This study will be carried out with four objectives. Firstly, to identify the appropriate parameters to be used to parameterize river water flow by using image processing. The second objective of this study is to develop the best model for measuring the river water flow surface velocity by using image processing. The third objective is to investigate the performance of the proposed method of river water flow. All these objectives can be achieved by estimate the water flow velocity.

3.8.1 Autoregressive Integrated Moving Average (ARIMA)

In this study, arima will be utilized in the training and testing of flow velocity data. Arima is a time series analysis model used for forecasting and understanding the patterns and trends in data that changes over time. In comparison to other time series techniques, the ARIMA model holds an advantage by being able to accommodate various data models (Rizkya et al., 2019). The formula for the model is given as ARIMA (p, d, q), where p represents the order of the autoregressive process, d signifies the order of the stationary process for the data, and q denotes the order of the moving average process (Zhang, 2003).

ARIMA combines three components:

1. Autoregression (*AR*): It refers to model that uses the dependent relationship between an observation and a certain number of lagged, or prior, values.

Equation:

$$AR(p) = \varphi_1 * Y(t - 1) + \varphi_2 * Y(t - 2) + \dots + \varphi_p * Y(t - p) \quad (3.1)$$

2. Integration (*I*): It represent the differencing of raw observations is to convert the time series to into a stationary form.

Equation:

$$I(d) = (1 - B)^d * Y(t) \quad (3.2)$$

3. Moving Average (*MA*): It depicts the relationship between an observation and a residual error resulting from the application of a moving average model to lagged observations.

Equation:

$$MA(q) = \theta_1 * \varepsilon(t - 1) + \theta_2 * \varepsilon(t - 2) + \dots + \theta_q * \varepsilon(t - q) \quad (3.3)$$

The usual notation is ARIMA (p, d, q), the parameters are replaced with specific integer values to represents the model's characteristics:

- p: Represents the number of lag observations included, also known as lag order.
- d: Denoted the number of times the raw observations are differenced, also known as the degree of differencing.
- q: Signifies the moving average window's size, commonly referred to as the moving average order.

Overall, arima model combines these components to form the final equation:

$$Y(t) = c + AR(p) + I(d) + MA(q) + \varepsilon(t) \quad (3.4)$$

Where:

- $Y(t)$ represents the value of the time series at time t .
- c is the constant term.
- $\phi_1, \phi_2, \dots, \phi_p$ are the autoregressive coefficients.
- $\theta_1, \theta_2, \dots, \theta_q$ are the moving average coefficients.
- $\varepsilon(t)$ represents the residual term or error at time t .

In conclusion, ARIMA model is a powerful tool for time series analysis with a wide range of applications (Rizkya et al., 2019). It offers several benefits, including the ability to forecast future values, comprehend historical data patterns, and facilitate informed decision-making. By training and testing flow velocity data using the ARIMA model, you can gain insights into the behavior of flow velocity over time, make forecasts for future values, detect anomalies, and evaluate the model's performance.

3.8.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) architecture appeared for the first time in (Hochreiter & Schmidhuber, 1997). LSTM is a sequential data model designed to address the issue of long-term memory decay in Simple Recurrent Neural Networks (RNNs) (Graves & Schmidhuber, 2005). Based on the Figure 3.7, LSTM includes three essential components: forget gate, input gate, and output gate. The horizontal line connecting C_{t-1} to C_t at the top of Figure 3.4 represents cell state in an LSTM architecture. This cell state plays a crucial role in preserving and transferring time series information throughout the entire sequence (Park et al., 2020). Unlike other recurrent neural networks, the LSTM's cell state is designed to prevent memory loss over time (Park et al., 2020).

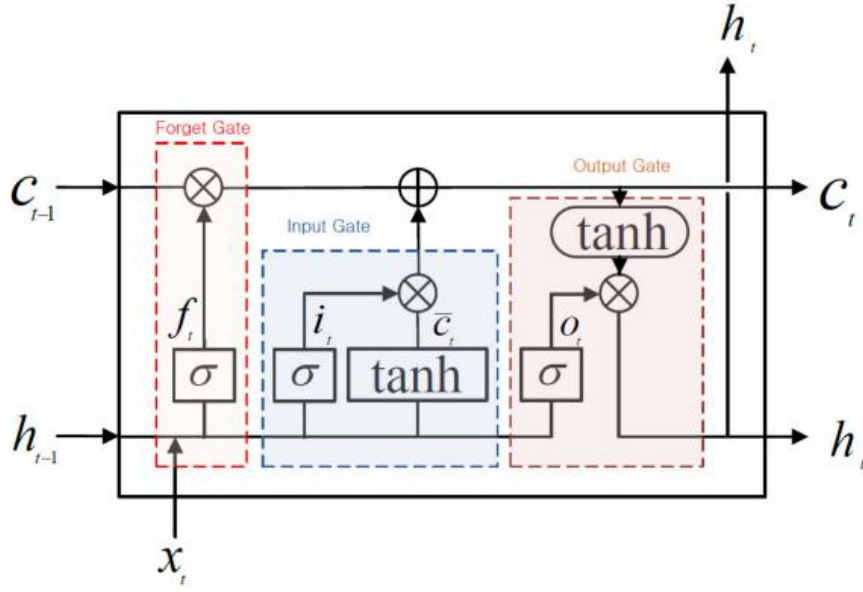


Figure 3.7: Long Short-Term Memory (LSTM)

Source: (Park et al., 2020)

The LSTM consists several components:

1. Forget Gate (f): The determination of which data from the former cell state should be eliminated. It calculates the forget gate value ($f(t)$) using the sigmoid activation function, considering the prior hidden state ($h(t-1)$), the current input ($x(t)$), and their corresponding weight matrix (W_f) and bias term (b_f).

$$f(t) = \sigma(W_f * [h(t-1), x(t)] + b_f) \quad (3.5)$$

2. Input Gate (i): It regulates the flow of new information from the previous cell state by considering the present input and the prior hidden state. It uses the sigmoid activation function to calculate input gate ($i(t)$), taking into account the prior hidden state ($h(t-1)$), the present input ($x(t)$), and their respective weight matrix (W_i) and bias term (b_i).

$$i(t) = \sigma(W_i * [h(t-1), x(t)] + b_i) \quad (3.6)$$

3. Cell State Update (C_{\sim}): The cell state update computes a new candidate cell state using the hyperbolic tangent activation (\tanh). It combines the prior hidden state ($h(t-1)$), the present input ($x(t)$), and their associated weight matrix (W_c) and bias term (b_c).

$$C_{\sim}(t) = \tanh(W_c * [h(t-1), x(t)] + b_c) \quad (3.7)$$

4. Cell State (C_t): The cell state ($C(t)$) represents the memory of the LSTM. It is updated by selectively forgetting and adding information. The forget gate output ($f(t)$) scales the prior cell state ($C(t-1)$), while the input gate output ($i(t)$) scales the new cell state ($C_{\sim}(t)$).

$$C(t) = f(t) * C(t-1) + i(t) * C_{\sim}(t) \quad (3.8)$$

5. Output Gate (o): The output gate determines how much of the cell state should be output as the hidden state. It computes the output gate value ($o(t)$) using sigmoid activation function, considering the prior hidden state ($h(t-1)$), the present input ($x(t)$), and their respective weight matrix (W_o) and bias term (b_o).

$$o(t) = \sigma(W_o * [h(t-1), x(t)] + b_o) \quad (3.9)$$

6. Hidden State (h): The LSTM cell's ultimate result is the hidden state ($h(t)$). It carries the relevant information learned from the input sequence and is computed by passing the cell state ($C(t)$) via the hyperbolic tangent activation function (\tanh) and scaling it with the output gate output ($o(t)$).

$$h(t) = o(t) * \tanh(C(t)) \quad (3.10)$$

By utilizing these components and their respective equations, LSTM networks can effectively capture and retain important information over long sequences, making them suitable for tasks involving sequential data analysis and prediction.

3.9 Evaluation

Model assessment refers to the systematic evaluation of a machine learning model's effectiveness, limitations, and advantages. It involves using diverse evaluation measures to gauge the model's performance. The assessment of river water flow velocity plays a critical role in evaluating the accuracy and reliability of the employed models. It involves comparing the estimated flow values with the actual or predicted flow values in the river to determine how well model perform. To ensure an assessment, model evaluation criteria derived from mathematical equations are utilized. These criteria encompass the mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE). The proximity of these metrics to the actual or predicted values signifies the model's performance, with smaller values indicating a better fit. By employing these evaluation metrics, we can effectively evaluate the performance.

Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) is a metric used to evaluate the accuracy of predictions in a set of observations. It measures the average magnitude of the errors without considering their direction (D. N. Moriasi et al., 2007). Otherwise put, it calculates the average absolute difference between the predicted values (x_i) and the actual observations (y_i) for a given set of data (n).

$$MAE = \frac{1}{n} \sum_{i=1}^n |x^i - y^i| \quad (3.11)$$

Mean Squared Error (MSE)

The Mean Squared Error (MSE) is a metric used to assess the accuracy of predictions by measuring the average of the squared differences between the predicted values and the corresponding actual observations (D.N. Moriasi et al., 2007). It computes the mean of the squared deviations between the estimated and observed data values.

$$MSE = \frac{1}{N} \sum_{i=1}^n (x^i - y^i)^2 \quad (3.12)$$

Root Mean Squared Error (RMSE)

The Root Mean Squared Error (RMSE) serves as a quadratic scoring rule employed to assess the average magnitude of errors in a set of predictions (D. N. Moriasi et al., 2007). The average squared difference between the anticipated values and the matching actual observations is used to compute it.

$$RSME = \sqrt{\frac{\sum_{i=1}^n (x^i - y^i)^2}{n}} \quad (3.13)$$

3.10 Deployment

In this study, the deployment of model can be done via public platform. By working closely with I Net Spatial, we can enhance and achieve better results for our study, especially when estimating river flow. When considering selection of a public platform for deployment, both Kaggle and GitHub emerge as excellent choices. GitHub boasts a large community of developers and offers seamless integration with various tools, making it an ideal platform for collaboration and code development. In contrast, Kaggle stands out as one of the premier platforms for deploying machine learning models. It boasts a user-friendly interface that simplifies the process of uploading and deploying models. Both platforms offer valuable features and vibrant communities that can contribute to the successful deployment of our application.

CHAPTER 4

EXPECTED OUTCOMES AND CONCLUSIONS

4.1 Introduction

In this study, our outcomes will be presented FlowVelo tool which are image-based flow velocity estimation. Amongst various available approach, this application quick and easy approach proves to be highly dependable. This study also highlighted an innovative image based technique to measure water surface velocity, which efficiently yields reliable results concerning river flow rates.

4.2 Data Preparation

In this study, the captured video dataset will be converted to image by using VLC media player. Based on Figure 4.1, it shows an image file for each frame.

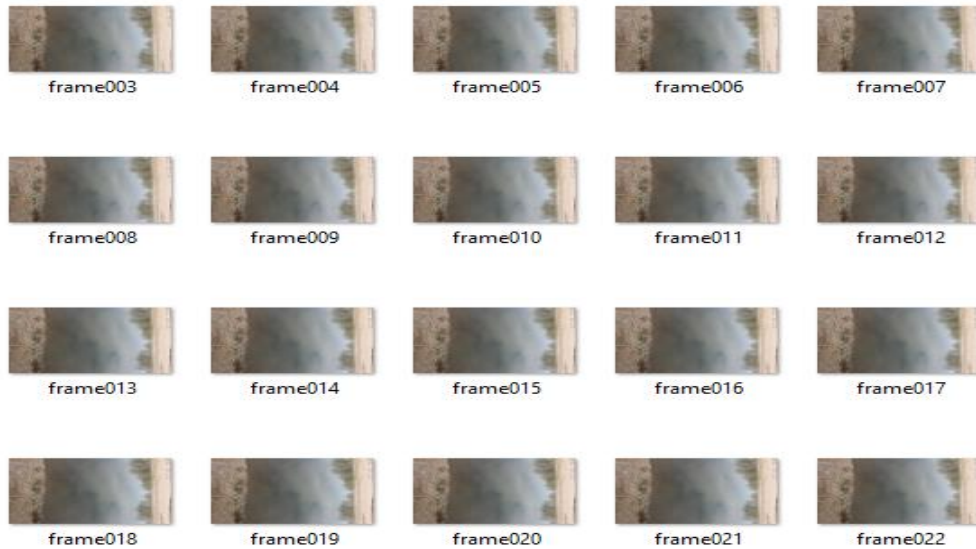


Figure 4.1: Image file of each frames

The video with a duration of 2 minutes will trim the video into 20 seconds with selected desired position and remove any content outside of this 20 second timeframe. Decide frame rate of 30 frames per second and extract frames from the 20-second video, 600 frames will be extract. After extract the images, save it into image file.

