USER MANUAL

Revision 1

March 7, 2025

SPEC hardware and software are designed, developed, and manufactured by Deep Analytics, LLC in Montpelier, Vermont.

Visit the public GitLab page for support and documentation.

gitlab.com/deepanalyticsllc/spec.git

Revision	Date	Summary
Release	February 10, 2025	--
Revision 1	March 7, 2025	Changed SPEC sled mount

Note to the User:

Setup PIV Edge Camera (SPEC) was a research and development project funded by Next Generation Water Observing System (NGWOS) of the United States Geological Survey (USGS). SPEC is a bank mounted camera system that uses Particle Image Velocimetry (PIV) to calculate the speed at which the water at the surface of a river channel is moving. If something about SPEC doesn't work right, or needs improvement, please let us know.

TABLE OF CONTENTS

Set up PIV (Particle Image Velocimetry) Edge Camera - SPEC 5

INTRODUCTION 5

SYSTEM CAPABILITIES 5

PREREQUISITES 5

Required/Suggested Hardware 6

SYSTEM POWER 6

MAJOR PARTS 6

Background/Backend Information 7

OPERATING SYSTEM 7

System Rules 7

Threading 8

Logs 8

PIV PROCESS 9

Image Preprocessing 9

PIV Algorithm 10

CONFIG.JSON 10

SAVE AND DELETING DATA 11

WI-FI AND CAPTIVE PORTAL 12

WEB APPLICATION 12

HOMOGRAPHY 13

Setup 18

HARDWARE INSTALLATION 18

IMU and Camera 18

Raspberry Pi Fan 18

Real Time Clock 19

3D Printed Sled 20

Camera Enclosure and Clip-ins 26

OPERATING SYSTEM INSTALLATION 28

SPEC SOFTWARE INSTALLATION 31

SPEC IMU CALIBRATION, CAMERA CALIBRATION AND CAMERA PARAMETERS 32

IMU Calibration 32

Gather Your Camera Specs 34
Perform Camera Calibration 34
Configure Camera Parameters 35
Optional: Customizing background image 36
Web Application 37
CONNECTING TO THE SPEC 37
CALIBRATION SPLASH 38
Calibrate 39
Main Menu 40
UTILITIES 40
Site Info 40
Live Video 40
Data Management 40
View Logs 41
Check Disk Space 41
Reboot System 41
SETUP AND CALIBRATIONS 41
Trapezoid Calibration 41
PIV Parameters 44
Masking Options 45
Digitize Mask 46
Generate Mask 48
CAMERA PARAMETERS 48
PIV FUNCTIONS 48
Run Test 48
Run PIV 49
Results 50
OFFSITE PROCESSING 52
DISCLAIMERS 53
PIV Accuracy Requirements 53
HOMOGRAPHY AND ITS IMPACT 53
NIGHTTIME PIV 54
References 54

SET UP PIV (PARTICLE IMAGE VELOCIMETRY) EDGE CAMERA - SPEC

INTRODUCTION

Setup PIV Edge Camera (SPEC) was a research and development project funded by Next Generation Water Observing System (NGWOS) of the United States Geological Survey (USGS). SPEC is a bank-mounted camera system that uses Particle Image Velocimetry (PIV) to calculate the speed at which the water at the surface of a river channel is moving. The system is a fully contained data collection and processing unit. Users can customize settings such as how often to capture a sequence of images, the duration of the image sequence, the time between each image in the sequence (frame rate), and other parameters related to the PIV calculations via a user interface (UI) that is integrated with the system and accessed through a Wi-Fi connection. The system was built with citizen scientists in mind, making the cost of materials as low as possible, providing all 3D print files, and providing the code as an open-source public GitLab repository.

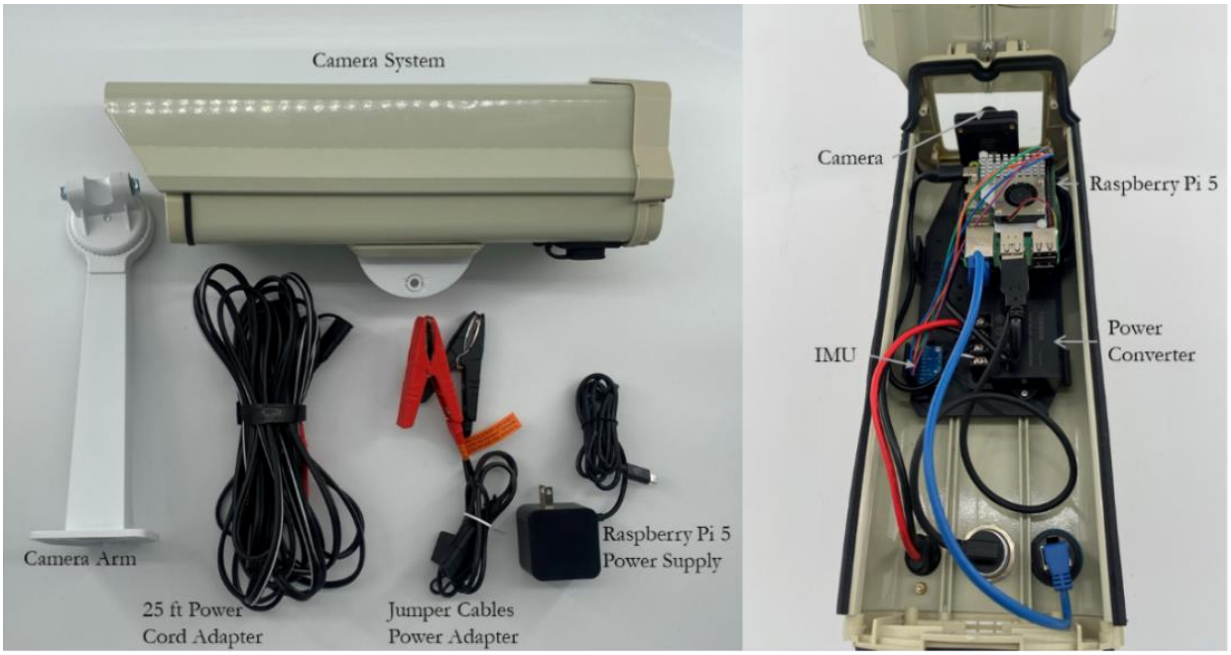

Figure 1. SPEC system.

SYSTEM CAPABILITIES

SPEC, as seen in Figure 1, was built to fit the needs of NGWOS by providing a bank-mounted camera that can be deployed in a remote location to measure river velocity, with the goal of detecting increased velocity events (flooding). This system is intended to be set up once, left in a stable position, and used to produce river velocity vector fields at a user-defined interval. The system saves video, inertial measurement unit (IMU) data, velocity outputs, and other important parameters for each run.

PREREQUISITES

- Raspberry Pi 5 (8 GB ram tested).
- IMU, camera and other parts as listed below and in the bill of materials (BOM).

- Ability to solder IMU header pins if required.
- Familiarity with Linux command line interface.
- Basic understanding of running Python scripts.
- Access to a local or online 3D printing service.
- A passion for moving water.

REQUIRED/SUGGESTED HARDWARE

Links are to sources from which we have ordered hardware previously and are only provided as suggestions, not specific requirements.

Please refer to the complete bill of materials (BOM) for all parts.

SYSTEM POWER

1. We have included a DC-DC power converter that accepts 12V/24V and outputs 5V to power the Pi. To power the system, you could use a car/marine battery, a USB battery pack, a solar power solution or "plug in" with existing power solutions at the field site.

MAJOR PARTS

2. If you want to connect your Pi to a monitor during setup, you will need a monitor cable with a micro HDMI connector on one end. You'll also need a keyboard and a mouse. It is also possible to perform a headless install where you will log into your Pi remotely.
3. Raspberry Pi 5 - 8 GB RAM (Pi 5 versions with less RAM and older versions have not been tested)
4. Official Raspberry Pi 27W USB C power supply.
5. Raspberry Pi 5 fan.
6. microSD card: Choose the size that fits your field-testing situation. We recommend 512GB for several months of storage, depending on your settings.
7. Camera. Here is the camera we tested.
8. You will need female-female jumper wires and if you want to breadboard the project before installing in the 3D printed carrier, you'll need a few more parts. There are many sources for this. As an example of one that has far more than you need for this project, check here.
9. IMU (Inertial Measurement Unit) MPU9250. Other IMUs could be used but this is the only one that has been tested.
10. Camera Enclosure This is the enclosure that our 3D printed "sled" fits. Other enclosures have not been tested.

11. 3D printed sled for mounting the Pi, camera and IMU. Download STL files – see Figure 2.

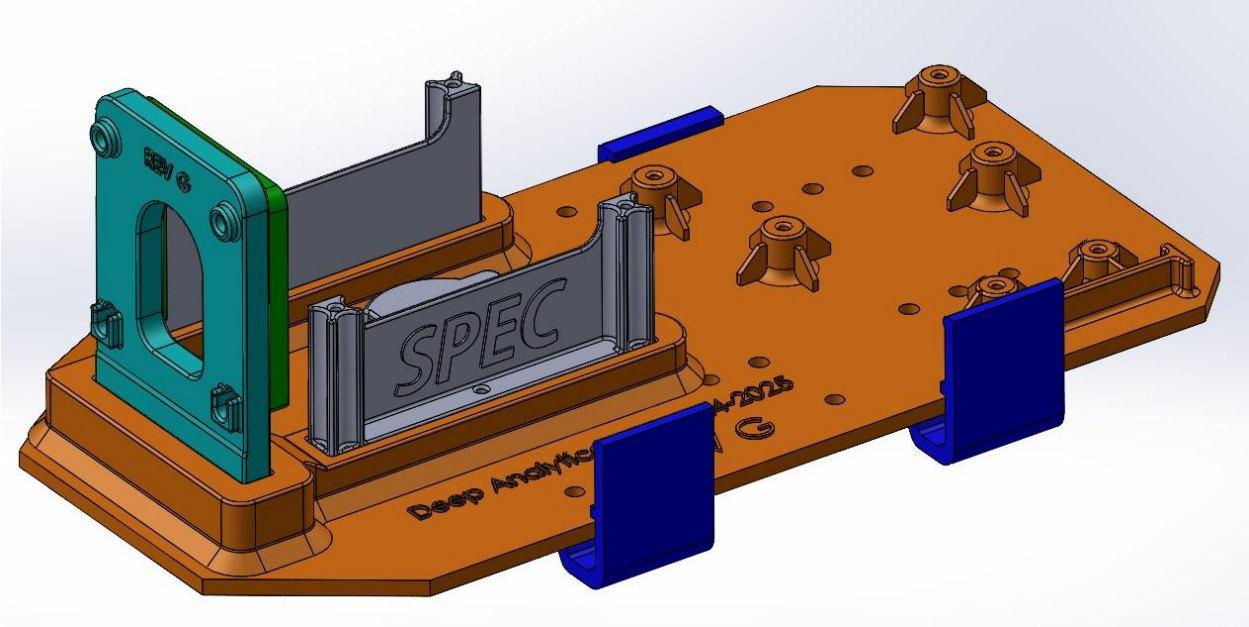

Figure 2. 3D printed sled.

12. Soldering iron and solder.

BACKGROUND/BACKEND INFORMATION

This system was designed to be a completely self-contained package for acquiring images and performing PIV calculations. Once set up, you can leave it out and it will measure velocities from the bank of the river continuously. The following are brief descriptions of the main processes as well as some background information on the use of homography.

OPERATING SYSTEM

The Raspberry Pi 5 images are configured with a Linux-based operating system.

SYSTEM RULES

The SPEC codebase operates using six systemd rules. A systemd rule is a script that the system executes automatically each time it reboots. This setup ensures simplicity and allows for seamless startup whenever the Raspberry Pi 5 is powered on.

These systemd rules are established through a setup script. Once configured, users simply need to power up the Raspberry Pi 5 in the field and the entire system will be ready to use.

The six systemd rules are as follows, as in Figure 3:

1. One rule sets up the two virtual cameras.
2. One rule streams the camera to the two virtual cameras.
3. One rule sets up the Wi-Fi access point.
4. One rule starts the web-based user interface.

5. One rule handles disk space management.
6. The final rule initiates a script to run the main velocity calculation process.

This design ensures the system is efficient, user-friendly, and field-ready with minimal effort

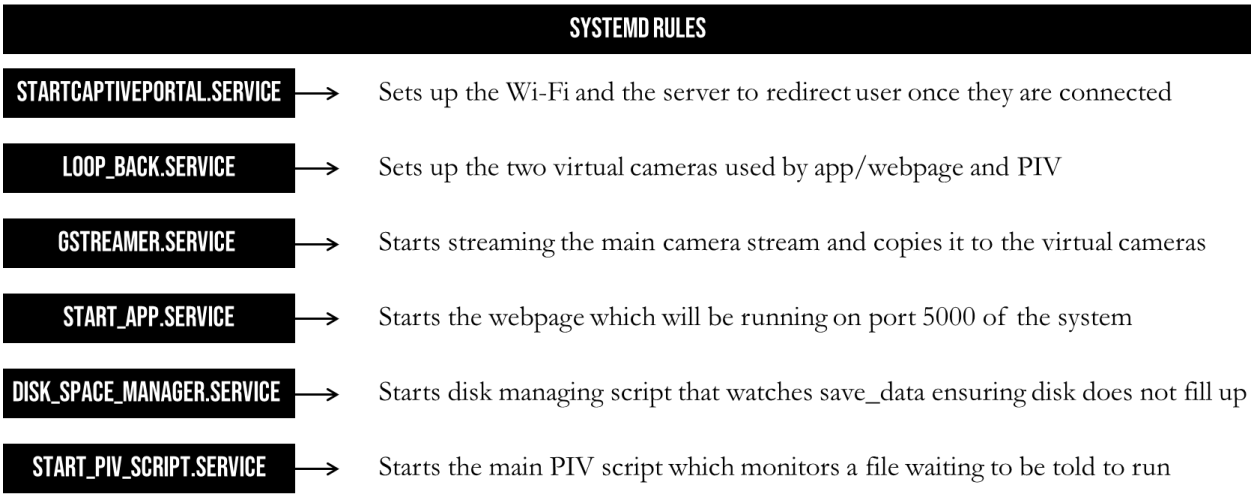

Figure 3. System rules block diagram.

THREADING

The SPEC system architecture leverages multi-threading in several key operations on the Raspberry Pi 5 (RPi5) to enhance computational speed and efficiency. Four main operations utilize threading:

1. Image Preprocessing: Accelerates image transformations and preparations.
2. IMU Reading in the Web Application: Ensures real-time sensor data is smoothly integrated.
3. Camera Frame Reading in the Web Application: Improves streaming performance for the user interface.
4. Fast Fourier Transform (FFT) in the PIV Algorithm: Speeds up velocity calculations for PIV processing.

Threading in the web application ensures a smoother user experience, while threading in image processing and PIV operations optimizes the velocity calculation process for maximum efficiency.

LOGS

All processes running on the RPi5 using SPEC's codebase generate logs, which are stored in the /var/log directory. Each log file corresponds to a specific systemd rule, as follows:

1. start_app: Logs all print and error statements from the web application to usgs_app.log.
2. start_PIV_script: Logs print and error statements from the main PIV process to piv_process.log.
3. loop_back: Logs details related to loopback operations to loopback.log.
4. Gstreamer: Logs camera streaming activities to Gstreamer.log.
5. disk_space_manager: Logs deleted file paths used to maintain sufficient disk space for the operating system in disk_space.log.
6. startCaptivePortal: Does not have a dedicated log file as the software manages its own logging.

