Handbook: On the creation of large-scale image velocimetry sites

Large-scale image velocimetry in rivers

Author: Martin Jolley

Institute: Newcastle University

Date: January 2022

Version: 1.1

With collaboration from the Scottish Environment Protection Agency (SEPA)

Contents

Introduction

More frequent, high intensity storms and flooding means that the understanding of flow regimes is as important as ever, and advances in the technology that we use has been seen with the use of novel introduction of new methods and techniques (e.g. deep learning models [1], satellite mapping [2]). One of the most widely adopted new technologies is the introduction of acoustic Doppler current profiler's (aDcp)[3]. Although this method allows users to accurately capture entire water columns along a profile [4], the issue is that it requires contact with the water surface, which can be problematic in areas witnessing high flows which are inaccessible due to safety concerns for either the user or the equipment. An alternative approach to this is to capture flow data in a non-contact manor. A technique that allows the user to do this is image velocimetry and in particular, large scale image velocimetry [5–8]. Large-scale image velocimetry can be an autonomous method of flow gauging a site using surface velocities captured via a camera, and processed using feature detection and tracking software [9–11]. Consequently, discharge can be calculated from the surface velocity using estimates of alpha values (depth-average velocity values), and cross-sectional data [12, 13].

This book is designed to be of use to those wishing to create large-scale image velocimetry sites. There are many methods to use to apply image velocimetry with (e.g. hardware used, software suites used, system settings). To ensure best-practice, it is important to fully understand which methods are best for a particular site, as well fully defining what it is you wish to capture (e.g. high flows, low flows, continuous flows, surface flow patterns).

Each section of this book adds a level of depth to the discussions around the best-practice for image velocimetry using a combination of experience, available literature, and study-based precedence. We begin in chapter 1 by looking at some of the key pieces of terminology that relates back to image velocimetry and the basics of what they mean, and how they are typically applied in a general sense. Chapter 2 is a summary of a literature review paper produced by Jolley et al. [14]. This section is dedicated to the more technical aspects of using PIV/PTV software and will expand on some of the terms introduced from chapter 1.

The subsequent three chapters (chapters 3, 4, 5), consider some more practical elements that are vital to the use of image velocimetry, including hardware considerations, software considerations, and the code requirements respectively. Finally, chapter 6 looks at providing an overall recommendation on how to best set-up a site depending on three key-factors: overall aim, budget, size of site. The key-factors identified have been chosen as they influence the main decisions around how to best capture data using either: fixed or mobile stations (aim & budget), software suite used (aim & budget), camera calibration settings (budget & size of site), processing and storage requirements (aim, budget & size of site), etc.

There are, of course, many variations of methods that can be used to acquire and process image velocimetry data, however, for years this list of methods has been expanding which does not help those looking to find the best solution to their individual needs. The aim of this handbook is to narrow down the methodology and decisions required to begin using image velocimetry. The methods seen in this handbook is non-exhaustive and it could be argued that there are better methods for particular sites/challenges. Despite this, the need for a generalised 'best-practice' is great, and with the information presented throughout this handbook, it should be much clearer for those new to the topic to understand what is required for them to undergo reliable gauging using PIV or PTV.

Chapter 1 Introducing Terminology

It is important to first establish a base understanding of some of the terminology that is to be used throughout this handbook. To do this, a list below has been created that summarises some of the key terms, abbreviations, and methods associated with the processing of image velocimetry data. Although this list is non-exhaustive, it includes the key parameters that users will need when first installing sites, how they can be used, and why they are important.

aDcp - *acoustic Doppler current profiler*

An aDcp is typically used to determine the flow across a cross-section in rivers, but also presents cross-sectional areas and vertical velocity columns. The aDcp's use sound waves by bouncing frequencies off of neutrally buoyant particles in the water and produce instantaneous current velocity [15].

Alpha Coefficient/Value

The alpha coefficient is a depth-average velocity value that is associated with bed friction. Alpha coefficients are specific to each site and should be calculated where possible. It is possible to calculate alpha values using aDcp's, but it has been shown that an average of 0.85 can be used with a good level of confidence, as research typically shows values between 0.79-0.91 [13].

Nadir

Nadir is a direction of which a camera is typically described should it be plum. This means that it is perfectly perpendicular to the plane and follows the line of gravity (towards the centre of the earth); essentially, pointing directly down.

Flow State

The flow state of a river can either be steady or unsteady. Steady flow has little acceleration across the area of interest, which is important for PIV methods. The fundamentals of PIV means that steady flow across the search areas is assumed, and localised areas of acceleration (and hence, velocity change), are ignored and an average is outputted. Unsteady flow has pockets of acceleration across the river, and can be more accurately represented using PTV methods that can track tracers throughout their whole movement.

GCP - *Ground Control Point*

GCP's are used for orthorectification of images. These are stationary referencing points that are visible throughout image sequences and have known coordinates, either in a localised or global grid. Typically, it is advised that there are more than at least 6 GCP's available to orthorectify to.

GSD - *Ground Sampling Distance*

GSD is a combination of several factors including: resolution, height of camera, angle of camera, and camera properties (sensor size, lens). GSD is the dimension that each pixel in frame is associated with, typically described as *"cm/px"*. Where a camera is at nadir and has been calibrated to remove lens distortion (often seen with most cameras), GSD is equal across the image. However, any imagery that has introduced an angle will experience an altering GSD across the image, with a stretch of pixels in the further distances from the camera (background). This is usually associated with orthorectification.

Image Manipulation

Image manipulation is a more advanced feature associated with image velocimetry. Where imagery may not be ideal, or where tracers are not as prevalent as they could be, then there are several steps that can be taken to edit the image sequences. The basics of image manipulation is to grey-scale imagery, usually done by software suites automatically. Other applications can be normalisation and equalisation, both manipulate the brightness variations across an image to make dark areas darker, so that light areas (typically tracers), can stand out more. For more information, see the image manipulation section in Jolley et al. [14].

IP - Internet Protocol

IP is a term used for uniquely identifying devices that are connected via a network. It acts as an address for the devices. Typical use of IP's day to day is your computer or your phones connected to the internet, but can also be used to connect IP cameras to networks or directly to computers.

module

For this research, any computing unit that can receive, process or transmit data will be referred to as a module . This includes RPI's, data loggers, and computers.

Orthorectification

Orthorectification is the process of georeferencing GCP's in frame and associating them with pixels so that each pixel is given a real-world length dimension. It is vital that orthorectification is done as accurately as possible, as it is these lengths associated with pixels that are used to determine how far a tracer moves between frames and its corresponding velocity. It is suggested that a minimum of six GCP's are used around the area of interest.

PIV - *Particle Image Velocimetry*

PIV is a method of which image velocimetry follows to calculate surface velocities. It is an alternative method to PTV. It essentially follows a time-average velocity approach, which means tracers are not tracked for the entirety, but are located and then searched for after a period of time (a certain number of frames). This method relies on search and interrogation areas to track tracers across the image. Outputs are direction-magnitude vectors, usually of equal space along the stretch of the river. The method has been used to provide highly accurate results compared with aDcp's and is less sensitive to tracer disfigurement than its PTV counterpart. However, discontinuity and inhomogeneity of tracers severely hinder PIV processing. When choosing between PIV or PTV, it is important to consider the average tracer availability and stability in various flows.

Platforms (Fixed or Mobile)

Platform choice depends on the type of study required from a site. For discharge rating curves over a long period of time, fixed stations are required. A fixed station is a permanent/semi-permanent set-up where cameras are installed and secured to not move over the period of time. Mobile platforms are more temporary set-ups, usually for capturing peak flows or areas of interest where fixed platforms are not an option. Mobile platforms are typically drones or UAS's, but can be any temporary platform (e.g. telescopic poles, mounting poles etc.).

PTV - *Particle Tracking Velocimetry*

PTV is a method of which image velocimetry follows to calculate surface velocities. It is an alternative method to PIV. PTV identifies and tracks a tracer for the entirety of its movement in frame. This method has been known to perform well even without a high seeding density, however, the more tracers present the more accurate that PTV can be. On the other hand, as each tracer is tracked throughout the entirety of its progression along the river, the processing required for high volumes of tracers may be more of a hindrance than it helps. Issues arise from PTV when tracers deform through the process of tracking. When choosing between PIV or PTV, it is important to consider the average tracer availability and stability in various flows.

RPi - Raspberry Pi

Raspberry Pi's are small computing modules that are known for being relatively cheap and powerful. They run off of a Linux system and can be used as a simple shell or can be connected to and used with a Graphical User Interface (GUI).

Stabilisation

All mobile platforms and some fixed stations will require stabilisation. Stabilisation is a process where all frames are referenced to a previous frame (usually the 1st frame), and the edges are cropped so that the image in the middle (where the area of interest usually is), looks to remain steady. There are several ways to perform stabilisation (e.g. pre-built programs, MATLAB code), most of which use types of detection and tracking software with consistently visible and stable features (e.g. corner points of walls or buildings etc.).

Tracers

tracers are the fundamentals to PIV and PTV methods. These are any trackable features on the surface of the river. These can be either artificially placed for specific studies (using wood chippings or decomposing materials), or where possible, naturally occurring. Natural tracers can be foam, or strong ripples seen on the surface caused by turbidity. The more tracers visible in the area of interest the better any algorithm can perform. There are recent studies on the density of tracers required for the best results, focused on Seeding Density Indexing (SDI). SDI is a benchmark system that disregards frames that do not meet the required SDI requirements [16].

UAS - Unoccupied Aerial System

UAS systems mostly used with image velocimetry are drone units. These systems allow a user to remote control a flying unit, most likely propeller based. This mobile method of image capture can be used where either fixed stations are not present and there are flows of interest, or where flows are too treacherous to obtain manually.

Chapter 2 Technical Considerations for Image Velocimetry Practices

2.1 Pre-Processing

Prior to capturing and analysing any velocimetry data, users should look to optimise the set-up of the station to ensure that any imagery captured has the best possible chance of being usable. To do this, there are several considerations to be made. Tracers, average environmental conditions, Ground Sampling Distances (GSD), lighting, and orthorectification are all vital components to consider when setting up a station. Here, we will break down each of these and explain how and why they need to be optimised. This is a condensed down version of what to consider, and mostly taken from research previously carried out in Jolley et al. [14].

2.1.1 Site Selection

When choosing where to place velocimetry systems or perform analysis, one must consider how the site characteristics match those needs that velocimetry has. Tracers and environmental conditions are two considerations that can be crucial to how well a system can perform, whereas the infrastructure of a site is something which is less crucial, but still needs consideration. If unsure on the availability of tracers, it is possible to take a short video clip (around 20-30 seconds), to initially run through RIVeR to check for compatibility; if successful, vectors will be prevalent on a majority of the surface.

2.1.1.1 Tracer Availability

Tracers are any artificial or naturally occurring feature that is free-flowing on the surface of the river, capable of being tracked by automated systems. Artificial tracers can be used (e.g. wood-chips or decomposing materials), in individual studies of particular flows, typically seen with low-flow scenarios where the flow can be laminar and without substantial flow patterns. Ideally, we would look for areas of high natural tracers which can include foam from turbidity, or prevalent surface patterns. As a rule of thumb, look to have at least 10-30% coverage of trackable features. Advance users of image velocimetry should look at the Seeding Density Indexing (SDI) of sequences, as it shows to reduce velocity error by between 16-39% with both PIV and PTV methods [16, 17].

2.1.1.2 Environmental Conditions

Adverse environmental conditions can disrupt either the capture of imagery, or the analyses of sequences. Fog, snow, rain and wind are currently the conditions that have been highlighted as problematic, with the addition of the glare across image also being questioned on its ability to impact results. Rainfall can disrupt tracers, as well as create false tracers where rainfall is heavy and flow is slow. Wind shears on the surface of the river and can also disrupt true paths of tracers, and in some cases, create more false tracers. Fog and snow can simply obscure the cameras view and the level of tracers that it can detect. There are no clearly defined ways of error correcting the environmental impacts, but it has been shown that strong headwinds relative to the direction of flow can cause up to 8% of error to surface velocities. More research is being carried out in this field. For now, it is best to limit situations where strong headwinds and adverse conditions are seen. This means ideal sites would be somewhat sheltered from many of the elements and brightness is consistent across the river for a majority of the day. Where this is not always possible, be more cynical with results gathered, and in some cases, disregard where extremities are witnessed (thick fog, heavy snowfall).

2.1.1.3 Infrastructure

When discussing the infrastructure for fixed image velocimetry, it can vary depending on what requirements the user has regarding the power requirements, networking, and structures that can be used for housing. Typically, it is easier for velocimetry stations to be set up at existing monitoring stations that have mains electricity, and are usually not too isolated to receive a mobile signal for networking. However, this does not limit where stations can be set up. Depending on the module used and the power requirements it has, solar panels and 12v batteries can be connected to create a completely in-dependant site. These may require more maintenance (battery changes and cleaning of solar panels), but are viable options, especially with low-power modules such as RPi's and simple data loggers.

Where networking is not viable from mobile networks or landlines, satellite broadband is usually available in most areas and can be used to upload data. Alternatively, data can be stored in-situ and manually collected routinely. This is not the ideal way of using the remote methods that image velocimetry can provide and should only be done as a final option. For the housing on structures, there are many viable options. Depending on the remoteness of the site, there are a few key things that need to be considered: the height that the camera can reach and its subsequent angle and GSD (see section 2.1.2.1), the security from both environmental conditions and vandalism, and the issues with legality concerning capturing imagery in general public areas. All of these are issues that are site specific, and cannot be generalised.

2.1.2 Camera Setup

Ensuring that the camera has been properly calibrated and ideally situated greatly increases the chances of accurately representing surface velocities. There are three main areas to consider prior to setting up your camera and housing: GSD, orthorectification , and lighting.

2.1.2.1 GSD

GSD is a vital consideration to a majority of image velocimetry approaches. Often, this can be overlooked and the GSD is an afterthought. However, should GSD be properly considered prior to station set-up, that it could improve the probabilities of capturing more useful data (e.g. detect more tracers, track tracers more effectively). GSD is a mix of a few elements, but effectively, it is the distance associated with each pixel. For example, a 200×200 pixel image capturing an area of 20×20m will have a GSD of 0.01m/px (1cm/px). The main elements you need to consider for GSD when capturing at nadir are: image resolution, height of camera, camera focal length, camera sensor width. However, where angles are introduced, GSD can be harder to estimate. Introducing an angle in any direction will skew GSD in the distance of imagery. This can get complicated, and does not need to be a perfect science. To help better understand how GSD is impacted, please see the excel spreadsheet created and readily available on GitHub at: `https://github.com/MJolley1/GSDCalc/blob/main/MJ_Oblique_CamAngle_Est.xlsx` [18]. This spreadsheet allows the user to input the camera properties (sensor and focal parameters), heights and angles of the camera, and resolutions. From this information, it can estimate the average GSD across the image.

As a general rule of thumb, you want to be aiming for as small of a GSD as possible, around 1-3cm/px, to match the size of the tracers that you typically see in the imagery (ideally a tracer would be slightly larger than 1 pixel so that it is clearly defined). It is best to begin by limiting the resolution that you use, as research shows that using high resolutions does not always significantly increase the accuracy of models, but does significantly increase the power and processing requirements [14]. This gets complicated when you have to consider the size of the area that is required to be covered and how you are able to get tracers clearly defined. For smaller channels (<10m) and capturing at nadir, it is a good idea to begin by using a resolution of around half full-HD (960×540), although it has been shown that lesser resolutions can obtain data just as well (something to consider depending on site limitations). For wider/larger channels, half full-HD can also be used initially for angles up to about 30°, for angles larger than this (previous studies use angles up to around 60°), perhaps consider using higher resolutions (up to full-HD). At larger angles, it may be hard to capture meaningful data in the far bank, but post-processing tools can help extrapolate this data where it may be missing/unreliable.

To set up the camera optimally, it is advisable to bare in mind that the closer to nadir a camera is the better. This means that the height of the camera can play a relatively large part in impacting GSD. However, large camera heights can be challenging in regards to maintenance and safely setting up the station. Therefore, it is advised to use a camera that has a variable focal length (zoom capability), and set up the camera in a safe and efficient manner at the highest possible height, with the smallest possible angle. It is possible to get focal lenses that provide 180° of view with a high level of resolution which can be used where heights cannot be reached accessibly.

2.1.2.2 Orthorectification

Orthorectification is a process of associating a real-world length with pixels by georeferencing stationary points (GCPs) in frame using either a localised coordinate grid, or a globalised grid. It requires a stable image frame throughout, which can be achieved from a secure fixed station (although can sometimes undergo small oscillations), or through stabilisation for mobile platforms, see section 2.1.2.4. There is much research on the best ways of implementing orthorectification, but for simplicity, users should look to implement an absolute minimum of six GCPs in the frame, with at least four immediately around the area of interest [14]. It is, however, advised to use more than six GCPs, as an increase in GCPs increases reliability. Where data is captured perfectly at nadir, orthorectification is not a necessity. Alternatively, it is vital that the user accounts for lens distortion and manipulates the imagery data to remove these by calibrating the camera. Often, cameras come with the parameters it uses when distorting images and this can be accounted for relatively simply (some models of camera come with software that allows them to be modified to remove lens distortion).

2.1.2.3 Lighting

A major issue with image velocimetry is its capability of working at night where channels are not illuminated. Infra-red cameras can be used and some studies show that image velocimetry can be applicable in these conditions, however, it is unclear on how effective it is at accurately representing surface velocity. This is something to be monitored at sites and evaluated at the time. Where it is not a viable option and results are not coming through, illumination can be used during the times that image captured is required. For example, if the user sets capture time to be 30s of capture every 15 minutes, then lights can be programmed to illuminate the surface for the 30s that are captured and then turn off.

2.1.2.4 Stabilisation

Stabilisation is a process that references each frame to a reference frame (usually the 1st or a previous frame), and crops out edges to give the appearance of a stationary frame. This is mostly used for mobile platforms, where stabilisation is a critical component. However, it is sometimes important for fixed platforms that witness some forms of movement through vibrations from traffic (when close to a road e.g. on a bridge), wind, or any other oscillations. To perform video stabilisation, some cameras have built in stabilisation (e.g. GoPro's), but pre-programmed software can be used, as well as MATLAB coding, as seen with this example available on GitHub: `https://github.com/MJolley1/stabilisation/blob/main/VideostabilisationExample.zip`.

2.2 Processing

There are two main ways of processing data, either remotely through uploads and downloads, or in-situ. Studies have looked at the amount of power required to process sequences in-situ using RPi's "at the edge" and have manipulated software to better suit these requirements [19]. This study concludes that a RPi 4 uses the least amount of energy, and can perform 160 full cycles of of Optical Tracking Velocimetry (OTV) on a single 12v battery charge (715$\times$540, monochrome). This methodology of in-situ processing, along with those that are similar, is incredibly useful where network connections are not reliable or upload speeds do not allow vast amounts of data transfer. However, where networking is efficient and can be done with relative ease, it is currently advised to not use in-situ processing. The reason being, with images being networked and stored online, it is available to be viewed and re-analysed in alternative methods where reliability is questioned.

Remote processing is currently the most common method of using image velocimetry. This typically involves capturing footage and uploading it to a cloud storage solution. Footage can be temporarily stored onto the module ready for uploading. This data is then downloaded to a computer for processing and a software suite can be used to produce the velocity outputs. This results in the module at the site not requiring large amounts of processing power and can be cheaper module units.

2.3 Post-Processing

Post-processing can be a generic term for anything that the surface velocity results are used for; most notably being calculating discharges. However, there are also some other processes that can be applied to the output data, including those that look to validate and correct the velocities calculated.

2.3.1 Calculating Discharge

Many software suites have the built in capability of producing discharges from the velocimetry data. For this, bathymetry (cross-sections), is usually required. The simplest method to use is a velocity-area method, which introduces typical fluid dynamics parameters, such as α. α is a depth-average value obtained from finding vertical velocity profiles (typically with aDcp's). Where it is not possible to obtain site specific α values, it is reasonable to assume a value of 0.85 [5, 13].

2.3.2 Error Correction

There are methods that can filter erroneous vectors for both PIV and PTV results. For PIV, vector validation is an option, which uses several statistical methods and referencing each cell to its nearest neighbour for comparison [20, 21]. These methods include vector difference tests, median tests, and normalized median tests [22, 23]. For PTV methods, validating data can be more challenging due to the unstructured layout of its vectors. The algorithms used for PTV validation stem from those that were created for PIV, however, will struggle to validate data that has more than 15% of outliers. For both PIV and PTV, validation and correction can be done using the Universal Outlier Detection Scheme which is available as a MATLAB code online [24]. For a majority of studies, it would be hoped that corrective methods need not be used, however, where more detail is required or where it is obvious that errors are occurring (often in more turbulent areas), then it may be advised to run corrective measures. RIVeR (one of the pieces of software we look at in section 7, has built in post-processing features that can easily be used by users to validate and replace vectors).

Chapter 3 Hardware Considerations

In this chapter, we will begin to discuss some of the pieces of hardware that can be used to set-up image velocimetry stations. These will be advisories, and lists will not be exhaustive, however, each piece of hardware will be chosen because of its capabilities to perform the type of analysis and optimal parameters listed in chapter 2.

3.1 Platform Type

As previously mentioned, platforms can be either fixed and more permanent features, or mobile platforms can be used for capturing individual flows (usually extremes). Each method is fully capable of providing reliable image velocimetry data, but each has its own limitations and advantages. Fixed platforms are generally used for long-time studies such as discharge-rating curves, and are typically CCTV cameras attached to a structure such as a pillbox monitoring station or scaffolding poles. Mobile cameras are more flexible and have their own criteria that needs to be met to ensure reliable data is captured. The most popular way of using mobile platforms currently, is with the aid of UAS's (drones). Here, we will discuss some of the popular methods that can be used for both fixed and mobile platforms, and begin to look at the considerations needed to produce reliable results.

3.1.1 Fixed Platforms

As previously stated, fixed platforms are stationary cameras, more permanently fixed to structures that will not undergo movement. These stations are capable of capturing a time-series of imagery that can either be processed in-situ or alternatively, networked and processed after. The issue with fixed stations is that once they have been established, you cannot alter the field of view without having to recalculate coefficients used for converting pixels to real world coordinates [14]. During this form of analysis, often it is important to know the stage of the river to understand the distance between the camera and the surface of the river. This can be done with pre-existing stage meters, or ultrasonic level sensors that are also recording the stage alongside the camera data. When using fixed stations, there are a couple of key parts that are required to be considered, namely the source of power for the module and camera, as well as the networking of the module should it be needed.