4.3 Data Pre-processing

The workflow for automatic retrieval of water flow information from image sequences is introduced in this chapter. Most of the data processing is realized in Python using the OpenCV library (Bradski, 2000).

```
(base) C:\Users\user>pip install opencv-python
Collecting opencv-python
  Downloading opencv-python-4.7.0.72-cp37-abi3-win_amd64.whl (38.2 MB)
    | 38.2 MB 432 kB/s
Requirement already satisfied: numpy>=1.17.3 in c:\users\user\anaconda3\lib\site-packages (from opencv-python) (1.21.5)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.7.0.72
```

Figure 4.2: OpenCV installing

The depicted Figure 4.2 demonstrates the command executed in the command prompt to initiate the installation of OpenCV. This command triggers the download and installation of packages associated with the OpenCV library. Upon completion, a message indicating a successful installation will be displayed.

```

import os
import numpy as np
import pandas as pd
import cv2
import imageio

import coregistration_functions as coregF
import photogrammetry_functions as photogrF
import input_output_functions as ioF
import PTV_functions as ptv

import Tkinter as tk
import tkFileDialog, ScrolledText
from ttk import *

```

Figure 4.3: Import all related libraries

Figure 4.3 above, install and import all the related libraries on the jupyter notebook. After import all the libraries, preprocessing may start.

```

class FlowVeloTool:

    def __init__(self, master):

        master_frame = Frame(master, name='master_frame')
        master.title('Image-based flow velocity estimation')
        note = Notebook(master_frame, name='note')
        master_frame.grid()

        #text box for display output
        self.textbox = ScrolledText.ScrolledText(master, height=10, width=20)
        self.textbox.place(x=700, y=50, width=300, height=800)

        '''-----frame flow velocity-----'''
        frame = Frame(note)
        note.add(frame, text="flow velocity")
        note.grid(row=0, column=0, ipadx=500, ipady=440)

        self.xButton = 370
        self.xText = 250
        self.xText2 = 350
        self.yAddText = 10
        self.yAddText2 = 10
        Style().configure("RB.TButton", foreground='blue', font=('helvetica', 10))

        currentDirectory = os.getcwd()

```

Figure 4.4: Main Frame

4.4 Data Analysis

In this step, there are 600 images will be used to extract information. The specific analysis techniques and methods applied to these images. The FlowVelo tool accept images in common image formats such as PNG (.jpg and.jpeg formats). All the images used in this study are in.png format; this is to reduce the amount of storage space required and speed up the FlowVelo tools processing.



Figure 4.5: Original Image (PNG)



Figure 4.6: Gray Scaled Image (PNG)

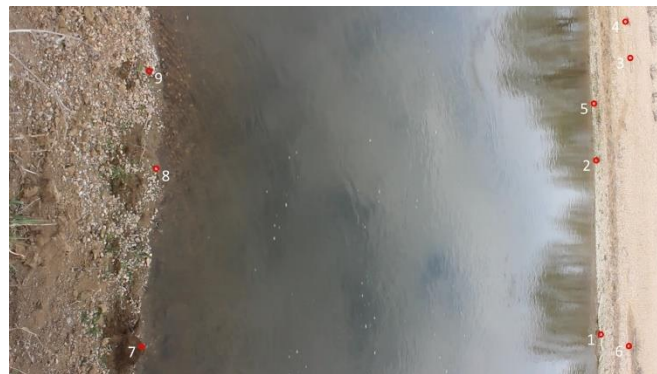
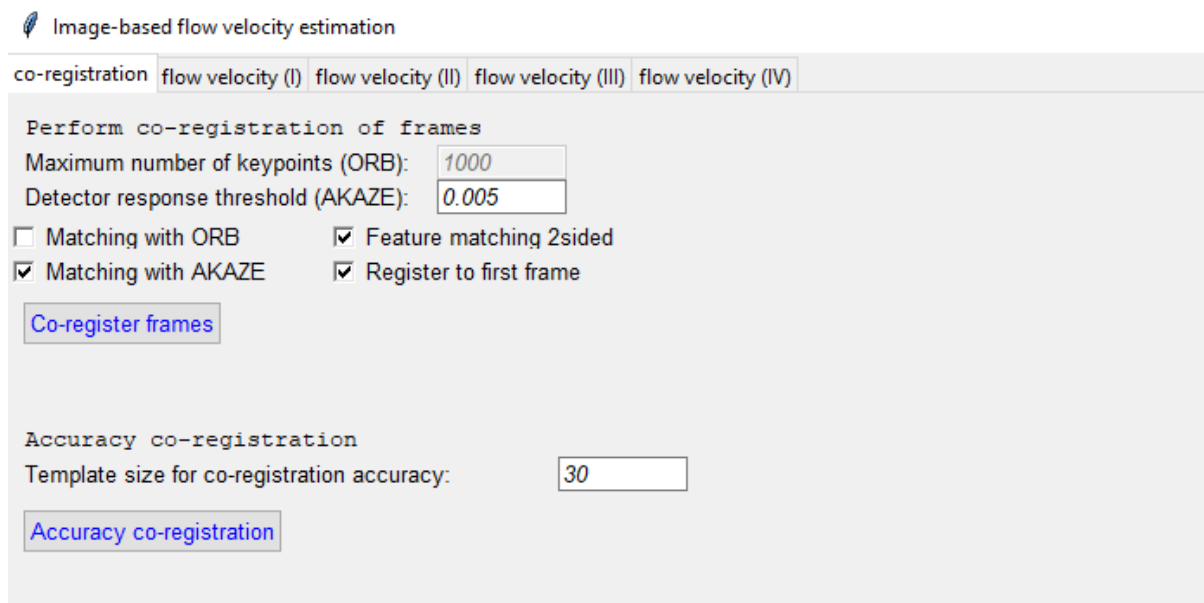


Figure 4.7: Location GCPs image (PNG)

Besides, FlowVelo tool also support the use of colour images in addition to grayscale images. This means that we can provide colour images as input to the FlowVelo tool for analysis and processing.

4.4.1 GUI Interface Image-based flow velocity estimation

In this chapter, the GUI interface will be implemented using Python in a Jupyter notebook. There are several steps to adjust and set the parameters on each interface. First of all, press the co-registration button to enter the co-registration interface. Based on Figure 4.8, there are several parameter settings required to perform co-registration (stabilisation) of frames when the camera has been moved during data acquisition.



The screenshot shows a web-based GUI titled "Image-based flow velocity estimation". It has a tabbed interface with four tabs: "co-registration", "flow velocity (I)", "flow velocity (II)", "flow velocity (III)", and "flow velocity (IV)". The "co-registration" tab is active. Under the heading "Perform co-registration of frames", there are two input fields: "Maximum number of keypoints (ORB):" with the value "1000" and "Detector response threshold (AKAZE):" with the value "0.005". Below these are four checkboxes: "Matching with ORB" (unchecked), "Matching with AKAZE" (checked), "Feature matching 2sided" (checked), and "Register to first frame" (checked). A "Co-register frames" button is located below the checkboxes. Further down, under the heading "Accuracy co-registration", there is an input field for "Template size for co-registration accuracy:" with the value "30". An "Accuracy co-registration" button is located below this field.

Figure 4.8: GUI Interface Co-Registration (FlowVelo tool)

These parameters include:

1. Maximum number of keypoints: This parameter is applicable when the ORB option is chosen as the matching variant. It allows you to specify the maximum number of keypoints (distinct features) to be detected and matched between frames.

2. Detector response threshold: This parameter is relevant when the AKAZE option is selected as the matching variant. It determines the threshold value for detecting keypoints. Lower values, typically ranging from 0.02 to 0.005, result in the detection of more features but can increase the computation time required.
3. Matching with ORB: If checked, this option utilizes the ORB feature descriptor for matching features between frames.
4. Matching with AKAZE: If checked, this option employs the AKAZE feature descriptor for matching features between frames.
5. Feature matching 2sided: Enabling this option enhances the robustness of feature matching by considering matches in both directions (from frame A to frame B and vice versa). However, this also increases the computation time.
6. Register to first frame: When checked, this option registers all frames of the sequence to the first frame of that sequence. On the other hand, if unchecked, each new frame is registered to the previous frame in the sequence.

Once you have configured all the necessary input and set the parameters for co-registration, you can proceed to the next interface. Press the flow velocity (I) to enter the second interface. Based on Figure 4.9, there are several settings required to set the data input and parameters for flow velocity measurement.

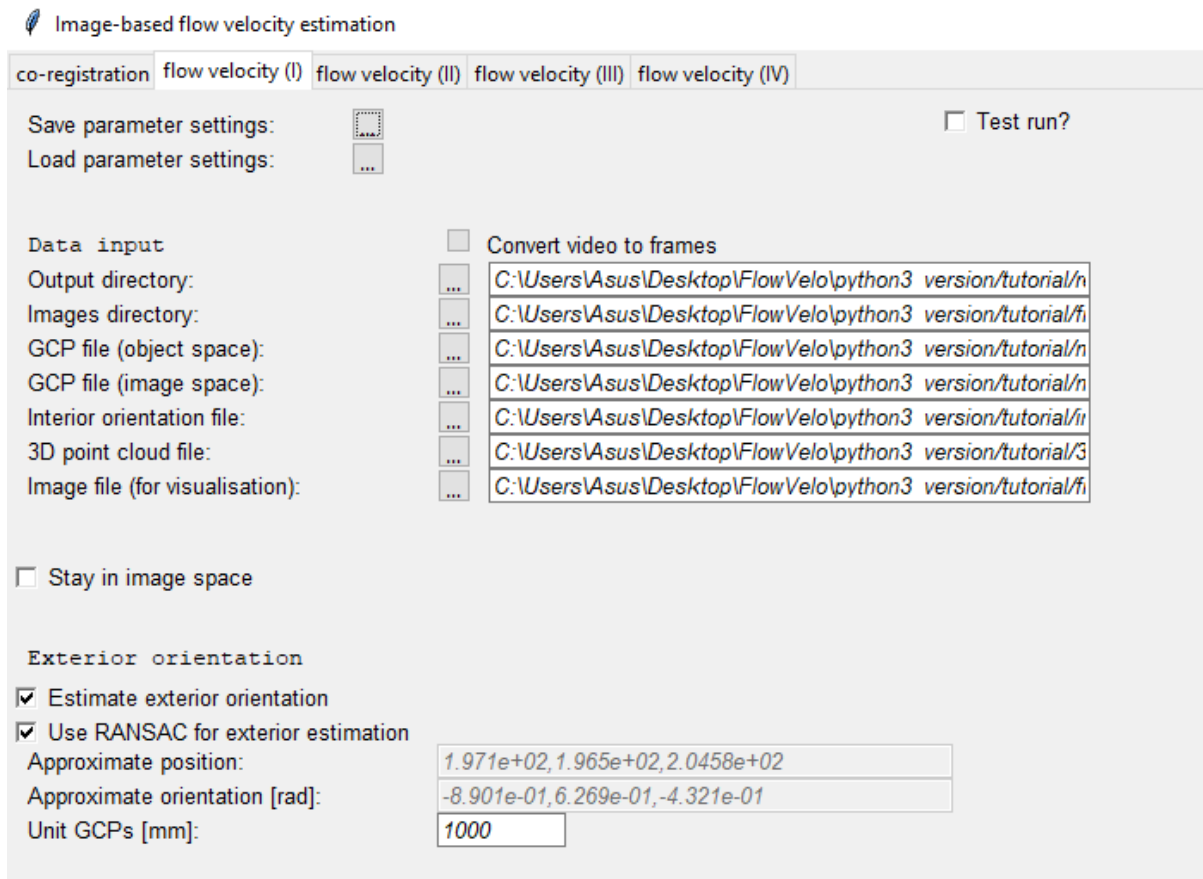


Figure 4.9: GUI Main Interface Flow Velocity I (FlowVelo tool)


These parameters include:

1. Save parameter settings / Load parameter settings: These options allow you to save and load all parameter settings of the flow velocity tool. This is useful when testing different parameter choices to find the optimal configuration for tracking.
2. Convert video to frames: If you have a video file instead of individual frames, you can use this built-in option to convert the video into frames, which can then be used for further processing.
3. GCP file (object space): An ASCII (txt) file that contains the point ID, X coordinate, Y coordinate, and Z coordinate of Ground Control Points (GCPs) in object space.