The logging system is designed to take advantage of the circular buffer functionality in the /var/log directory. This ensures that log files are recycled once they reach a predefined size, preventing the logs from taking up too much disk space.

PIV PROCESS

The following details how images are processed for PIV and briefly describes what happens on the backend during PIV calculations.

IMAGE PREPROCESSING

For the PIV algorithm to function correctly, raw frames from the camera must undergo preprocessing (Figure 4) This process consists of five steps:

1. Removing distortion:
    - Uses the camera matrix and distortion coefficients to correct radial distortion, which is common with wide-angle lenses or certain cameras.
    - Ensures all pixels represent consistent spatial areas across the image.
2. Homography:
    - Transforms the bank-mounted camera view into a nadir perspective, so that the area of the river surface represented by a single pixel is consistent throughout the image.
3. Masking:
    - Applies a user-defined mask to focus on specific areas, typically isolating the river section for analysis.
4. Enhancement:
    - Uses a CLAHE filter to enhance the visibility of distinct features on the water surface features, improving the algorithm's ability to detect and track them.

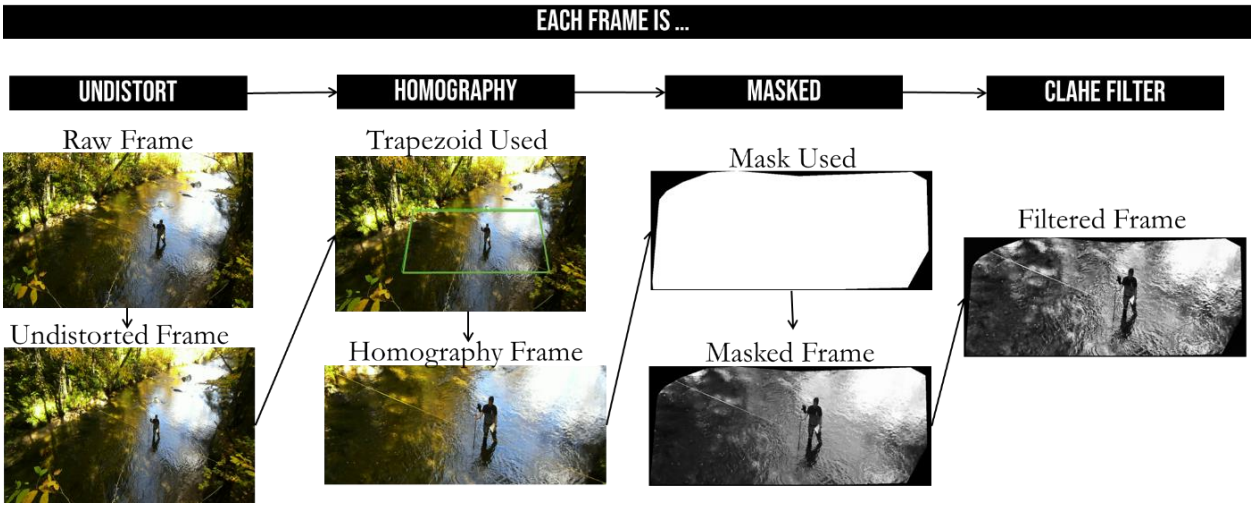

Figure 4. Preprocessing example.

PIV ALGORITHM

The PIV algorithm used to calculate river velocity is based on the Toolbox for River Velocimetry using Images from Aircraft (TRiVIA) developed by the USGS [1] and was ported from MATLAB to Python. All related scripts are available from the PIV folder within the SPEC GitLab repository. The primary scripts driving the process are callPivLab.py and ensemblePIV.py, while the remaining scripts serve as helper functions for these main components.

PROCESS OVERVIEW

callPivLab.py:

- Loads image file paths and parameters.
- Prepares the data for processing in ensemblePIV.py.

ensemblePIV.py:

- Processes images in batches to prevent the RPi5 from running out of RAM and crashing.
- Performs the Fast Fourier Transform (FFT) and correlation steps on each batch.
- Combines batch results and applies post-processing steps, including peak finding, smoothing, and infilling.
- Produces a vector field of velocity in pixels.
- The resulting vector field is returned to callPivLab.py, where it is scaled to meters per second and masked to a user-defined region. The final velocity field is then saved as CSV files.

CONFIG.JSON

To feed in the parameters to the backend code from the web app, we use a config.json file that contains all user input. This is the file that the PIV calculation reads from to perform a run based on the user input settings. This file is saved along with the resulting data that was acquired based on these settings. This allows the user to reproduce runs or look back at the parameters used to get the data (Figure 5).

Figure 5. An example of the config.json file where all parameters are stored.

SAVE AND DELETING DATA

Because SPEC is intended to be a riverbank-mounted camera left to run continuously, a lot of data could be stored. With this in mind, we designed a directory architecture for the **save_data** that keeps your data organized and reproducible. Every time you choose to run a new PIV run, whether it be a test or a continuous run, a new folder is made in the **save_data** folder named by the date. If

it is a test, it lists the data_test001 where 1 refers to the test number you are on. Inside that folder a timestamped and dated folder is located that has the video, IMU data, and the outputs from the PIV run performed at that time. If this is a continuous PIV run, then there will be multiple run folders in the overall config folder. Lastly, the config folder contains the config.json with all the parameters used for each run and the mask used for each run (Figure 6).

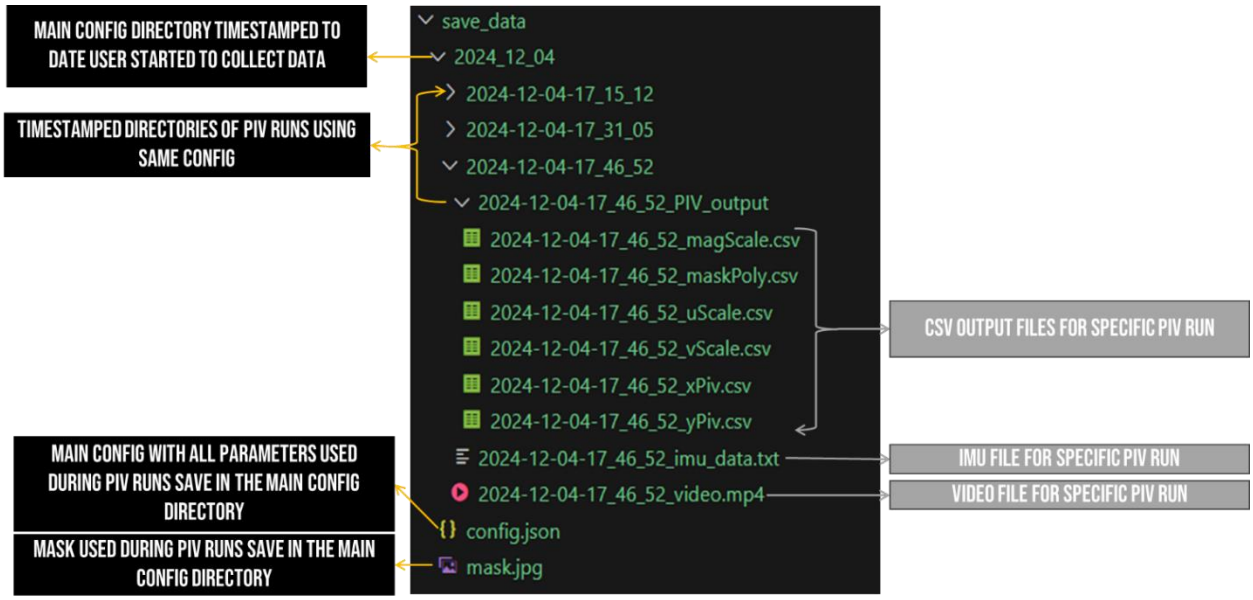

Figure 6. A graphic explaining the architecture of the save_data folder.

There is a **Data Management** function in the web application that will give you the option to save the data to a USB and then delete specific directories of data.

WI-FI AND CAPTIVE PORTAL

The web application is accessible by connecting to the Wi-Fi signal broadcast by the RPi5. Two key software tools enable this functionality:

- Hostapd: Allows users to configure the Wi-Fi name and password during setup and then broadcasts the signal.
- DNSMasq: Assigns IP addresses to devices connected to the Wi-Fi.

To streamline access, a captive portal is implemented. When users connect to the Wi-Fi, they are automatically redirected to the app's login page. This is managed by Nginx, a server that detects Wi-Fi connections and handles redirections.

WEB APPLICATION

The app serves as the main user interface (UI) for setting up PIV calculations on-site. It is powered by Flask, a Python-based framework, for the backend, with the frontend built using JavaScript, HTML, and CSS.

- Backend: Located in the app.py script.
- Frontend:

- Templates directory: Contains all HTML and layout files.
- Static directory: Stores images and other media displayed in the app.

The entire application is contained within the app folder of the GitLab repository that has been cloned to the Raspberry Pi, ensuring a streamlined and portable design.

HOMOGRAPHY

The original PIV algorithm the system is modeled after was adapted from TRiVIA [1], which produces PIV-based velocity estimates from videos and sequences of images with a nadir perspective acquired from aircraft or bridge-mounted cameras. To use this algorithm, we must transform the images obtained from the SPEC system mounted on the bank to look as though they were taken from nadir. We do this via the homography calculation as shown visually in Figure 7.

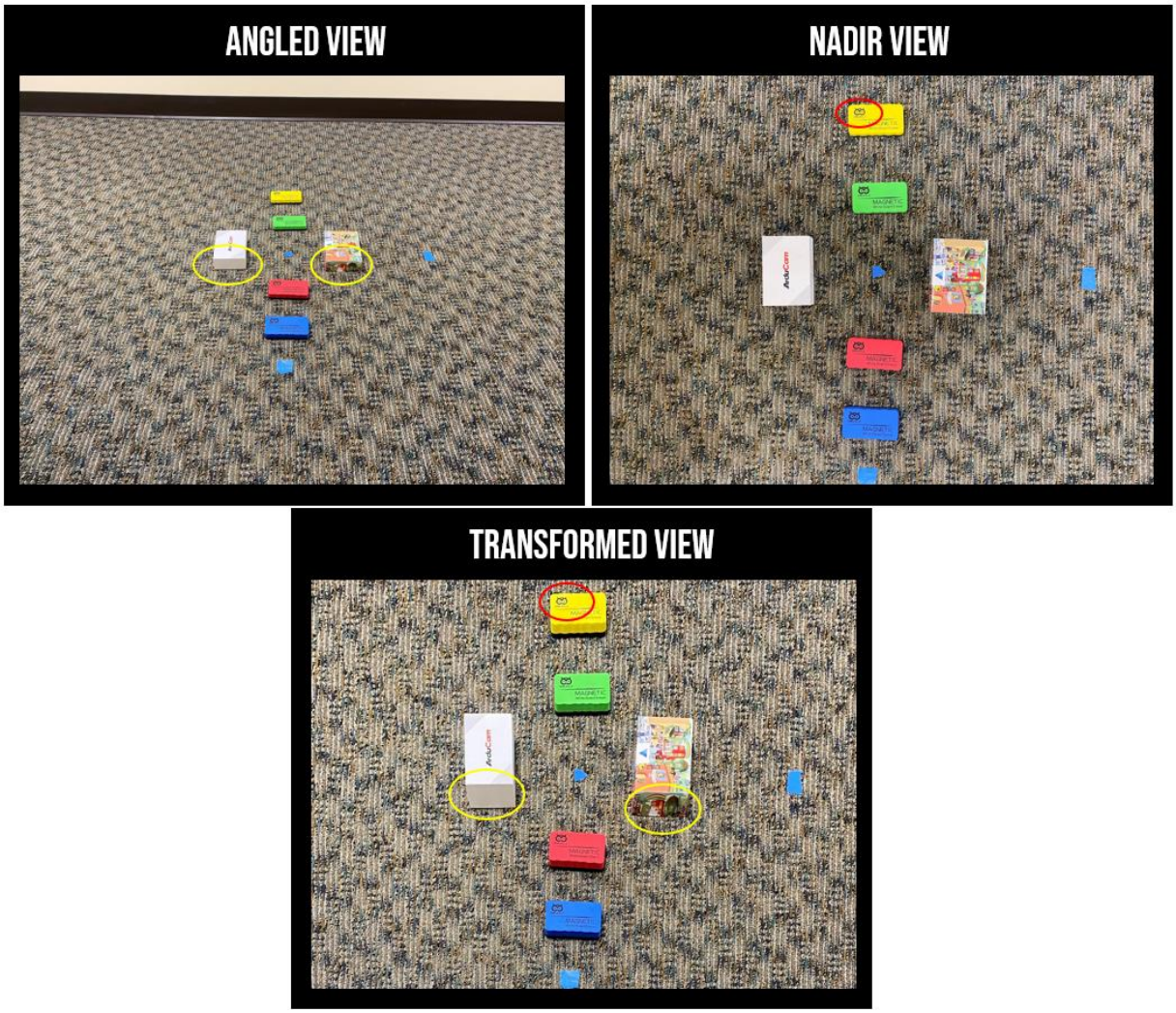

Figure 7. An angled image and nadir image were acquired and then a homography matrix was calculated. The angled image was transformed to a nadir perspective using the homography matrix. The yellow circles show features from the angled image and the red show features from the nadir image.

Homography is a projective transformation between two planes or a mapping between two planar projections of an image. In normal operations when performing homography two photos are taken

and the mapping is calculated to transform one photo to the same view as another. For example, taking an angled view of a scene and a nadir (bird's eye) view of the same scene we can find the mapping between the two (homography matrix) to map the angled view to look as if it was taken at nadir (Figure 7). We can use homography on a river because we are treating the surface of a river as a plane.

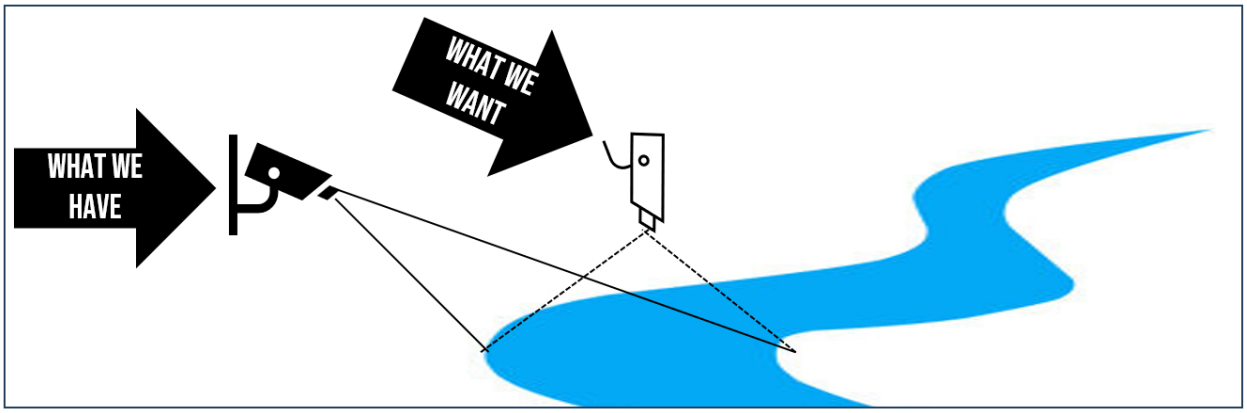

Figure 8. Graphic explaining the camera position we want and the camera position we have.