3.1.1.1 Power Considerations

The easiest way to provide power to fixed stations is through the use of mains that are connected up to a previously installed monitoring station. However, how often will it be required to get further information at stations that are already monitored? There may be occasions that it is needed to extend discharge rating curves or validate them, however, a majority of the time we will be looking to capture data in either more remote locations not previously monitored, or in areas where mains is simply not viable. For this, it should be noted that systems can be battery operated depending upon the module and camera model used. Without going into detail on the module used (as this will be discussed later in section 3.4.1), a typical method of providing power to monitoring stations is using a 12v battery (such as a car battery), and a solar panel. This method may require some regular maintenance (e.g. swapping out batteries, cleaning solar panels), however, the time spent maintaining these sites could outweigh the time usually spent monitoring the site manually. The key to using batteries for this is to ensure they are raised off the ground and are fairly well insulated to prevent the cold from draining them. Solar panels may be enough at times to fully recharge the batteries, however, there may be circumstances where foliage and the time of year prevent the panels from sufficiently keeping the battery charged, it is in these instances that they will likely need replacement and manually recharging.

Something to think about?...

Alternatively, depending on the state of the site, it may be possible to install a small hydro-power charging unit to supplement or replace the solar panels. Hydro-power will be relatively consistent throughout the year, more-so that solar power. However, this has not been a tried and tested method in regards to image velocimetry, so there is no precedence. With that said, there is no reason why it could not be a viable option for some sites, as technology advances have been able to create portable hydro-chargers that are no larger than a car battery themselves. A perfect example of such a contraption could be the Blue Freedom portable charger seen here: `https://blue-freedom.net/hydro/portable/`. As previously stated, this is not a tried and tested method of charging stations, but with a little more research and in comparison with site studies, there is no reason as to why something like this could not be implemented.

3.1.1.2 Networking Considerations

Where stations are not going to be visited to manually collect data, data will need to be networked via some method from the module to off-site storage for remote processing. The most common method here can be mobile networking where it is possible. For many locations that are not completely remote, there can be stable mobile network connections that can be used via a networking module hat (addition to modules e.g. RPi hats), a sim card, and a suitable antenna. The usual cost of this per month for the internet connection is around £20-30 for unlimited internet. This could be adjusted depending on several factors such as data size, type of processing (in-situ or off-site), and frequency of monitoring. However, there will be some cases where mobile networks are not sufficient. For these areas, a majority of the time satellite internet will be available and can be used. This is more expensive than mobile internet, usually around double the price per month, but is available in even the most remote of places. Where satellite broadband is used, if cost minimisation is a priority, it may be worth looking at in-situ processing and only communicating results rather than uploading full videos, especially where videos are a high-resolution and hence require a large bandwidth. However, if budget is not a major concern, it could be advisable to still communicate full videos so that data can be checked and alternatively processed.

3.1.2 Mobile Platforms

Mobile platforms are ever more present in the modern era, and are becoming increasingly affordable. Historically, mobile methods of image velocimetry would include using tools such as telescopic poles, but modern methods allow users to implement drones and UAS's. Mobile methods are predominately used to capture specific flows such as extremely high flows where it is too dangerous to deploy aDcp's or other contact methods. The advantage it has over other non-contact methods is that it still provides a full area of velocimetry data rather than point specific data. The reason for using drones or telescopic poles instead of simply setting up a camera say on a tripod, is to prevent the need for orthorectification by capturing at nadir angles. Using a temporary setup on a tripod will still require the user to orthorectify the images. This can be done at a later time if the camera is geo-referenced to a global grid (using GPS), and surveys are later carried out using the same system. This would be done if storm chasing flows and it is not safe to perform a GCP survey during the high flow.

3.1.2.1 UAS

UAS systems such as DJI Phantom drones now cost less than £1500 per unit. These come installed with gimbals and cameras that are able to capture high-definition footage for periods of around 30 minutes of flying time. The main issue with using UAS systems is the need for accurate and reliable stabilisation to remove apparent movement (see section 2.1.2.4). As stated, orthorectification is not required when capturing data at nadir , however, it will not always be the case, and where cameras are required to be angled (height of the drone is outside that of which you are comfortable flying it needed for capturing the full width of the river), orthorectification is still required. This can be accomplished by laying out several GCPsalong the bank of the river, such as simple checked board sheets. Many UAS systems have the capability of automatically storing their location using GPS for determining camera location. A good example of using UAS systems for capturing image velocimetry data is seen in Perks et al. (2016) [8] and Ioli et al. (2020) [25].

3.1.2.2 Telescopic Pole

Telescopic poles have been used several times in previous studies [13, 26]. They often include a remote controlled platform at the end of the mast, and can use a majority of cameras, however, a GoPro is usually the preferred choice due to its cost, weight, and capture capabilities, (see section 3.3). The use of a telescopic pole allows a user to elevate the camera to an advantageous height that reduces the angle of capture and subsequently the distortion of GSD in far field zones. Much like that of UAS based captured, stabilisation will need to be applied to the images and orthorectification where angles are non-nadir.

3.1.2.3 Out of Bank Flows

At times of substantially high flows, where rivers burst their banks, analysis may be wanted to be carried out. Issues with out of bank flow measurements is, that with fixed cameras, the out of bank flow must already be within frame as it is not possible to change the Field Of View (FOV) without having to re-calibrate imagery. An alternative that has been soft-trialed is the use of fish-eye lenses that capture 360° views, so that should a river burst its banks, it has some chance of being captured. This still, however, requires further research. It may be possible to capture and process the out of bank flows with mobile platforms, but the challenge would be understanding how flow traverses the landscape. Alpha valuesmay be well established for in-bank flows, however, there is a lot less research around depth-average velocities for out of bank flows. Where surface velocities have successfully been captured, it could still be possible to estimate the discharge, but note there may be substantial error bounds.

3.2 Surveying Requirements

As previously mentioned, image velocimetry requires a series of GCPs for orthorectification. For post-processing discharges, cross-sections of the river are also usually required. Therefore, the best way to navigate this is to do a full site survey at the initial time of setting up. Once the camera has been installed, lay out your 8+ GCPs within frame and ensure they will not move during your survey. Once set-up, capture these GCPs in a video that can later be used for analysis, or simply capture an image so you can clearly see each GCP. Next, do a full site survey that includes capturing all of the GCPs, the camera location, and 2 or 3 cross-sections along with water level. Using a base-station as a reference point could also mean that you can locate your local coordinate grid to a global grid using GPS. This process needs to be only done occasionally, namely when the camera undergoes any sort of movement (tilts, shifts, rotations), or after a large event (as large events can disrupt river beds, through scour and other processes). For advanced users, 3D laser scans can be used to capture the full channel in frame, along with camera locations and cross sections. This gives a much more dynamic range of locations that can be used for orthorectification, and if captured in low flows, can give more accurate cross-sections. Cross-sections can also be captured using an aDcp if also geo-referenced to the same grid.

3.3 Camera

When it comes to image velocimetry, the type of camera used could simply be personal preference, or limited by budget. The important factors when it comes to the camera choice are its lens lengths, max resolution, sensor size, and cost. There are also other limiting factors that could be problematic depending on the type of site being developed such as weight, power consumption, and size. Where a site has ideal conditions (i.e. has no limiting factors), within reason, most cameras could be used and still obtain usable data. The key to selecting the camera is deciding what the limiting factors are, how they are to be used, and ensuring there is not a large amount of redundancy in terms of the capabilities of the camera and what is required from it. For example, using a 4k camera to capture imagery that is going to be no higher than 720p. For mobile image velocimetery methods, where cameras are not already provided, the weight, size and compatibility are also important considerations.

3.3.1 Camera Types

The parameters of a camera are the main variants between camera models (e.g. lenses, resolutions, etc.), however, the type of camera also extends to the way that the camera runs and how it communicates to a module. For years, cameras have been able to run off of low-power connections, such as USB's (such as webcams). Alternatively, larger cameras such as security cameras still require a higher voltage input. The availability of power will be a major consideration in the type of camera to be used. The following cameras could all be used for fixed platforms, but not all are also compatible with mobile platforms. With that being said, although they may be compatible with both, they may not be the best solution for both, and some cameras better lend themselves to the flexibility that fixed stations may need.

3.3.1.1 USB

Many modern webcams are cheap, reliable and capable of capturing high resolution imagery, typically to at least HD (1080p). Webcams connect via USB for their means of communication and power, which can be an efficient way of capturing data as it minimises the number of connections required. USB based cameras, however, are not typically built to withstand outside conditions and will need to be protected via casing. With that said, the cost of the these cameras means that they are easily replaceable should they be broken or damaged. Other issues with using USB methods are that they are typically not very flexible in what they can offer regarding range, FOV, and ultimately GSD. Previously, GSD was mentioned that it is one of the most important considerations regarding image velocimetry practice. A camera that provides a flexible GSD means that it can be optimised on site with minimal effort to capture tracers in the highest possible detail. There are a number of case studies that implement USB based cameras, with some options available listed here:

Table 3.1: Examples of small, USB cameras that are readily available and could be effectively used in the field for both mobile and fixed platforms.

Camera	Max resolution	Focal Length /FOV	Sensor Size	Features	Estimated Cost (£)
Olympus Tough TG-6	4K	4.5-18mm	12MP 1/2.3 inch	USB Connection, Weatherproof	369.00
GoPro Hero10	5.3K	SuperView, Wide, Linear, Linear + Horizon-Leveling, Narrow	23MP 1/2.3 inch	Lightweight, weatherproof, USB connection	380.00
Arducam B020201	1945x1109 pixels	160 degree	2MP 1/2.8 inch	Not weatherproof, very lightweight, vast compatibility, USB connection, 0.001lux	46.00
raspberry pi HQ Camera	1080p30, 720p60 and 640 × 480p60/90	Depends on lens	12MP 6.287mm x 4.712mm	Multiple lenses available, vast compatibility, USB connection, very compact	47.27 + lens

3.3.1.2 Network Cameras

Network cameras are digital cameras that operate over network connections, rather than relying on local storage for capturing imagery. Network cameras (or IP cameras), can be connected to and viewed live, via computer or even mobile methods. IP cameras can be connected via wireless, Ethernet and even USB. These cameras provide high quality imagery that can be used for velocimetry, but are typically larger and heavier than the more mobile methods listed in Table 3.1. Some possible IP cameras that can be used are listed in table 3.2, and are picked for their flexibility and capabilities of being used for image velocimetry.

Table 3.2: Examples of network cameras that are readily available and could be effectively used in the field for fixed platforms.

Camera	Max resolution	Focal Length /FOV	Sensor Size	Features	Estimated Cost (£)
HIKVISION DS-2CD2647G2-LZS	2688×1520	3.6 to 9 mm	1/1.8 inch	0.0005 Lux, 60m illumination, colourised low light imagery, weatherproof	370.00
Wisenet QNP-6230H	1920x1080	4.44-102.2mm	1/2.8 inch	colour: 0.2 lux (F1.6), b/w: 0.01 lux, 360 degree pan, 23x optical zoom, weatherproof	950.40
Vivotek FD9368-HTV	1920x1080	2.8-12mm	1/2.9 inch	IR capabilities (30m), <0.005 lux b/w, weatherproof	200.40
LILIN ZHR8182EX2	4K	3.6mm–10mm	1/1.8 inch	IR capabilities (35m), colour: 0.4 lux, b/w 0.08 lux, weatherproof	776.40
ReoLink RLC-523WA	2560x1920	2.7-13.5mm	1/2.7"	Wifi connection, IR capabilities (60m), weatherproof, 360 degree pan	205.59
Reolink Go	1080p HD	110 degree diagonal	-	Wire-free, battery, solar compatible, 5V power, IR capabilities (10m), fixed lens (digital zoom), sim network	173.73
Hikvision EXIR Fixed Bullet Solar Power 4G Network Camera	1920×1080	2.8mm,4mm,8mm	1/2.8"	IR capabilities (30m), wire-free, battery, solar included, colour: 0.004 Lux, b/w: 0.002 Lux, fixed lens, sim network	1,800.00 + VAT

3.3.2 Analogue Cameras

Analogue cameras are the typical type of camera used for CCTV. They require power and data cables and feed data to modules. They have a history of being reliable and cheap sources of video surveillance, but have recently became second best to IP cameras in most fields regarding quality, reliability, and in some cases, price. There are instances, however, where this is not true. Some analogue cameras provide good quality imagery in low-light conditions, for a reasonable price.

Table 3.3: Examples of analogue cameras that are readily available and could be effectively used in the field for fixed platforms.

Camera	Max resolution	Focal Length /FOV	Sensor Size	Features	Estimated Cost (£)
RS PRO Analogue Camera	1312 x 1069	5-50mm	2.12 MP	IR capabilities (40m), colour: 0.08 Lux, B/W: 0.008 Lux, small in size	335.20
Vicon Analogue Outdoor IR CCTV Camera	1080p	2.8-12mm	1/2.8 inch	IR capabilities (30m), colour: 0.03 Lux, B/W: 0.01 Lux, 360 degree view	230.77
Twilight Pro CAM HD VFC 2 G	1080P	2.8-12mm	1/2.9 inch	IR capabilities (30-50m)	63.65

3.4 Processing Raw Data

Video data capture is the start of image velocimetry, and whatever methods you use to capture that data, feeds into the second step; processing. There are two main ways of performing analysis of the raw data, either in-situ or off-site. In-situ processing can be done primarily where network connections are not available for data transfer, or where it is limited in such it may only be able to transfer output data (e.g. text files of results, where the size of the files may be relatively small).

3.4.1 Modules

Modules are systems that are capable of communicating with the camera, and are used to capture, store and transfer data. There are many different methods that could be used to capture, store and transfer data, and sometimes multiple devices can be used in tandem. The main module systems to use are one-board processors (such as RPi's), simple data loggers (such as the Iris wireless logger), or full systems which can be complete computer set-ups.

3.4.1.1 Raspberry Pi

RPi's have several models, the latest being the RPi 4B. RPi's are single board computers that have a good amount of processing power in relation to their cost. Typically, depending on model, a RPi unit costs between £5 and £60 for the basics, with additional costs for SD cards and alternative hats. All RPi's can be used as a controller to capture data, most can be used to store some data, and some can be wired to provide a data transfer and sometimes even process some image velocimetry data. The advantage of using a RPi is the cost effectiveness of them. RPi's are Linux based operators, and can be coded to run any script. If connected to a mobile network, it is also possible to remotely control them, which can be significant if setting up a full network of modules. raspberry pi Pico's are the cheapest models at £3.60, and is specifically designed as a controlling module . With this, it should be possible to use it to control cameras and other hardware, however, it is not possible to do many more advanced pieces of processing such as running programs or uploading data. For a more flexible and complete station, it is advised that the modules used when using an RPi is a raspberry pi 4, or whatever the latest model is. Pi4's have the capability of being fully remote accessed, programmable to run code and software, and have the capability to be added to in terms of different hat modules (for mobile networking, power capabilities and other connections). Pi4's have also been trialled at running a PIV software (OTV), relatively successfully. In addition to this, it is also possible to purchase and apply camera modules to the RPi4 which have the potential to be used for image velocimetry analysis. RPi4's cost around £34, with power hats costing around £20, and sixfab mobile networking hats around £30.

3.4.1.2 High-Performance Methods

The running definition for a full system is one that is able to compute and transmit data as one unit, simply a laptop or desktop computer. If processing in-situ is required and the conditions are hospital (e.g. power supply, dry housing), then it may be best to set up a laptop with sufficient RAM to capture, record, and process the data. USB mobile network dongles are available to connect the laptops to mobile networks if possible so that they can transfer results, or hard-drives can be connected to increase internal storage so that all data is captured, processed and stored on-site ready for retrieval. Alternatively, single board computer alternatives to the RPi are available that provide desktop computer results. These can be the equivalent of a super-computer that processes up to 3.6 GHz, such as the UDOO Bolt V8, capable of running windows 10 and the software suites used for image velocimetry, with the need to adapt the programs for running on lower-power modules.

3.4.1.3 Trigger Sensitive Systems

Trigger sensitive systems can prevent modules and systems from drawing too much power from being active despite not being needed. Examples of this would be to use a trip linking the camera and the stage of the river so that there is only data captured for particular flows. This can be done using ultrasonic level meters or equivalent. Alternatively, Iris data-loggers, if used, have a 5V power out switch that can be triggered to deliver power to a 5V camera unit. There are several other methods that can perform the same task and should be explored by the user.

3.5 Networking

Networking data from the site can be a challenge, especially in remote locations where mobile networks are inconsistent or non-existent. The type of networking done will depend upon the type of module used and the type of data needed to be transferred. A major advantage of networking to a module is the capability of remote accessing fully. Not only can you remotely and automatically transfer data from the site to places such as online storage, but it is also possible to log in to some modules and control them without being on site, should code or software need updating.

3.5.1 Mobile Networking

The simplest of methods for connecting modules to the internet for networking would be to use mobile internet. Mobile internet for modules works very much the same way as a mobile phone, with typically a networking adapter (either a USB stick or module hat), a sim card, and an antenna. To use mobile networks, the sim card needs to be registered to a mobile network and activated with some allowance for mobile roaming included. The amount of data required depends on the characteristics of the site and what output you want from it. Large rivers that require higher resolution images will need more data to upload the videos, however, if you process the imagery in-situ and only send across the results of the analysis, then you would not need as much. It is best to decide what you want to accomplish at the site and adjust for that where budgets are not large, otherwise, sim cards that come with unlimited data are the best way forward, mainly to avoid a data loss should it run out. The main issue with mobile networking, however, is the signal strength. Image velocimetry is a very useful technique for sections of river that are isolated, where remote working is a major benefit. In these areas, especially those that are hilly or mountainous, there may be very little to no signal. In these areas, alternative solutions are required.

3.5.2 Satellite Networking

Satellite networks work similarly to those of mobile networks, but instead of connecting to the nearest mobile network mast, satellite connections are used for the transfer of data. Historically, satellite connections were relatively slow to that of wired and mobile networks, however, the advances in technology have pushed up the speeds that are capable from satellite networking tremendously. The major benefit of using satellite networking is its capability to connect virtually anywhere. Where locations are remote, and mobile networking is not available, satellite networking is the only alternative for connecting to the station. The main disadvantage of using satellite networking is the sheer cost of running. The set-up usually requires a satellite dish, which can be portable, but are still fairly substantial in size and set-up. The cost of running satellite internet is also nearly double that of mobile internet, and is still capped on monthly usage (there are, however, large data packages available that could still meet requirements).

3.5.2.1 Bandwidth

The bandwidth of a site usually is not a major consideration, as uploads are done automatically and can be done in their own time. Problems with bandwidth occur where a backlog of data begins, should it not be uploading at a speed that is quicker than the time between capturing data. Also, issues may persist where sites require remote working, and streaming the camera through the mobile network for any reason. Typically, HD streaming requires 2.8Mbps. Anything close to 2.8Mbps should not result in any uploading issues for the videos, but could perhaps impact any streaming via remote connection that may be required. With that said, it is not very often a remote connection to live stream the camera will be required, and therefore should not impact the decision significantly. Where the bandwidth is not sufficient, the interval between cameras should be lengthened to allow for a full upload of the previous video before the next is stored. If this is impacts the study significantly, there alternative solutions should be found.

3.6 Storage

Storage is a significant part of the image velocimetry cycle. Decisions should be made from the start on what data should be stored, how it should be stored, and who has access to that storage. When storing video imagery, there are particular problems that may persist legally. Where there are images of people, permission is needed from those people to store the data, unless there are measures in place which already accounts for this. Secondly, the amount of storage required for some sites is substantial, and will ultimately determine what solution to storage is used.

3.6.1 Hard Storage

Hard storage is defined as a physical location for the data, e.g. on hardrives or portable storage units. Portable storage units now have the capability to store terabytes of data on them, which can be more than enough to store over a years worth of HD videos to, and results. Hard storage can be used either in-situ, or data can be sent across a network to a hard storage solution off-site. The advantage of a hard storage solution is that there is a non-networked copy of the data, which could mean limited access to people, as well as having the data in one physical place. However, should that hard storage device become corrupt, lost, or damaged, then it can result in all of the data being lost instantly. Many camera units have the capability of storing data on them as a hard copy on micro-SD which is placed directly inside the unit. This could be a valuable asset where hard storage is used, as it can allow for some data to be retrieved should the original hard storage unit become damaged. With that being said, it is also possible to make multiple copies of the data (back-up copies), however, this instantly doubles the demand for storage and also creates issues of where to store data, and who may get access to the data.

3.6.2 Cloud Storage

Cloud storage is a relatively new piece of technology, and has seemingly endless amounts of storage space. Personal services such as Googledrive, Dropbox, and OneDrive have flourished over the past decade, as it allows users to store all of their data off of a single device (away from hard storage), and into a cloud, where it is accessible from anywhere and can be shared instantly. There are now various cloud services available for business users, that boast about the quality of the security that they have, which can be essential for such things as risk assessments for capturing and storing video data. Cloud services can be uploaded to directly from a module , and once uploaded, accessible from anywhere. This could be a major benefit for automated systems where code is used to upload/download data remotely, and then results re-uploaded to either storage or websites for up to date data. Previous studies conducted at Newcastle University uses Amazon Web Services, S3, for all of the cloud storage requirements. Modules can automatically connect to S3 services and create 'buckets' (folders) for data to be routinely dumped directly via mobile networking.

Chapter 4 Software Considerations

4.1 Introduction to Software

In this chapter, we are to discuss the analysis processes that turns raw video data into image velocimetry outputs. The main focus of this work will focus on PIV and PTV methods, such as LSPIV and LSPTV. A little bit of detail will be provided on how these schemes work, as it provides the fundamentals on which method is best to use for the specific site, as each has its own advantages and disadvantages. We will also begin to look at some of the software suites that are currently available, and begin narrowing them down for practical uses.

4.1.1 Detection and Tracking Schemes

Detection and tracking schemes are fundamentally what builds a PIV or PTV algorithm. Tracers are initially detected by the algorithm, which can range from patterns on the surface that are stable across the timeline, or individual particles such as pieces of foam (for more, see tracers at section 2.1.1.1). Tracking schemes are what allows algorithms to follow the detected tracers between frames, and there are numerous way to both detect and track said tracers. The way algorithms work is explained in Jolley et al. (2021) [14], but is outside the scope of this handbook. What is required, is to understand that software varies in the way that they detect and track tracers, and this is the main difference between many.