4. GCP file (image space): An ASCII (txt) file that contains the point ID, X image coordinate (in pixels), and Y image coordinate (in pixels) of GCPs in image space.
5. Interior orientation file: An ASCII (txt) file that contains the focal length, principal point, radial distortion parameters, tangential distortion parameters, affinity and shear parameters, sensor size, and image resolution of the camera's interior orientation.
6. 3D point cloud file: An ASCII (txt) file that contains the X, Y, and Z coordinates of the 3D point cloud.
7. Image file (for visualization): Select an image file (.img) where the results of feature detection and tracking will be displayed for visualization purposes. Camera pose estimation: Choose the unit of Ground Control Points (GCPs) and provide the camera pose information if known, instead of using GCPs. You can also combine approximate camera pose information with GCP measurements.
8. Stay in image space: By checking this option, you can use unscaled tracking results, meaning no additional information regarding the exterior orientation, water level, and 3D point cloud file is needed.
9. Unit GCPs: Specify the unit of the Ground Control Points (GCPs) measurements. All measurements in the FlowVelo-Tool are performed in millimeters (mm), so the interior camera geometry should also be provided in mm.
10. Estimate exterior orientation: Check this parameter if you want to perform camera pose estimation using GCP coordinates measured both in object space (e.g., using a total station) and in the images.
11. Use RANSAC for exterior estimation: If you don't have approximate values of the camera orientation and position, checking this option allows the tool to estimate the camera pose using spatial resection. It also helps identify and exclude outliers among the GCPs during the pose estimation process.

12. Alternatively, if you have a priori known information about the camera pose and do not want to use GCPs for defining the exterior orientation, uncheck both "Estimate exterior orientation" and "Use RANSAC for exterior estimation." You can then input the known exterior orientation parameters in the approximate position/orientation fields.

Once you have entered all the required input parameters and files, you can proceed to the next interface for processing. Press the flow velocity (II) to enter the third interface. Based on Figure 4.10, there are several parameters that need to set for feature detection, feature tracking and iterations.

 Image-based flow velocity estimation

co-registration flow velocity (I) **flow velocity (II)** flow velocity (III) flow velocity (IV)

Feature detection		Feature tracking	
<input type="checkbox"/> LSPIV	<input checked="" type="checkbox"/> PTV	<input type="checkbox"/> LK	<input type="checkbox"/> Initial Estimates LK
Maximum number features:	<input type="text" value="1000"/>	<input checked="" type="checkbox"/> NCC	
Minimum feature brightness:	<input type="text" value="70"/>	Template width:	<input type="text" value="7"/>
Neighbor search radius:	<input type="text" value="30"/>	Template height:	<input type="text" value="7"/>
Maximum number neighbors:	<input type="text" value="5"/>	Search area size x direction:	<input type="text" value="24"/>
Sensitivity feature detection:	<input type="text" value="0.02"/>	Search area size y direction:	<input type="text" value="24"/>
PIV cell width:	<input type="text" value="200"/>	Shift search area in x:	<input type="text" value="0"/>
PIV cell height:	<input type="text" value="200"/>	Shift search area in y:	<input type="text" value="8"/>
		<input checked="" type="checkbox"/> Subpix	<input type="checkbox"/> LSM
		<input checked="" type="checkbox"/> Plot results	<input checked="" type="checkbox"/> Save gif

Iterations	
FD every nth frame:	<input type="text" value="10"/>
Track for n frames:	<input type="text" value="20"/>
Track every nth frame:	<input type="text" value="2"/>

Figure 4.10: GUI Interface Flow Velocity II (FlowVelo tool)

These parameters include:

1. LSPIV: This option defines interrogation areas in a raster pattern, and the content within these areas is then tracked.
2. PIV cell width: It defines the distance between the center points of interrogation areas along x-axis. This option is specific to LSPIV.
3. PIV cell height: It defines the distance between the center points of interrogation areas along y-axis. This option is specific to LSPIV.
4. PTV: This option involves detecting individual particles and then tracking them.
5. Maximum number features: This parameter defines the maximum total number of detected good features to track. It determines how many features will be kept for subsequent filtering steps.
6. Minimum feature brightness: This parameter sets the threshold for the minimum brightness required for a feature to be considered a particle for tracking. Increasing this value results in fewer detected features. This option is specific to PTV.
7. Neighbor search radius: This parameter sets the radius (in pixels) within which nearest neighbors are searched for subsequent cluster filtering. Increasing this value includes more features in the filtering step. This option is specific to PTV.
8. Maximum number neighbors: This parameter determines the maximum number of neighbors allowed within the previously defined neighbor search radius. Increasing this value retains more features. This option is specific to PTV.
9. Sensitivity feature detection: This parameter sets the quality level for features to be kept. It considers the quality measure defined by the minimal eigenvalue (corner score).
10. NCC: Perform feature matching using Normalized Cross Correlation.
11. LK: Perform feature matching using the Lucas-Kanade approach.

12. Initial Estimate LK: Enable this option if the Lucas-Kanade method should use the tracked position as an initial estimate for matching in the next frame.
13. Template width: Set the width of the template (in pixels) used for matching. The template size should be odd. A wider template increases processing time but reduces the chances of ambiguities.
14. Template height: Set the height of the template (in pixels) used for matching. The template size should be odd. A taller template increases processing time but reduces the chances of ambiguities.
15. Search area size x direction: Define the width of the search area (in pixels) within which the template is moved to find the corresponding region (e.g., particle). A larger search area increases processing time and the chances of ambiguities.
16. Search area size y direction: Define the height of the search area (in pixels) within which the template is moved to find the corresponding region (e.g., particle). A larger search area increases processing time and the chances of ambiguities.
17. Shift search area in x: Define the approximate flow speed (in pixels) in the x direction to improve matching accuracy. Higher flow velocities result in larger shifts.
18. Shift search area in y: Define the approximate flow speed (in pixels) in the y direction to improve matching accuracy. Higher flow velocities result in larger shifts.
19. Subpixel: Enable this option to refine the matching results to subpixel accuracy using a weighted centroid fit to locate the peak position. "LSM": This option is still in development. It performs matching via a least square adjustment method, resulting in subpixel accurate positions. "
20. Plot results: Choose whether to display the matching results.
21. Save gif: Choose whether to save the tracking results as an animation in a GIF format.
22. FD every nth frame: Set the interval at which new features are detected. Features will be detected once every nth frame, where n is the specified value.

23. Track for n frames: Determine the duration for which features are tracked. Features will be tracked across n frames, where n represents the specified value.
24. Track every nth frame: Specify the frame skipping interval for feature matching. Only every nth frame will be used for feature matching, where n is the specified value. For example, if you want to track features in every frame, set the value to 1. If you want to track features every second frame, set the value to 2. Similarly, set the value to 3 for tracking every third frame, and so on.

After entered all the required input parameters, you can proceed to the next interface for processing. Press the flow velocity (III) to enter the fourth interface. Based on Figure 4.11, there are several parameters need to set for track flitering.

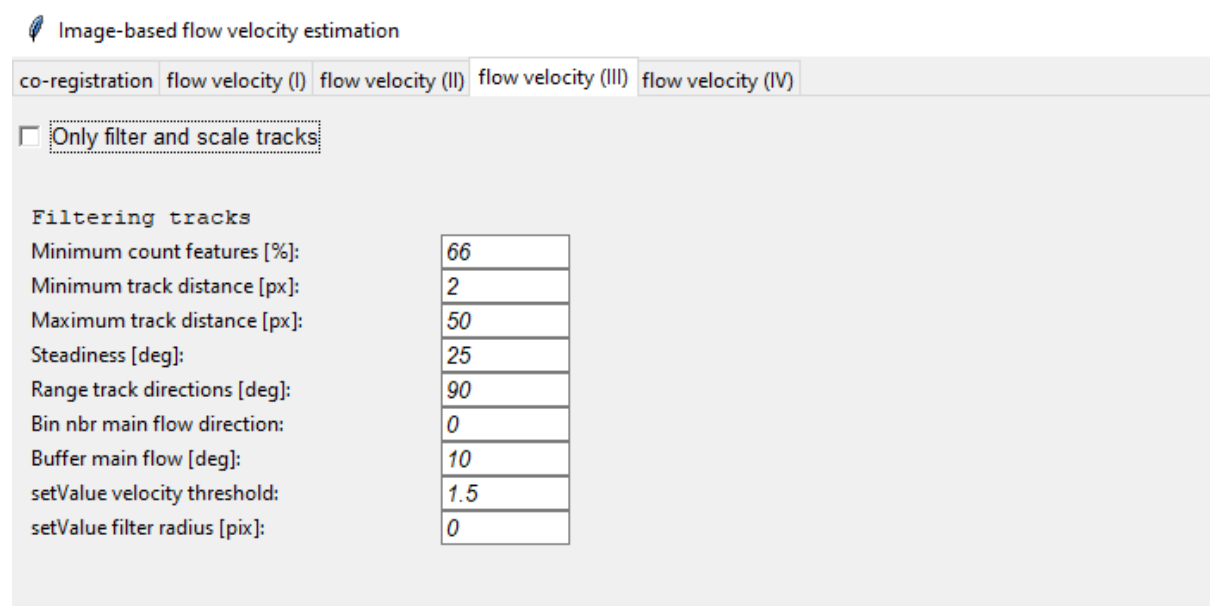


Figure 4.11: GUI Interface Flow Velocity III (FlowVelo tool)

These parameters include:

1. Only filter and scale tracks: Select this option if you have already successfully performed tracking and want to adjust the filtering options. When this option is checked, click the "Estimate flow velocity" button in the "flow velocity (IV)" frame and load the "Tracking_FT_nbrFrames_FD_nbrFrames.txt" file.
2. Minimum count features: Set the minimum number of frames that a feature must be tracked across to be considered a reliable track. This value is expressed as a percentage of the total traceable number (i.e., the "Track for n frames" value).
3. Minimum track distance: Define the minimum distance (in pixels) that features must move between consecutive frames. This corresponds to the required minimum length of a sub-track. A track will be considered reliable if all sub-tracks meet or exceed this threshold.
4. Maximum track distance: Specify the maximum distance (in pixels) that features can move between consecutive frames. This corresponds to the maximum allowed length of a sub-track. A track will be considered reliable if all sub-tracks are below this threshold.
5. Steadiness: Determine the maximum allowed change in direction (in degrees) for features between frames. This value represents the maximum allowed standard deviation of directional changes for all sub-tracks within a track.
6. Range track directions: Define the maximum range (in degrees) within which sub-tracks of a track can change their direction.
7. Bin nbr main flow direction: This parameter needs to be tested. It is recommended to set it to 0.
8. Buffer main flow: Specify the allowed deviation (in degrees) of an entire track from the main flow direction. The main flow direction is calculated as the average direction of all track directions.