Collecting a nadir image of a river can be challenging, generally one either needs a UAS (uncrewed aerial system) or crewed aircraft with a camera to ensure the entire river width is captured by the camera. Our goal was to find a way to transform an angled image, like those from a bank-mounted camera, to appear as though it was taken from nadir without using a second nadir image to transform it (Figure 8). There are already ways to do this, one being to use GPS devices to measure real world points and transform the bank-mounted images using that information. These devices are usually costly and not readily available to the average citizen scientist. With these individuals in mind, we explored possibilities of using measurements that are easier to obtain to serve as reference points.

The measurements we need to make this calculation are the height between the camera and the water, and the angle that the camera is tilted at, which can be recorded from an inertial measurement unit (IMU) connected to the camera. Our approach follows Hartley and Zisserman [2], who explain the ideas behind projective geometry. By following real world parallel lines as indicators, we can find points to run homography that will give the nadir view we want (Figure 9). With this idea in mind, we needed to come up with an angle representation of how parallel lines in an image move based on the angle of the camera.

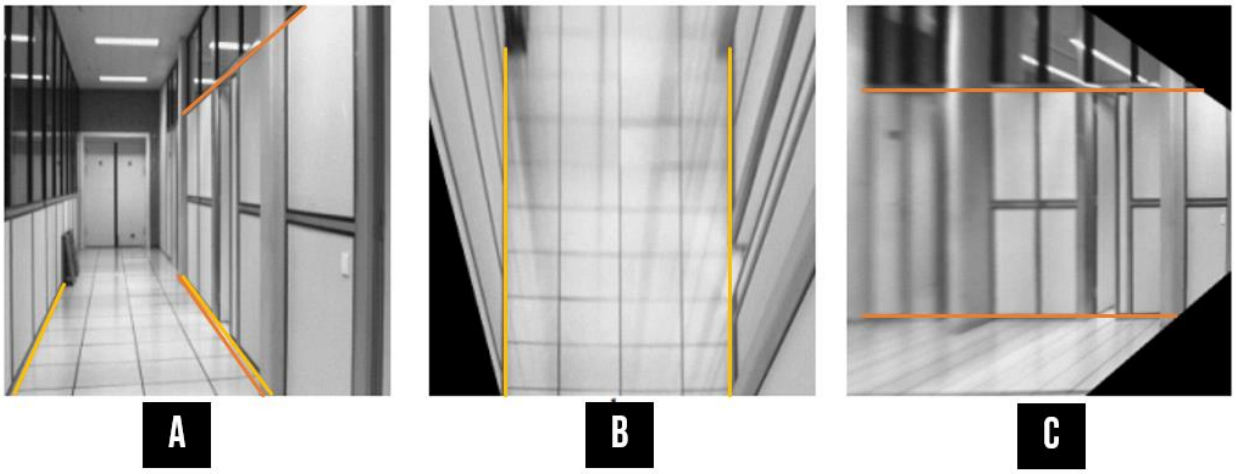

Figure 9. Image A is the original image; image B was transformed with homography with points along the yellow lines in image A. Image C was transformed with homography with points along the orange lines in image A [2].

This led us to the keystone effect. The keystone effect is mainly seen when using projectors. If your projector is pointed straight at the wall, the image on the wall should be perfectly square. As you tilt the projector at an angle, the image on the wall slowly obtains a trapezoid shape as in Figure 10.

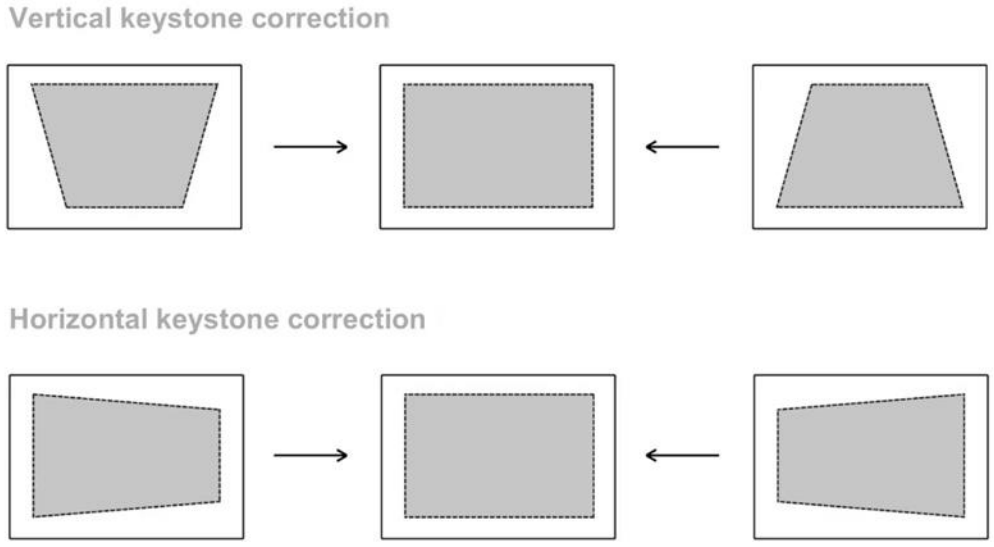

Figure 10. Graphic representing the keystone affect [3].

We are using the ideas of this effect to help map the parallel lines in an image based on the camera tilt angle.

In order to find how the angle of the parallel lines moves, we have to look at the side view and top view of the image plane. This allows us to understand the values we have and the values we need to calculate – see Figure 11. The top view shows the image field of view from an angled camera. We are using the movement of the sides of the top view, line AB in Figure 11, to represent the parallel lines movement in our image field.

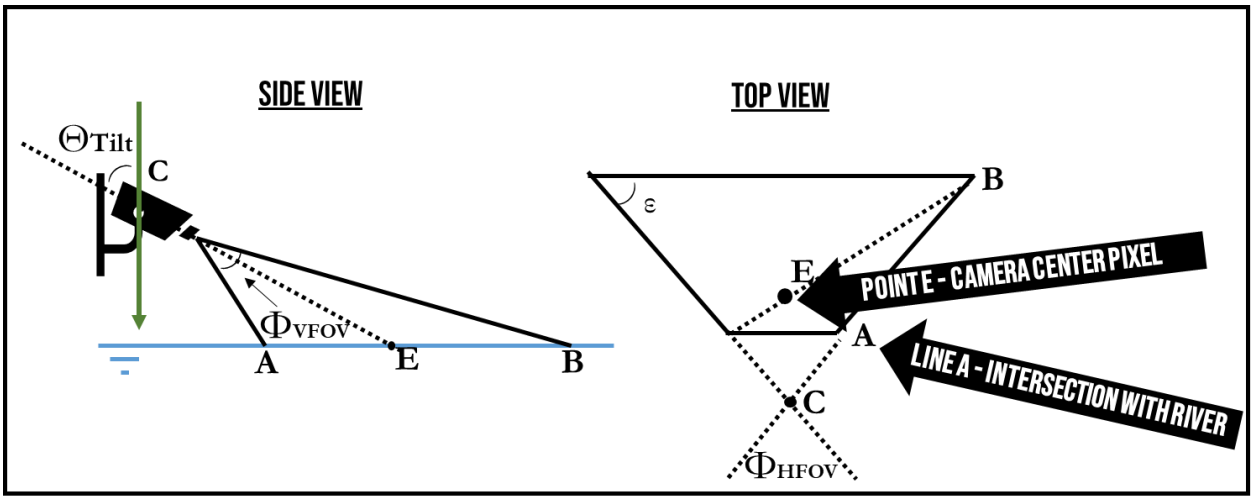

Figure 11. Field of view perspectives.

With these in mind, we treat the field of view (top view) as two normal planes, in relation to the camera, stacked on top of each other. It's similar to taking an image of a wall, then stepping back and taking another image. The two images are what we call the two normal planes. (Figure 12).

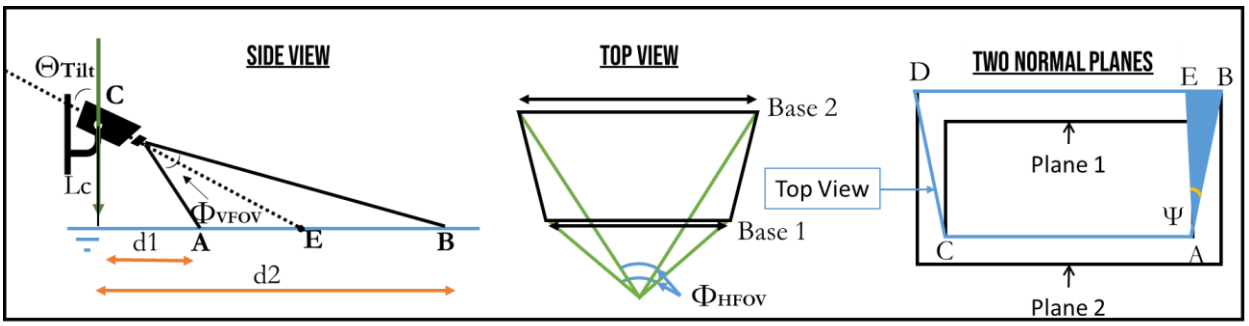

Figure 12. Set up field of view represented as two normal planes, one like the camera is close (small rectangle) and the other like the camera is far (large rectangle).

Using trigonometry, we will set up all the variables needed to calculate the final angle of $\sphericalangle EAB$ (Ψ) (Figure 13 and Figure 14).

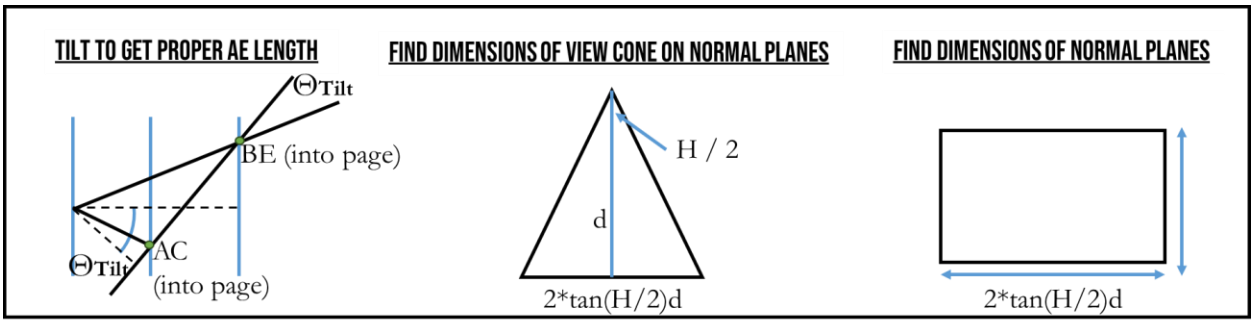

Figure 13. Tilt side-view (left) to get proper length of AE. Use trigonometry to set up the variables needed for future calculations (middle and right)

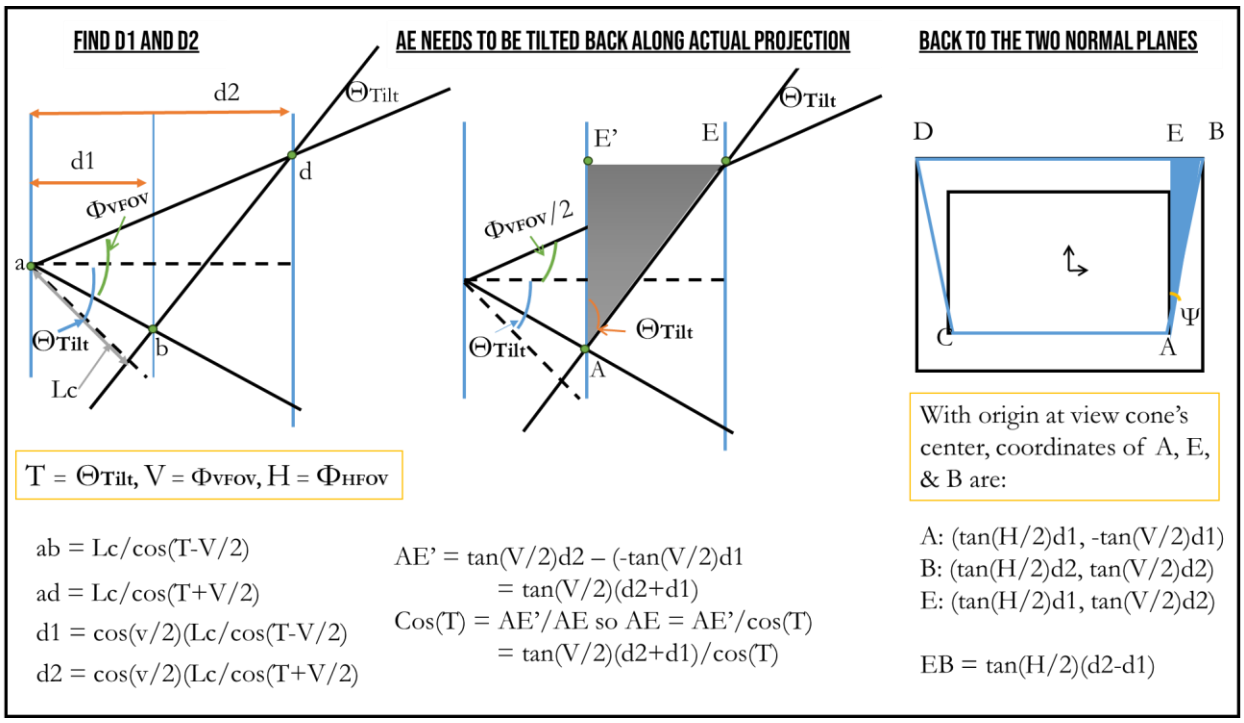

Figure 14. Calculate the distance to the two normal planes (left), tilt AE back along projection (middle) and use values found and plug back in to the two normal plan representation (right).

Using this set up, we solve for the desired angle (Equation 1)

$$\psi = \tan^{-1}\frac{EB}{AE} = \frac{\tan\left(\frac{H}{2}\right)\left(\cos\left(\frac{V}{2}\right)\left(\frac{Lc}{\cos\left(T+\frac{v}{2}\right)} - \frac{Lc}{\cos\left(T-\frac{V}{2}\right)}\right)\right)}{1/\cos(T)\left(\tan\left(\frac{V}{2}\right)\left(\cos\left(\frac{V}{2}\right)\left(\frac{Lc}{\cos\left(T+\frac{v}{2}\right)} + \frac{Lc}{\cos\left(T-\frac{V}{2}\right)}\right)\right)\right)}$$

$$= \frac{\cos(T)\tan\left(\frac{H}{2}\right)\left(\frac{1}{\cos\left(T+\frac{v}{2}\right)} - \frac{1}{\cos\left(T-\frac{V}{2}\right)}\right)}{\left(\tan\left(\frac{V}{2}\right)\left(\frac{1}{\cos\left(T+\frac{v}{2}\right)} + \frac{1}{\cos\left(T-\frac{V}{2}\right)}\right)\right)}$$

Equation 1. Parallel line angle equation.

With this angle calculation, a trapezoid is projected onto a live video whose sides are representative of the parallel lines of the image based on the camera angle. The user then selects the area for which PIV will be performed and those points are saved to transform all images.

SETUP

HARDWARE INSTALLATION

IMU AND CAMERA

To properly run the SPEC software, the IMU and camera need to be connected to the Pi. The camera should be plugged into a USB port, and the IMU should be wired with jumper wire as in Figure 15.