4.1.1.1 PIV

PIV is a method of image velocimetry that basically breaks down a frame into search and interrogation areas. Once established, each search area detects and tracks tracers independently to the next using the interrogation areas to find particle displacement. This results in each search area producing a time-average velocity, usually provided in the form of a direction and magnitude velocity vector. There are a few key areas that must be considered when using PIV methods, mainly that the search areas must be large enough to successfully detect and track tracers without losing them (e.g. if tracers are moving too fast they can fall outside a search area and not be successfully tracked). PIV is a method that works well and can be highly accurate (compared to aDcp measurements), where tracers are prevalent throughout the timeline, and can be less sensitive to the deformity of the tracers [27]. Problems arise, however, where tracers are inhomogeneous or are discontinuous, and can negatively affect the quality of the analysis. Where this is the case, PTV methods may be better suited.

4.1.1.2 PTV

PTV methods are known for being a little more complex in the way that algorithms perform. Instead of focusing on areas of a river, PTV methods detect and track individual particles for the entirety of the video (where they are not lost). PTV methods tend to work better than PIV methods where there are low seeding densities. However, the more tracers that are successfully tracked, the higher the probability of producing a reliable surface velocity profile. With that said, the main disadvantage of using PTV methods is the processing power required, especially when tracking large amounts of tracers of a long period of time, as the amount of tracers and processing time are somewhat proportional. At times, large amounts of tracers can also impact the tracking capabilities of the algorithms, as it can make tracers much more ambiguous [28].

4.2 Available Software

There are many different variations of software that use either PIV or PTV methods, many are free or open source. As these methods have developed, they have began to become much more user friendly, with easy to operate GUI's and more advance features that allow the user to do more with them. It is not proven currently proven that commercial software provide better results from imagery, however, they may be developed to be more user friendly and intuitive. With that said, the cost of commercial licences can be very expensive (around the £10,000 mark per licence), for what usually is a cheap system (around £1000 per system depending on set-up). This section will begin to explore those software suites and look at some of the requirements for each.

4.2.1 Open Source

Free and/or open source software allows for rapid development to drive the advances of a software. The main benefits of open source software is the community support that surrounds the piece. Many of the pieces of software that are open source are used in many varying pieces of literature. Suites such as LSPIV that have been around for years, have been a focus method of image velocimetry for pieces of work that do not just focus on outcomes, but also parameters such as the impact of tracer densities, mobile platforms, and other papers that compare one piece of software to another. This sort of development and critique of software is not always readily available to pieces of software that have been commercialised, and the general community of those that are knowledgeable in using the free software will greatly outnumber those using the commercialised. A list of some free software can be seen in table

Table 4.1: A non-exhaustive, generalised, overview of software suites that are currently available with some of their requirements and functions listed. Taken from Jolley et al. 2021 [14]

Software	Overview	Requirements and Functions
PTVlab [9]	PTVlab is also a MATLAB code, using particle identification to track movement. PTVlab enhances the images to better show particles and their centroids. With this data, it performs integrated cross-correlation and iterative relaxation labelling techniques to better represent varying seeding densities and track their trajectories.	Image enhancement is performed to highlight the flowing particles and their centroids which are tracked by the software. This tracking requires the user to input interrogation area sizes.
Fudaa-LSPIV [10]	Fudaa-LSPIV is a free software created using a Java GUI. It begins with image pre-processing taking video clips and converting image sequences into PGM ASCII formats. It then sequentially performs orthorectification, PIV analysis, result processing, and discharge computation. Within the software is the ability to visualise and export the data in different formats.	For accurate results, users will need to input GCPs , water level estimations and the positioning of the viewpoint. For discharge measurements, bathymetry profiles area required.

PIVLab [29]	PIVLab is available as an open source GUI within MATLAB, using several of the built-in processes that MATLAB has to offer regarding image pre-processing, image analysis, post-processing and data exploration, producing outputs in video, image or vector map forms.	The software uses image enhancement, camera calibration and orthorectification improve the images. It relies on the user to choose interrogation and search area sizes to perform direct cross-correlation and direct Frourier transformations.
RIVeR [30]	RIVeR is a standalone MATLAB code that uses the theories behind PIV/PTV software but has been designed to provide a simple user-friendly application. It extracts images from video footage, processes them using PIVLab/PTVlab algorithms, then uses GCPs to orthorectify the images and produce velocity and/or discharge results.	Video image extraction rates, frame rates, colour depths, and camera and lens parameterisation is required as an initial input to the software. GCPs also need to be referenced in the images for orthorectification. For discharges, section bathymetry is required.
OTV [31]	OTV processes data through feature detection, feature tracking (using the Lucas-Kanade algorithm), and trajectory-based filtering. This improves the reliability of the velocities as it only retains what are believed to be reliable feature trajectories to reduce error. This produces an average surface velocity estimation.	The software automatically detects features and does not rely on inputting tracers. The direction and length of the flow is determined by the user, who can also determine the quantity of trajectories. Can be mildly affected by lower resolutions and image frequency.
Photrak. SSIV [32, 33]	SSIV uses cross-correlation like most other PIV methods, but unlike other PIV methods, SSIV filters erroneous vectors rather than attempting to correct them. Photrak has two systems available for use: DischargeKeeper which is a fixed IP camera with local processing for continuous measurements, and the DischargeApp for smartphone compatibility . Initially designed for use with consumer-grade surveillance equipment in urban environments.	SSIV requires full camera calibration to take place, with several reference points in the frame of view for orthorectification of the images. Search and interrogation areas need to be defined by the user. Without water level information, SSIV cannot provide discharge but only flow velocity.

KLT-IV [34]	KLT-IV is a user friendly interface that aids users with a step by step process, taking video sequences from fixed or mobile platforms and GCP data and producing surface velocity results and discharges (with the addition of alpha values and cross-sectional data). The software allows users to stabilise videos and orthorectify frames. KLT-IV uses GFTT and subsequently the Kanade-Lucas-Tomasi tracking scheme for tracking. This is a pyramidal tracking scheme that searches 30 iterations within interrogation areas to find the new location of each point until it reaches convergence, discarding any results that produce an error of 1px or more.	For accurate results, camera calibration is required. This relies on knowing the model of the camera for removal of lens distortion and orthorectification of images. This software then tracks individual trajectories through a known 'extraction rate', block sizes, and known reference points. Variations occur dependant upon extraction rates.

4.2.2 Commercial

The majority of software currently available for PIV and PTV methods have been developed and published for free use under general public licence. However, there is one piece of software called Hydro-STIV that is currently being sold for commercial use. This software does not use the typical PIV or PTV method, but instead, uses what is known as Surface-Time Image Velocity (STIV). This is different to PIV and PTV, and uses brightness variation across frames to determine flow. Instead of tracking particles, it follows a particular search line and measures its change in brightness to produce what is known as a 'gradient'. This gradient produced by each search line gives the space-time average velocity along the river, the steeper the gradient, the slower the velocity and vice-versa. The brochure for this software can be found here: `https://hydrosoken.co.jp/en/service/img/Hydro_STIV_brochureA4_en.pdf#zoom=100`, which gives a good breakdown on how the software works. There is also several pieces of research surrounding the use and development of STIV and similar software [35–37]. For the initial phases of development of techniques that this handbook focuses on, STIV is not a consideration for use. The free software suites currently available that uses PIV and PTV methods can more than aptly represent surface velocity fields, as has been shown in numerous pieces of research over the past 20 years. Perhaps, once stations have been successfully created and stabilised, other software suites can be trialled and compared to aDcp and PIV/PTV data to determine which is more user friendly and requires less training/validation.

4.3 Image Pre-Processing

In some cases, video imagery needs to be manipulated so that tracers are as prevalent as possible on the river surface. All software suites require imagery to be captured or converted in gray-scale, which can then be further enhanced using several different methods. The overall aim of image pre-processing is to reduce the noise around the area of interest, reduce the probabilities of erroneous vectors, and enhance the chances of a tracer being detected and successfully tracked.

4.3.1 Image Manipulation

There are several methods that can be used to enhance tracer visibility ready for tracer detection and tracking. These methods all effectively darken the areas that are not required, and brighten the visibility of tracers to make them more prevalent on the surface. Code can be written for these methods to automatically apply manipulation to raw data ready for analysis where it is needed. In many cases, it can be advised to use image manipulation to enhance tracer visibility, so long as the module that is being used for analysis is capable of processing the raw images and applying said transformations. The transformations that have previously been studied are: intensity normalisation, histogram equalisation, contrast limited adaptive histogram equalisation (CLAHE), and image binarisation. It is stated that the most important application of manipulation would be the enhancement of contrast by using intensity normalisation, histogram equalisation, and CLHE. To do this, scripts can be used to take a video, break it down into frames, apply the transformation, and stitch together the frames to recreate the video. Examples of how these manipulations impact imagery once applied can be seen in figure 4.1. It is worth noting, that in some software suites (especially those created in MATLAB), image pre-processing comes as standard (e.g. RIVeR and CLAHE).

4.4 Image Post-Processing

Post-processing of data can generally mean anything that is done with the data after it has been analysed for surface velocity. Typically, discharges are wanted from the surface velocity, so we would apply depth-average flow analysis to the data to get out our discharge. This will be the case for a majority of studies, therefore, this section will not focus on this too much, as it is typically a main part of most PIV and PTV software suites and comes as standard. This section will focus on the validation and correction of velocity vectors, prior to the analysis of discharge.

4.4.1 Corrective Software

Typically, when an image output of surface velocity is provided, it is easy to spot where spurious vectors may have appeared. These vectors go against the general direction of flow, or completely misidentify the velocity of the surface compared to its nearest neighbours. Where this is the case, it is vital to post-process the analysis using corrective methods. There are several algorithms available that can undergo these transformations, however, some software include post-processing steps as standard. PIVLab and Fudaa-LSPIV (to name a couple), allow users to apply vector validation automatically, and remove any spurious vectors from the field. PTVlab (similarly a MATLAB PTV add-on), can also apply vector validation methods at the users digression. In general, post-processing techniques are usually available as a built-in script that can be applied where the user wants. This step is not always necessary, but is useful to apply where confidence in producing accurate flow patterns is low, or where errors continuously occur, particularly at highly turbid sites where vectors can be more often seen.

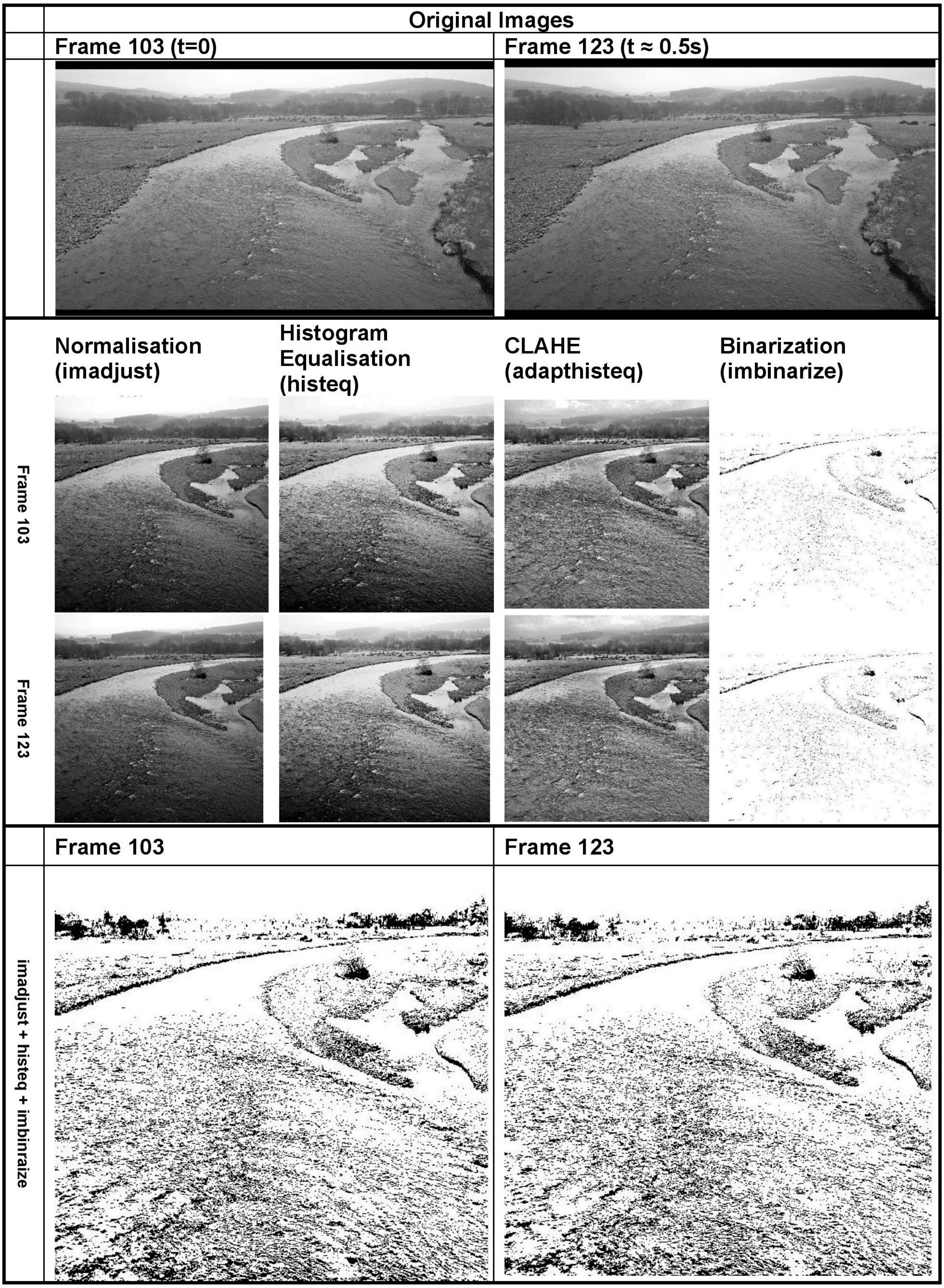


Figure 4.1: The impacts of using each image manipulation transformation onto a frame taken from drone footage, with a time-step between each for comparison.

Chapter 5 Code and Module Setup

5.1 The Code Explained

5.1.1 Introduction

A code is required so that the automation of capture and upload is automated and does not require a user input at any stage. A code has been created using python script that does just this. It begins by creating new folders that store back up data and new data. The code then begins to capture the data using the camera IP address, and finishes by uploading the data and deleting anything that has been uploaded. The code requires users to input several pieces of information before it can be used, including the IP camera address, destination folder, and site name; the code should do the rest. Begin by downloading the code created for windows devices here: `https://github.com/MJolley1/Code`.

5.1.2 Installing the code

Begin by downloading both the "Code" file and the "S3cmd" files and saving them to a folder in your documents. Open both of these python scripts by right clicking the files and "open with" Notepad.

5.1.2.1 Setting up "Code"

Once you have opened the code in notepad, scroll down to a section listed "User input". As you can see, the whole script is commented throughout to describe what each line does, and where things are needed editing. We begin with the save destination. Go on to your documents, find an area that you wish to save all the new data, and highlight the destination at the top, seen highlighted in figure 5.1. Copy this text to the code within the quotation marks and be sure to double backslash all the backslashes.

Set up the next part of the code by setting "zz" (the time between captures), to what you wish. It is advised to not make it much more regular than 15 minutes (reduce the chance of a backlog). Set the FPS to what you desire also, 20-30fps is recommended as default. Next, you need to set the IP camera address. This should be a default value of 192.168.1.64:554 if you are using a Hikvision and have set up using the "Hikvision camera setup" instructions. If not, you can find the IP of your camera and enter that there instead.

Finally, we have to set up the file containing our AWS keys. Open the file "S3cmd" in notepad by right-clicking and 'open with'. Within the code, you will see 6 lines that contain multiple "#'s", these need removing and replacing with your access key and secret access key. To find these, go onto AWS S3, go on to your account, and find "My security credentials". Find the "Access keys" section, and click on "Create new access key". Download the key files so that you can use them again in the future for other stations, but be sure they are secure. Click on "Show access keys" and you will be shown the two keys you need to copy and paste into the S3cmd script. Be sure to save both scripts and these can now be closed. Right click on either of the files, click on 'properties' and look on the 'general' tab. Where it says "opens with", ensure it is "Python IDE", typically Python 3.9 (see figure 5.2).

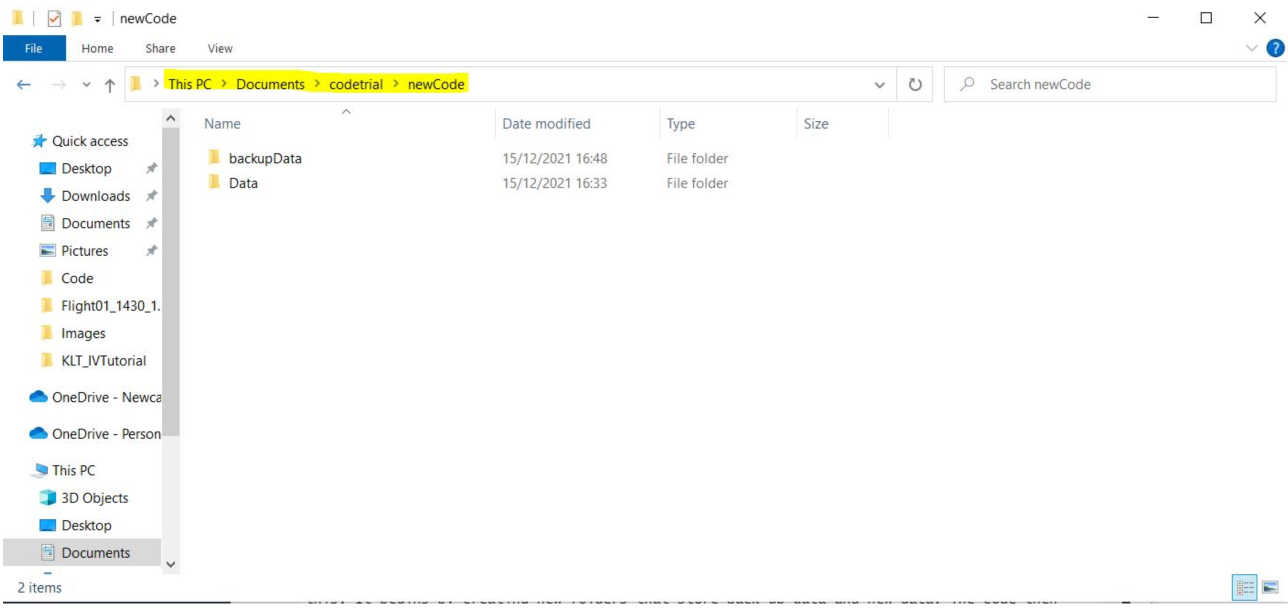


Figure 5.1: Click the area highlighted here, select all, and copy to the code. Be sure to replace all back-slashes with double backslashes

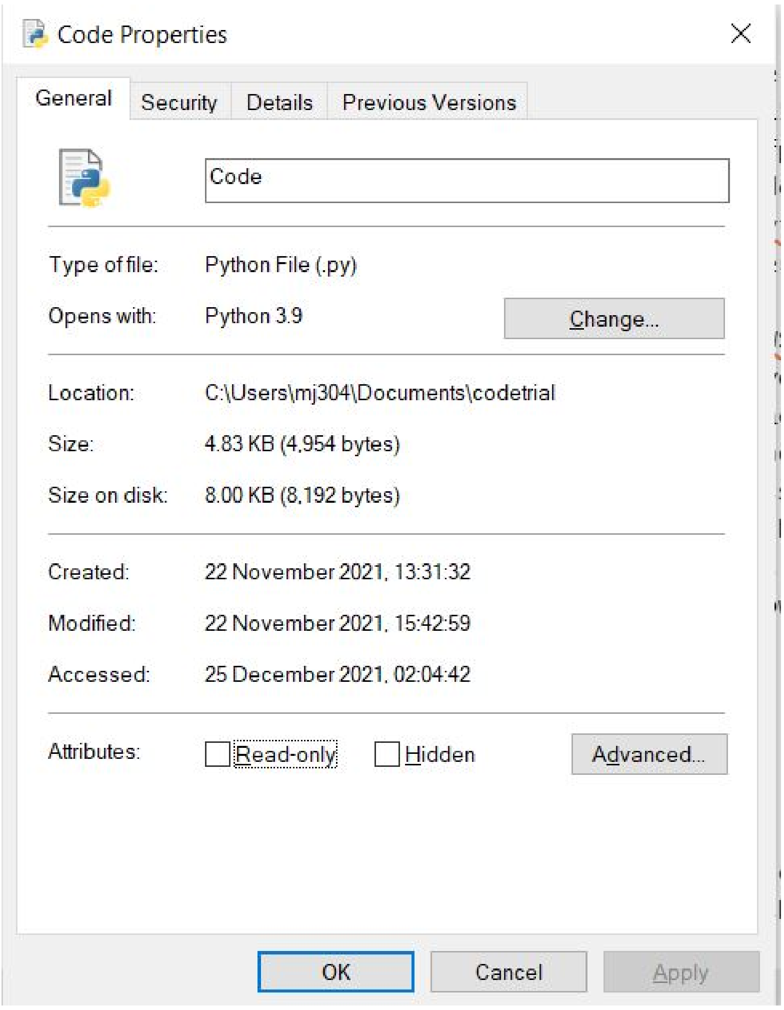


Figure 5.2: Ensure that the "opens with" is opening with Python IDE, here seen as Python 3.9.

5.2 Using the Code

Once the code has been set up, we can place it into the windows OS start up folder so that when the module boots, the code automatically begins to run. This is needed for a multitude of reasons, but the most important being, if the power drops out then the module picks up where it left off without any interaction. To do this, select and copy the two files that you edited from before and navigate to the following path: "C:\Users\"USER"\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup" Paste the two files "S3cmd" and "Code" to this folder. To run the code manually, double click the Code file. Close the file explorer and everything should now be ready to work.

Chapter 6 Site Design

6.1 Overview

Now that we have explored many of the different aspects that must be considered when designing an image velocimetry site, we are better poised to begin determining what is best for new sites. To do this, we will explore some of the already developed studies and sites that are currently or previously created, and look at the hardware/software used for those, as well as some of the challenges and limitations that they faced. I will also draw from some of the experience that I have had out in the field, and what equipment we use at Newcastle University to acquire and process imagery.

6.2 Existing Case Studies

A general list of PIV and PTV experiments was explored in Jolley et al. (2021) [14] and displayed in the table seen below (table 6.1). When new to setting up image velocimetry stations, it is advised that you familiarise yourself with some of the previous studies and how they extracted data from the videos, also note the type of hardware that they used and what worked/did not work with the software choice.