9. setValue velocity threshold: Apply a statistical outlier filter to remove features. The threshold is calculated as the mean plus the specified value multiplied by the standard deviation. Adjust the setValue to determine the threshold above and below which features will be removed. Increasing the value will result in more tracks being retained.
10. setValue filter radius [pix]: For the local outlier filter, set the maximum radius (in pixels) within which features will be considered for statistical filtering.

After entered all the required input parameters, you can proceed to the next interface for processing. Press the flow velocity (IV) to enter the fifth interface. Based on Figure 4.12, there are several settings to set the parameter and file for scale velocities and water area.

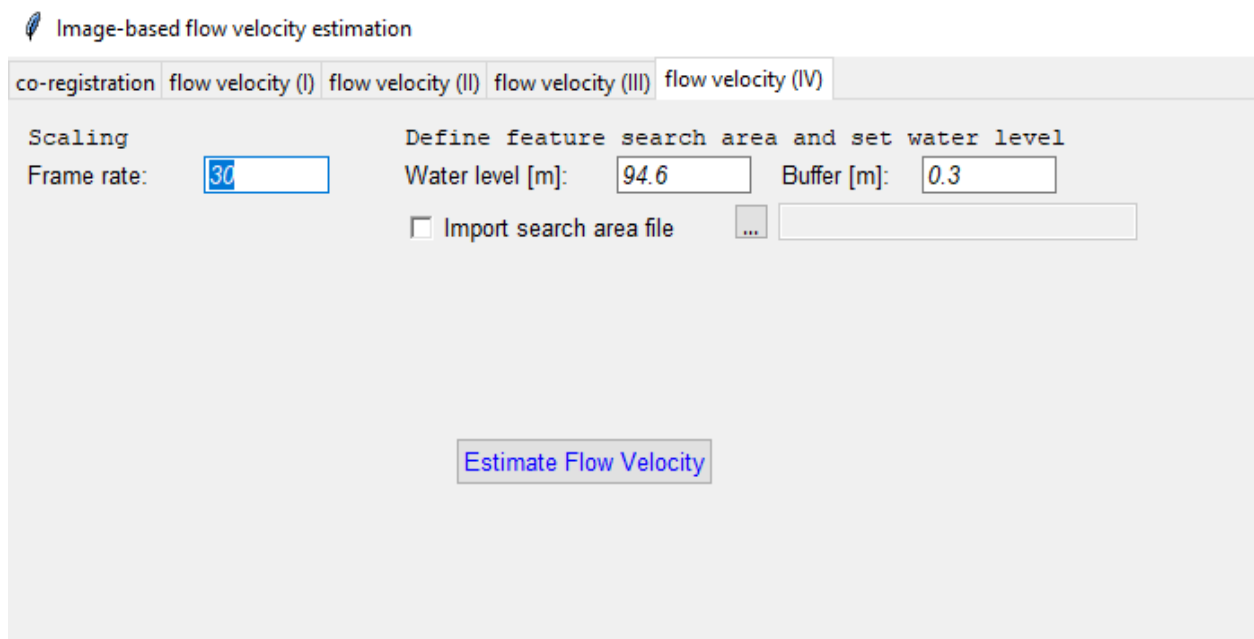


Figure 4.12: GUI Interface Flow Velocity IV (FlowVelo tool)

These parameters include:

1. **Frame rate:** Specify the frame rate (in seconds) at which the videos or images were captured. This information is important for accurate timing and synchronization of the data.
2. **Water level:** Define the water level in meters. This parameter represents the height of the water surface or the desired reference level for the analysis.
3. **Buffer:** Set the buffer value for the water level. This parameter is particularly useful in complex terrains or situations where the camera pose estimation is not very accurate. It allows for a margin or buffer around the water-shore borders to account for uncertainties or variations in the water level estimation.
4. **Import search area file:** If you have manually defined the feature search area using an external image editing tool like GIMP, you can import a mask image of the masked water area. The tool will automatically derive the contour that contains the image coordinates of the border of the mask. This contour information can be saved as a result for further analysis or visualization.

These all steps allow you to configure the data input and set parameters to adjust the settings according to your specific requirements.

4.5 Result Water Flow Velocity

Once all the input parameters and files have been set, click on the "Estimate Flow Velocity" button to initiate the flow velocity estimation process. The tool will generate the results in text and image, which can be found in the file named "resultFlowVelo.file". This file contains the computed flow velocities based on the provided inputs and parameters settings.

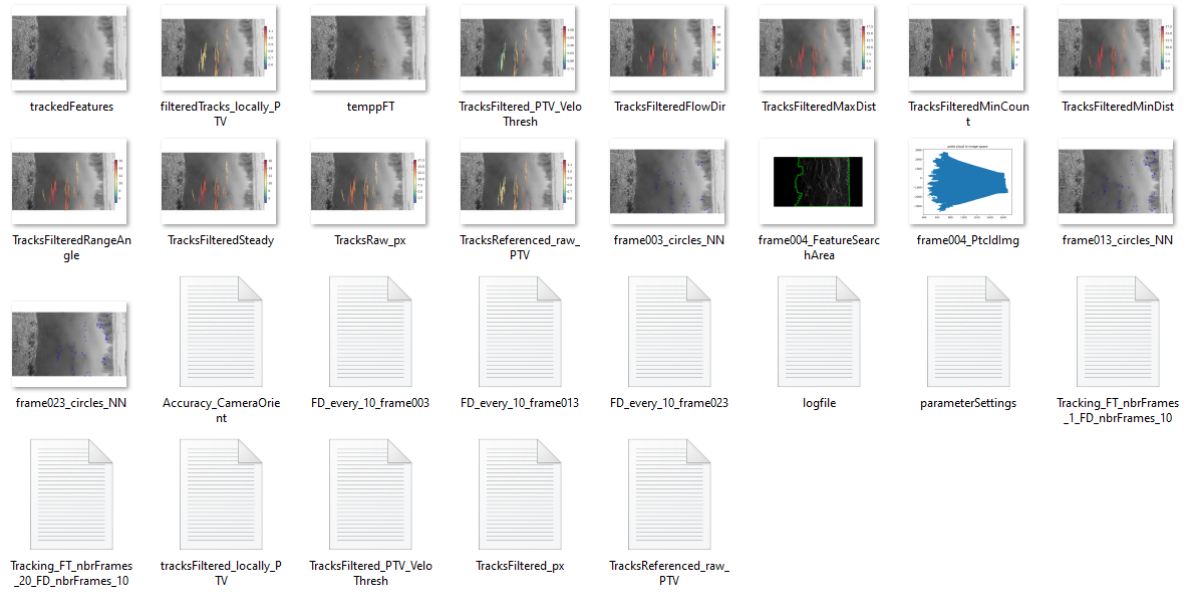


Figure 4.13: Result Flow Velocity in resultFlowVelo.file.