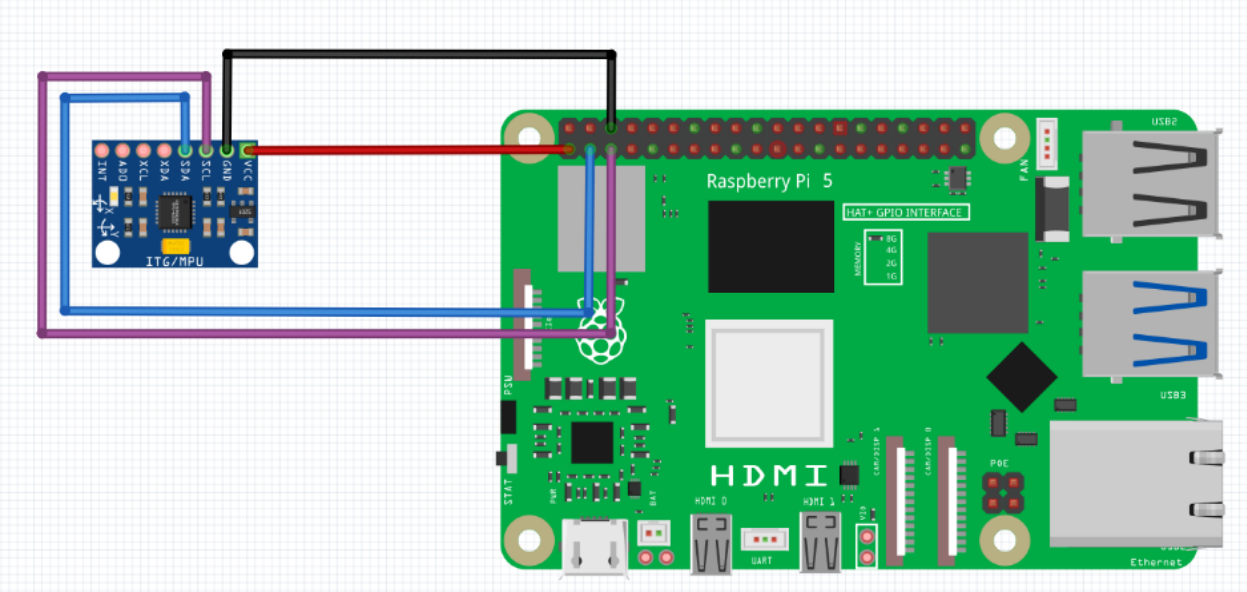

Figure 15. Wiring diagram with IMU.

Notes: The IMU in the wiring diagram does not have as many pins as the one we used for development. Also, the colors of the jumper used in the diagram do not have to be the ones you use. The diagram is for wire placement the colors chosen were used to help distinguish between the four wires. Here is the GPIO pinout for the Raspberry Pi 5.

RASPBERRY PI FAN

To keep the Pi at a safe operating temperature, we have added a fan and heatsink. To install this fan (Figure 16):

1. Peel off the heatsink pad's wrapper
2. Line up the heatsink-fan with the Pi – the cord of the heatsink-fan should be on the side of the Pi with the USB plug-ins
3. Press down on the heatsink-fan corner connectors and push them through the predesigned holes
4. Gently press down on the rest of the heat sink.
5. Plug in heatsink/fan. Note: the plug on the Pi may have a cap on it; you simply need to take it off.

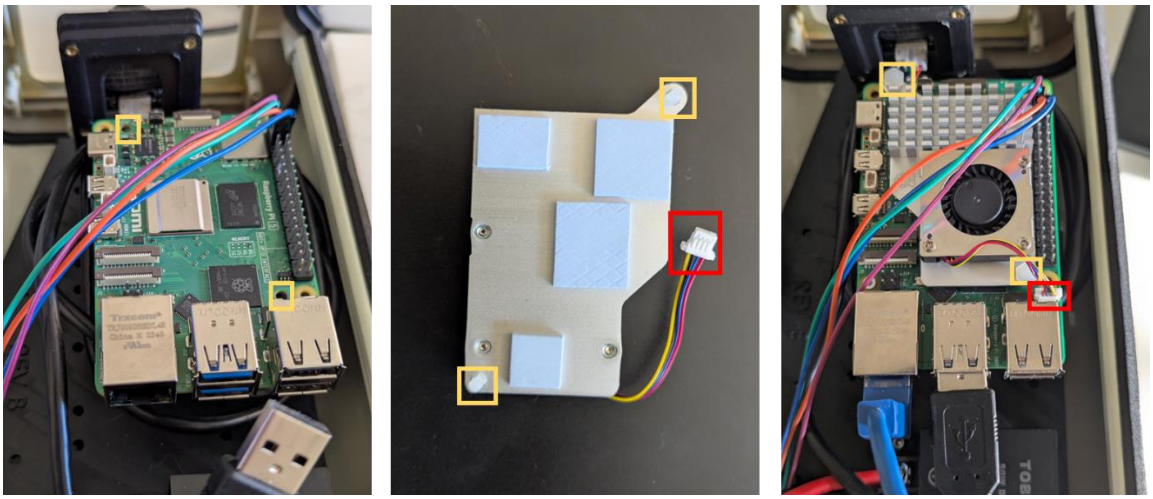

Figure 16. Left: Pi before heatsink/fan, Middle: bottom of heatsink/fan, Right: Pi with heatsink/fan. The yellow squares represent the areas that need to be connected together. The red is where the cord is plugged in

REAL TIME CLOCK

The Raspberry Pi 5 includes a real time clock (RTC) module that will keep track of the time that is set in the operating system even if the unit is disconnected from power. The Pi will retrieve the correct time when it connects to the internet. You must plug a small coin battery into the Pi for the RTC to work. In the bill of materials, the official rechargeable battery with the appropriate 2-pin JST plug is listed. It is not recommended to use a non-rechargeable battery as the current consumption of the RTC module is high and will result in a short life for the battery. Figure 17 shows where to connect the battery, just to the right of the USB-C power connector.

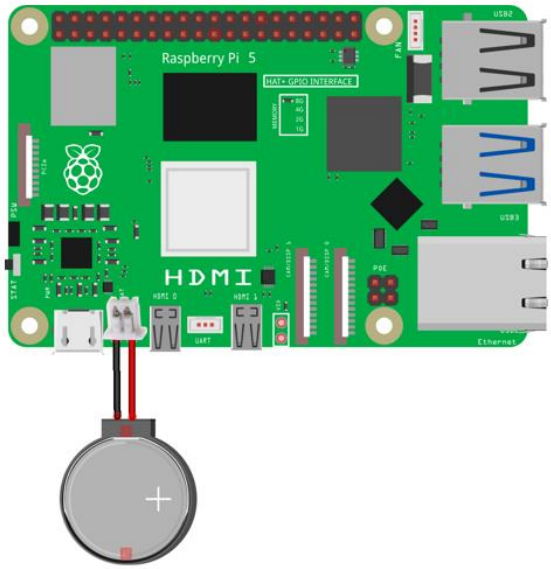

Figure 17. Raspberry Pi 5 and rechargeable coin battery for RTC.

3D PRINTED SLED

In order to mount all of the pieces of the SPEC system together, we have a 3D-Printed sled that holds all the devices (Figure 18). Once the sled is printed, all the pieces should go together nicely. Below are some images showing assembly. You might need to use a bit of force to screw into the 3D printed standoffs.

1. Gather all the needed pieces.

Figure 18. All the pieces needed to assemble the SPEC sled.

2. First screw in SPEC sled mount pieces (Figure 19)

Figure 19 SPEC sled mount pieces

a. Screw in Camera holder (Figure 20).

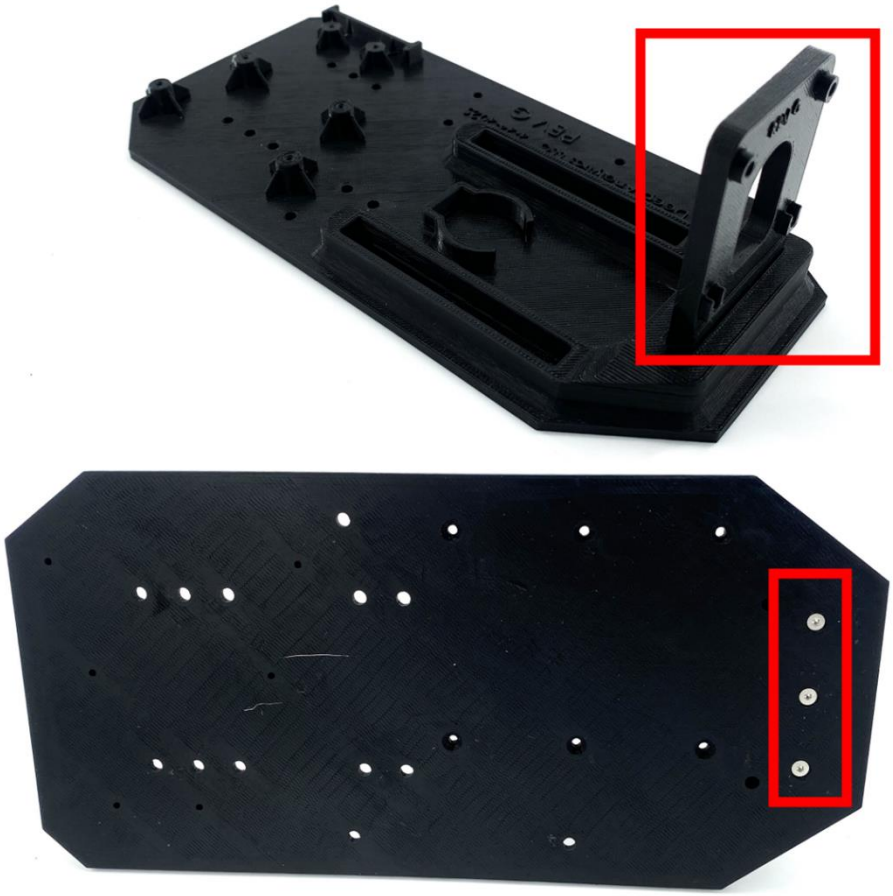

Figure 20 Camera holder assembly

b. Screw in Raspberry Pi 5 holder (Figure 21).

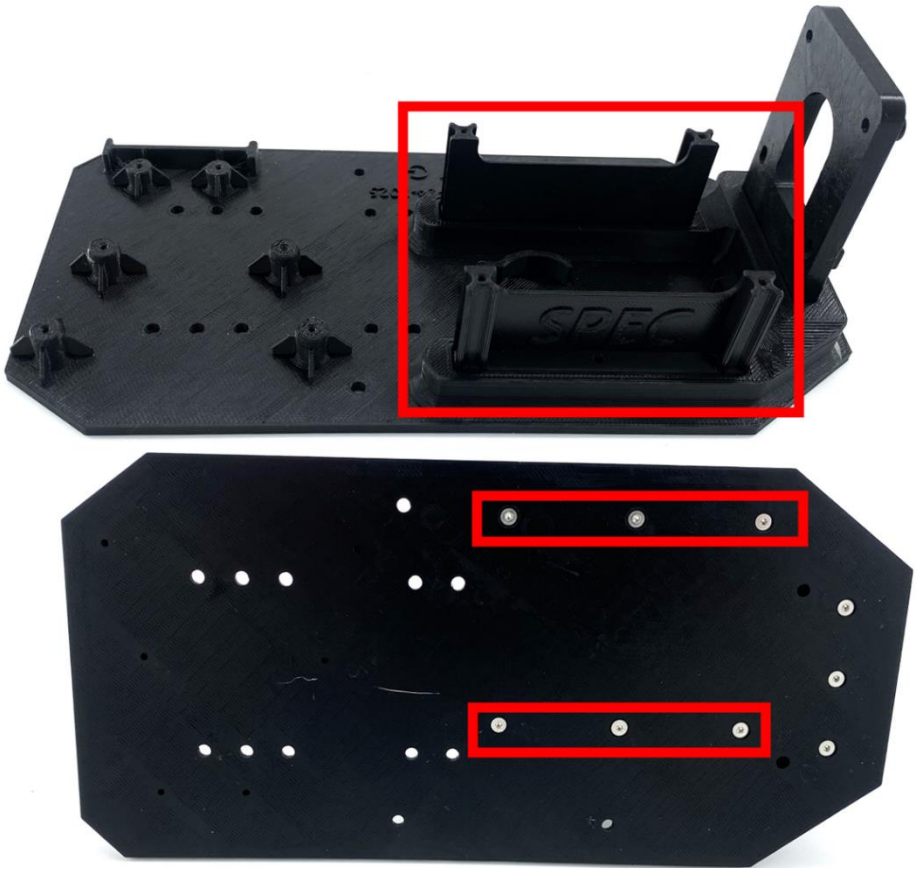

Figure 21 Raspberry Pi holder assembly

c. Place RTC battery in holder and add cover (Figure 22).

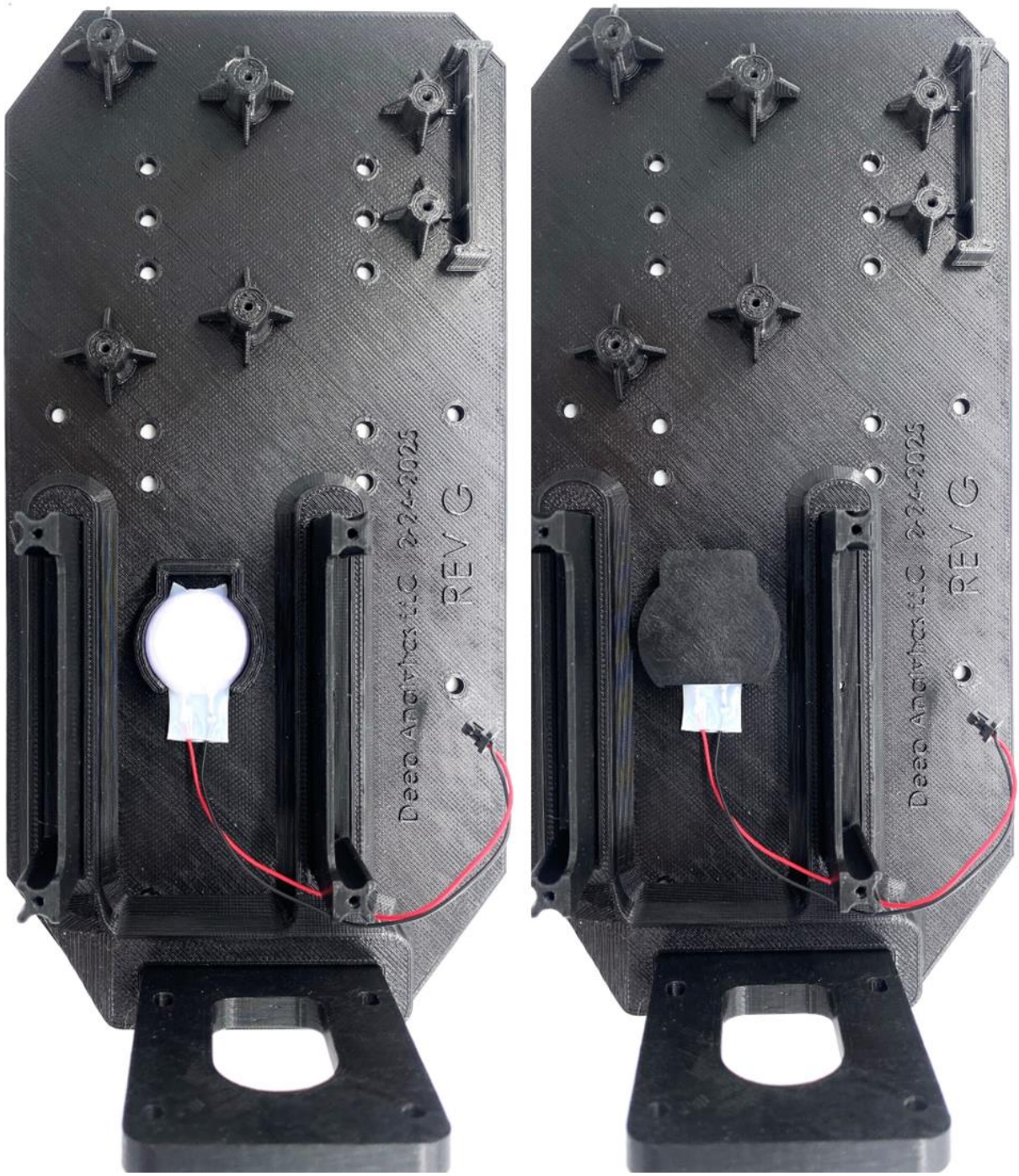

Figure 22 RTC battery placement.