Table 6.1: A selection of application papers chosen due to their variation of hardware and software suites. Each has been briefly summarised for their backgrounds and their results. Originally published in Jolley et al. (2021) [14]

Details	Background	Results
Author: [5] Case Study: Iowa River, Iowa Software Application: In-house algorithm (PIV)	"Proof of concept" research paper. PIV was used without artificial tracers. GCPs used with a 70m wide channel. Ten, 10 minute, recordings used for processing. Sony DCR-TRV320 camera used at an angle of 60° and 14m above the river level. GSD was 6.4 pixel/m. Interrogation areas set at 64×64 pixels and search areas set at 24 pixels.	Discharges from the ten sequences ranged from 50-300m^3s^{-1}. Results were compared to 215 current meter measurements and PIV results agree with the defined rating curve. Average velocities mostly agreed, except for two measurements. Disregarding these two results, error ranged between -11.6% and +4.2%. The study highlighted the issue of results being sensitive to light, with reflections and shadows making it hard to detect movement.
Author: [6] Case Study: River Arc, France Software Application: LS-PIV	Dam flushing, discharge rate between 10m^3/s to 150m^3/s. Survey area roughly 60×40m. Tilt angle roughly 20° to the horizontal. Calculated alphato be 0.79 (0.85 is default). DEM used for bed elevations.	Bias (thought to be due to friction coefficients used), towards the shallow areas of the river; under-estimated compared to a 2D-model. Lack of tracer visibility on right bank due to tree shadow and lack of tracers. At smaller levels of discharge (60m^3/s), calculated error (compared to a rating curve) was large (30%), however, at higher estimated discharges (77m^3/s), estimated discharge was within 0-6% of the rating curve.

Author: [38] Case Study: Alaska, USA Software Application: LS-PIV (Thermal)	Benchmarked using aDcp(velocity, discharge, depths). Thermal camera (FLIR SC8340) used to capture data (1280×720px). Frame rate at 10Hz captured for 1min. Cameras were fixed, with most angles set to nadir, however, some instances 13 degrees off nadir was used. 25mm focal lens used (provided a view of 41.2°x24.8°). ROI of 512×512px used (roughly 6×3.5m, GSD 0.005m).	Angling the cameras resulted in issues; the distortion of the FOV (between 2% and 9%), but also reduced the near-field thermal energy received. The paper does not track 'tracers' per se, but instead thermally distinct patterns. Results skewed from not accounting for camera distortions (e.g. camera tilts), however, the edges of the frames were trimmed when processing (where most of the distortion would occur). Velocities (from both aDcp's and PIV), ranged between 0.5m/s and 3m/s for all the sites. The mean error for all sites was within one standard deviation (range of error between 0.09-0.187m/s with a ± 1 standard deviation of between 0.132-0.247m/s)
Author: [31] Case Study: Brenta River, Italy Software Application: OTV (PTV based)	Camera used was a GoPro Hero 4 Black, set to capture at full HD (1920×1080 pixels) at a capture rate of 50 Hz. Out of a 4 minute video, twelve 20s clips were extracted with a resolution of 1430×1080 pixels (equalling a field size of 7.1×5.3 m^2) at a rate of 25 Hz. Image manipulation was used, including grayscaling and gamma correction to darken mid-tones.	Processing time was measured using OTV and compared against PTVLab and PIVLab and was shown to be two magnitudes quicker using a simple personal computer (2 minutes for OTV and 17±7 minutes for PIVLab/PTVLab). RMSE and R^2 values were used for comparisons with results. These results were compared against a benchmark from a deployed current meter (3cm below the surface). FAST-based OTV produced a R^2 value of 0.83 compared to reference data, with ORB-based OTV 2nd best with 0.74. For comparison, PIVLab produced a R^2 value of 0.49 and PTVLab 0.6 compared to the reference data.
Author: [7] Case Study: Noce Rivers, Italy Software Application: PTV-Lab	Field study after numerical experimentation on optical settings for PTV solutions. Average channel width 28m, slope gradient 1.1%. Waded current meter benchmarking. DJI Phantom 3 Quadcopter with a 4k UHD mounted camerawith 3 axis stabilisation system. 10m hover above surface with orthogonal angle to surface (nadir). Captured in HD (1920×1080px) at 24 FPS. Surface captured was 17.0×9.6 m^2. Artificial tracers were used.	Reconstruction of velocities at banks show higher uncertainty due to bank effects, and a lack of tracer movement; produces significant underestimation. Emphasis on frame rate being such that a particles displacement is larger than that of the particle being tracked, and if not, to subsample frames (reduce framerate to increase displacement). Results compared to benchmark show a good comparison in the middle of the reach (1% error), compared to that of the edges which are much less reliable (48% error). Benchmark velocities show a median of 0.437m/s, and a max of 0.48m/s, whereas PTV-Lab produced a RMSE of 0.125m/s for the full width, and 0.04m/s when considering only the areas with good seeding concentration.

Author: [34] Case Study: River Feshie, Scotland Software Application: KLT-IV (PTV based)	Hikvision AcuSense 4mMP IF network camera (20Hz, 2688 $\times$ 1520px), and DJI phantom 4 (29.97fps, 4096$\times$2160px). High flow conditions (stage range between 0.785m-1.762m). GCPs were used throughout detection and the images were stabilised to negate platform movement.	A range of stage was used to reconstruct the flow (between 0.785m-1.762m). Mobile and fixed cameras produced deviations from a rating curve at 4% and 1% at stages above 1.5m. Vegetation obstructed mobile imaging, had to extrapolate edge, but had over one-million trajectories. Fixed camera had 7433 trajectories (10s of video), mostly in centre, both banks needed extrapolating.
Author: [39] Case Study: Milwaukee River, USA Software Application: LS-PIV	Flow ranged between 6-60m^3/s. Camera used was a NikonD7000 with HD 1080p imaging and a distortion free lens (50mm F1.4). Oblique angle used (between 79° and 86°). Images were orthorectified using 11 GCPs . Water level found by marking extra GCP just above water level. ADV used for benchmarking along cross-section. 25FPS used over a 10min period. Images used have GSD of 0.02m/px over an area of 70$\times$23m.	Median tests used, average error between 29% and 36% over the whole study. The RMS of the velocity agrees well with the ADV benchmarking after the median tests were performed to remove outliers. Good correlation between the ADV and the PIV magnitude (0.12m/s and 0.14m/s from the ADV and PIV respectively). Typically, results were larger than that of the ADV and it is thought to be due to the ADV measurements coming from 5cm under the surface of the river, resulting in less turbulence compared to that of the surface velocity. Velocity index for depth average calculations set at 0.85.
Author: [40] Case Study: Kolubara River, Serbia Software Application: KLT-IV, LSPIV, LSPTV, OTV, SSIV	Comparison paper between software suites. Low flow experiments (approx. 3.4 m^3s^{-1}, benchmarked with aDcp. Mobile platform used (DJI Phantom 4 Pro) with a standard fitted camera sensor (1" CMOS 20MP). Artificial tracers used. Six GCPs were used along the channel. Resolution of 4k (4096$\times$2160 pixels) and a FPS of 23.98. Two videos of 30s were sub-sampled captured from 26m elevation (GSD of 0.77 cm/px), and 32m (GSD of 0.95 cm/px).	The aDcp was used across three cross-sections. KLT-IV and LSPIV provided the best correlation with the results from the aDcp averaging a range between 0–0.07 ms^{-1}. On average, KLT-IV, LSPIV, OTV and SSIV software suites are in better agreement with the aDcp than LSPTV. The Nash-Sutcliffe results for the softwares compared to the aDcp are for video one: 0.535 (KLT-IV), 0.3592 (OTV), 0.4905 (LSPIV), 0.1609 (LSPTV), and 0.3875 (SSIV). A sensitivity study was also produced. The results show that KLT-IV is generally insensitive to changes in configurations. OTV had the lowest sensitivity scores across software suites, however, can be sensitive to the particle trajectory length threshold.

6.3 Equipment Recommendations

This section will begin to look at a very generalised example of what can be achieved via image velocimetry methods, and what sort of considerations can be made. The ultimate aim of this section is to come away with a decision tree that can be followed to allow the user to decipher the best possible way to capture and process data, in the most efficient way. The preferred pathway will be highlighted where limitations of the site are minimal, and generally conditions are met. For this specific section, the assumption that a fixed platform is required for analysis, as mobile platforms are much more simplistic in the overall needs, as they are temporary structures.

6.3.1 Design Aspects

The main considerations for designing image velocimetry sites has been broken down into 6 variables: power supply, mobile network availability, accessibility to site, type of monitoring, channel width, and in-situ processing. These variables influence the equipment that is likely to feature on a site, and the way that data collection will work. These variables have been chosen because they are the most influential impacts on the set-ups. The suggestions made through this handbook have been selected for ideal circumstances, and cannot always be applied to every site. With that in mind, it is sometimes a requirement to use personal judgement on developing a site because of site specific limitations or specific output requirements. For example, in the handbook, an ultrasonic level is suggested where only high flows are required, however, it may be possible to use existing level meters, or optically observe the river level using the camera footage and deep-learning algorithms.

6.3.2 Costs

On average, the cost of a suggested set-up in this handbook is £999.32. This is an estimated cost that does not take into consideration some of the smaller items (e.g. wiring needs, securing cameras, securing modules etc.). It does, however, include a years worth of networking (mobile or satellite), which does take up a significant amount of the budget. The items used in the suggestions to follow are listed in table 6.2.

Table 6.2: Equipment list, prices, and URL's

Item Name	Estimated Price	URL
ReoLink RLC-523WA	£205.59	https://reolink.com/product/rlc-523wa/
Vivotek FD9368-HTV	£200.40	https://www.vivotek.com/fd9368-htv
Reolink Go	£173.73	https://reolink.com/product/reolink-go/
HIKVISION DS-2CD2647G2-LZS	£370.00	https://www.hikvision.com/en/products/IP-Products/Network-Cameras/colorvu-series/ds-2cd2647g2-lzs/
RSPi 4B	£54.00	https://thepihut.com/products/raspberry-pi-4-model-b
UDOO Bolt V8	£339.91	https://shop.udoo.org/en_eu/udoo-bolt-v8.html
LattePanda Delta	£181.93	https://www.mouser.co.uk/ProductDetail/DFRobot/DFR0543?qs=sGAEpiMZZMtTz4c6chlxkDfAvxRwNyzNMFfwzYMkEuIUBveNfvwNkA%3D%3D
Konnect Sat. Lmt	£408.88	https://europe.konnect.com/en-GB?gclid=CjwKCAjwkvWKBhB4EiwA-GHjFqxVR88Az0LSV1P8g94lXu1b4rN-dGDzBR5uOYmV4_trD-cqA_dRkRoCl4MQAvD_BwE
Konnect Sat. Unl	£840.00	https://europe.konnect.com/en-GB?gclid=CjwKCAjwkvWKBhB4EiwA-GHjFqxVR88Az0LSV1P8g94lXu1b4rN-dGDzBR5uOYmV4_trD-cqA_dRkRoCl4MQAvD_BwE
Mob_Vodafone_USB (Including data costs)	£360	www.vodafone.co.uk/mobile-broadband-deals?icmp=uk 1_consumer topnav 1_shop 3_broadband 5_mobile_wi-fi&linkpos=topnav 1 1 3 5
Mob_Vodafone_PiHat (Including data costs)	£380.37	https://sixfab.com/product/raspberry-pi-4g-lte-modem-kit/
RSPi 4B_SSDUnit (Including data costs)	£43.00	https://thepihut.com/collections/raspberry-pi-ssd-storage/products/wd-green-240gb-2-5-ssd
		https://thepihut.com/collections/raspberry-pi-ssd-storage/products/ssd-to-usb-3-0-cable-for-raspberry-pi
12V_Battery_and_Case	£200.00	User Specific
RS Solar Panel Unit	£100.00	RS PRO 20W Polycrystalline solar panel | RS Components (rs-online.com)
MaxSonar	£222.00	https://www.maxbotix.com/Ultrasonic_Sensors/MB7586.htm

6.3.3 Decision Tree

A decision tree helps make decisions based on typically yes or no questions. For the purpose of this handbook, the 6 variables listed in section 6.3.1 are used as questions, and have two options given for each to try and determine the needs for specific sites. The equipment listed in table 6.2 has been suggested because it provides a flexible approach to working. This does mean, however, that there may be better, more specific set-ups for each site, but in general can be used efficiently. This section looks at the match-ups with the questions and the site requirements. A spreadsheet has been created and is available at `https://github.`

com/MJolley1/Handbook_EquipmentList/blob/main/ImageVelocimetry_EquipmentLists.xlsx. A manual version of the tool can be seen in table 6.3 and table A.1. Both of these require the user to select one of two options for each of the six variables to create a 6-string binary output number which can be matched to its subsequent set-up equipment list. An equipment list for setting up a site using a latte panda is available, also at https://github.com/MJolley1/Handbook_EquipmentList. These are the items required for setting up most of a site that uses either battery or mains. This is a basic set-up and should be altered to better represent the needs of the site at hand. A mounting pole (diameter around 80mm), is also required to be secured at site to situate the camera.

Table 6.3: Variables listed as questions, find your binary output (111111, 000000, 110010 etc.) and compare it to the results table. Option 1 binary result is a 1, whereas option 2 produces a 0. For example: mains, no, low, high only, <10m, no, is 111111, whereas battery, yes, high, any, 10m+, yes, is 000000 and any variation of this.

Question	Subject	Option 1	Option 2
1	Power supply available	Mains	Battery
2	Mobile internet available?	No	Yes
3	Accessibility to site	Low	High
4	Flow to monitor	High Only	Any
5	Channel width (approx.)	<10m	10m+
6	Process the data on-site?	No	Yes

Chapter 7 Using PIV and PTV

In this chapter, we will look at three of the pieces of software freely available for use. Two of the three methods are user-friendly GUI's designed primarily for PIV use, while the final one is also a user-friendly GUI, but uses PTV methods. Before you go to site and capture any data, it is highly advised that you familiarise yourself with the following bits of software so it is completely clear on what information is required for each one so that once captured, the process can run smoothly and accurately. Pieces of information such as GCP coordinates and cross-sections can be used multiple times so long as the camera position and river bed stays constant throughout the processing period, so it is important that these are done carefully and accurately to ensure the best possible chance of producing reliable data.

7.1 RIVeR (PIV)

7.1.1 Introduction

RIVeR is a Matlab compiled app that uses the fundamental calculation algorithm of PIVLab, and has a user-friendly GUI to walk-through the process of extracting images from video, through to computing discharge from a cross-section. To use RIVeR, matlab runtime (2015b) is required, as well as the RIVeR application file, both of which can be found here: https://www.mathworks.com/supportfiles/downloads/R2015b/deployment_files/R2015b/installers/win64/MCR_R2015b_win64_installer.exe and here: https://www.dropbox.com/sh/pe5lnybrsn64qgu/AAC-WS_cUQPWwu2qA8em8RmYa?dl=0 respectively. For the purpose of this walkthrough, we will be using RIVeR version 2.5. An example will also be provided for use as a tutorial that can be used to experiment with. For the tutorial files, please visit here. If using the tutorial file, then you can find a copy of RIVeR2.5 in the folder, simply extract this and use this copy after downloading and installing Matlab Runtime.

There are two main ways to process the data; automatically or manually. The automatic feature of RIVeR uses some default values and the suggested settings for the video after entering some of the basic information such as resolution, region of interest, and masks. The manual method of processing data is not any more complicated than using the automated feature, but allows much more of a variation and control over the processing of the data. In this section, manual processing will be broken down into each of its features and discussed on how best to process data that you may have. The following sections provide a more in-depth look at how to use the app so it is easier to process your own data. There is a section at the end designed to run through the tutorial "in a nutshell", as it gives you every click and input you need to process the data the same way that was initially done. Use this tutorial to get a feel for the software, and play about with the features of RIVeR that this walk-through does not cover.

7.1.2 Image Extraction

The process starts with opening the software. Once Matlab Runtime (2015b) has been installed, extract the RIVeR folder from the .Zip File, and open the application file using administrative permissions (right click on the file), if possible (this is not vital, but is suggested). Once RIVeR is open, the first job is to extract frames from the video that you are processing (tutorial named "TutorialVideo1"), see figure 7.1. To do this, the resolution of the video is required. If this information is not readily available from the properties of the video, or known prior to the capture of the video, then it is possible to find out using VLC player (usually comes with windows computers but is freely available online). Simply open the video in the app, and press CTRL+J to open the "codec data", this is where the resolution data is available (see figure 7.2). Enter this resolution and apply a gray scale to the video as standard. Finally, go to "file - PIVLab - PIVLab GUI" to open up the processing segment of the software.

Once open, the first step is to load and import the images that we have just extracted from the video file. Click on "Load Images", followed by "Browse" to find the **folder** that the images are stored in, usually named the same as the video file; click on it and then click "Select Folder" to bring up a side menu of all the images located in the folder. Here you can select the images that you wish to load, but by default, click on any of the images on the left hand menu, select all by pressing CTRL+A, and press the "Import" button to move them to the right menu (see figure 7.3). Delete from the right menu any images that do not belong and ensure that the sequences of the images are in order (usually numbered at the end of the stringed name). Once loaded, it should bring up an image of the river which can now begin to be processed.

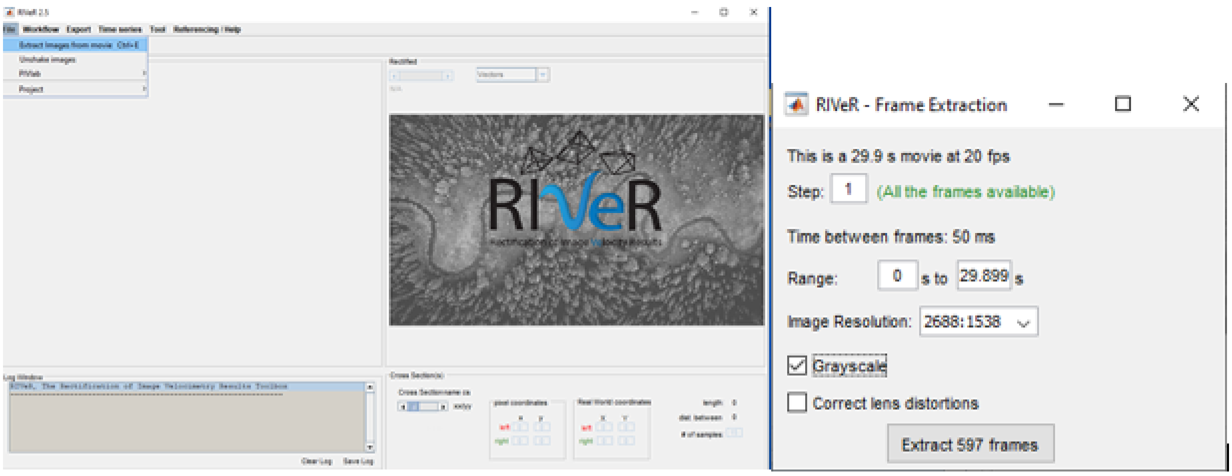


Figure 7.1: Extracting images from a video

7.1.3 Selecting the region of interest

To start, a region of interest (ROI) is required along with any masks that are used to hide objects that may impact results (e.g. any structures or foliage that falls into the region of interest). A ROI is an area that highlights the section of surface of the river to be processed by the software. The ROI should encompass at least one area that has a cross-section, and should have at least four GCPs in or immediately around it (see section 2.1.2.2). To begin, click on "Image Settings - Exclusions (ROI, Masks)". If it is the first time processing the data, select manually the area that you wish to process by clicking "Select ROI" and note down the "x,y, width, height" values that it produces so that you can use them again later (alternatively, we are going to save the PIVLab settings at the end which will store most of the input data for ease). Once selected, click on "Draw masks for current frame", which allows you to create a polygon shape around instances of obstruction to flow. It is import to cover any foliage or structures that are in the ROI that are not to be processed. Once you have covered them, save the masks for use at a later time. When you are happy with the region of interest and the masks that have been applied, be sure to click the "apply current masks to frames" and have "1:end" selected, this will simply apply the masks that you have selected to each frame, otherwise it will only be applied to the reference frame.

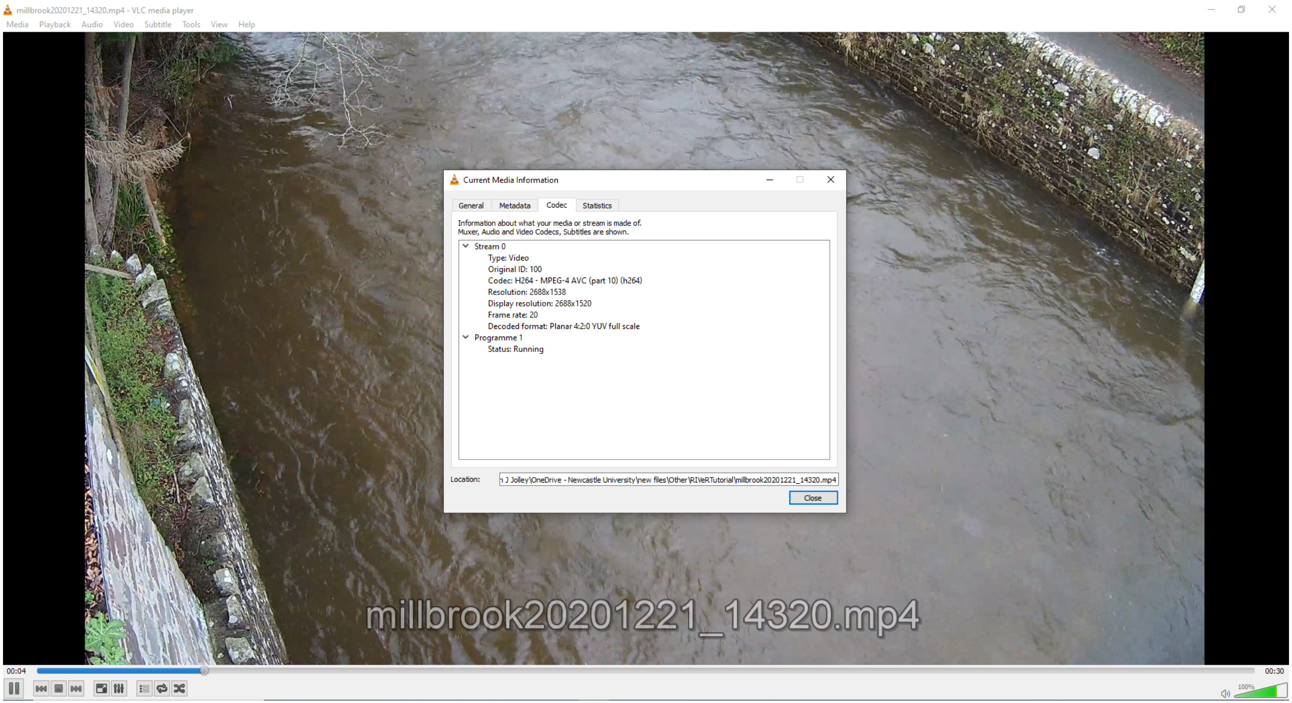


Figure 7.2: How to find the resolution of the video using VLC player

7.1.4 Image pre-processing

Next up on "Image settings", is the pre-processing settings found at "Image settings - Image pre-processing". In this menu, it is possible to apply a number of pre-processing steps to the imagery, and the type of filters needed depends on the quality of raw data collected, see figure 7.5. If there are plenty of visible to the eye tracers and the background (e.g. riverbed), is not prevalent, then not much is typically needed and CLAHE can be applied. Where tracers are not so prevalent, then applying the kernal high pass filter and the other available options may make it easier, but this will be dependant upon the type of footage captured and should be experimented with. For images that feature a highly visible background, background subtraction (the available GUI listed on image pre-processing), can be used. This compares two sequential frames and removes any pixels that may be constant between the two, leaving only the changes (which could typically be the tracer information that we are trying to capture). Again, the effectiveness of this will vary depending on the input data, as most videos captured from fixed stations will hopefully not need this level of pre-processing.