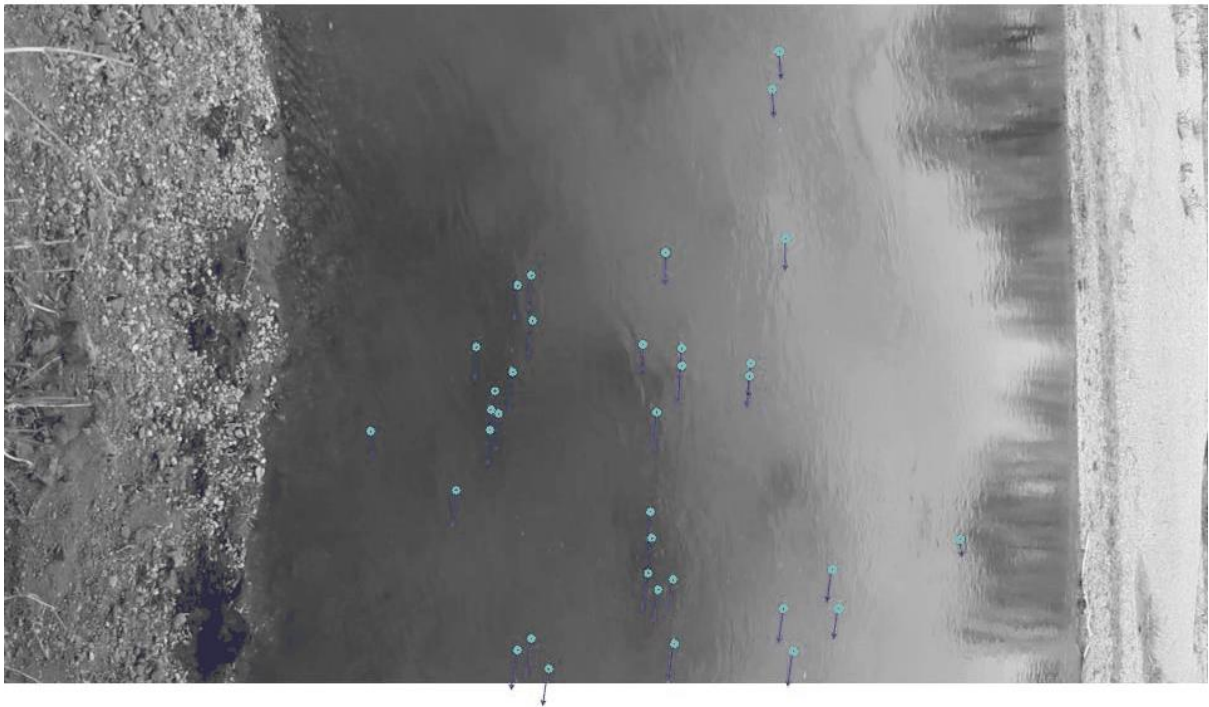


Figure 4.14: Result of tracked feature

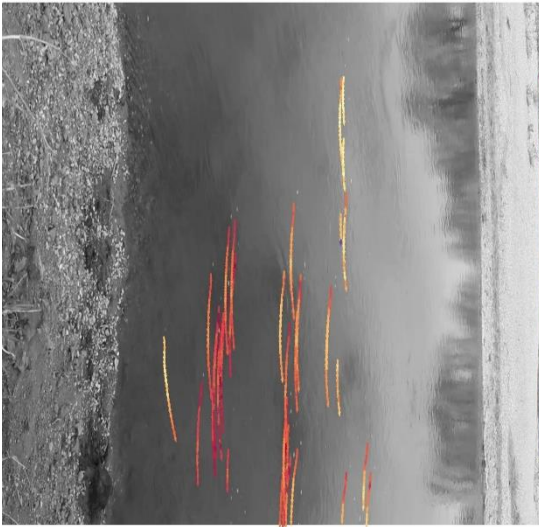


Figure 4.15: Result of minimum count filter

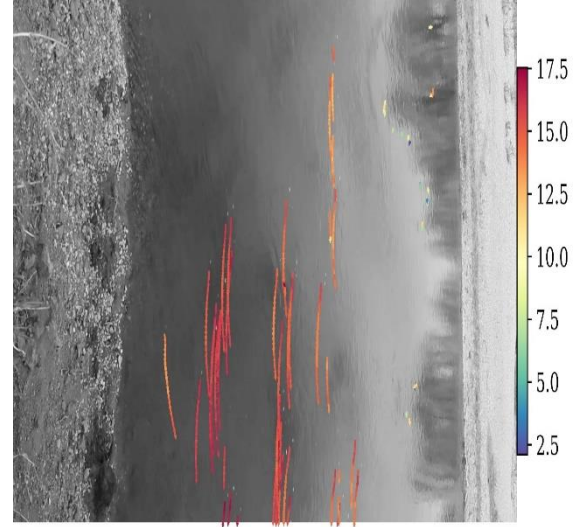


Figure 4.16: Result of minimum distance filter



Figure 4.17: Result of maximum distance filter

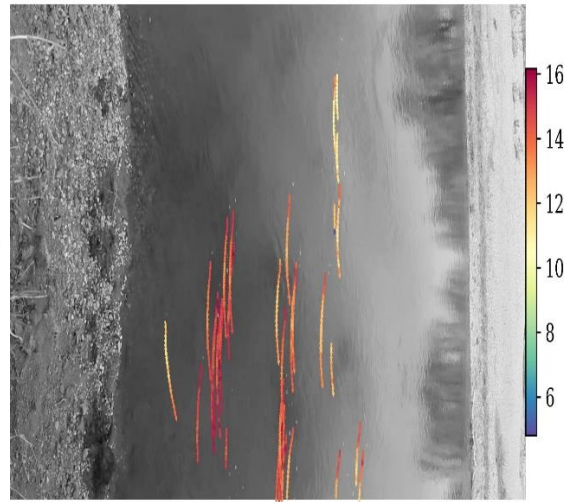


Figure 4.18: Result of steadiness filter



Figure 4.19: Result of flow direction filter

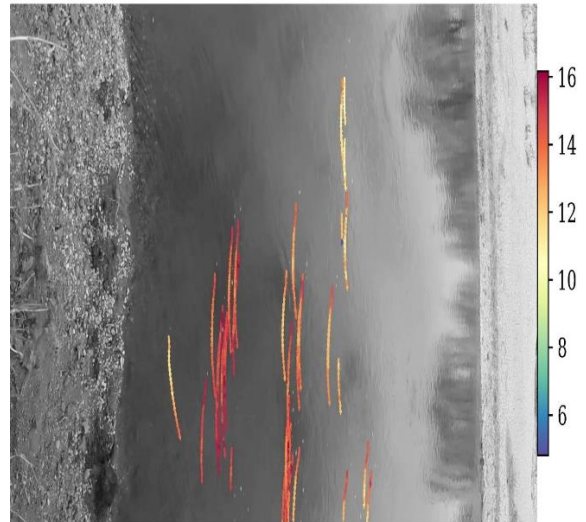


Figure 4.20: Result of flow angle range filter

The outcome of feature tracking after the image dataset has been filtered. The result of tracked features represents the detected points in Figure 4.14. Figure 4.15 illustrates the filtered tracks after applying a minimum count of subtracks requirement. Tracks that have been filtered after applying minimum and maximum distance thresholds in Figure 4.16 and 4.17. Filtered tracks after considering steadiness filtered tracks in Figure 4.18. Tracks that have been filtered after taking into account departure from the average flow direction and the variety of sub-track orientation angles in Figure 4.19 and 4.20.

Table 4.1 Discharge estimated using flow velocities and cross sections retrieved from UAV data

	Average surface flow velocity (ms^{-1})	Standard deviation (sd)	Cross-section area (m^2)	Average Discharge (m^3s^{-1})	Track count
UAV camera	0.88	0.066	9.19	8.09	9.48

The water discharge can be calculated by multiplying the cross-section area by the average surface flow velocity to obtain the discharge, which represents the volume of water passing through the cross-section per unit of time. The average surface flow velocity (0.88 ms^{-1}) and its standard deviation (0.066), indicating the velocity and variability of the water flow. The cross-section area (9.19 m^2) gives an idea of the river's size and capacity. The average discharge ($8.09 \text{ m}^3\text{s}^{-1}$) indicates the volume of water flowing through the cross-section. Additionally, it mentions the track count (9.48 units), representing the number of measurements taken.

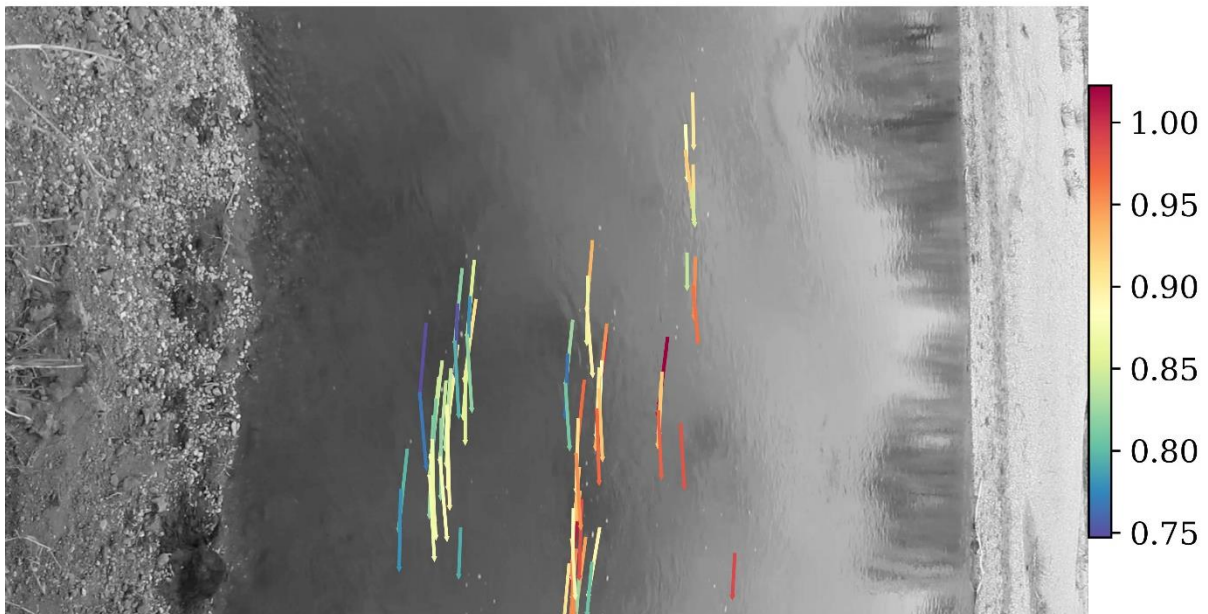


Figure 4.21: Results of flow velocity after application of statistical (threshold) velocity filter

The estimation of river flow is conducted by analyzing video frames captured through a UAV camera. Figure 4.21 showcases the resulting tracks after the implementation of a statistical outlier filter. After applying the statistical outlier filter, the Figure 4.21 showcases the final tracks that have been refined and cleaned from the presence of outliers. These tracks provide a more accurate representation of the flow patterns and dynamics within the river, enabling further analysis and interpretation of the flow characteristics.

4.6 Modelling

Based on the obtained results, the next step involves extracting the results dataset into a CSV file and proceeding to the modelling phase. In this phase, time series analysis will be employed to extract flow velocity information from the data. In this study, ARIMA and LSTM will be used to do the modelling. The resulting dataset will be loaded in Jupyter Notebook in CSV format. Since the data is absence of timestamps and the dataset has only few second of time intervals, using sequence instead of timestamp can improve the performance.

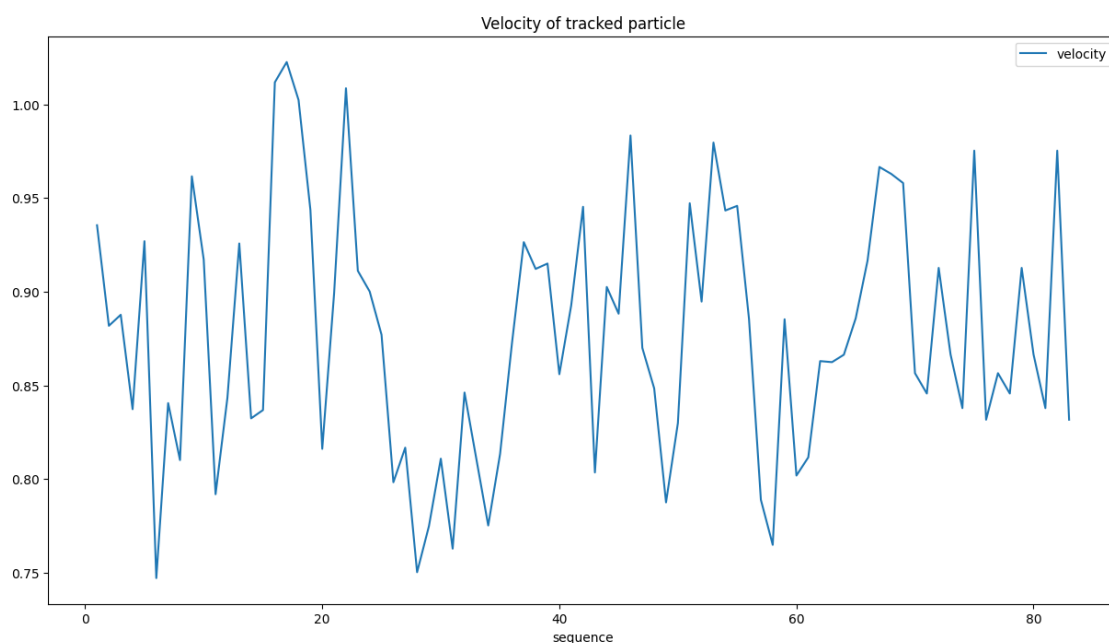


Figure 4.22: Velocity of tracked particle by each sequence

The data series exhibit an overall

- a) upward and downward trend
- b) irregular pattern
- c) non-seasonal trend

4.6.1 Data Partitioning

The partition of data into train and test set is a necessary step before training the model. In this study, the data partition was processed in Jupyter Notebook. From the data, there are 83 observations. Split the data into in-sample (training data) and out sample data (Testing). In sample is 80% of the observation, out-sample is 20% of the observation.

4.6.2 Training

The primary objective of this chapter is to employ both ARIMA and LSTM models for the training and prediction of flow velocity through time series analysis. However, before proceeding with model training, it is essential to find out the best-fitting model for ARIMA.

```
import pmdarima as pm

# Assuming df contains your time series data

# Convert the DataFrame column to a series if necessary
df = data['velocity']

# Find the best ARIMA model without considering seasonal component
model = pm.auto_arima(df, seasonal=False, trace=True)

# Print the model summary
print(model.summary())
```



```
Performing stepwise search to minimize aic
ARIMA(2,0,2)(0,0,0)[0] : AIC=inf, Time=0.55 sec
ARIMA(0,0,0)(0,0,0)[0] : AIC=216.118, Time=0.03 sec
ARIMA(1,0,0)(0,0,0)[0] : AIC=inf, Time=0.02 sec
ARIMA(0,0,1)(0,0,0)[0] : AIC=inf, Time=0.07 sec
ARIMA(1,0,1)(0,0,0)[0] : AIC=-202.468, Time=0.08 sec
ARIMA(2,0,1)(0,0,0)[0] : AIC=-208.844, Time=0.17 sec
ARIMA(2,0,0)(0,0,0)[0] : AIC=inf, Time=0.08 sec
ARIMA(3,0,1)(0,0,0)[0] : AIC=-206.911, Time=0.18 sec
ARIMA(1,0,2)(0,0,0)[0] : AIC=inf, Time=0.15 sec
ARIMA(3,0,0)(0,0,0)[0] : AIC=inf, Time=0.10 sec
ARIMA(3,0,2)(0,0,0)[0] : AIC=inf, Time=0.28 sec
ARIMA(2,0,1)(0,0,0)[0] intercept : AIC=-217.084, Time=0.08 sec
ARIMA(1,0,1)(0,0,0)[0] intercept : AIC=-219.077, Time=0.12 sec
ARIMA(0,0,1)(0,0,0)[0] intercept : AIC=-218.932, Time=0.07 sec
ARIMA(1,0,0)(0,0,0)[0] intercept : AIC=-220.811, Time=0.04 sec
ARIMA(0,0,0)(0,0,0)[0] intercept : AIC=-212.048, Time=0.03 sec
ARIMA(2,0,0)(0,0,0)[0] intercept : AIC=-219.083, Time=0.06 sec

Best model: ARIMA(1,0,0)(0,0,0)[0] intercept
Total fit time: 2.237 seconds
```

Figure 4.23: Best ARIMA model

The analysis presented in Figure 4.23 reveals that the ARIMA (1,0,0) (0,0,0) model is determined to be the most suitable model for the flow velocity dataset. With this information, we can now proceed to train the ARIMA model.