3. Screw in the Pi and IMU (two screws for IMU four screws for Pi) and connect them using the jumper wires (Figure 15), also plug in the RTC battery as shown in Figure 17. (Figure 23).

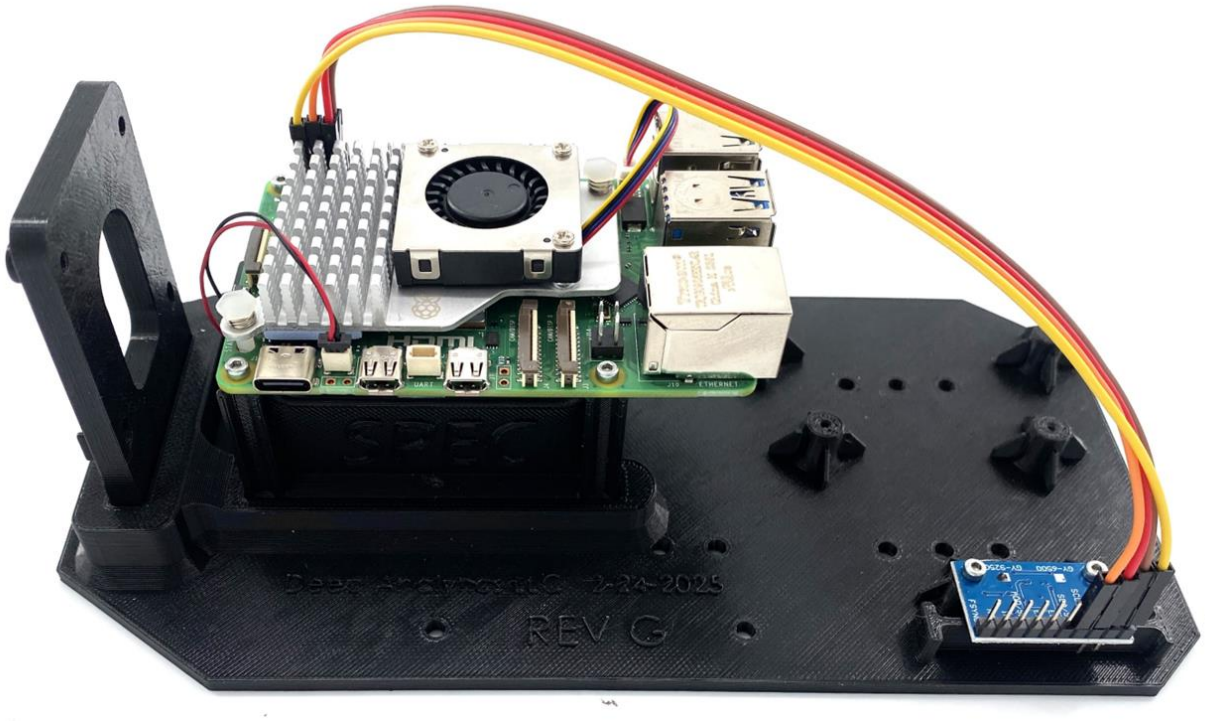

Figure 23. Connect Pi and IMU.

4. Screw in the camera, make sure it is right side up (white cord plug on bottom shown in red square) (Figure 24).

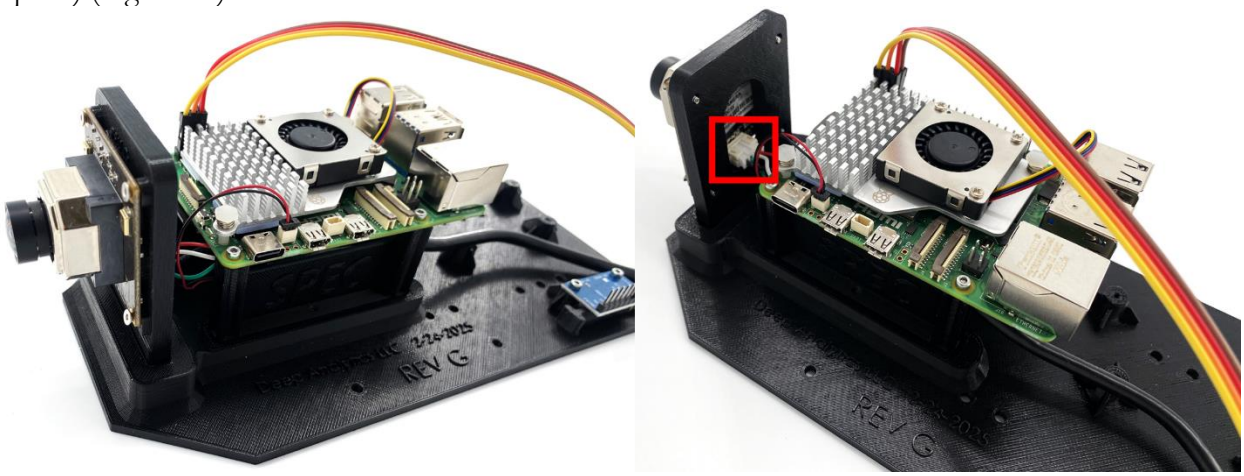

Figure 24. Connect camera.

5. Since the DC-DC converter is connected to the enclosure we will install that in the next section.

CAMERA ENCLOSURE AND CLIP-INS

The suggested camera enclosure comes with two predrilled holes for the ethernet and power passthrough connectors. To also include a USB pass through power connector, we need to drill another hole in the center. You will also need to connect the DC-DC converter with the power passthrough and USB-C plug-in for the Pi (Figure 25 and Figure 26).

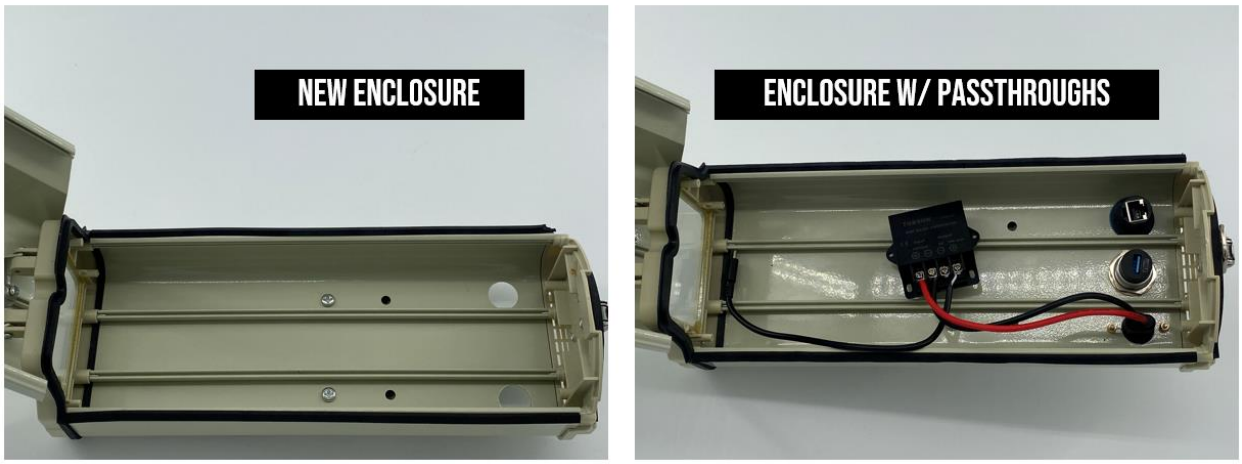

Figure 25. Before and after - creating passthroughs on camera enclosure.

Figure 26. DC-DC converter connections.

Once all the devices are secured to the sled and the passthrough connectors are in place the two pieces are ready to be put together. To make this an easily adjustable install we designed 3D-printed clip-ins for the sled that allow you to adjust without any extra tools (Figure 27 and Figure 28).

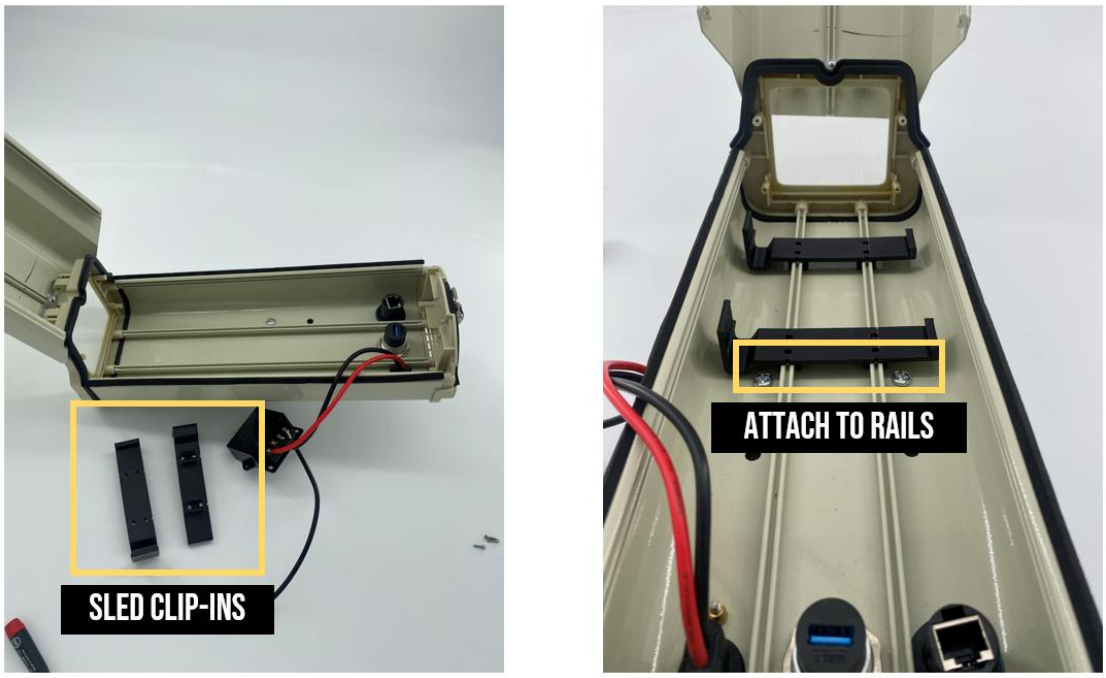

Figure 27. Attach sled clip-ins to camera enclosure.

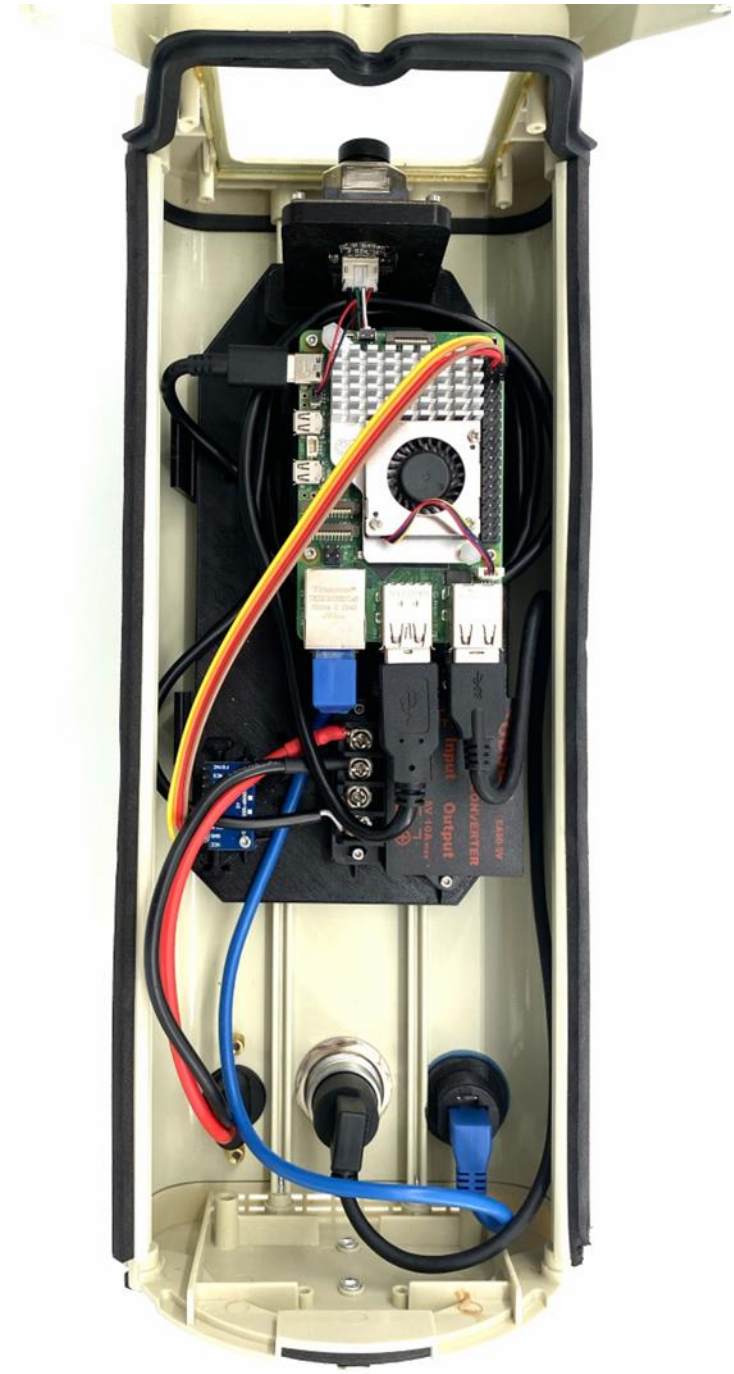

Figure 28. Finished assembled SPEC.

OPERATING SYSTEM INSTALLATION

1. Download and install the Raspberry Pi Imager on your computer and insert the microSD card into the Pi.

2. Select the operating system: with or without the desktop. The desktop version is not necessary and will use more power (though you can turn off booting to desktop) but is helpful with the initial setup (Figure 29).