7.1.5 PIV settings

Now is time to choose the PIV processing settings, go to "Analysis - PIV settings". From here, we are able to choose the PIV algorithm to be used (suggested FFT window deformation as default), the interrogation and search area sizes, the number of passes to be completed, and the correlation parameters. As a general rule of thumb, a search area about 50% the size of the interrogation area is usually suggested, but for simplicity, the app comes with a "suggest settings" feature. Generally, it is easiest to use this feature to decide on your starting pass sizes, as it will vary from site to site, and ultimately depends on the average size of the tracers that the software will use to produce the surface velocity. With each "pass", data is carried through and recalculated to increase the accuracy of results, but this also can substantially increase the processing time required. Typically, the idea is to start big and get smaller as you pass through the steps (e.g. 164 for pass 1, 82 for pass 2, and so on). With this in mind, although smaller interrogation areas may increase the resolution, it can also increase the noise.

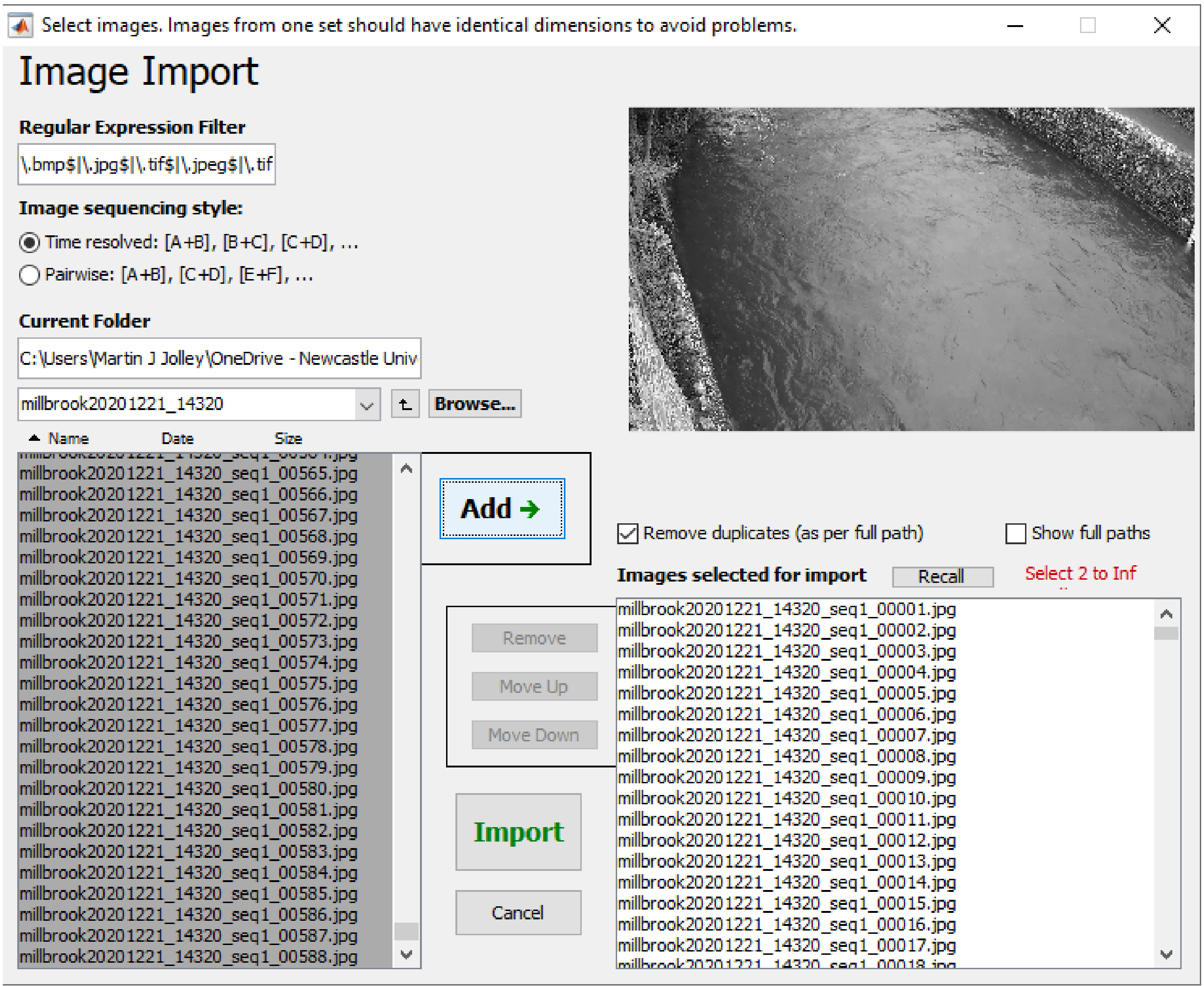


Figure 7.3: Importing the frames

Now the fun begins, go to "Analysis - ANALYZE!", followed by "Analyze all frames" to begin watching the magic happen. The time for processing to complete ultimately depends on the computer processing power you have at your disposal, the length of the video to be analysed, and the processing steps previously mentioned (e.g. interrogation areas, number of passes, etc.). Typically, with the computer used to analyse the tutorial data, processing time lasted between 35-40 minutes.

7.1.6 Post-processing

Once complete, you should be left with a vector map that shows the direction of the flow for each area, as well as a magnitude depicted by the size of the arrow vector. You could just leave it as that if you are happy and confident in the typical direction that is shown on the map. However, if you want to calculate discharges, it is vital to remove as much erroneous information as possible. To do this, PIVLab has some built in features that can be used, found in "Post-processing - Vector validation", see figure 7.7. Usually, for flows that are not greatly turbulent, it is quite evident which direction the vectors should be heading, and any that look obviously wrong can be corrected easily. To do so, it is initially recommended that a correlation filter should be used and I believe that for any flows that are not relatively largely turbulent, 3 standard deviations should be used to encompass 99.7% of all values, and it is reasonable to assume the 0.3% are outliers. Even if this assumption

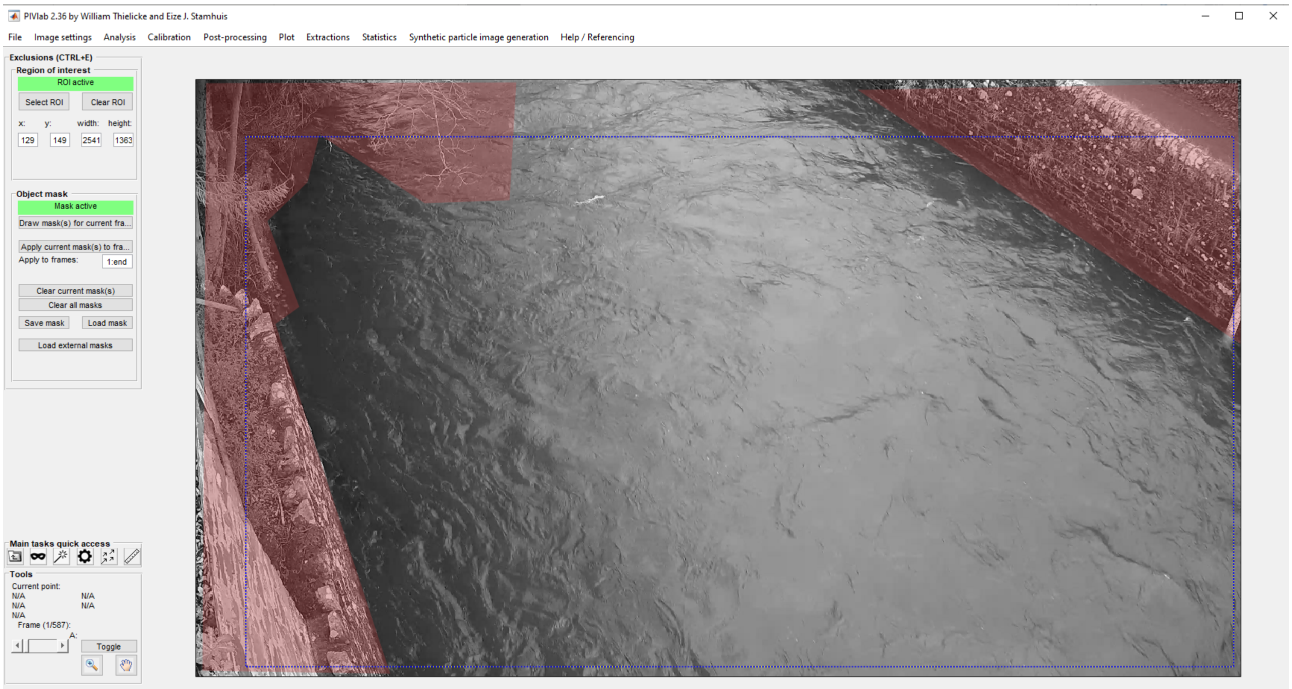


Figure 7.4: Applying a ROI and a mask

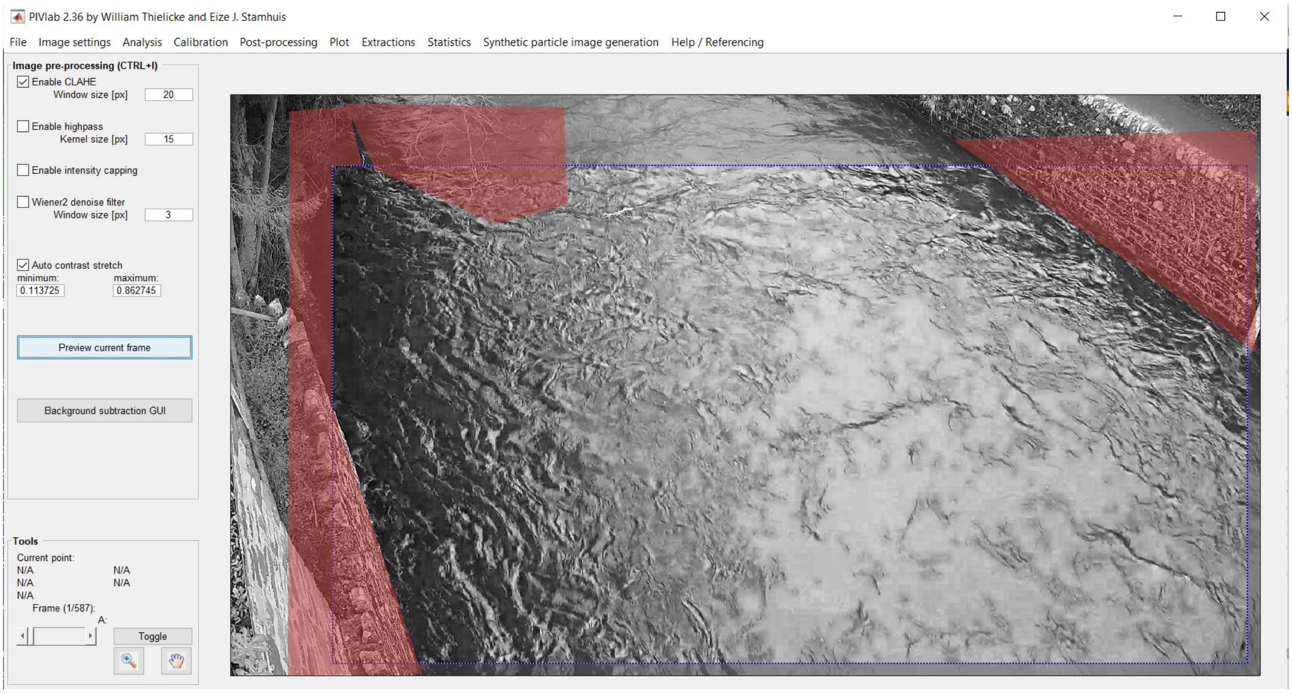


Figure 7.5: The pre-processing parameters required for the tutorial video. Play with these values until tracers are as prevalent as possible.

is incorrect, any missing values for velocity can be filled using interpolation techniques. A wrong value could produce a larger deviation from the true discharge than using an interpolated value as the calculated value is done so using average velocities surrounding the missing area. As general good practice, it is advised to use the "standard deviation filter threshold", set to 3, and the "local median filter threshold" set to default (5 and 0.1), and then finish by clicking "interpolate missing data" and "apply to current frame" to check the new vectors highlighted as orange. You could also "refine velocity limits" manually by selecting an area of the velocities

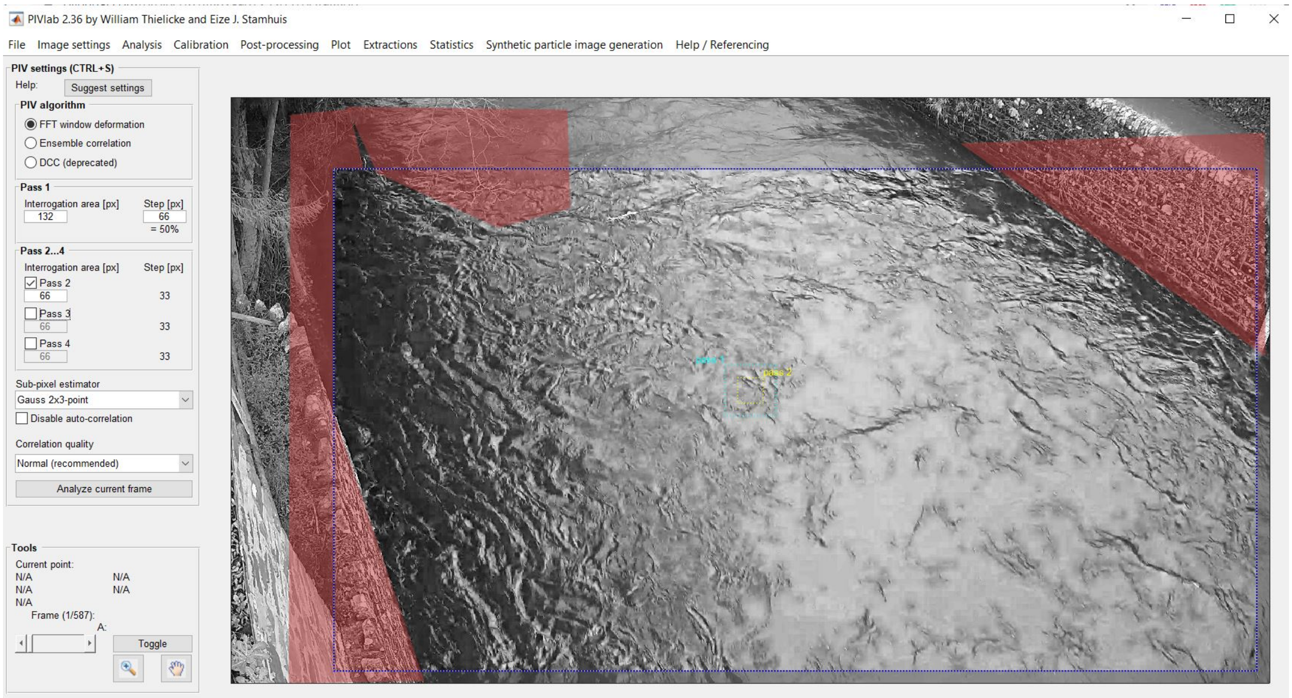


Figure 7.6: Some typical values for the PIV settings used in the tutorial. These can be changed to include more passes and larger initial areas.

believed to be most credible, however I would advise this to only be used once users are fully comfortable with how everything works. When happy with the post-processing applied to the frame, be sure to apply the changes to all the frames.

7.1.7 Saving data

Now it is time to save your settings and your session. Saving your settings means that next time you want to analyse a video from the same camera (e.g. when you have a fixed site and you want to process multiple videos). Saving the session means that you can reopen the analysis of that video with all the settings previously set so that you can change and re-analyse the footage. To do this, simply go to "file - save" and do so for both. From here, there are other pieces that can be done in PIVLab, however, for the purpose of this walk-through, they are not required as we want to simply calculate our discharges. Feel free to explore the other menus and features once you have saved your session and can revert back to it if needed. You can now close the PIVLab GUI window (not RIVeR).

7.1.8 Orthorectification

Back in RIVeR, we are now ready to calculate our surface velocity values (not just their magnitudes and directions), as well as our discharge. To begin, go to "workflow - load PIV analysis - load PIVLab session" and browse to where you saved your PIVLab session from the PIVLab GUI. Each time you successfully complete a step, the step turns green in the drop-down menu. Next, go to "workflow - load background image" and let it automatically choose the image. Similarly, go to "workflow - load CP's image" and browse to a frame from the video used to capture the GCPs .

It is now time to rectify the images. Go to "workflow - define CPs/ region of interest" and we are going to be redefining it in 3D as hopefully we have more than 8 GCPs located in the frame; so go to "define in 3D -

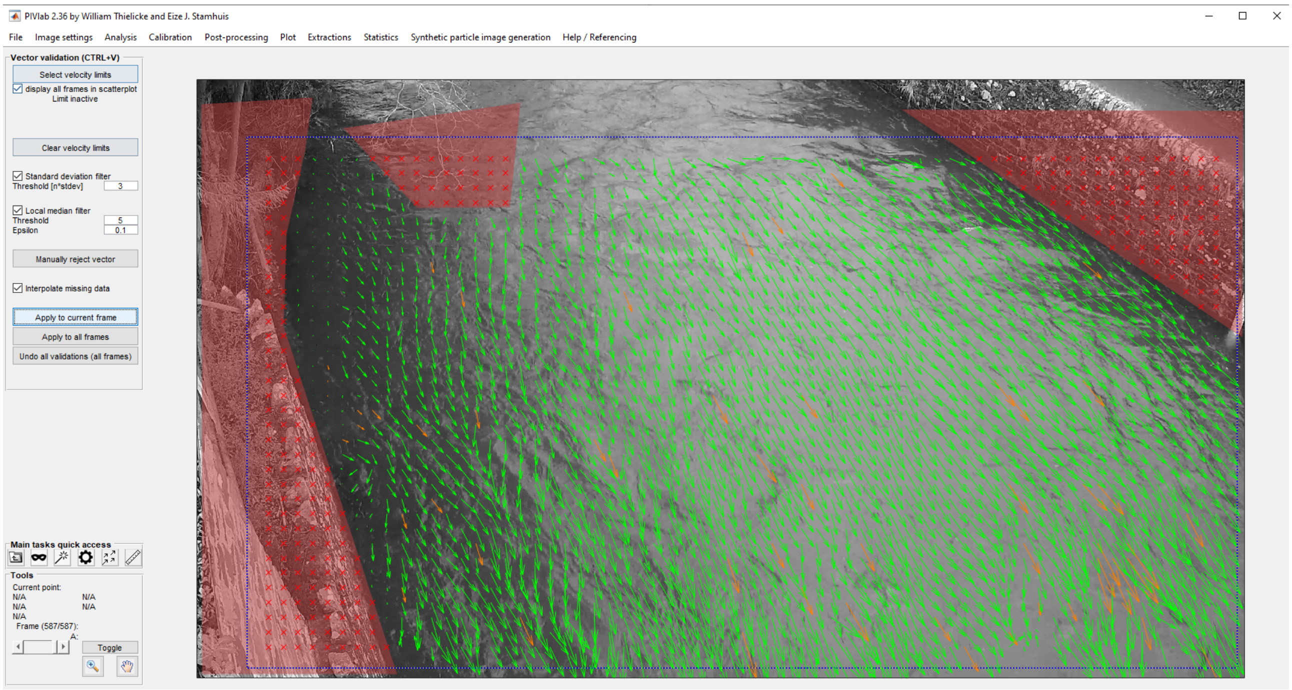
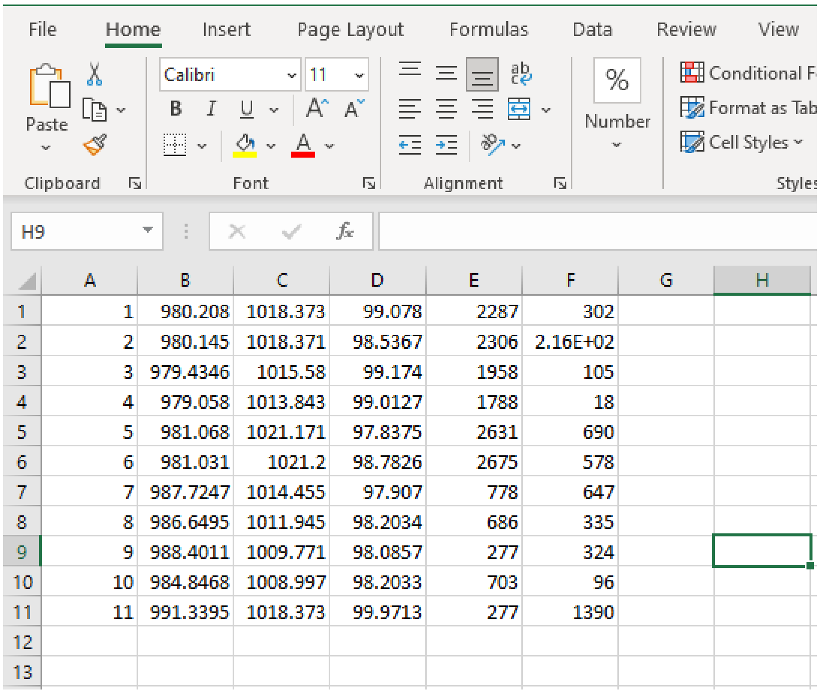


Figure 7.7: Post-processing considerations for the data, including vector validation and interpolation. Erroneous green vectors are removed and replaced with statistically valid orange vectors depending on the thresholds set at the menu on the left side of the pane.

import ID,X,Y,Z,x,y file" and find the GCP file for the site. This file should be a .xls file that contains, in order, ID, the real-world X,Y,Z coordinates of the GCPs, and the local coordinates x,y of each in frame too, see figure 7.8. To find the pixel coordinates of the GRPs, simply isolate a frame of the video that has the GRPs in frame and use the open source "GIMP" image processor (available here: `https://www.gimp.org/downloads/`), to find the pixel location in frame (see figure 7.9. If the GCP video you collected has been processed in RIVeR, then it will already be broken down into frames, and any of these frames can be used to find local coordinates. If the GCP video has not been broken down, open the video using VLC media player (free application, usually comes with windows PCs), and save a snapshot by going to "Video - Take Snapshot"; open this screenshot in GIMP and find the GCP coordinates that you need.