Dep. Variable:	velocity	No. Observations:	67			
Model:	ARIMA(1, 0, 0)	Log Likelihood	90.944			
Date:	Mon, 19 Jun 2023	AIC	-175.889			
Time:	10:10:42	BIC	-169.275			
Sample:	0	HQIC	-173.272			
	- 67					
Covariance Type:	opg					
	coef	std err	z	P> z	[0.025	0.975]
const	0.8758	0.013	67.353	0.000	0.850	0.901
ar.L1	0.4195	0.133	3.161	0.002	0.159	0.680
sigma2	0.0039	0.001	5.080	0.000	0.002	0.005
Ljung-Box (L1) (Q):	0.01	Jarque-Bera (JB):	0.46			
Prob(Q):	0.93	Prob(JB):	0.79			
Heteroskedasticity (H):	0.56	Skew:	0.08			
Prob(H) (two-sided):	0.18	Kurtosis:	2.63			

Figure 4.24: Results of fitting the ARIMA(1, 0, 0) model

The Figure 4.25 shows the implementation of an LSTM (Long Short-Term Memory) model using the Keras library for sequence forecasting. In this case, the “n_input” is set to 16, it means that LSTM will take into account the 16 preceding timesteps. The “n_features” will set to 1, indicating that the flow velocity dataset has one feature. In the LSTM layer, the parameter "20" is specified as LSTM units or memory cells in the layer. The activation function used for the LSTM units is "ReLU" (Rectified Linear Unit). ReLU is a well-liked option for activation functions as it introduces non-linearity and helps the model learn complex patterns and relationships. The Figure 4.26 shows the LSTM model will be trained for a total of 20 epochsa using the provided generator object and the training data.

```

from keras.preprocessing.sequence import TimeseriesGenerator

n_input = 16
n_features= 1
generator = TimeseriesGenerator(scaled_train_data, scaled_train_data, length=n_input, batch_size=1)

from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM

lstm_model = Sequential()
lstm_model.add(LSTM(20, activation='relu', input_shape=(n_input, n_features)))
lstm_model.add(Dense(1))
lstm_model.compile(optimizer='adam', loss='mse')

lstm_model.summary()

Model: "sequential"

```

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 20)	1760
dense (Dense)	(None, 1)	21

```

=====
Total params: 1,781
Trainable params: 1,781
Non-trainable params: 0

```

Figure 4.25: Parameters of LSTM model

```

Epoch 1/20
51/51 [=====] - 2s 6ms/step - loss: 0.2035
Epoch 2/20
51/51 [=====] - 0s 7ms/step - loss: 0.0801
Epoch 3/20
51/51 [=====] - 0s 6ms/step - loss: 0.0651
Epoch 4/20
51/51 [=====] - 0s 6ms/step - loss: 0.0647
Epoch 5/20
51/51 [=====] - 1s 11ms/step - loss: 0.0619
Epoch 6/20
51/51 [=====] - 1s 13ms/step - loss: 0.0633
Epoch 7/20
51/51 [=====] - 1s 13ms/step - loss: 0.0601
Epoch 8/20
51/51 [=====] - 1s 13ms/step - loss: 0.0605
Epoch 9/20
51/51 [=====] - 1s 13ms/step - loss: 0.0600
Epoch 10/20
51/51 [=====] - 1s 13ms/step - loss: 0.0585
Epoch 11/20
51/51 [=====] - 1s 12ms/step - loss: 0.0567
Epoch 12/20
51/51 [=====] - 0s 6ms/step - loss: 0.0560
Epoch 13/20
51/51 [=====] - 0s 6ms/step - loss: 0.0553
Epoch 14/20
51/51 [=====] - 0s 6ms/step - loss: 0.0545
Epoch 15/20
51/51 [=====] - 0s 6ms/step - loss: 0.0538
Epoch 16/20
51/51 [=====] - 0s 6ms/step - loss: 0.0559
Epoch 17/20
51/51 [=====] - 0s 6ms/step - loss: 0.0534
Epoch 18/20
51/51 [=====] - 0s 6ms/step - loss: 0.0522
Epoch 19/20
51/51 [=====] - 0s 6ms/step - loss: 0.0505
Epoch 20/20
51/51 [=====] - 0s 6ms/step - loss: 0.0512

```

Figure 4.26: LSTM model trained

4.6.3 Result ARIMA and LSTM model predictions

	velocity	ARIMA_Predictions	LSTM_Predictions
sequence			
68	0.962832	0.891786	0.894397
69	0.957976	0.882517	0.887283
70	0.856528	0.878629	0.882919
71	0.845674	0.876998	0.879213
72	0.912705	0.876313	0.876038
73	0.866548	0.876026	0.873340
74	0.837886	0.875906	0.871081
75	0.975255	0.875855	0.869185
76	0.831703	0.875834	0.867639
77	0.856528	0.875825	0.866379
78	0.845674	0.875822	0.865361
79	0.912705	0.875820	0.864554
80	0.866548	0.875819	0.863908
81	0.837886	0.875819	0.863396
82	0.975255	0.875819	0.863001
83	0.831703	0.875819	0.862703

Figure 4.27: Comparison Actual flow velocity between ARIMA predictions and LSTM predictions

Comparing the actual values with the model predictions, we can see that the ARIMA predictions and LSTM predictions are relatively close to the actual values. However, there are slight differences between the actual and prediction values for each sequence.

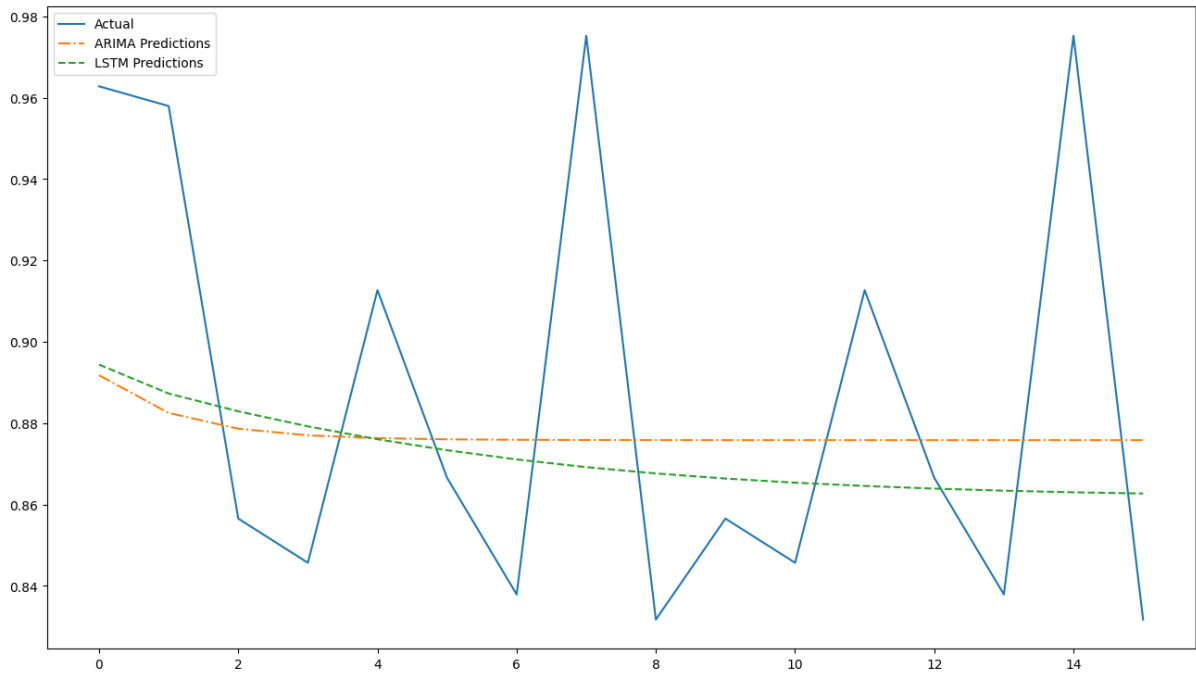


Figure 4.28: Comparison actual flow velocity between ARIMA predictions and LSTM predictions

4.6.4 Evaluation

Evaluating the result of the ARIMA and LSTM models in predicting water flow velocity entails a comprehensive analysis of how accurately their forecasts align with the actual observed values. This assessment helps determine the effectiveness and reliability of the models in capturing the underlying patterns and dynamics of the water flow velocity data.

Table 4.2 Errors of surface flow velocity between ARIMA and LSTM.

Model	MAE	RMSE	MSE
ARIMA	0.042394	0.049488	0.002449
LSTM	0.041676	0.052177	0.002722

Considering the RMSE and MSE values, a lower value shows better model performance in terms of prediction accuracy. Comparing the two models, the ARIMA model has a lower RMSE (0.049488) and MSE (0.002449) compared to the LSTM model's RMSE (0.052177) and MSE (0.002722). This suggests that the ARIMA model is providing slightly better predictions on this small dataset. Additionally, looking at the MAE values, we observe that the LSTM model has a lower MAE of 0.041676, while the ARIMA model has a slightly higher MAE of 0.042394.

Overall, considering all the metrics, both models perform relatively well on this dataset, but the ARIMA model showcases slightly better accuracy based on RMSE and MSE, while the LSTM model demonstrates slightly better accuracy based on MAE.

4.7 Discussion

This study explores the application of FlowVelo to PTV using UAVs is presented, since it is a reliable measurement method. It shows how the presented measurement technique can be used to determine the flow velocity. The UAVs were deployed at a flight height of 15m to capture video for surface velocity measurement. The findings reveal that the average velocity measured was 0.88 m/s, considering a cross section area of $9.19m^2$. The maximum velocity recorded was 1.02 m/s while the minimum was 0.747 m/s. The MAEs were calculate for both arima and lstm models, resulting of 0.0424 and 0.0417. This study underscores the influence of flight height on the analysis of surface velocity. The results indicate that varying flight heights can lead to variations in the measured velocities at different heights. Such insights contribute to a better understanding of the impact of flight parameters on the accuracy and precision of surface velocity measurements.

The application requires the video frame, camera pose, interior camera geometry, and GCP information to processing. The image sequences needed to be projected onto a consistent spatial reference to maintain accurate image velocimetry procedures (Detert, 2021). In this study, interior camera parameters and camera poses of header frames were calculated using Ground Control Points (GCPs). The focal length, sensor size, and resolution are needed based on the UAV information. It is also necessary to evaluate the frame co-registration to ensure

that the tracked particle movements are indeed those of the river instead of the camera. In the image velocimetry analysis, geometrically stable images were used to compensate for horizontal plane movement (McIlvenny et al., 2022). In addition, if enough shorelines are shown in the frames, any camera movement that occurred during the video capture can be automatically corrected. Raising the flying height could be necessary to encompass sufficiently large and stable areas for larger rivers, but it will reduce the visibility of feature tracking. Hence, a camera with wider opening angles might be required, which might lead to significant lens distortion.

The FlowVelo tool requires experience and some trial and error to find the most appropriate filtering parameters for successful filtering in riverine environments. The filter might lose valid velocity traces if the parameter is selected too precisely, which is especially true in rivers that have complicated flow patterns. Measuring the surface velocity is to outside influences such as wind, waves, or raindrops, which can distort the established relationship between the water surface and average flow velocity. This is because the surface velocity decreases or increases depending on wind and wave direction. Traceable particles can be used to increase the seeding density to make surface velocity measurement more accurate and reliable (Detert et al., 2017).