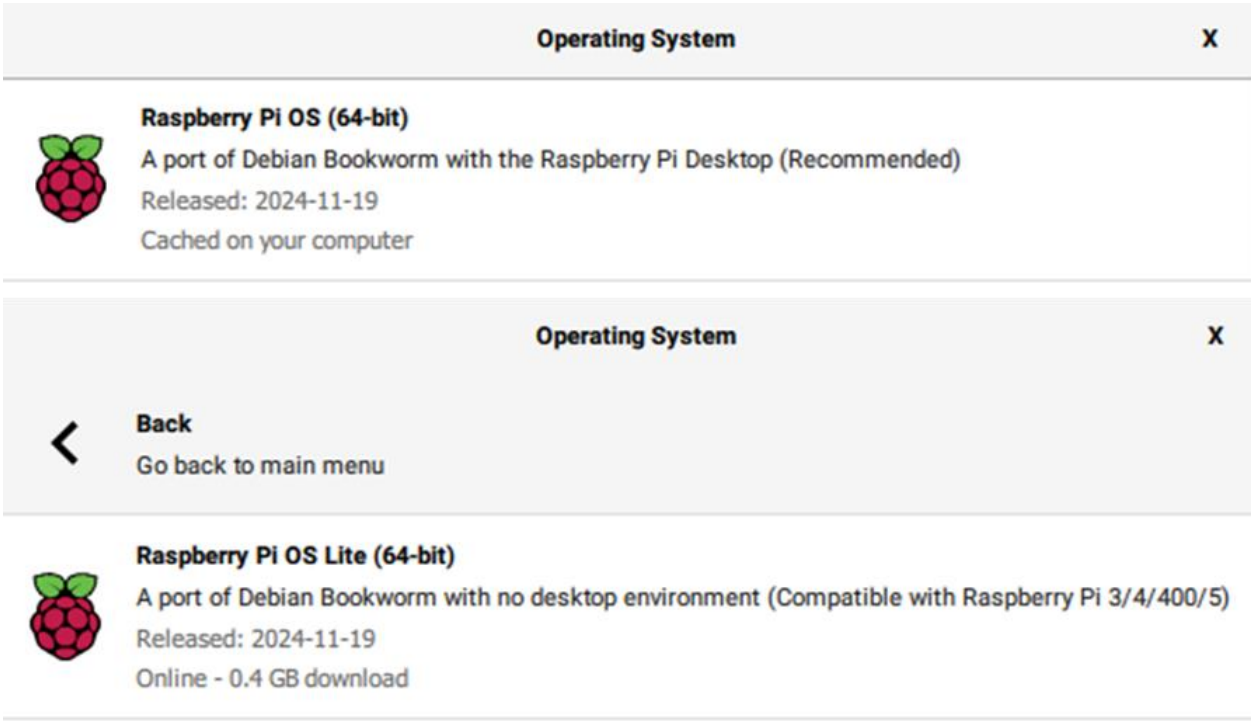

Figure 29. Desktop operating system (top) or headless operating system (bottom).

3. During the imaging setup, you will be asked to apply customization settings. Select Yes (or Edit settings if this is not the first time using the software). (Figure 30)

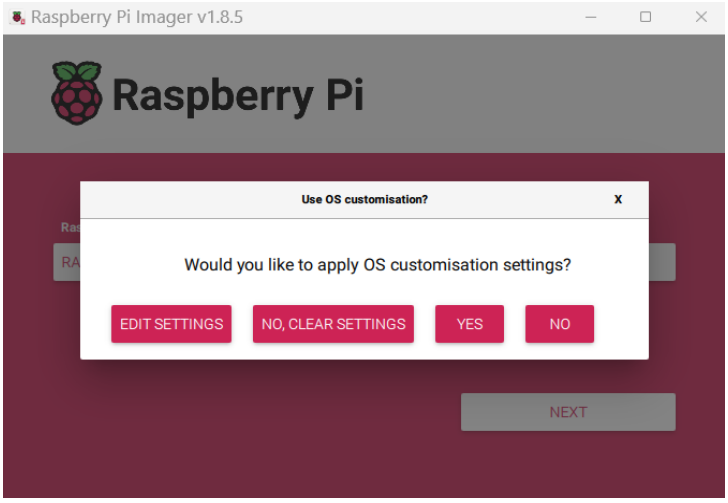

Figure 30. Select Yes for custom settings.

4. Fill in the hostname, username, and password fields in the General tab, and select Enable SSH with Use password authentication on the Services tab (Figure 31).

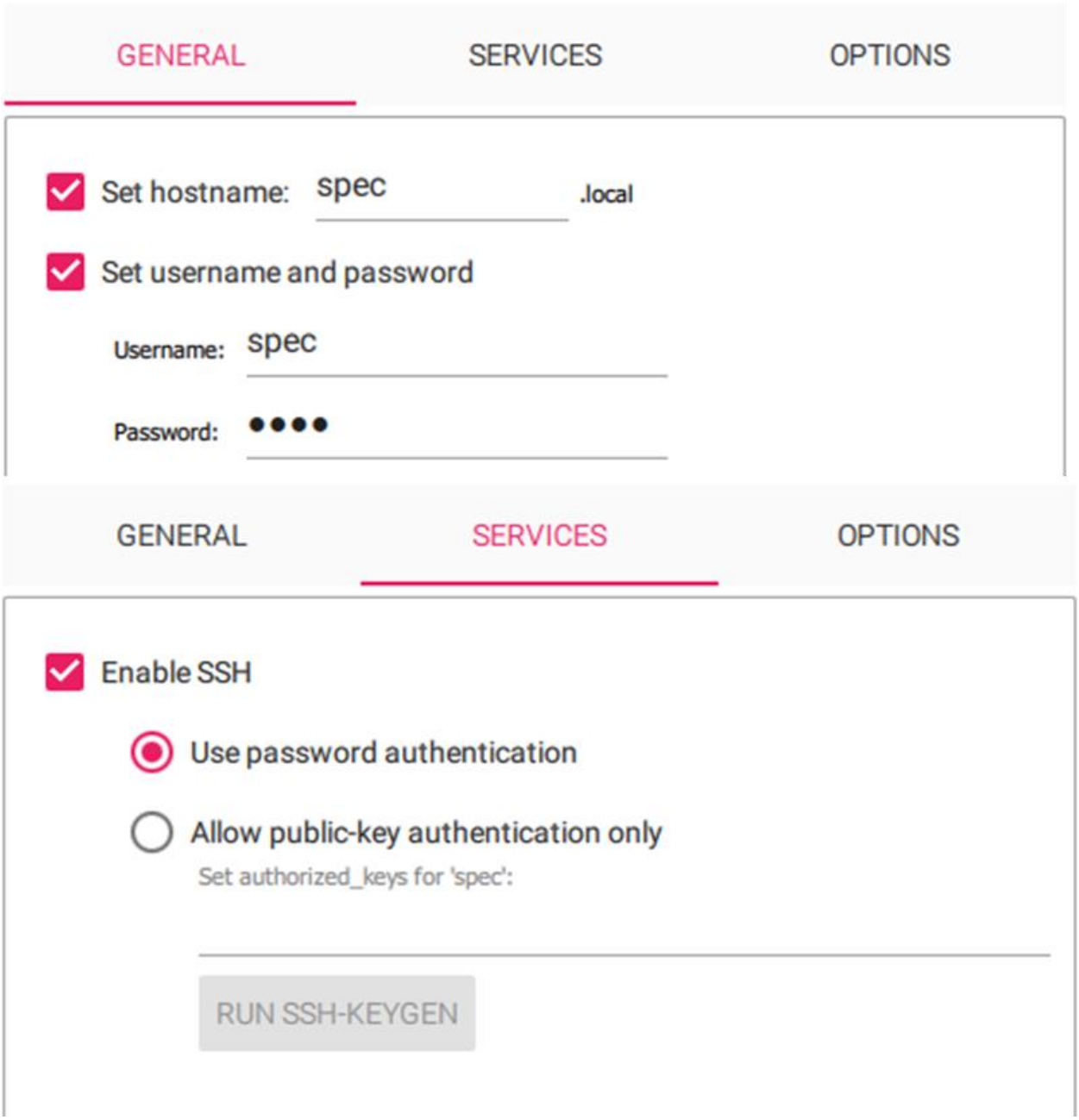

Figure 31. Choose custom settings.

5. When settings are complete, select Save and you will be prompted to create the image (Figure 32).

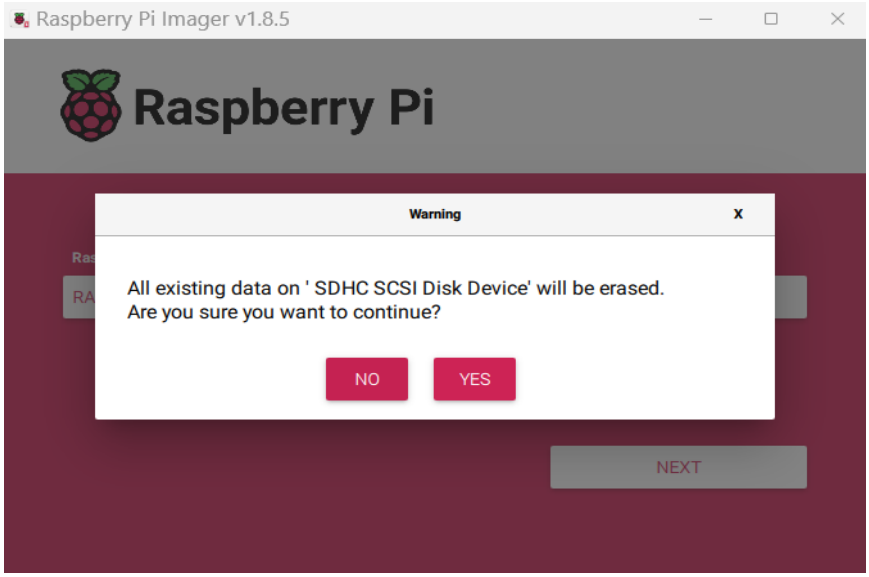

Figure 32. Select YES to create the image.

6. Power Up
    - This step assumes that you either have a monitor/keyboard/mouse plugged into the Pi, OR you have connected "headlessly" to the Pi via a remote login (SSH).
    - Once the image has been created, plug everything into the Pi EXCEPT the power.
    - Place the SD card into the Pi and insert the power cable from the official Raspberry Pi power supply.
    - Confirm Pi is broadcasting its SSID by inspecting available wireless networks on your device.

SPEC SOFTWARE INSTALLATION

1. Clone the SPEC repository.
    - **git clone https://gitlab.com/deepanalyticsllc/spec**

2. Run the setup script that will install the SPEC software by doing the following
    - `cd spec`
    - `./System/set_up_system.sh`
        - The script will ask for four items of user input:
            - Wi-Fi name
            - Wi-Fi password
            - Web application username
            - Web application password

3. To confirm the installation is successful, connect a device to its Wi-Fi. If you reach the SPEC login page, the system is working.

SPEC IMU CALIBRATION, CAMERA CALIBRATION AND CAMERA PARAMETERS

It is CRITICAL to calibrate the camera and the IMU before going out to your field site to set up the system. The camera calibration will enable you to obtain the focal length of your camera experimentally, which is especially useful if the camera manufacturer does not provide this information.

IMU CALIBRATION

1. To perform the IMU's accelerometer calibration, you will run the IMU/misc/calibrate_imu.py script and place the system in 6 different positions for 2 seconds each.

    a. If this is challenging (and for some of us it was!) you can modify the IMU code to change the number of seconds in between positions.

    *Navigate to /usr/local/lib/python3.11/dist-packages/imusensor/MPU9250/MPU9250.py. Scroll to line 292 and edit `time.sleep(2)' by changing the numeric value to be the number of seconds you want in between periods of data collection. Save the file.

    b. To make things easy, the 6 positions you can use are illustrated in Figure 33.

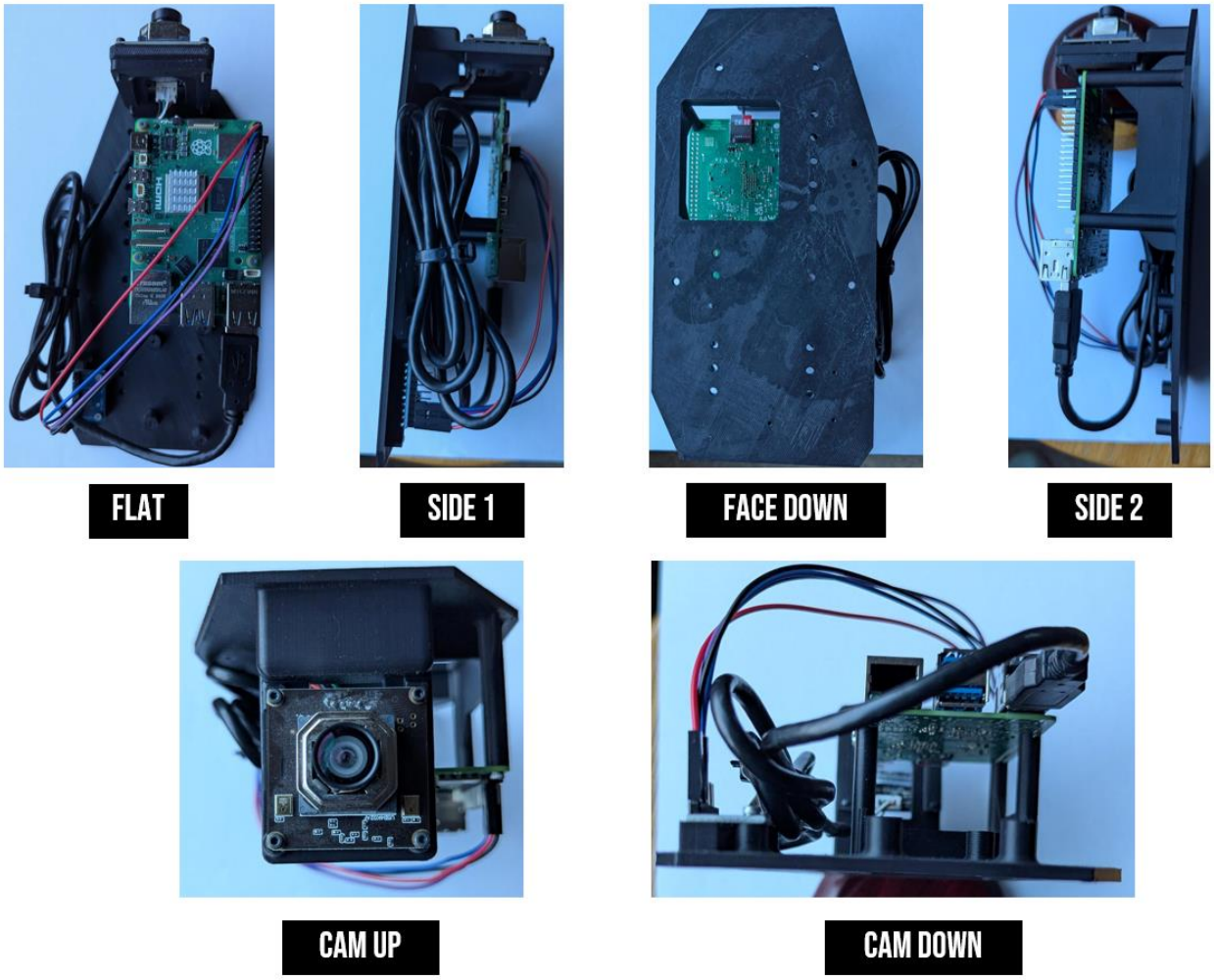

Figure 33. IMU calibration positions for system.