The next two definitions of the plane of interest is the water level (real world coordinates), and the region of interest (real world coordinates). Go to "workflow - define CPs/ region of interest - Define in 3D", and for the water level "Define gage", and for the region of interest " Define real world region of interest". The tutorial images were defined using a cloud point, meaning there is more than enough coordinate data to cover a range of regions of interests and water levels. For the purpose of the tutorial, the water level is showing 5m on the gauge on the video, which in coordinates is around the 98.15 mark, so this is our gauge. The region of interest, much like the GCPs, can be input as a .xls file. The layout for this is X,Y,Z (all real-world coordinates), and in the order of point 1 being a corner of a rectangle, with point 2 being the next corner of the rectangle going clockwise (see figure 7.10. The next step is simply defining the time-step of the video, which is usually detected automatically; but if it is not, then it is simply the time between frames (e.g. 20fps would be 50ms). Finally, go to "workflow - rectify results" to process the orthorectification.



	A	B	C	D	E	F	G	H
1	1	980.208	1018.373	99.078	2287	302		
2	2	980.145	1018.371	98.5367	2306	2.16E+02		
3	3	979.4346	1015.58	99.174	1958	105		
4	4	979.058	1013.843	99.0127	1788	18		
5	5	981.068	1021.171	97.8375	2631	690		
6	6	981.031	1021.2	98.7826	2675	578		
7	7	987.7247	1014.455	97.907	778	647		
8	8	986.6495	1011.945	98.2034	686	335		
9	9	988.4011	1009.771	98.0857	277	324		
10	10	984.8468	1008.997	98.2033	703	96		
11	11	991.3395	1018.373	99.9713	277	1390		
12								
13								

Figure 7.8: This is the layout required from the GCP .xls file. From left to right, there is ID, X (real world), Y (real world), Z (real world), x (local pixel), y (local pixel).

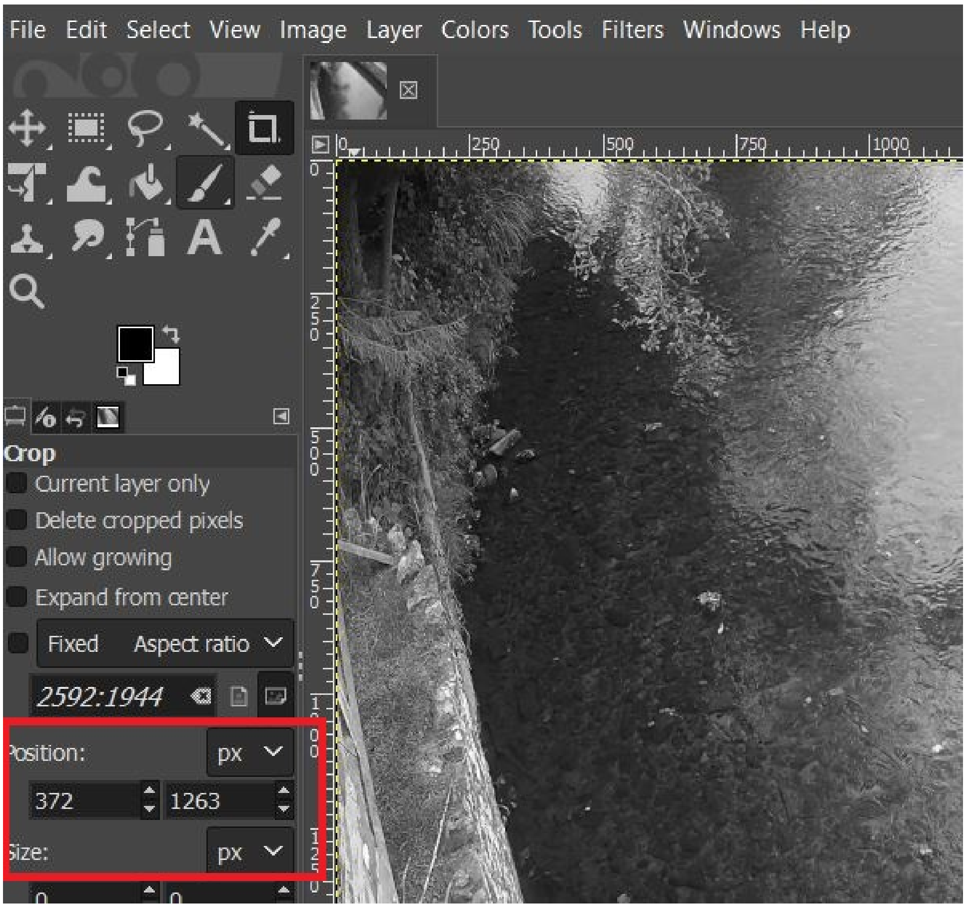


Figure 7.9: How to easily find GRP local coordinates (x,y or i,j), from a still image using GIMP.

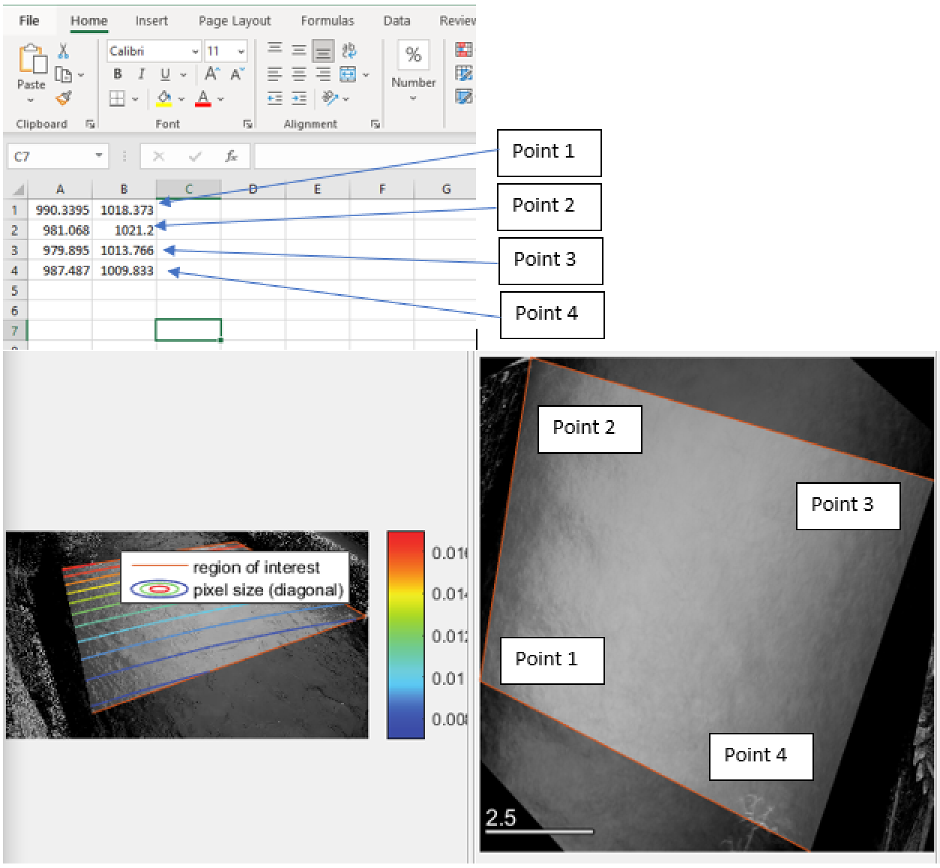


Figure 7.10: The input file and the output ROI as seen in the RIVeR app.

7.1.9 Cross-sections

To calculate the discharge, we need to define a known cross-section. To do this, a definition is needed either in the form of drawing on to the image (workflow - cross-section - add new - define cross section), or defining a known cross-section (workflow - cross-section - add new - define coordinates). For the tutorial, the file CS1 holds the location data for the cross-section we are to calculate for using real-world coordinates on the surface of the water (X and Y), see figure .

7.1.10 Calculating Discharge

If everything is completed, you will be able to open the discharge menu found at "Workflow - add cross section - compute discharge". From this menu, we enter our cross-sectional depth values along the cross-section previously defined. To do this, it is advised you have the bathymetry in a .CSV file format as seen in figure 7.12. Go to "File - import - CSV file" and find the bathymetry file (named bathy for the tutorial), and select "Depths" so that the software can plot the cross-section along with its estimated velocity values. Next up, enter the stage value and velocity coefficient (alpha value) parameters. With the tutorial cross-sectional depths, all values were referenced to the local zero plane, so the stage is now directly taken from the rule and does not have to be in the real-world coordinate grid. For simplicity, you could calculate everything to the real-world coordinates captured at the time of the rest of the data (such as GCPs), and use the real coordinates for depths and stage. Either way is fine so long as you are sure on the location of the cross-section and the consistency of coordinate grids. The velocity coefficient (alpha value, see section 2.3), can be assumed to be a default value of 0.85, but can be calculated precisely by using an aDcp. You could also associate an error bar with results by calculating the

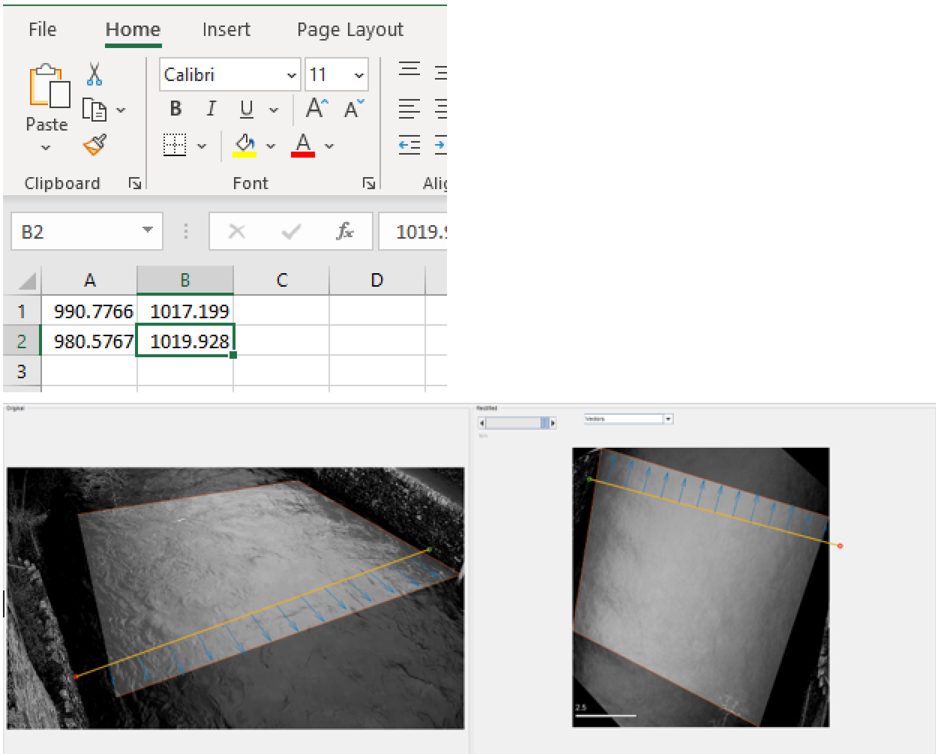


Figure 7.11: The cross-section input data along with the defined cross-section image imposed onto the river, along with the velocity vectors that apply to the cross-section.

discharge with the range of alpha coefficients between 0.79-0.91, as these are the typical values for alpha. For the purpose of this tutorial, the edges of the river have purposely been left out of calculation so that the program can interpolate these values and you can see what it does. When calculating this for yourself, you should really try and calculate velocity across the whole river and only interpolate where sections are not available. So, with that said, tick the box that says "inter. NaNs" and look at the difference in velocity profiles. Now click the all important "Compute" button to see the estimated discharge. Once you are happy with everything, save it as the AreaComp file, close it up and save the project.

7.1.11 Summary

RIVeR is a very simple tool to use once you have understood what the program needs for raw data. It is heavily advised that you take a full 3d laser scan of channels so you have more than enough data to use for referencing. Ideally, you should capture all coordinate data in one go so that everything is referenced to the same grid. This includes the GCPs, of which you should capture AT LEAST 8, but ideally around 12 or more with 4 or more directly around the ROI. You need to capture the ROI coordinates, but this could be later calculated using the 3D cloud if captured, if not, capture these at the time of the GCPs. Cross-sections are also required in real-world coordinates, so be sure to capture these at the same too (XYZ data). Capture at least 1 cross-section but ideally 3 so they can be compared and validated between one another. The number of data points along the cross-section is up to the user and depends on the method used. They can be manually captured or captured using an aDcp, so long as the coordinate grids can be matched accurately. It is also important that you capture a video that encompasses all of the GCPs. The GCPs could be stationary objects that are clearly visible in the images, or can be temporary points that you have placed, but is still captured in the video; this is the method used

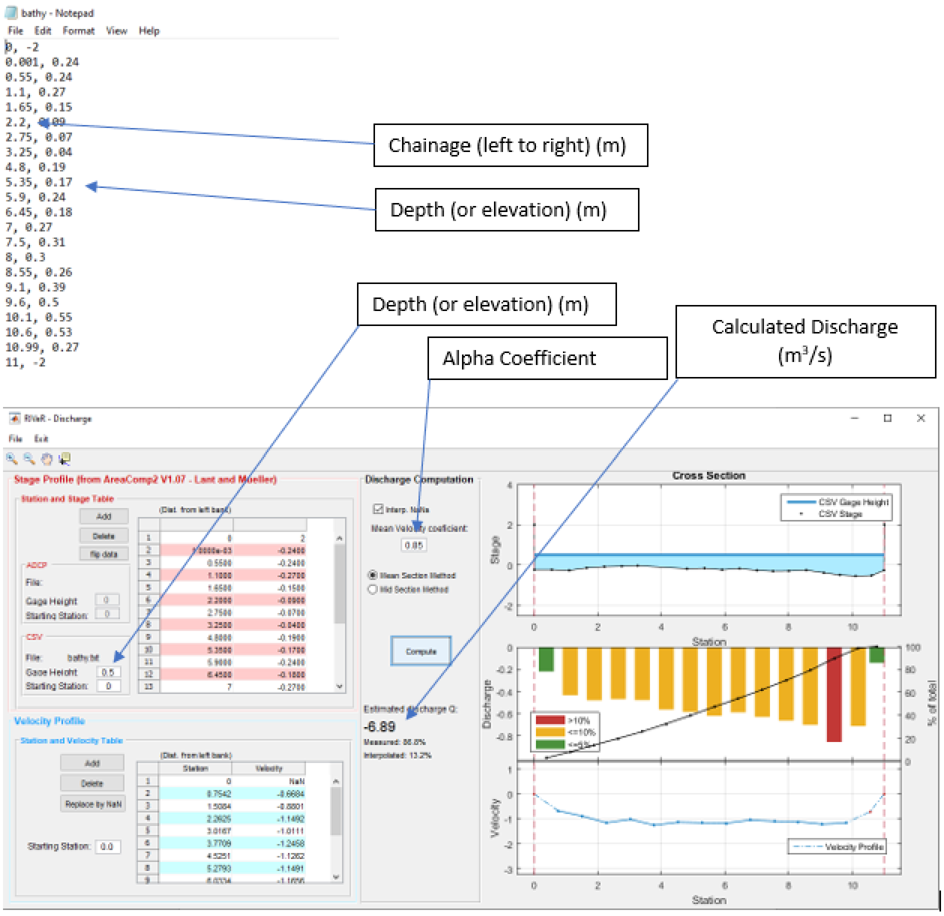


Figure 7.12: This is the required data for inputting cross-sectional data and the parameters for calculating discharge for the tutorial. The alpha coefficient has been shown to average around the 0.85 mark, but is specific for each site. Be sure to define whether your cross-sectional data has either depths or elevations in the program (comes up as a text box during inputting).

for the tutorial. The stage is also required for calculation, therefore, either a stage capturing device is required and some points along the water surface (e.g. when capturing a cross-section, capture a few points of the water surface along the same cross-section so the stage can be converted to the real-world coordinate grid), or if there is a stage rule already at the site, this can be captured within the video and used (the same as the tutorial). It is still important to know the coordinate value of the water surface, so it is still advised to capture some points so that it can be calculated. For example, within the tuturial, the rule gives an indication of the water level and each level is associated with a Z coordinate such as 0.5m = 98.15, 0.6m = 98.25, 0.7m = 98.35 and so on (this was captured during a 3d laser scan of the channel).

There are several other features of RIVeR such as exporting data in different formats, but this can be up to the user on how they use these, and can be played about with at your own discretion. For the purpose of this walk-through and tutorial, we will end with calculating discharges, but if you wish to know more, there is some more information on some other bits in the handbook that is found in the folder containing the GUI, which is automatically downloaded with RIVeR 2.5.

7.1.12 Tutorial in a nutshell

To make things simpler and save some time, the following points will walk you through the tutorial files to come up with a full process using given values. Simply follow these steps carefully to fully extract images and process the data to find an estimated discharge.

1. Download the files required for RIVeR including the software and the tutorial files, found here (available until Sunday May 8 2022).
2. Download and install Matlab Runtime (2015b), available here.
3. Extract and open the file containing the the tutorial files, and extract the copy of RIVeR2.5 found inside ready for use.
4. Open the extracted folder for RIVeR2.5, right click on "RIVeR_GUI" and open with administrative permissions (if possible), if not possible, just open by double clicking. This should open up the RIVeR2.5 GUI.
5. Click "File" and "Extract images from movie". Find the tutorial folder and double click on the file named "TutorialVideo1" to bring up the extraction menu. We want to step each frame (so keep at 1), the time between frames should be automatically detected at 50ms, the range is the full video (just under 30s), the image resolution needs set to (2688:1520), and the frames need gray-scaling. Then click "Extract frames"
6. Go to "File - PIVLab - PIVLab GUI" to open up the PIVLab GUI.
7. Click "Load Images" and then "Browse". Go to the folder that contains the tutorial data, and click on the newly created folder containing the images "TutorialVideo1" followed by "Select folder" to load up the image sequences.
8. Click on the left hand side menu of one of the images, press "Ctrl + A" to select all and then "Add" to move the frames to the right hand side menu. Now click "Import".
9. Click "Image settings - Exclusions" to bring up the ROI and Masking settings.
10. Enter the following details for the ROI: x=149, y=141, width=2460, height= 1346. This will create the rectangle for the ROI over the river.
11. Click on "Load mask" and select the file in the tutorial folder named "PIVlab_mask_Tutorial" to bring in the masks that have already been preset. These mask the stationary/ moving areas that we do not want processed. Finally, click on "apply current masks to all frames".
12. Go back onto "Image settings" and click "pre-processing". Check that "Enable CLAHE" is selected and at a size of 20px, as well as "Auto contrast stretch". Preview this effect to ensure it works.
13. Go to "Analysis" and click on "PIV settings". From the top down, ensure that "FFT window deformation" is selected, Pass 1 is set to 136 and 68 for interrogation and step respectively, Pass 2 is on and is set to 68 and 34, "Gauss 2x3-point is selected, and the correlation quality is normal. Analyse the current frame to see the impact.
14. Click back on "Analyse", and go to "ANALYSE!". Press on "Analyse all frames" to begin the processing. Depending on processing power, this can take between 20-50 minutes.
15. Once complete, go to "Post-processing" and select "Standard deviation filter threshold" and set it to 3. Click on the "Local median filter" and set it to 5 and 0.1. Also ensure that "Interpolate missing data" is selected. Apply to all frames.
16. Go onto "File" and save the PIVLab session to the tutorial folder. Close the PIVLab GUI and go back onto RIVeR.
17. Go onto "Workflow" and ensure that "Load PIV/PTV session" is green, meaning it has the session you just processed loaded.
18. Click on "Workflow" and go onto "Load background image" and let it automatically select a frame. Open "Workflow" again and select the "Load CPs image" option and select any of the frames (the CP image is not provided so is not needed, any image will do here).

19. Next up, go onto "define CPs/ region of interest" and "Define in 3D", followed by "Import CPs file", select the "MillbrookGCPs" file. Select "OK" to the pop up. Go back to the same menu, only this time "Define Gage". Set the water surface plane to "98.15" (this is the Z coordinate location of the water surface relating to a gauge reading of 0.5m).
20. Next up is the "Define real world region of interest". Find the ROI file in the tutorial folder and select it and click "OK".
21. On "Workflow", next step is defining the time step. This should be set to 50ms, but should automatically be detected.
22. Rectify the results by clicking on "Workflow" and "Rectify results".
23. Next is defining a cross-section. This has been supplied as a excel file. Go to "Workflow" and go down to "Cross-section - add new - define coordinates" to open the browsing menu. "Import from excel" and go to the tutorial folder. On the drop down menu at the bottom right which says "(*.xls), click on it and select "(*xlsx)", this should bring up the "CS1" file. Double click, followed by "OK" and name it CS_1. This will bring up the cross-section and the results associated with them.
24. Next up, we need to calculate the discharges associated with the cross-section. Go to "Workflow - Cross-section - compute discharge". This will bring up the discharge GUI.
25. Go to "File" and "Import" and "CSV File" to bring up the bathymetry associated with the cross-section; it is ok to clear the data already entered (click yes ot a pop up window). The file is in the tutorial folder and named "Bathy". Double click this to open it, it has 0 headers and contains "Depths".
26. Back on the discharge GUI, the gauge height associated with the video is 0.5m. We also want it to interpolate data, so click on the checkbox named "Inter. NaNs". The mean velocity coefficient we default to is 0.85 as suggested in literature (so change the 1 to 0.85).
27. Finally, click on "Compute" to bring up the estimated discharge, which should hopefully be -6.89 (the negative can be ignored).
28. Save everything if you want to keep them all for a later date. RIVeR can now be closed.