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Introduction

In this chapter, we will present our conclusions and recommendations derived from the application of the FlowVelo tool. Through our analysis and experimentation, we have gained valuable insights that contribute to our understanding of flow velocity prediction and its practical implementation.

5.2 Conclusions

In conclusion, this study presented a workflow for measuring flow velocity and discharge estimation with video captured by a UAV. It can be applied to both terrestrial and aerial imagery. A co-registration algorithm automatically stabilizes the camera movements during the video capture. Flow velocity is estimated by detecting and tracking particles on the water surface using PTV method. The average flow velocity and the wetted cross section are used to retrieve the discharge. According to the results, surface flow velocities calculated with a maximum error of 6% have a high accuracy potential compared to arima and lstm models. The FlowVelo tool focuses on the camera calibration process to increase accuracy and able to provide a realistic flow velocity. The field measurement calibration method that is described can assist in overcoming challenges when shifting to large-scale and more complicated measurement environments.

This tool has low error for measuring the parameter of river water flow at a low cost and with ease of deployment and maintenance. As a result of the velocity tracking, surface velocity can be measured spatially and discharge can be estimated in previously unmeasured and ungauged areas, making it ideal for flood event assessment. Additionally, this research assists the community and government in making flood prevention plans. Thus, our workflow can be

very valuable for future applications observing water flow changes due to changing water levels.

5.3 Recommendations

In future studies, it is desirable to reduce the parameter settings so that the video can be captured directly in the field next to the river to reduce the loss of probability of system settings. It can also systematically and statistically increase the observation density of suitable cross-sections for flow estimation. This workflow has the potential to be expanded to include direct georeferencing and capturing data on river arrival velocity even during extreme flood events. This feature is particularly valuable for understanding and managing rapidly changing water flow dynamics. Furthermore, the field measurement calibration method that is described can assist in overcoming challenges when shifting to large-scale and more complicated measurement environments.

REFERENCES

A STUDY ON CAUSES OF FLOOD AT SUNGAI ISAP AND JALAN TUN ISMAIL,. (n.d.).

- Aleixo, R., Soares-Frazão, S., & Zech, Y. (2011). Velocity-field measurements in a dam-break flow using a PTV Voronoï imaging technique. *Experiments in Fluids*, 50, 1633–1649.
- Bradski, G. (2000). The Opencv Library. In *Dr. Dobb's J. Softw. Tools* (Vol. 25).
- Brunton, S. L., Noack, B. R., & Koumoutsakos, P. (2020). Machine Learning for Fluid Mechanics. *Annu. Rev. Fluid Mech.*, 52(1), 477508. <https://doi.org/10.1146/annurev-fluid-010719-060214>
- Charles, R. Q., Su, H., Kaichun, M., & Guibas, L. J. (2017). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 77–85. <https://doi.org/10.1109/CVPR.2017.16>
- Chen, Y., Aggarwal, P., Choi, J., & Jay, C. C. (2018). A deep learning approach to drone monitoring. *Proceedings - 9th Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2017, 2018-February*, 686–691. <https://doi.org/10.1109/APSIPA.2017.8282120>
- Chetverikov, D. (2001). Particle Image Velocimetry by Feature Tracking. In W. Skarbek (Ed.), *Computer Analysis of Images and Patterns* (pp. 325–332). Springer Berlin Heidelberg.
- D. N. Moriasi, J. G. Arnold, M. W. Van Liew, R. L. Bingner, R. D. Harmel, & T. L. Veith. (2007). Model Evaluation Guidelines for Systematic Quantification of Accuracy in Watershed Simulations. *Transactions of the ASABE*, 50(3), 885–900. <https://doi.org/10.13031/2013.23153>
- de Oliveira Fleury, G. R., do Nascimento, D. V., Galvão Filho, A. R., Lima Ribeiro, F. D. S., de Carvalho, R. V., & Coelho, C. J. (2020). Image-based river water level estimation for redundancy information using deep neural network. *Energies*, 13(24). <https://doi.org/10.3390/en13246706>
- Detert, M. (2021). How to Avoid and Correct Biased Riverine Surface Image Velocimetry. In *Water Resources Research* (Vol. 57, Issue 2). Blackwell Publishing Ltd. <https://doi.org/10.1029/2020WR027833>
- di Cristo, C. (2011). Particle Imaging Velocimetry and Its Applications in Hydraulics: A State-of-the-Art Review. *GeoPlanet: Earth and Planetary Sciences*, 1, 49–66. https://doi.org/10.1007/978-3-642-17475-9_3
- Di, L., Yu, E. G., Kang, L., Shrestha, R., & BAI, Y. qi. (2017). RF-CLASS: A remote-sensing-based flood crop loss assessment cyber-service system for supporting crop statistics and insurance decision-making. *Journal of Integrative Agriculture*, 16(2), 408–423. [https://doi.org/10.1016/S2095-3119\(16\)61499-5](https://doi.org/10.1016/S2095-3119(16)61499-5)
- Dobriyal, P., Badola, R., Tuboi, C., & Hussain, S. A. (2017). A review of methods for monitoring streamflow for sustainable water resource management. In *Applied Water Science* (Vol. 7, Issue 6, pp. 2617–2628). Springer Verlag. <https://doi.org/10.1007/s13201-016-0488-y>

- Douglas, I., Alam, K., Maghenda, M., McDonnell, Y., Mclean, L., & Campbell, J. (2008). Unjust waters: climate change, flooding and the urban poor in Africa. *Environment and Urbanization*, 20(1). <https://doi.org/10.1177/0956247808089156>
- Eltner, A., Sardemann, H., & Grundmann, J. (2020). Technical Note: Flow velocity and discharge measurement in rivers using terrestrial and unmanned-aerial-vehicle imagery. *Hydrology and Earth System Sciences*, 24(3), 1429–1445. <https://doi.org/10.5194/hess-24-1429-2020>
- Foley, J. D., Fischler, M. A., & Bolles, R. C. (1981). *Graphics and Image Processing Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography*.
- Fujita, I., Notoya, Y., Tani, K., & Tateguchi, S. (2019). Efficient and accurate estimation of water surface velocity in STIV. *Environmental Fluid Mechanics*, 19(5), 1363–1378. <https://doi.org/10.1007/s10652-018-9651-3>
- Fulton, J. W., Mason, C. A., Eggleston, J. R., Nicotra, M. J., Chiu, C. L., Henneberg, M. F., Best, H. R., Cederberg, J. R., Holnbeck, S. R., Lotspeich, R. R., Laveau, C. D., Moramarco, T., Jones, M. E., Gourley, J. J., & Wasielewski, D. (2020). Near-field remote sensing of surface velocity and river discharge using radars and the probability concept at 10 U.S. geological survey streamgages. *Remote Sensing*, 12(8). <https://doi.org/10.3390/RS12081296>
- Genc, O., Ardiclioglu, M., & Ağiralioglu, N. (2015). Calculation of mean velocity and discharge using water surface velocity in small streams. *Flow Measurement and Instrumentation*, 41, 115–120.
- Gleason, C. J., & Durand, M. T. (2020). Remote Sensing of River Discharge: A Review and a Framing for the Discipline. *Remote. Sens.*, 12, 1107.
- Grant, I., & Pan, X. (1995). An investigation of the performance of multi layer, neural networks applied to the analysis of PIV images. *Experiments in Fluids*, 19(3), 159–166. <https://doi.org/10.1007/BF00189704>
- Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5), 602–610. <https://doi.org/https://doi.org/10.1016/j.neunet.2005.06.042>
- Griesbaum, L., Marx, S., & Höfle, B. (2017). Direct local building inundation depth determination in 3-D point clouds generated from user-generated flood images. *Natural Hazards and Earth System Sciences*, 17(7), 1191–1201. <https://doi.org/10.5194/nhess-17-1191-2017>
- Han, X., Chen, K., Zhong, Q., Chen, Q., Wang, F., & Li, D. (2021). Two-Dimensional Space-Time Image Velocimetry for Surface Flow Field of Mountain Rivers Based on UAV Video. *Frontiers in Earth Science*, 9. <https://doi.org/10.3389/feart.2021.686636>
- Harris, C. G., & Stephens, M. J. (1988). A Combined Corner and Edge Detector. *Alvey Vision Conference*.
- Hauet, A., Morlot, T., & Daubagnan, L. (2018). Velocity profile and depth-averaged to surface velocity in natural streams: A review over a large sample of rivers. *E3S Web of Conferences*, 40. <https://doi.org/10.1051/e3sconf/20184006015>

- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. *Neural Computation*, 9, 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Jiwani, R., Steffen Lucas, P., & Eng, D. (n.d.). *Methods of Flow Measurement for Water and Wastewater*.
- Khalid, M., Pénard, L., Mémin, E., Khalid, M., Pénard, L., & Mémin, E. (2019). *Optical Flow For Image-Based River Velocity Estimation. Flow Measurement and Instrumentation*. 65, 110–121. <https://doi.org/10.1016/j.flowmeasinst.2018.11.009>
- Kimura, N., Liu, W., Wu, C. H., Bechle, A. J., Chen, W.-B., & Huang, W. (2011). Flow measurement with multi-instrumentation in a tidal-affected river. *Water and Environment Journal*, 25.
- Koutalakis, P., & Zaimes, G. N. (2022a). River Flow Measurements Utilizing UAV-Based Surface Velocimetry and Bathymetry Coupled with Sonar. *Hydrology*, 9(8). <https://doi.org/10.3390/hydrology9080148>
- Koutalakis, P., & Zaimes, G. N. (2022b). River Flow Measurements Utilizing UAV-Based Surface Velocimetry and Bathymetry Coupled with Sonar. *Hydrology*, 9(8). <https://doi.org/10.3390/hydrology9080148>
- Lee, C.-C., Yang, C.-H., Liu, H.-C., Wen, K.-L., Wang, Z.-B., & Chen, Y.-J. (2008). A Study of the hydrogeological environment of the lishan landslide area using resistivity image profiling and borehole data. *Engineering Geology*, 98(3), 115–125. <https://doi.org/https://doi.org/10.1016/j.enggeo.2008.01.012>
- Lewis, Q., & Rhoads, B. (2018). LSPIV Measurements of Two-Dimensional Flow Structure in Streams Using Small Unmanned Aerial Systems: 1. Accuracy Assessment Based on Comparison With Stationary Camera Platforms and In-Stream Velocity Measurements. *Water Resources Research*, 54. <https://doi.org/10.1029/2018WR022550>
- LI, D., ZHONG, Q., YU, M., & WANG, X. (2013). Large-scale particle tracking velocimetry with multi-channel CCD cameras. *International Journal of Sediment Research*, 28(1), 103–110. [https://doi.org/https://doi.org/10.1016/S1001-6279\(13\)60022-0](https://doi.org/https://doi.org/10.1016/S1001-6279(13)60022-0)
- Lin, F., Chang, W. Y., Lee, L. C., Hsiao, H. T., Tsai, W. F., & Lai, J. S. (2013). Applications of image recognition for real-time water level and surface velocity. *Proceedings - 2013 IEEE International Symposium on Multimedia, ISM 2013*, 259–262. <https://doi.org/10.1109/ISM.2013.49>
- Liu, P. S., & Chan, N. W. (2003). The Malaysian flood hazard management program. *International Journal of Emergency Management*, 1(3), 205–214. <https://doi.org/10.1504/IJEM.2003.003303>
- Lloyd, P. M., Stansby, P. K., & Ball, D. J. (1995). Unsteady surface-velocity field measurement using particle tracking velocimetry. *Journal of Hydraulic Research*, 33, 519–534.
- Mallery, K., Shao, S., & Hong, J. (2020). Dense particle tracking using a learned predictive model. *Experiments in Fluids*, 61(10). <https://doi.org/10.1007/s00348-020-03061-y>
- Mansour, H. A., Abdallah, E. F., & Gaballah, M. S. (2015). IMPACT OF BUBBLER DISCHARGE AND IRRIGATION WATER QUANTITY ON 1- HYDRAULIC PERFORMANCE EVALUATION AND MAIZE