2. Run the calibration script (in the IMU/misc folder).

   ```
   sudo python calibrate_imu.py
   ```

   The script walks you through placing the IMU/system in 6 different positions. When this step is finished, it will display the message "Accelerate calib success". If you are not successful, an error message will appear from the IMU software asking you to try again. Wait until the script completes calibration of the magnetometer (not used in SPEC but done for completeness). When finished, you will see that the calibration data has been saved and loaded properly.

3. To test the results, run the IMU/misc/testimu.py script. This will output accelerations in x, y, z. If your system is in the flat position, as in Figure 33, and on a flat surface, you should expect outputs like this:

   Accel x: -0.04689449376828755 ; Accel y : 0.2007838316150461 ; Accel z : 9.803405120620532

   roll: 1.1735931389023007 ; pitch : 0.2740142540259086 ; yaw : -149.5553912913779

Acceleration in z (downward) should be about 9.8 m/s^2 and the pitch and roll angles should be about 0°. If this is not the case, try IMU calibration again.

GATHER YOUR CAMERA SPECS

- f = Focal Length.
    - f may be provided by the camera/camera sensor manufacturer. If not, we will determine f via the camera calibration matrix results and parameters set in Camera Parameters page of the web app.
- Physical Sensor Height and Width in mm
    - These parameters are either specified in the sensor documentation or calculated from the sensor diagonal and the sensor's aspect ratio.
- Image Resolution
    - The default is 1920x1080, even though our system's camera is capable of 3840x2160. The reduced resolution will decrease image size and improve processing times. We performed our testing at 1920x1080 and it is the maximum setting available.
- Sensor Pixel Size in mm
    - This parameter should be provided by the manufacturer.

PERFORM CAMERA CALIBRATION

A brief description of how camera calibration works is found here. Following is a description of how to calibrate the camera usings files we provide. All these steps should be run on the Raspberry Pi in the SPEC repository directory.

1. Download and print a chessboard for the process.
2. Navigate to Image_Processing/Camera_calibration
3. Run `sudo python take_photos.py`
    - Take about 10-15 photos of the chessboard while the chessboard is in various positions.
        - We recommend attaching the chessboard to a piece of thick cardboard and moving it, rather than moving the camera.
    - You can see that these photos are saved in Image_Processing/camera_calibration/images_for_calibration.
4. Once the photos are taken, open the calibration.py script
    - If you are using the calibration chessboard linked above, your settings should remain as
        - chessboardSize = (9,6)

        - frameSize = (1920,1080)
        - size_of_chessboard_squares_mm = 25
    - If you used a different chessboard:
        - Count the inner corners - the intersections where the black and white squares meet. Count along the rows and along the columns to determine the size.

5. Run the calibration.py script
    - The output of this script is your camera calibration matrix like this:

[[1.02444441E+03 0.00000000E+00 9.91167483E+02]

[0.00000000E+00 1.02434832E+03 6.04988705E+02]

[0.00000000E+00 0.00000000E+00 1.00000000E+00]]

Some notes on the camera calibration:

- The number in the [first row, first column], or the [0,0] element, of your camera calibration is the focal length in the x-direction in pixels. The number in the [second row, second column], or the [1,1] element, is the focal length in the y-direction in pixels. These should be close, so choose one.
- On the camera parameters page, we calculate your focal length in mm by multiplying your reduced resolution camera pixel size (also calculated for you on the camera parameters page). For example, element [0,0] is 1.02444e+03. Our camera's reduced resolution pixel size is 0.0029 mm so focal length = 1024.44*0.0029 = 2.97 mm.

CONFIGURE CAMERA PARAMETERS

- Connect to the SPEC app via wifi.
- Log in with the user credentials you selected.

- Proceed to Main Menu, Setup and Calibrations, and Camera Parameters as seen in Figure 34

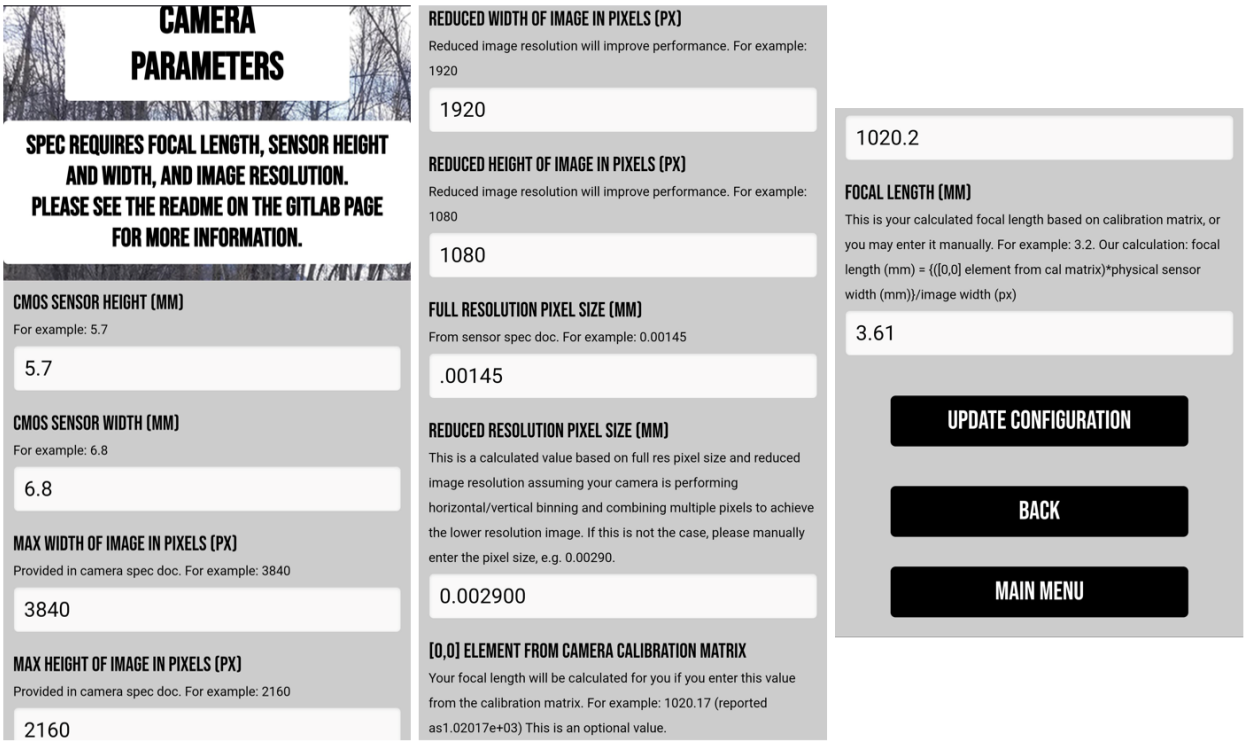

Figure 34. Camera Parameters page as seen from left to right.

- Fill in your sensor height, width and max resolution parameters. The reduced width and height should be set at 1920x1080 for best system performance, and is the maximum setting available.
- Fill in your full resolution pixel size in mm as provided in manufacturer specs.
- Enter the [0,0] element from the camera calibration matrix. - The SPEC app has now calculated your reduced resolution pixel size as well as your focal length.
- Hit UPDATE CONFIGURATION to save these settings to config.json, and you're done with setup.

OPTIONAL: CUSTOMIZING BACKGROUND IMAGE

The SPEC web application has a default background image that you may wish to change. Rename your image to app_bg.jpg and place it in the /app/static/images folder. You will need to restart the app service manually via

```
sudo systemctl restart start_app.service
```

or simply reboot or power cycle the Pi.

WEB APPLICATION

CONNECTING TO THE SPEC

Once the SPEC is fully set up and the camera and IMU calibrations are complete, the system is ready to be deployed in the field. Follow these steps to get started:

1. Reboot the SPEC:
    - After completing setup and calibration, restart the SPEC. If you are using a desktop view, you can disconnect the HDMI, mouse, and keyboard at this point.
    - To reboot, simply unplug the SPEC from its power source and reconnect it. Ensure the SPEC is connected to a power source only during operation.
2. Connect to the SPEC Wi-Fi:
    - Once the SPEC powers back on, connect your device to the Wi-Fi access point you previously configured with an SSID and password.
3. Access the Web App:
    - Laptop: Your default browser should automatically open and display the login page.
    - Android Device: After connecting to the Wi-Fi, open a browser, and you should be redirected to the login page.
    - iPhone/iPad:
        - The iOS captive portal browser will pop up upon connecting to the Wi-Fi, displaying the login page.
        - **Note**: The iOS captive portal can slow down the app and may not redirect you to a standard browser. To bypass this:
            - Tap "Cancel" and select "Use Without Internet."
            - Open a browser and enter 192.168.0.1 in the address bar to access the login page.

Once directed you will be able to type in the username and password you selected previously for the webpage. Figure 35 shows the SPEC login page as viewed on different devices.

Figure 35. Login page as viewed on a tablet or laptop (left) and phone (right).

CALIBRATION SPLASH

When you log in, you will be directed to a page called the calibration splash page. On this page, a message will be displayed with either **"DEVICE NEEDS TO BE CALIBRATED"** or **"LAST CALIBRATION: [date]"**, depending on whether the device has been previously calibrated to run PIV (Figure 36).

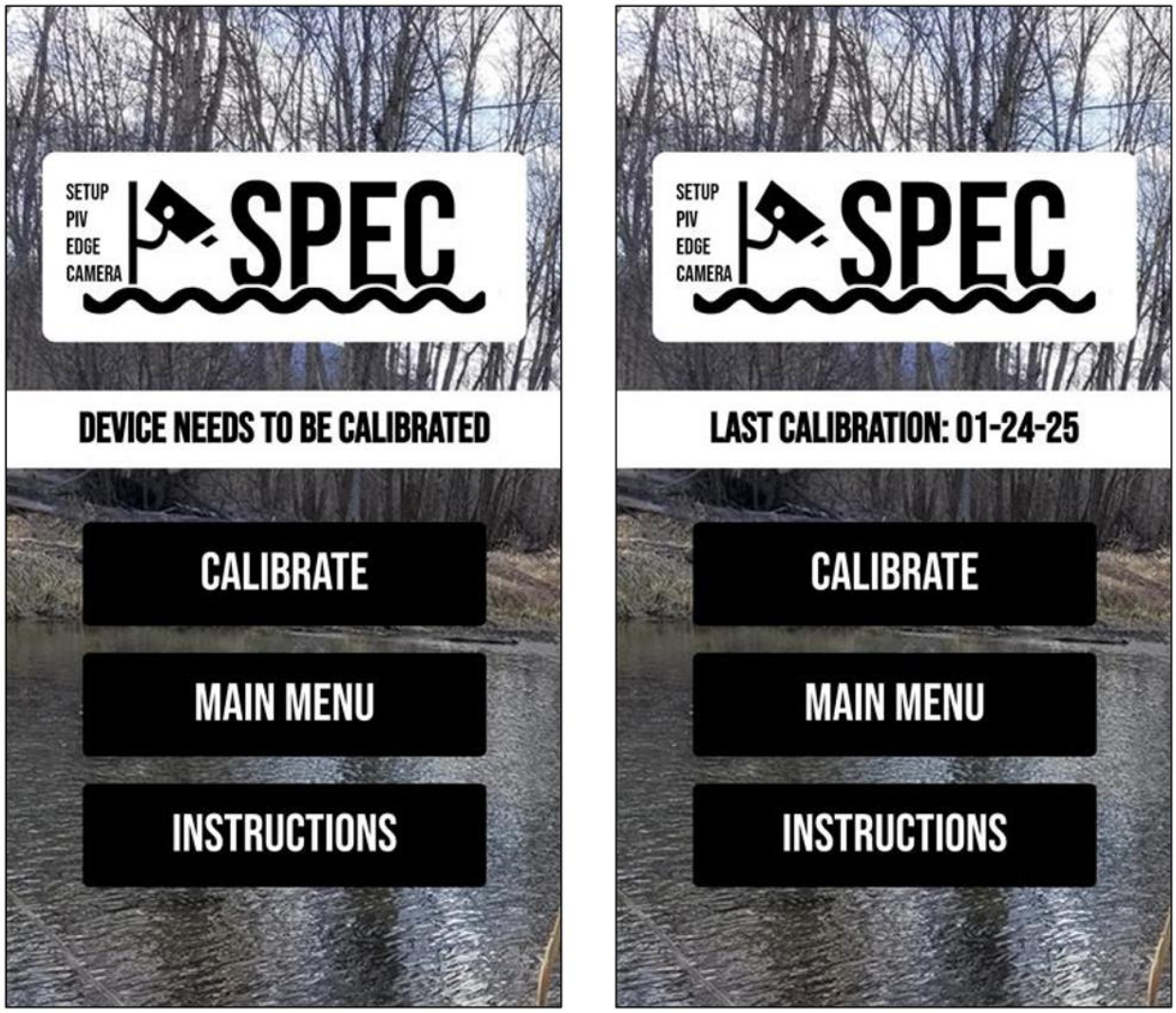

Figure 36. Calibrate splash page display showing a non-calibrated system (left) and a calibrated system (right).

CALIBRATE

The **CALIBRATE** button will guide you through three calibration functions required for the PIV calculations. Each calibration function on the app will present instructions on a page about how to use it. Upon completion of a calibration function, the app prompts you to complete the next one.

There are three main calibration pages:

- **Trapezoid Calibration:** Define the area of interest for PIV.
- **PIV Params Configuration:** Set the parameters for your PIV calculations.
- **Mask Calibration:** Define the area to exclude from PIV analysis.

The same instructions displayed here will also be on the pages before the calibration functions, as well as on the Gitlab for review if needed. To see further information on this process go to the Setup and Calibrations section.

It is recommended that if you have never calibrated the system or want to recalibrate the system, press the **CALIBRATE** button and walk through the steps in order.

MAIN MENU

The first page you will come to is the **MAIN MENU** splash page. On this page, you will have three options to navigate to other sections and a logout button. Additional instructions on using the main menu are provided on that page. Below are the available options and their corresponding functions:

UTILITIES

The utilities page provides services for you to use that are not related to PIV or calibration. Below are the different options available in this section:

SITE INFO

Site Info allows you to manage details about the PIV field site, including:

- **Site Name**: Assign a name to the field site for identification.
- **Site ID**: Assign an ID to uniquely identify the field site.
- **Operator**: Specify the operator's name for tracking purposes.
- **Site Comments**: Add any relevant notes or comments about the field site that can be stored for future reference.

LIVE VIDEO

The Live Video option allows you to view the camera's live feed without any processing. This is simply the raw, undistorted video as seen by the camera.

DATA MANAGEMENT

The Save Data page allows you to save all data to a USB drive, delete data, whether from tests or main PIV runs, and unmount the USB drive. The process works as follows:

- The system will detect if a USB drive is connected.
- If found, it will copy all files from the **save_data** folder, which includes all PIV runs and test files.
- A directory called **Offsite_Processing** will also be saved. This directory contains a script that allows you to reprocess the data offsite if needed.
- You then have the option to delete specific directories.
- ***Important note: to avoid writing a "dirty" bit to the USB drive that will prevent its further use, press the Unmount USB drive button prior to removing the USB drive.***

VIEW LOGS

The View Logs page provides four log options for troubleshooting and diagnostics:

- **App Logs**: Logs related to the overall application performance.
- **PIV Logs**: Logs specific to PIV operations and calculations.
- **Gstreamer Logs**: Logs for the Gstreamer process that streams the camera feed.
- **Loopback Logs**: Logs for the loopback process used to create virtual cameras.
- **Diskspace Logs**: Logs for the disk space manager, will tell what files are deleted if disk becomes full.