7.2 KLT-IV (PTV)

7.2.1 Introduction

KLT-IV has been created as a stand-alone program created in MATLAB and packaged using ffmpeg. The installation of the software requires MATLAB2019 Runtime, and will be installed during the download if not already present on the device. The version 1.0 of the software is available here: https://sourceforge.net/projects/klt-iv/ (accessed Oct 2021), with the corresponding paper defining its background and application here: https://gmd.copernicus.org/articles/13/6111/2020/ [34]. For the latest version (V1.01), please contact us (m.jolley1@newcastle.ac.uk), and I can forward it on to the creator (Matthew Perks). For the purpose of this tutorial and walkthrough, we will be using the files found here (available until June 2022). These are from the paper that comes with KLT-IV (V1.0) and is an example from the River Feshie, using a fixed camera at an oblique angle to the river. A disclaimer here: it is not a perfect example of a video we would usually look to run through KLT, we are simply using it as an example for simplicity. Usually we would look to capture more GCPs, at a less oblique camera angle, and for a longer duration.

To use KLT-IV, it is required to know the following parameters:

1. GCP coordinates (real or local, but must be consistent with the rest of the coordinates)
2. Camera location (coordinates, see above)
3. Camera angles (tilt, yaw, pitch)
4. Water level (WSE)
5. Camera model (if listed), to remove distortion

KLT-IV has been shown to be a reliable method of capturing surface velocity and calculating discharges, but also is less sensitive than other methods to user inputs such as defining interrogation parameters. The most important parameters to get as accurate as possible are the GCP coordinates, camera location, and the WSE, as these define the transformation applied to the images during orthorectification. The camera model and angles are somewhat corrected during the orthorectification stages and are not vital for accurate results, but are preferable to have, especially on long-term periodic data. Where imagery is not steady, stabilisation can be carried out automatically in-app by the software, done by defining the orientation and setting it as one of the "dynamic" settings (usually "GCPs + stabilisation").

7.2.2 Video Inputs

Once MATLAB2019 Runtime is installed, then you are able to open and run KLT-IV 1.01. There are several parameters to define in the video inputs section including the video/s to be processed and the camera data. The first job to complete is to select the mode (usually a single file, but multiple can be selected to process a batch of videos). For the purpose of this walkthrough, we will be processing a single file. Select Mode - Single Video, and then go to Define Video(s) and browse to your file (the tutorial file is called "KLTIVTutorialVid". It will ask if you want to re-encode your video, but usually the file format videos come in are compatible with Matlab, so default is 'No'. Next you need to select your camera type if it is known and listed; if not, select "Not listed" (select not listed for the sake of this example). The orientation of your camera depends on the type of the platform you are using (mobile or fixed), and the angle of capture (nadir or angled). Where a platform is fixed but undergoes movement (e.g. wind oscillations etc.), then stabilisation should still be selected and dynamic should be selected. This can sometimes greatly increase the processing time. For the purpose of the tutorial, we should selected stationary and GCP.

Next we need to define some of the coordinates and angles that we are capturing from. For the tutorial, these have been provided as follows: 284954, 804757, 238.47 (X,Y,Z), and can be seen in the GCP spreadsheet "GCPS and camera location". For the angle of the camera, it does not have to be exact as these will be nullified by the orthorectification process if there are enough tracers, but does help the process if given accurately. For the purpose of this study, we will set our values (given in radians), to 0, 0.43, and 0, see figure 7.13 below.

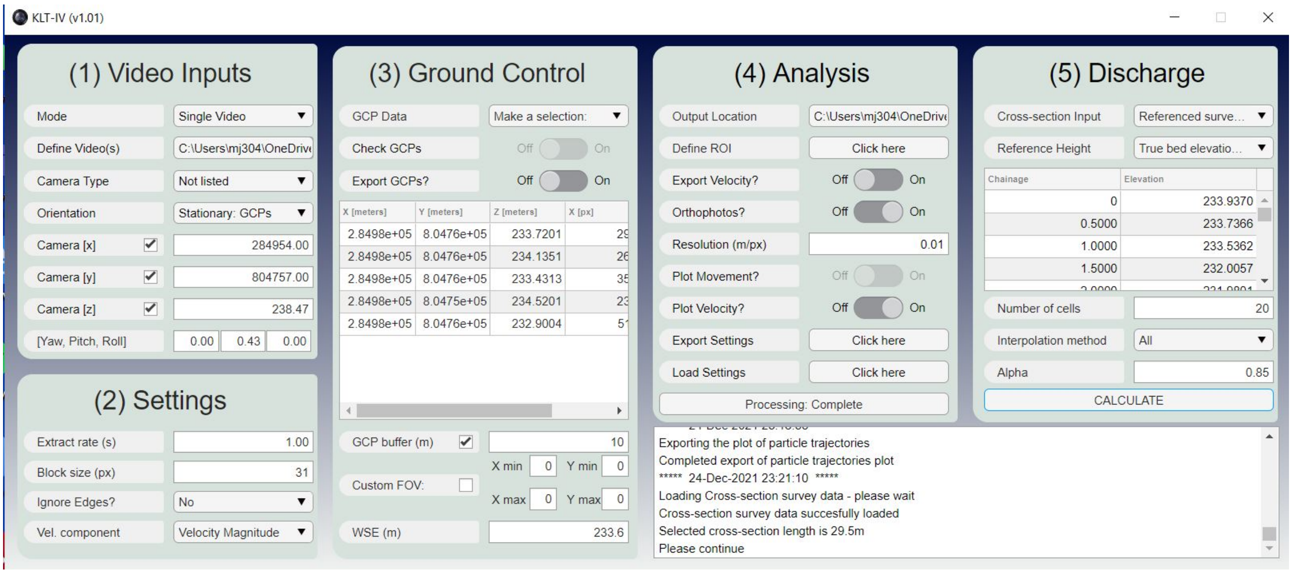


Figure 7.13: KLT GUI with the full inputs shown.

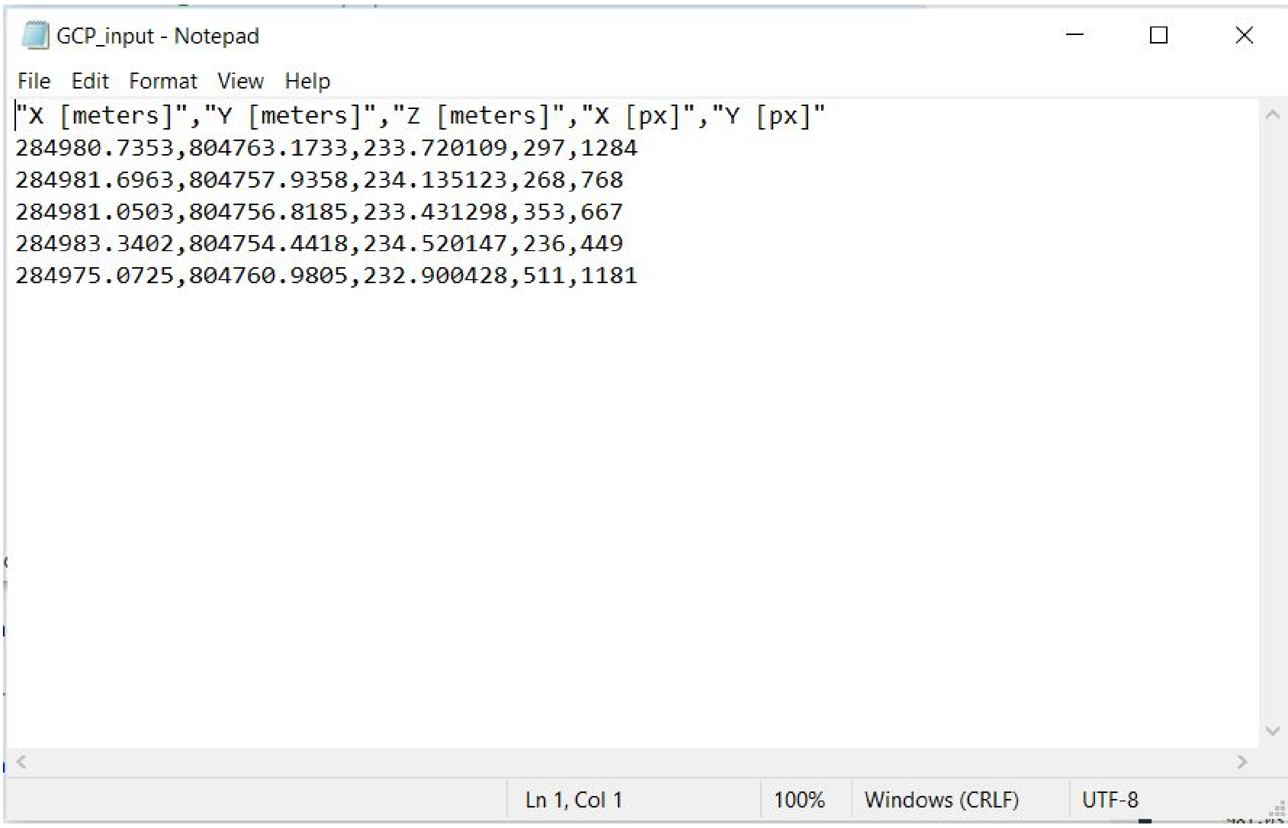
7.2.3 KLT Settings

Settings in KLT define how you want to go about processing the data, and includes the extract rates, block sizes (in pixels), including or ignoring the edges of the videos, and what component you want to measure (velocity or normal). To understand more on how to use these settings, see the paper [11] which explains them all in detail, but I will briefly summarise them here too. The extract rate is the time between tracking features and displaying their displacement, where flows are fast, it will be more advisable to decrease the time between capturing displacements, as the smaller the number the greater the number of trajectories. Alternatively, the larger the extract rate, the smoother the trajectories will be and possibly the more robust. Anything between 0.5s and 2s is advised. Due to the nature of KLT, the block size input for settings is insensitive to input, and can be anything larger than the minimum (5px), but the default which is recommended most of the time is 31px. Ignoring the edges usually is advised where camera parameters such as lens distortion are not confidently removed. It simply ignores the outer 10% of the frame so that these trajectories do not influence the overall accuracy of the output. The software provides velocity magnitudes in metres per second and in X and Y components. You can decide how to input these settings, play about with them but for the sake of the tutorial, let's leave them as default (1.00, 31, No, Velocity Magnitude).

7.2.4 Ground Control

The GCP input for KLT is an important aspect of the processing. Data can be input in three different ways: Manually (as a table format), from a CSV file (X,Y,Z,x,y format like with RIVeR, see figure 7.8), or selected from the image. Manually means you have to enter the known coordinate points along with the corresponding pixel coordinates if known. From CSV requires you to have the GCPs in a format of XYZ (real world coordinates) and xy (pixel coordinates). The software then loads in these values as if they were manually inputted. The "From image" option is a useful tool that KLT has over RIVeR. It opens up an image from the sequence that allows a user to click on a pixel and define its real world coordinates one at a time. Once finished, it will load the values into the table, again, as if done manually. These can then be exported by selecting "export GCPs?". Where "Dynamic: GCPs" was selected earlier in case of mobile platforms, then GCPs are tracked between frames iteratively. It is advised to select "Check GCPs" where they are hard to visually see (and therefore hard to track).

The GCP buffer option allows the user to extend the FOV around the GCPs for the defined distance (e.g. the default of 10m extends the FOV out by 10m past the GCP network). However, if a specified FOV is defined, then the software will only extend to this field of view. Finally we need to define the WSE. This means working in the same coordinate grid as the camera and the GCPs. It is simply the Z coordinate of the camera, minus the difference between the camera and the water level, e.g. camera Z coordinate of 100m and the water surface is 6m below that, then the WSE is 94m. For the tutorial, the GCP CSV is provided (separate to the one with the camera location), this is named "GCP_input" in the tutorial files; let us set our buffer area to 4m.




```
"X [meters]","Y [meters]","Z [meters]","X [px]","Y [px]"
284980.7353,804763.1733,233.720109,297,1284
284981.6963,804757.9358,234.135123,268,768
284981.0503,804756.8185,233.431298,353,667
284983.3402,804754.4418,234.520147,236,449
284975.0725,804760.9805,232.900428,511,1181
```

Figure 7.14: To format a simple CSV file that contains the GCP data required for processing, then it needs to be laid out as X,Y,Z,x,y; where X,Y,Z are real-world coordinates and x,y are their pixel coordinate counterparts.

7.2.5 Analysis

The analysis section is the final section we need to edit before extracting velocity data. Firstly, simply select a destination that you wish to store all of your data by pressing on the "Output Location" and finding the folder you want to save to. Next, the ROI needs defining, and this is a simple process in KLT. Click on "Define ROI" and right click a polygon area that you wish to process. This is not a necessity (unless running stabilisation, in which case, it is), and the program will simply revert back to the area defined by the GCP buffer instead. Choose whether or not you wish to export velocity outputs and orthophotos, for the purpose of the tutorial, draw an ROI along the river (as seen in figure 7.15 and check both "Export velocity" and "Orthophotos" so you can see how it outputs this data; these can be played about with after if you wish. For resolution, there are maximum values, but the software sorts this for you if it is ever exceeded; anything around 0.01 is usually fine. The "Plot movement" function is used for dynamic platforms and tracks the movement of the camera through time, for the tutorial, this is not needed. Finally, lets plot the velocities and click "RUN" to begin the program tracking. The results that you get can be exported to a spreadsheet, but can also be plotted on the orthorectified image as seen in figure 7.16. For this tutorial, we are more interested in the discharge outputs.

Figure 7.15: When setting out a ROI, be sure to only include areas of the river surface, and try to leave out any vegetation that may disturb surface velocity results.

7.2.6 Discharge

Within KLT is the built in ability to apply cross-sectional data to the velocimetry results either using a 'referenced survey' (in the same coordinate grid as the GCPs), or as 'relative distances' where two points can be identified in frame to work from. For a referenced survey, your CSV file needs to have a heading in the first row in the order of X,Y,Z(depth). For relative distances, after picking the start and finish point of the cross-section, you need to define the chainage (distance along the line e.g. 0m,0.5m,1m,1.5m etc. for a cross-section with a reading every 50cm), and the Z(depth). Regarding the depth, you need to tell the software if you are defining a bed elevation, or the water depth.

A search distance is used here for pairing velocity data ready to find the median around each "node" (a measurement along the transect). This is followed by the type of interpolation you wish to do where tracers

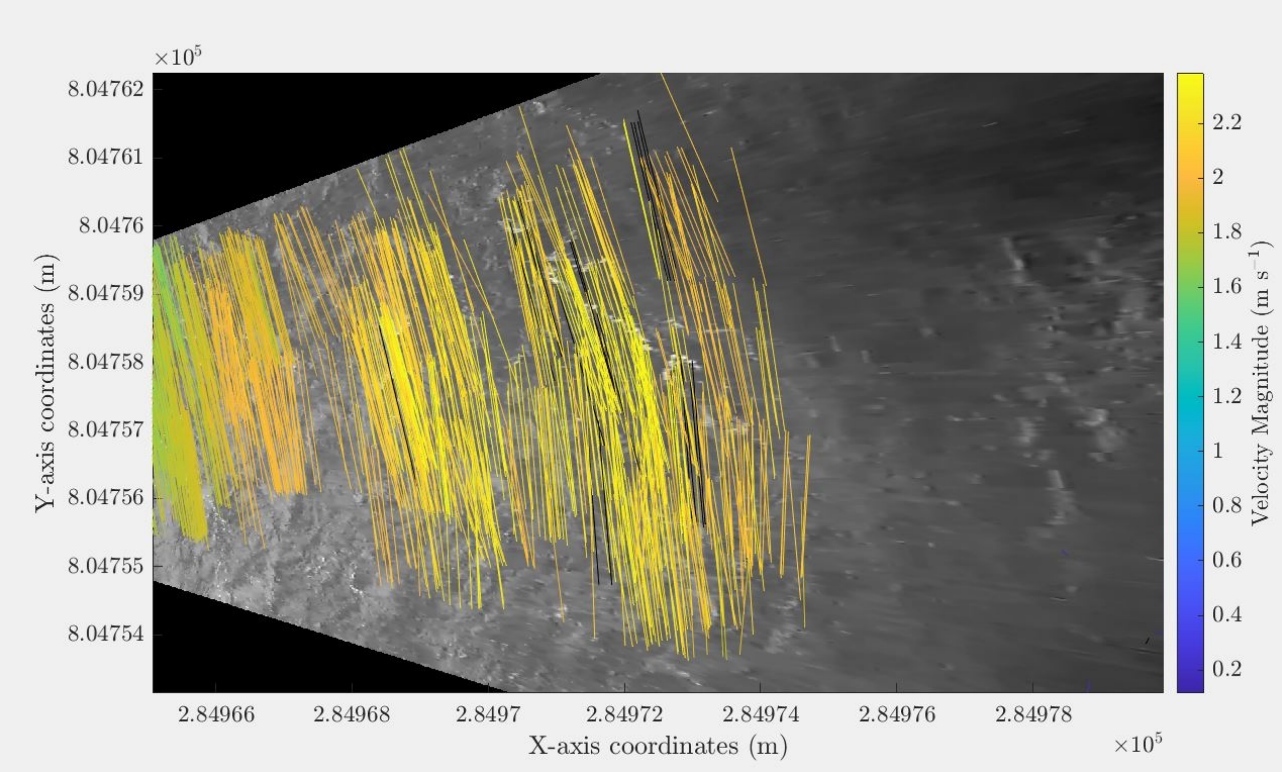


Figure 7.16: Velocity results that have been tracked across the surface. Colour shows indication of speed, while the vector shows the vector trajectory and tracked distance.

are not present. For channels that are fairly uniform, in that the velocity is fastest in the middle and slowest at the edges, quadratic can be used. For flows that are asymmetric or have secondary peak flows, polynomial may work best. For any other type of flow, try using the constant Froude method. Finally, you need to define the depth-average velocity constant (or Alpha value), which studies show can default at 0.85 but if you have specific data for the cross-section then use that. For the purpose of this tutorial, a csv file is prepared ready for use regarding a referenced survey cross-section that shows bed elevation, this is named "XS_input". For the number of cells, keep it at 20, set the interpolation to 'all', and the alpha value to 0.85. Finally, hit "Calculate". You should get around 16-17m $^3/s$ for flow depending on how you play about with any of the numbers (see figure 7.17.

7.2.7 Summary

The main pieces of information required to process videos in KLT-IV are a full survey of the site, including water levels, camera locations, and a minimum of 8 GCPs. Cross-sections are also required if the user wishes to calculate discharges from the velocity data. Optional inputs are the camera type if it is listed in the program, and a more specific alpha value (if obtained using means such as an aDcp). Stabilisation is done automatically for the user if they have defined that the platform is dynamic (not stationary).

KLT-IV is a standalone piece of software created on MATLAB by Matt Perks at Newcastle University. The aim of the software was to provide a user-friendly PTV software that can be used for mobile and fixed platforms alike. The user-friendly GUI walks the user through each of the steps required for processing, from pre-processing of the video, to processing, and finally, through to post-processing and discharges. The simple simplicity of KLT-IV makes it a very useful tool for processing of videos to achieve discharges.

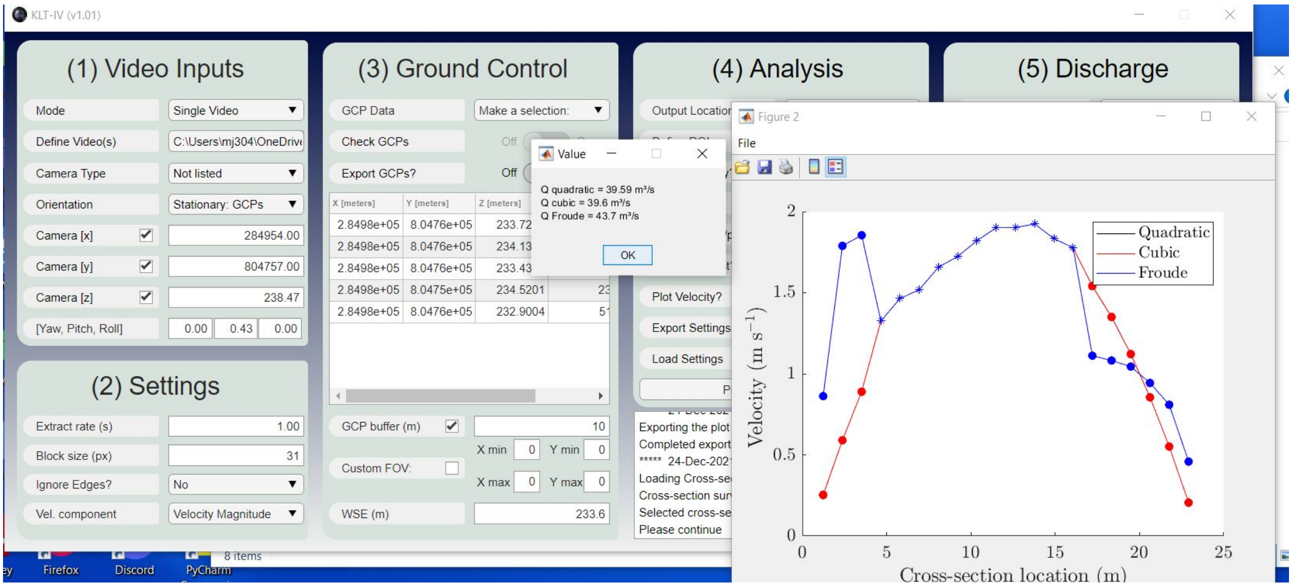


Figure 7.17: The discharge results after running the full process are shown here.

If you want to trial more datasets and have a go at processing either multiple videos at once, or using dynamic platforms, then there is a repository found here: `https://zenodo.org/record/3882254#.YcW3cGjP1Pb` that was created with the original KLT-IV that contains dozens of files ready to run through the software for training.

7.2.7.1 Troubleshooting

Sometimes with an unexpected crash or closure of the program, it can cause some of the MATLAB Runtime paths to corrupt. An error that can be common with unexpected/forced shutdowns is a "ctfroot" error (see figure 7.18). To resolve this error, we need to remove some temporary files that have not been cleaned up from the previous run. To do this, open "My documents" and find the temp folder for KLT-IV. The path for this is something along these lines:
"C:\Users\"USER"\AppData\Local\Temp\"USER"\mcrCache9.7"
Next, once you have found this folder, highlight all the files inside the mrCache9.7 folder, and delete them. Run the software again and hopefully these temp files come back and the program starts as usual. For Mac and Linux operators, the process will be the same but the path may be different.