- BIOMASS YIELD. *International Journal of GEOMATE : Geotechnique, Construction Materials and Environment*, 9, 1538–1544.
- McIlvenny, J., Williamson, B. J., Fairley, I. A., Lewis, M., Neill, S., Masters, I., & Reeve, D. E. (2022). Comparison of dense optical flow and PIV techniques for mapping surface current flow in tidal stream energy sites. *International Journal of Energy and Environmental Engineering*. <https://doi.org/10.1007/s40095-022-00519-z>
- Mittal, P., Singh, R., & Sharma, A. (2020). Deep learning-based object detection in low-altitude UAV datasets: A survey. In *Image and Vision Computing* (Vol. 104). Elsevier Ltd. <https://doi.org/10.1016/j.imavis.2020.104046>
- Mohanty, M. P., Mudgil, S., & Karmakar, S. (2020). Flood management in India: A focussed review on the current status and future challenges. *International Journal of Disaster Risk Reduction*, 49, 101660. <https://doi.org/https://doi.org/10.1016/j.ijdrr.2020.101660>
- Niakšu, O. (2015). CRISP Data Mining Methodology Extension for Medical Domain. In *Baltic J. Modern Computing* (Vol. 3, Issue 2). <https://www.researchgate.net/publication/277775478>
- Park, K., Jung, Y., Kim, K., & Park, S. K. (2020). Determination of deep learning model and optimum length of training data in the river with large fluctuations in flow rates. *Water (Switzerland)*, 12(12). <https://doi.org/10.3390/w12123537>
- Pereira, F., Stürer, H., Graff, E. C., & Gharib, M. (2006). Two-frame 3D particle tracking. *Measurement Science and Technology*, 17(7), 1680. <https://doi.org/10.1088/0957-0233/17/7/006>
- Reyes-Avila, J., Razo-Flores, E., & Gomez, J. (2004). Simultaneous biological removal of nitrogen, carbon and sulfur by denitrification. *Water Research*, 38(14–15), 3313–3321. <https://doi.org/10.1016/j.watres.2004.04.035>
- Rizkya, I., Syahputri, K., Sari, R. M., Siregar, I., & Utaminingrum, J. (2019). Autoregressive Integrated Moving Average (ARIMA) Model of Forecast Demand in Distribution Centre. *IOP Conference Series: Materials Science and Engineering*, 598(1). <https://doi.org/10.1088/1757-899X/598/1/012071>
- Rozos, E., Dimitriadis, P., Mazi, K., Lykoudis, S., & Koussis, A. D. (2020). On the Uncertainty of the Image Velocimetry Method Parameters. *Hydrology*.
- Sanders, L. L. (1998). *A manual of field hydrogeology*. Prentice Hall.
- Shi, J., & Tomasi. (1994). Good features to track. *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 593–600. <https://doi.org/10.1109/CVPR.1994.323794>
- Sirenden, B. H., Mursanto, P., & Wijonarko, S. (2022). *Dynamic Texture Analysis of Water Flow Video Using Temporal Binary Pattern Image Histogram Neural Network*. <https://doi.org/10.20944/preprints202205.0233.v1>
- Sun, C., Zhan, W., She, J., & Zhang, Y. (2020). Object Detection from the Video Taken by Drone via Convolutional Neural Networks. *Mathematical Problems in Engineering*, 2020. <https://doi.org/10.1155/2020/4013647>

- Tauro, F. (2016). Particle tracers and image analysis for surface flow observations. *Wiley Interdisciplinary Reviews: Water*, 3.
- Tauro, F., Piscopia, R., & Grimaldi, S. (2019). PTV-Stream: A simplified particle tracking velocimetry framework for stream surface flow monitoring. *Catena*, 172, 378–386. <https://doi.org/10.1016/j.catena.2018.09.009>
- Tauro, F., Tosi, F., Mattoccia, S., Toth, E., Piscopia, R., & Grimaldi, S. (2018). Optical Tracking Velocimetry (OTV): Leveraging Optical Flow and Trajectory-Based Filtering for Surface Streamflow Observations. *Remote. Sens.*, 10, 2010.
- Trieu, H., Bergström, P., Sjö Dahl, M., Hellström, J. G. I., Andreasson, P., & Lycksam, H. (2021a). Photogrammetry for free surface flow velocity measurement: From laboratory to field measurements. *Water (Switzerland)*, 13(12). <https://doi.org/10.3390/w13121675>
- Trieu, H., Bergström, P., Sjö Dahl, M., Hellström, J. G. I., Andreasson, P., & Lycksam, H. (2021b). Photogrammetry for free surface flow velocity measurement: From laboratory to field measurements. *Water (Switzerland)*, 13(12). <https://doi.org/10.3390/w13121675>
- Vieira Do Nascimento, D., Rodrigues, A., Filho, G., De Oliveira Fleury, G. R., Carvalho, R. V., De Souza, F., Ribeiro, L., & Coelho, C. J. (n.d.). *Automatic measurement of river water-level using image-based computer vision*.
- Wu, H., Zhao, R., Gan, X., & Ma, X. (2019). Measuring surface velocity of water flow by dense optical flow method. *Water (Switzerland)*, 11(11). <https://doi.org/10.3390/w11112320>
- Yu, K., & Lee, J. (2023). Method for Measuring the Surface Velocity Field of a River Using Images Acquired by a Moving Drone. *Water (Switzerland)*, 15(1). <https://doi.org/10.3390/w15010053>
- Zaimes, G. N., Iakovoglou, V., Syropoulos, D., Kaltsas, D., & Avtzis, D. (2021). Article assessment of two adjacent mountainous riparian areas along nestos river tributaries of greece. *Forests*, 12(9). <https://doi.org/10.3390/f12091284>
- Zhang, P. (2003). Zhang, G.P.: Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model. *Neurocomputing* 50, 159-175. *Neurocomputing*, 50, 159–175. [https://doi.org/10.1016/S0925-2312\(01\)00702-0](https://doi.org/10.1016/S0925-2312(01)00702-0)

APPENDICES

Appendix A: Times Series model ARIMA and LSTM for Flow Velocity

```
# import libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.seasonal import seasonal_decompose
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, LSTM, SimpleRNN
from sklearn.preprocessing import MinMaxScaler
# from pmdarima import auto_arima
from sklearn.metrics import mean_squared_error, mean_absolute_error,
    r2_score
from statsmodels.tools.eval_measures import rmse
df = pd.read_csv("flowvelocity.csv", header = 0, index_col = 0)
df.rename(columns={'velo': 'velocity'}, inplace=True)
df.rename(columns={'dist': 'distance'}, inplace=True)
df

df.isna().mean()
df.mean()
data = df.iloc[:, [3]]
data

# plot the time series data
from matplotlib import pyplot
data.plot(figsize = (15, 8), title='Velocity of tracked particle',
    fontsize=10)
pyplot.show()

#
import pmdarima as pm

# Assuming df contains your time series data

# Convert the DataFrame column to a series if necessary
df = data['velocity']
```

```

# Find the best ARIMA model without considering seasonal component
model = pm.auto_arma(df, seasonal=False, trace=True)

# Print the model summary
print(model.summary())

#Split data in-sample & out-sample
training = data[:len(df)-16] #67 in-sample data
testing = data[len(df)-16:] #16 out-sample data

training
testing

arma_model = ARIMA(training['velocity'], order = (1,0,0),
                    seasonal_order = (0,0,0,0))
arma_result = arma_model.fit()
arma_result.summary()

arma_pred = arma_result.predict(start = len(training), end =
                               len(data), typ="levels").rename("ARIMA Predictions")
arma_pred

testing['velocity'].plot(figsize = (16,5), legend=True)
arma_pred.plot(legend = True);

testing['ARIMA_Predictions'] = arma_pred
testing

# Assuming `testing['velocity']` and `arma_pred` have different
lengths
# Trim `arma_pred` to match the length of `testing['velocity']`
arma_pred = arma_pred[:len(testing['velocity'])]
arma_rmse_error = rmse(testing['velocity'], arma_pred)
arma_mse_error = arma_rmse_error ** 2
mean_value = data['velocity'].mean()
arma_mae_error = mean_absolute_error(testing['velocity'], arma_pred)

print(f"MAE: {arma_mae_error}")
print(f"MSE          Error:          {arma_mse_error}\nRMSE          Error:
      {arma_rmse_error}\nMean: {mean_value}")

# LSTM
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()

scaler.fit(training[['velocity']])

```

```

scaled_train_data = scaler.transform(training[['velocity']])
scaled_test_data = scaler.transform(testing[['velocity']])

from keras.preprocessing.sequence import TimeseriesGenerator

n_input = 16
n_features= 1
generator = TimeseriesGenerator(scaled_train_data, scaled_train_data,
                                length=n_input, batch_size=1)

from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM

lstm_model = Sequential()
lstm_model.add(LSTM(20, activation='relu', input_shape=(n_input,
                                                         n_features)))
lstm_model.add(Dense(1))
lstm_model.compile(optimizer='adam', loss='mse')

lstm_model.summary()

lstm_model.fit_generator(generator,epochs=20)

losses_lstm = lstm_model.history.history['loss']
plt.figure(figsize=(12,4))
plt.xticks(np.arange(0,21,1))
plt.plot(range(len(losses_lstm)),losses_lstm);

lstm_predictions_scaled = list()

batch = scaled_train_data[-n_input:]
current_batch = batch.reshape((1, n_input, n_features))

for i in range(len(testing)):
    lstm_pred = lstm_model.predict(current_batch)[0]
    lstm_predictions_scaled.append(lstm_pred)
    current_batch = np.append(current_batch[:,1:,:],[[lstm_pred]],axis=1)

lstm_predictions_scaled

lstm_predictions = scaler.inverse_transform(lstm_predictions_scaled)
lstm_predictions

testing['LSTM_Predictions'] = lstm_predictions

```

```

testing['velocity'].plot(figsize = (16,5), legend=True)
testing['LSTM_Predictions'].plot(legend = True);

lstm_rmse_error = rmse(testing['velocity'],
    testing["LSTM_Predictions"])
lstm_mse_error = lstm_rmse_error**2
lstm_mean_value = data['velocity'].mean()
lstm_mae_error = mean_absolute_error(testing['velocity'],
    testing["LSTM_Predictions"])

print(f"MAE: {lstm_mae_error}")
print(f"MSE Error: {lstm_mse_error}\nRMSE Error:
    {lstm_rmse_error}\nMean: {lstm_mean_value}")

rmse_errors = [arima_rmse_error, lstm_rmse_error]
mse_errors = [arima_mse_error, lstm_mse_error]
mean_absolute_error = [arima_mae_error, lstm_mae_error]
errors = pd.DataFrame({"Models" : ["ARIMA", "LSTM"], "MAE" :
    mean_absolute_error, "RMSE" : rmse_errors, "MSE" : mse_errors})

import matplotlib.pyplot as plt

plt.figure(figsize=(16,9))
plt.plot(range(len(testing)), testing["velocity"], linestyle="-",
    label="Actual")
plt.plot(range(len(testing)), testing["ARIMA_Predictions"],
    linestyle="-. ", label="ARIMA Predictions")
plt.plot(range(len(testing)), testing["LSTM_Predictions"],
    linestyle="--", label="LSTM Predictions")
plt.legend()
plt.show()

print(f"Mean: {testing['velocity'].mean()}")
errors

testing

```