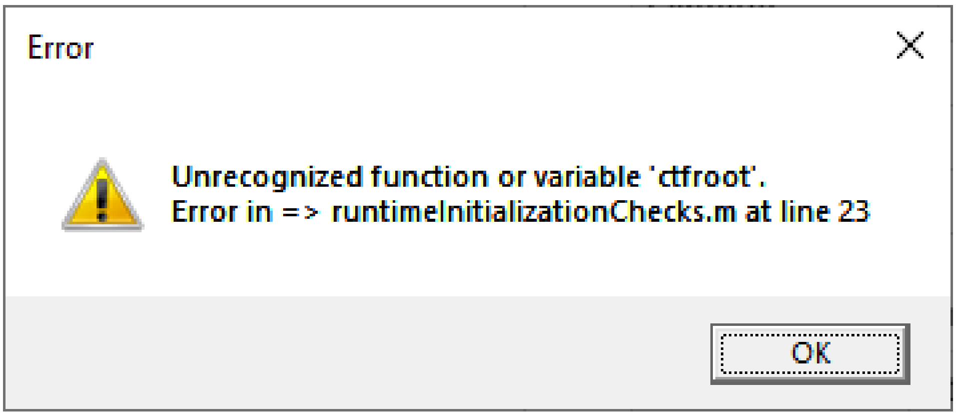


Figure 7.18: A possible error code you can observe with numerous unexpected closures or shutdowns of Runtime.

7.2.8 Tutorial in a nutshell

To make things simpler and save some time, the following points will walk you through the tutorial files to come up with a full process using given values. Simply follow these steps carefully to fully extract images and process the data to find an estimated discharge.

1. Download the files required for KLT-IV including the software and the tutorial files, found here (available until June 2022).
2. Open KLT and wait for the program to fully boot. Select "single file", then define the video by finding it in the tutorial folder.
3. Select "not listed" from the camera list. Although the camera may be defined in the list, and it is usually the best way to proceed, we are just going through as default.
4. Enter the camera location coordinates as follows: 284954, 804757, 238.47 (X,Y,Z respectively). The camera angles to be entered are 0, 0.43, and 0 (Yaw, Pitch, Roll).
5. For the settings, we will be leaving them as default (1, 31, no, velocity magnitude)
6. The GCP data has been provided as the CSV file named "GCP_input". Import this data, and set the buffer to 10m, leaving the custom FOV, and setting the WSE to 233.6m.
7. Next, define where you want to save your output files, followed by defining the ROI as seen in figure 7.15. Export the velocities and orthophotos if you wish.
8. Leave the resolution to 0.01 for the first run but feel free to play about with this, or any other value to see how it impacts results. Finally, plot the velocity and analyse the data.
9. Once finished, lets find the discharge. Import the cross-section input using the file XS_input. This has True bed elevations in it, so set the reference height to that.
10. Lets set the number of cells to 20, and look at how all of the interpolation methods change the data. Leave the alpha value at 0.85 as default, but play with this too to see how it impacts results. Press calculate when this is done.

Chapter 8 Conclusions

This handbook provides an insight into how image velocimetry operates, including the overarching theory as well as the practical exercise required to set-up stations. The initial chapters break down image velocimetry into its key stages and what practically is needed to be considered at each of the stages. By doing it this way, the hope is that a process is formed for when users arrive at a site, and begin to go through a sort of checklist for determining whether or not a site is compatible with the methods. The checklist should run through the order of the contents; note, that it always begins with tracer availability. Environmental conditions and infrastructure available are smaller considerations, as it is possible to work around these by creating new structures, using battery packs, and protecting the camera from external conditions (e.g. placing a hood over the camera to keep away rain). Once a site is determined feasible, the specifics of the set-up need to be considered, such as the camera to be used and at what height it needs to be set to get an optimum GSD. Good surveys of the site need to be taken at some stage prior to processing the data. This can be done after the site set-up if the conditions on the day are too challenging, and data can be captured and later adapted with orthorectification. Where movement is detected in these captured images, it is also to consider performing stabilisation to remove any spurious movements and vectors. Where errors are still evident, post-processing correction methods should be used, and are often available within software suites. This all needs to be done prior to calculating any discharges.

The type of hardware used can be user defined; there is no single best way of creating a site as it fully depends on the conditions and the desired outputs. This handbook has looked into some typically used pieces of hardware and methods to acquire footage and get them to the stage of processing. It is always up to the user to see if there are more specific ways of achieving results, as this research provides a generic set-up that can be used in a majority of cases. The important steps are capturing the footage (camera), and delivering that data to a place it can be processed either using in-situ processing (module), or remotely (network). The advantage of using a network connection to a site, as well as using an IP camera as suggested in the handbooks methods, is that it is possible to control the site fully remotely as well as view and capture data live.

The software used by the user, again, is user defined. Both PIV and PTV have their own advantages and disadvantages as listed throughout this handbook. For highly seeded flows, PIV may be the better option because of the relatively smaller processing required. But for other flows, it may be better to consider a PTV method. RIVeR and KLT-IV are the respective PIV and PTV methods suggested for using at sites because of their availability and performance track record. For the purpose of initial stages of implementing these sites, it is advised to use RIVeR to begin with, and where outputs are not reliable, then try other methods (KLT-IV).

The modal set-up considered in this handbook is the use of a Reolink RLC camera which provides a great amount of freedom of movement, good IR capabilities, illumination built in if needed, and the ability to connect to a router wirelessly which is ideal where satellite routers have been installed at a site and mains power is available. Typically, a LattePanda Delta has been advised where in-situ processing is not required. This is because of its simplicity, with windows capabilities, it should be relatively easy to plug and go. It is typically only used to control the camera and convey the data from site to cloud storage, which means something as powerful as the UDOO Bolt V8 is not needed and would be mostly redundant, whereas it is easier to configure on average than a RPi4B. In terms of networking, mobile internet is always the best method of connectivity because of its reliability, availability, and price. Where sites are not covered by mobile internet, it may be best to reconsider where you place the site. If it is not possible to do so, consider if it is more feasible to process the data on site or simply just store the image data on site, and manually collect it routinely for bulk processing.

The alternative to this is to install satellite internet at the site to be used for connectivity and data transfer. This process can be more complicated and costly, but once installed, could be useful for a number of things regarding image velocimetry methods (e.g. wireless cameras like the Reolink GO and RLC). This process also suggests the use of an ultrasonic level meter where only high flows are required, but cameras and deep learning algorithms can be used to determine stage where depth boards are already in place. Alterations like this are the bits that are down to the user and are mostly site specific.

8.1 What Next?

Version 1 of this handbook has been written in line with the first phase of several in the effort to develop a systematic way of using image velocimetry in everyday hydrometry. The methods described here are for beginners in the field, and provide only the basics of how to produce reliable results using imaging methods. There are more advanced features that can be used in conjunction (e.g. dual camera systems for 3D modelling of flow), but these are still being developed in research, and will not feature in any of the immediate phases.

The second phase of the roll-out of image velocimetry should begin to think about how to mass produce sites based on the results that come from this initial stage. The use of image velocimetry on sites can massively benefit the automated capture of larger flows, usually missed by other methods. Sites across the country that require the capturing of these higher flows should immediately be considered for installation of a site, especially as the cost of implementing a site costs on average £1000, which is relatively cheap compared to alternative hydrometry methods. A second benefit that could come from the second phase would be introducing some more smart technology, such as using deep learning on all of the cameras to detect the water stage without the need of any other implements. Not just this, but anything along the lines of it that can begin to reduce the need of secondary items and increase the automated efficiency of sites. A review of the use of RIVeR/KLT-IV should also be made. This software was recommended because of its wide availability and ability to produce reliable results. However, it is noted that specialist software like these can be hard to understand and use efficiently. Although these two pieces of software have been developed to be user-friendly with accessible GUI's, there may be alternatives that are better suited, and hence, a review could benefit. Further studies should also begin to include mobile methods (e.g. drones), ready for use in the field. Many of the methods for mobile platforms are the same as fixed, with the addition for a few extra steps such as stabilisation.

Two model sites are to be created in January 2022, using Hikvision cameras and latte pandas. Once these sites are set up and fully operational, this handbook should be edited to include all of the details and pictures that came from these set-ups, as well as any insights into issues we run into.

Bibliography

[1] Haytham Assem et al. "Urban Water Flow and Water Level Prediction Based on Deep Learning". In: (2017). ISSN: 16113349. DOI: 10.1007/978-3-319-71273-4_26.

[2] Mallikarjun Mishra et al. "Precision mapping of boundaries of flood plain river basins using high-resolution satellite imagery: A case study of the Varuna river basin in Uttar Pradesh, India". In: *Journal of Earth System Science* (2019). ISSN: 0973774X. DOI: 10.1007/s12040-019-1146-1.

[3] Ray Kostaschuk et al. "Measuring flow velocity and sediment transport with an acoustic Doppler current profiler". In: *Geomorphology* 68 (1-2 May 2005), pp. 25–37. ISSN: 0169-555X. DOI: 10.1016/J.GEOMORPH.2004.07.012.

[4] Simon P. Neill and M. Reza Hashemi. "In Situ and Remote Methods for Resource Characterization". In: *Fundamentals of Ocean Renewable Energy*. 2018. DOI: 10.1016/b978-0-12-810448-4.00007-0.

[5] J. D. Creutin et al. "River gauging using PIV techniques: A proof of concept experiment on the Iowa River". In: *Journal of Hydrology* (2003). ISSN: 00221694. DOI: 10.1016/S0022-1694(03)00081-7.

[6] M. Jodeau et al. "Application and evaluation of LS-PIV technique for the monitoring of river surface velocities in high flow conditions". In: *Flow Measurement and Instrumentation* (2008). ISSN: 09555986. DOI: 10.1016/j.flowmeasinst.2007.11.004.

[7] S F Dal Sasso et al. "Exploring the optimal experimental setup for surface flow velocity measurements using PTV". In: *Environ Monit Assess* 190.8 (2018), p. 460. DOI: 10.1007/s10661-018-6848-3. URL: https://www.ncbi.nlm.nih.gov/pubmed/29998453.

[8] Matthew T. Perks, Andrew J. Russell, and Andrew R. G. Large. "Technical Note: Advances in flash flood monitoring using unmanned aerial vehicles (UAVs)". In: *Hydrology and Earth System Sciences* 20.10 (Oct. 2016), pp. 4005–4015. ISSN: 1607-7938. DOI: 10.5194/hess-20-4005-2016. URL: https://www.hydrol-earth-syst-sci.net/20/4005/2016/.

[9] W. Brevis, Y. Niño, and G. H. Jirka. "Integrating cross-correlation and relaxation algorithms for particle tracking velocimetry". In: *Experiments in Fluids* (2011). ISSN: 07234864. DOI: 10.1007/s00348-010-0907-z.

[10] J. Le Coz et al. "Image-based velocity and discharge measurements in field and laboratory river engineering studies using the free FUDAA-LSPIV software". In: (2014). DOI: 10.1201/b17133-262.

[11] M. T. Perks. "KLT-IV v1.0: Image velocimetry software for use with fixed and mobile platforms". In: *Geoscientific Model Development Discussions* (2020). DOI: 10.5194/gmd-2020-187.

[12] Y. Kim et al. "Stream discharge using mobile large-scale particle image velocimetry: A proof of concept". In: *Water Resources Research* (2008). ISSN: 00431397. DOI: 10.1029/2006WR005441.

[13] J. Le Coz et al. "Performance of image-based velocimetry (LSPIV) applied to flash-flood discharge measurements in Mediterranean rivers". In: *Journal of Hydrology* (2010). ISSN: 00221694. DOI: 10.1016/j.jhydrol.2010.05.049.

[14] Martin J. Jolley et al. "Considerations When Applying Large-Scale PIV and PTV for Determining River Flow Velocity". In: *Frontiers in Water* 3 (2021), p. 151. ISSN: 2624-9375. DOI: 10.3389/frwa.2021.709269. URL: https://www.frontiersin.org/article/10.3389/frwa.2021.709269.

[15] Jongkook Lee et al. "Accuracy Improvement of Discharge Measurement with Modification of Distance Made Good Heading". In: *Advances in Meteorology* 2016 (2016). DOI: 10.1155/2016/9437401.

[16] Alonso Pizarro, Silvano Fortunato Dal Sasso, and Salvatore Manfreda. "Refining image-velocimetry performances for streamflow monitoring: Seeding metrics to errors minimization". In: *Hydrological Processes* (2020). ISSN: 10991085. DOI: 10.1002/hyp.13919.

[17] Silvano Fortunato Dal Sasso et al. "Increasing LSPIV performances by exploiting the seeding distribution index at different spatial scales". In: *Journal of Hydrology* (2021). ISSN: 00221694. DOI: 10.1016/j.jhydrol.2021.126438.

[18] Martin J Jolley. *GSD CamAngle Est*. https://github.com/MJolley1/GSD_CamAngle_Est/blob/main/MJ_Oblique_CamAngle_Est.xlsx. 2021.

[19] Arsal-Hanif Livoroi et al. "On the Deployment of Out-of-the-Box Embedded Devices for Self-Powered River Surface Flow Velocity Monitoring at the Edge". In: *Applied Sciences* 11.15 (2021), p. 7027. DOI: 10.3390/app11157027.

[20] Markus Raffel et al. *Particle Image Velocimetry: A Practical Guide*. 2018. ISBN: 978-3-319-68852-7.

[21] D. P. Hart. "PIV Error Correction". In: *Laser Techniques Applied to Fluid Mechanics* (2000), pp. 19–35. DOI: 10.1007/978-3-642-56963-0_2.

[22] J. Westerweel. "Efficient detection of spurious vectors in particle image velocimetry data". In: *Experiments in Fluids* (1994). ISSN: 07234864. DOI: 10.1007/BF00206543.

[23] Jerry Westerweel and Fulvio Scarano. "Universal outlier detection for PIV data". In: *Experiments in Fluids* (2005). ISSN: 07234864. DOI: 10.1007/s00348-005-0016-6.

[24] J Duncan et al. "Universal outlier detection for particle image velocimetry (PIV) and particle tracking velocimetry (PTV) data". In: *Measurement Science and Technology* (2010). ISSN: 0957-0233. DOI: 10.1088/0957-0233/21/5/057002.

[25] F. Ioli et al. "EVALUATION OF AIRBORNE IMAGE VELOCIMETRY APPROACHES USING LOW-COST UAVS IN RIVERINE ENVIRONMENTS". In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (2020). DOI: 10.5194/isprs-archives-xliii-b2-2020-597-2020.

[26] Guillaume Dramais et al. "Advantages of a mobile LSPIV method for measuring flood discharges and improving stage–discharge curves". In: *Journal of Hydro-environment Research* 5.4 (Dec. 2011), pp. 301–312. ISSN: 1570-6443. DOI: 10.1016/J.JHER.2010.12.005. URL: https://www.sciencedirect.com/science/article/abs/pii/S1570644311000049.

[27] M. Muste, I. Fujita, and A. Hauet. "Large-scale particle image velocimetry for measurements in riverine environments". In: *Water Resources Research* (2008). ISSN: 00431397. DOI: 10.1029/2008WR006950.

[28] Xin Zhang, Charles S. Cox, and X. Wang. "Further evaluation of feature correlation for PIV and PTV". In: 3172 (1997). Ed. by Soyoung Stephen Cha, James D. Trolinger, and Masaaki Kawahashi, pp. 411–419. DOI: 10.1117/12.279750. URL: https://doi.org/10.1117/12.279750.

[29] William Thielicke and Eize J. Stamhuis. "PIVlab – Towards User-friendly, Affordable and Accurate Digital Particle Image Velocimetry in MATLAB". In: *Journal of Open Research Software* (2014). ISSN: 2049-9647. DOI: 10.5334/jors.bl.

[30] Antoine Patalano, Carlos Marcelo García, and Andrés Rodríguez. "Rectification of Image Velocity Results (RIVeR): A simple and user-friendly toolbox for large scale water surface Particle Image Velocimetry (PIV) and Particle Tracking Velocimetry (PTV)". In: *Computers and Geosciences* (2017). ISSN: 00983004. DOI: 10.1016/j.cageo.2017.07.009.

[31] Flavia Tauro et al. "Optical tracking velocimetry (OTV): Leveraging optical flow and trajectory-based filtering for surface streamflow observations". In: *Remote Sensing* (2018). ISSN: 20724292. DOI: 10.3390/rs10122010.

[32] Maxence Carrel et al. "Evaluation of the DischargeApp: a smartphone application for discharge measurements". In: *HydroSenSoft, International Symposium and Exhibition on Hydro-Environment Sensors and Software.* (2019). URL: http://www.photrack.ch/images/DApp%7B%5C_%7Deval%7B%5C_%7Dvaw.pdf.

[33] João P. Leitão et al. "Urban overland runoff velocity measurement with consumer-grade surveillance cameras and surface structure image velocimetry". In: *Journal of Hydrology* (2018). ISSN: 00221694. DOI: 10.1016/j.jhydrol.2018.09.001.

[34] M. T. Perks. "KLT-IV v1.0: Image velocimetry software for use with fixed and mobile platforms". In: *Geoscientific Model Development Discussions* (2020). DOI: 10.5194/gmd-2020-187.

[35] Ichiro Fujita, Hideki Watanabe, and Ryota Tsubaki. "Development of a non-intrusive and efficient flow monitoring technique: The space-time image velocimetry (STIV)". In: *International Journal of River Basin Management* (2007). ISSN: 18142060. DOI: 10.1080/15715124.2007.9635310.

[36] I. Fujita et al. "Efficient and accurate estimation of water surface velocity in STIV". In: *Environmental Fluid Mechanics* (2019). ISSN: 15731510. DOI: 10.1007/s10652-018-9651-3.

[37] Ryota Tsubaki. "On the Texture Angle Detection Used in Space-Time Image Velocimetry (STIV)". In: *Water Resources Research* (2017). ISSN: 19447973. DOI: 10.1002/2017WR021913.

[38] Carl J. Legleiter, Paul J. Kinzel, and Jonathan M. Nelson. "Remote measurement of river discharge using thermal particle image velocimetry (PIV) and various sources of bathymetric information". In: *Journal of Hydrology* 554 (2017), pp. 490–506. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2017.09.004. URL: https://doi.org/10.1016/j.jhydrol.2017.09.004.

[39] Tong Jin and Qian Liao. "Application of large scale PIV in river surface turbulence measurements and water depth estimation". In: *Flow Measurement and Instrumentation* (2019). ISSN: 09555986. DOI: 10.1016/j.flowmeasinst.2019.03.001.

[40] Sophie Pearce et al. "An Evaluation of Image Velocimetry Techniques under Low Flow Conditions and High Seeding Densities Using Unmanned Aerial Systems". In: *Remote Sensing* 12.2 (Jan. 2020), p. 232. ISSN: 2072-4292. DOI: 10.3390/rs12020232. URL: https://www.mdpi.com/2072-4292/12/2/232.

Appendix A Equipment Lists

This chapter is in reference to the spreadsheet tool available on GitHub at https://github.com/MJolley1/Handbook_EquipmentList/blob/main/ImageVelocimetry_EquipmentLists.xlsx. Use it to generally decide the sort of equipment you would need to set up an image velocimtery site at several locations, depending on 6 given variables seen in table 6.3. The full list of results can be seen in the table below for a manual method (see table A.1).

Table A.1: The full results table for the binary output collected from the variable list table. Choose one of the two options for each of the 6 variables to produce your binary result (between 000000-1111111), and match it to the corresponding equipment suggestion in this table.

000000	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
000001	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
000010	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
000011	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
000100	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1326.37
000101	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1326.37
000110	Reolink Go-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1130.1
000111	Reolink Go-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1130.1
001000	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
001001	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
001010	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
001011	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	1104.37
001100	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1326.37
001101	Hikvision DS-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1326.37
001110	Reolink Go-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1130.1
001111	Reolink Go-+-RSPi 4B-+-Mob_Vodafone_PiHat-+–+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	1130.1
010000	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	597.4
010001	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	597.4
010010	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	597.4
010011	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	597.4
010100	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	819.4
010101	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	819.4
010110	Reolink Go-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	792.73
010111	Reolink Go-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	792.73
011000	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	597.4
011001	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	597.4
011011	Reolink Go-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+–+–+-	570.73
011100	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	819.4
011101	Vivotek HTV-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	819.4
011110	Reolink Go-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	792.73
011111	Reolink Go-+-RSPi 4B-+–+-RSPi 4B_SSDUnit-+-12V_Battery_and_Case-+-RS Solar Panel Unit-+-MaxSonar-+–+-	792.73
100000	Hikvision DS-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+–+–+-	1066.11
100001	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+–+–+-	738.53
100010	Hikvision DS-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+–+–+-	1066.11
100011	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+–+–+-	738.53
100100	Hikvision DS-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	1288.11
100101	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	960.53
100110	Hikvision DS-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	1288.11
100111	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	960.53
101000	Vivotek HTV-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+–+–+-	896.51
101001	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+–+–+-	738.53
101010	Vivotek HTV-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+–+–+-	896.51
101011	Vivotek HTV-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+–+–+-	896.51
101100	Hikvision DS-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	1288.11
101101	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	960.53
101110	Hikvision DS-+-UDOO Bolt-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	1288.11
101111	Vivotek HTV-+-LattePanda-+-Mob_Vodafone_USB-+–+–+–+-MaxSonar-+–+-	960.53
110000	Hikvision DS-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+–+–+-	1118.79
110001	Vivotek HTV-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+–+–+-	949.19
110010	Reolink RLC-+-LattePanda-+-Konnect Sat. Unl-+–+–+–+–+–+-	1227.52
110011	Reolink RLC-+-LattePanda-+-Konnect Sat. Unl-+–+–+–+–+–+-	1227.52
110100	Vivotek HTV-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1171.19
110101	Reolink RLC-+-LattePanda-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1018.4
110110	Reolink RLC-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1176.38
110111	Reolink RLC-+-LattePanda-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1018.4
111000	Reolink RLC-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+–+–+-	954.38
111001	Reolink RLC-+-LattePanda-+-Konnect Sat. Unl-+–+–+–+–+–+-	1227.52
111010	Reolink RLC-+-LattePanda-+-Konnect Sat. Lmt-+–+–+–+–+–+-	796.4
111011	Reolink RLC-+-UDOO Bolt-+-Konnect Sat. Unl-+–+–+–+–+–+-	1385.5
111100	Reolink RLC-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1176.38
111101	Reolink RLC-+-LattePanda-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1018.4
111110	Reolink RLC-+-UDOO Bolt-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1176.38
111111	Reolink RLC-+-LattePanda-+-Konnect Sat. Lmt-+–+–+–+-MaxSonar-+–+-	1018.4

Index