CHECK DISK SPACE

This button will display the remaining disk space and the date you will run out of storage based on the size of the last PIV run. If there is no PIV run or the run is still happening, the time until run out will not display. Once the disk space limit is reached (default is 4 GB left), the system will begin deleting the oldest datasets one at a time until there is 4 GB free.

REBOOT SYSTEM

If the system is not functioning correctly (e.g., camera is not on, or errors are occurring), you can use the **REBOOT SYSTEM** option to restart the system. Rebooting can often resolve issues by giving the system a fresh start. All services are programmed to restart automatically after a reboot. If the system starts up incorrectly, a reboot may be required to get everything working properly.

SETUP AND CALIBRATIONS

This page on the main splash will provide the three processes that you need to calibrate your system for PIV calculations.

TRAPEZOID CALIBRATION

This page is used to select the region of the image that you would like to run a PIV calculation. This is not a mask but the area that will be transformed to give a bird's-eye or nadir view of the bank-mounted image for the PIV algorithm to run correctly.

Roll

On this page, you will see a bubble level at the top displaying the roll angle of the SPEC based on the IMU. The roll angle is the angle you tilt the device side to side – see Figure 37 for a visualization of the roll concept. You want the roll angle to be 0°, in other words you want the camera to be level, as other values may introduce errors in PIV calculations. The app provides the bubble level to zero the roll value – see Figure 38.

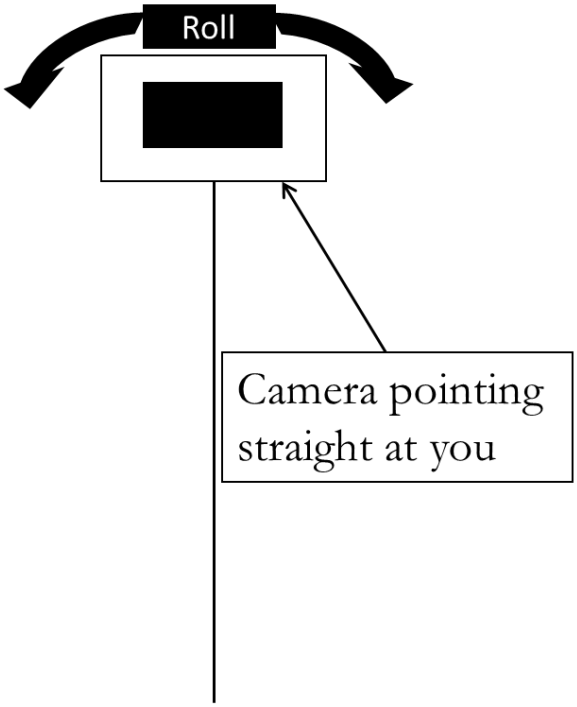

Figure 37. Graphic describing camera roll.

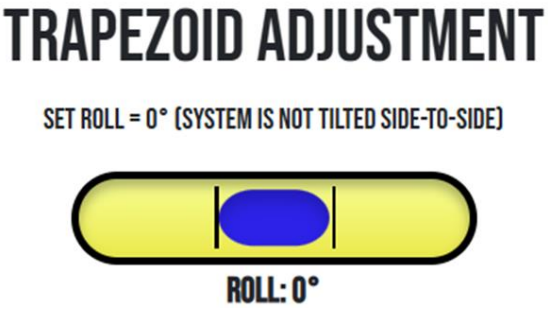

Figure 38. Bubble level display on Trapezoid page, should be 0° for PIV.

Trapezoid View

The next item displayed is a live video with a trapezoid overlaid on it. The area inside the trapezoid will be used for PIV calculations. As you incline the camera up and down, the trapezoid's left and right sides adjust based on the IMU's position relative to nadir (Figure 39). At nadir (pitch=0° or camera pointing straight down), the trapezoid appears rectangular, but as the incline (pitch) approaches 90°, the sides converge as depicted in.

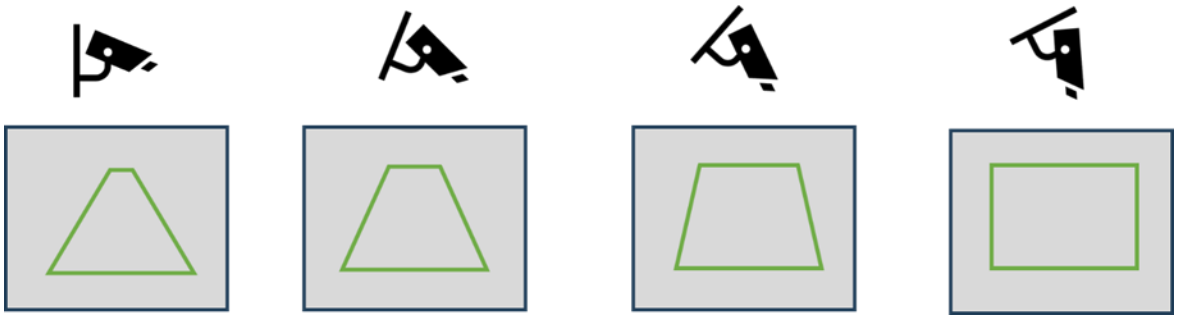

Figure 39. Graphic demonstrating how the sides of the trapezoid changes with the angle of the camera.

To move the trapezoid, use the sliders below the live view – see Figure 40. Adjusting one point recalculates the trapezoid and resets all points. Aim to include as much of the river or PIV region of interest as possible within the trapezoid. Banks or other objects can be masked out later.

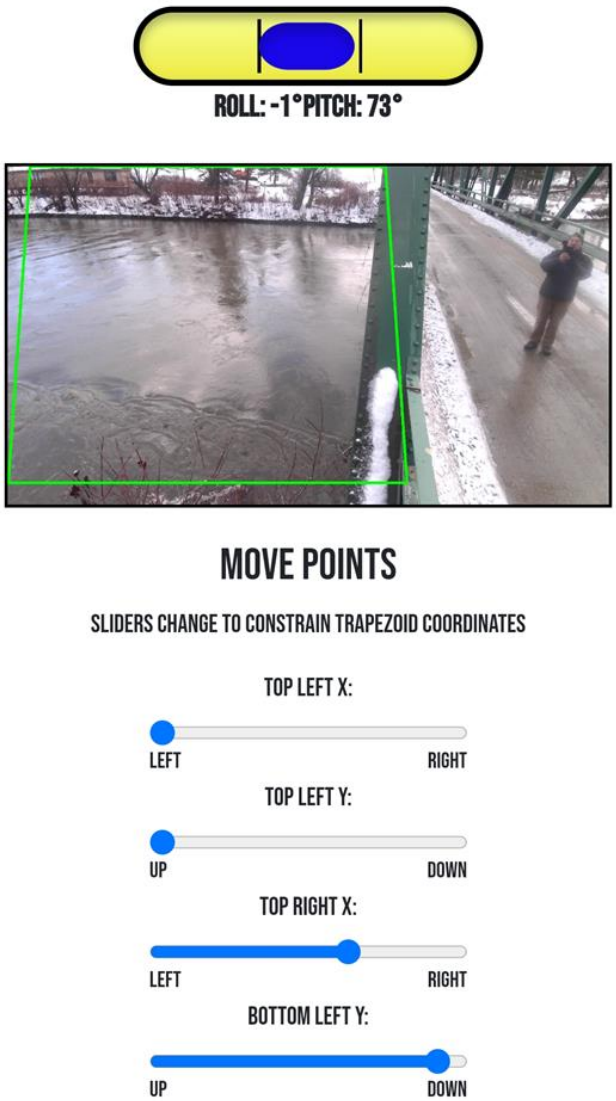

Figure 40. Trapezoid slider display.

Once satisfied with the trapezoid, click SHOW TRANSFORMED IMAGE to view the transformed image. Ensure the desired area for PIV is visible.

Repeat the following steps as needed:

- Adjust camera tilt (up-down angle)
- Set roll (side-to-side angle) to 0°
- Move trapezoid points
- Show transformed image

When finished, click SAVE POINTS to save the trapezoid configuration.

PIV PARAMETERS

The PIV PARAMETERS page is used to input all the parameters needed for your specific PIV situation. These parameters influence the PIV outputs and should be adjusted carefully [4] [5]. The first five parameters in Table 1 are the most important and require special attention.

For the most part, parameters that come after "Pixel Size Ground Sampling Distance" in the table can be left at the default values. However, the **Desired Output Vector Spacing** and **Capture Interval** are important parameters and if after running a test, you are not satisfied with the results, think about adjusting these parameters to assist in collecting good data. There are times that the output spacing is too large or small for a particle to be properly tracked and thus no output is produced. Similarly, the capture interval must allow enough time for features to move a measurable distance (i.e., number of pixels between frames). To get a good understanding of this look into the TIPS feature of TRiVIA [4] [1].

Adjust the parameters as needed. Once satisfied, click Update Configuration to save your settings.

Table 1. PIV parameters and their descriptions.

PARAMETER	DESCRIPTION
DESIRED OUTPUT VECTOR SPACING	This is the spacing the user wants between vectors in the PIV outputs, this variable affects the output a lot and is key for setting up the system. E.g. input of 1 refers to 1 meter output spacing.
CAPTURE INTERVAL	This is the time between each image capture, e.g. for 10 frames a second user should put 0.1s.
DURATION OF IMAGE SEQUENCE	This is the time you want to capture images for each PIV run. E.g. input of 10 is 10 seconds of image capture.
TIME BETWEEN PIV RUNS	This is the time the system waits between PIV runs. E.g. 15 tells the system to run every 15 minutes.
SENSOR HEIGHT ABOVE WATER	This is the height in meters above the water that the camera stands. Measure from the camera straight down to the water. E.g. input of 1 is 1 meter above the river.
PIXEL SIZE GROUND SAMPLING DISTANCE	This is calculated in the backend based on the sensor height. It is the ground sampling distance of the camera and cannot be modified by the user. It is calculated in meters.
MIN VELOCITY	This is the minimum velocity that is used to post-process the initial PIV output, anything less will be filtered out. Default 0.01 m/s
MAX VELOCITY	This is the maximum velocity used to post-process the initial PIV output, anything more will be filtered out. Default 5 m/s.
STANDARD DEVIATION THRESHOLD	This is the maximum standard deviation used to filter outliers from the initial PIV output. Default is 4
LOCAL MEDIAN THRESHOLD	This is the local median value used to filter outliers from the initial PIV output. Default is 1.5 m/s.
VECTOR INFILLING	Yes: Fill missing vectors in No: Do not fill missing vectors in
VECTOR SMOOTHING	Yes: Smooth vector field No: Do not smooth vector field
MASK	Yes: If plan to use a mask No: If no mask Disclaimer: Yes should almost always be used.

MASKING OPTIONS

The next step in the calibration process is masking. Masking allows us to exclude parts of the transformed image that are not the water, so that the PIV does not run on stationary areas. This can speed up the calculations depending on how much is masked out. Note that you want to mask out everything except the river and include as much of the river as possible to maximize the area with PIV output.

Masking Methods

The first page you will come to when you hit next is a page that asks if you want to digitize or generate a mask – see Figure 41.

Figure 41. Masking options splash page.

DIGITIZE MASK

This is a user-selected custom mask. You will be shown a transformed image and just need to click around the area you want to include in your mask. Start from the top right of the polygon you wish to create and proceed counterclockwise. This will show green dots where you click, and once finished, click **SUBMIT POINTS**. This will perform the masking process in the backend and show you three images: the original transformed image, the binary mask, and the final masked image – see Figure 42 for an example. If you like the mask, hit **SAVE MASK PATH**. The mask will be saved and then used in all following PIV calculations until a new mask is saved. Figure 43 shows an example of the masking process for a river example.

Figure 42. Digitize mask example.

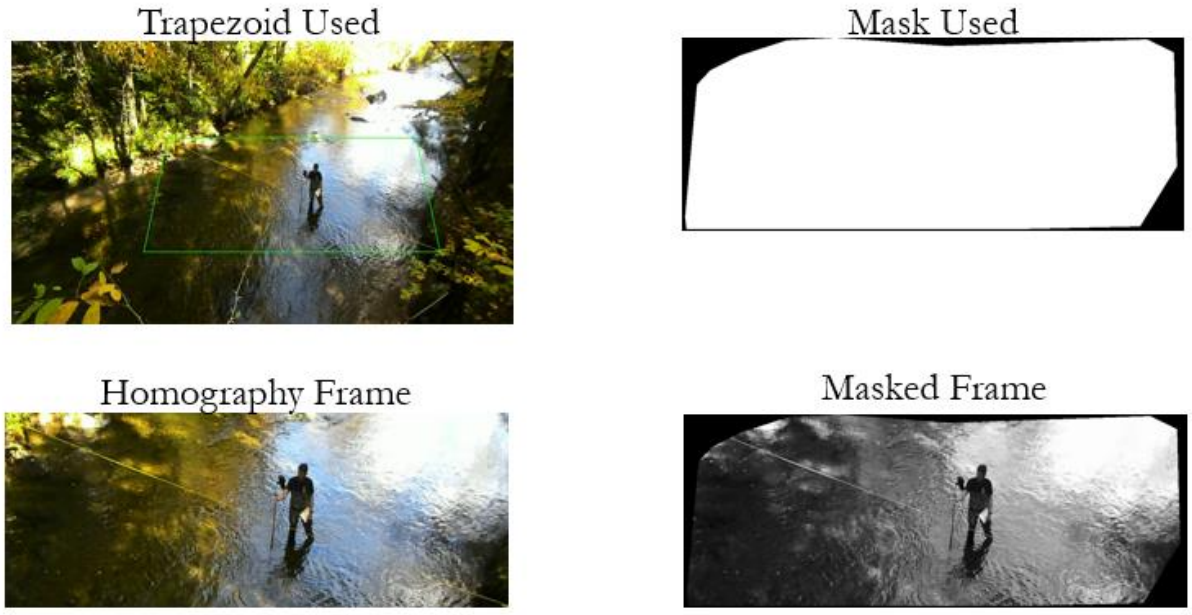

Figure 43. Masking out riverbanks at a field site.

GENERATE MASK

This is an automatic masking process. It creates a mask based on the largest contoured area. If your river is distinctly a different contour than its surroundings, then this is a good option. Once done finding the largest contoured area, it will display three images: the original transformed image, the binary mask, and the final mask. If you like the mask, hit **SAVE MASK PATH.** The mask will be saved and used in all subsequent PIV calculations until a new mask is saved.

CAMERA PARAMETERS

This page is only intended to be set up once, as it is specific to the camera hardware and nothing about the site itself. This page and the camera calibration (as detailed in the set up documentation) must be completed prior to going out to a field site.

This is the place where you put your camera specs: focal length, camera sensor height and width, max array availability, reduced resolution, original sensor pixel size. All these should be specs that you can find online or with your camera or camera sensor datasheet.

PIV FUNCTIONS

This page is where you can view PIV results, run tests, or finally run the main continuous PIV function.

RUN TEST

This button performs a single PIV calculation based on the selected parameters. This is useful for previewing how PIV outputs will look for the current system configuration.

- Each test generates a new folder of data, enabling comparison between test results.
- During the test, you will be directed to a waiting screen that provides live updates on the calculation process – see Figure 44.
- After completion, you will be redirected to the results page to view the most recent test files.

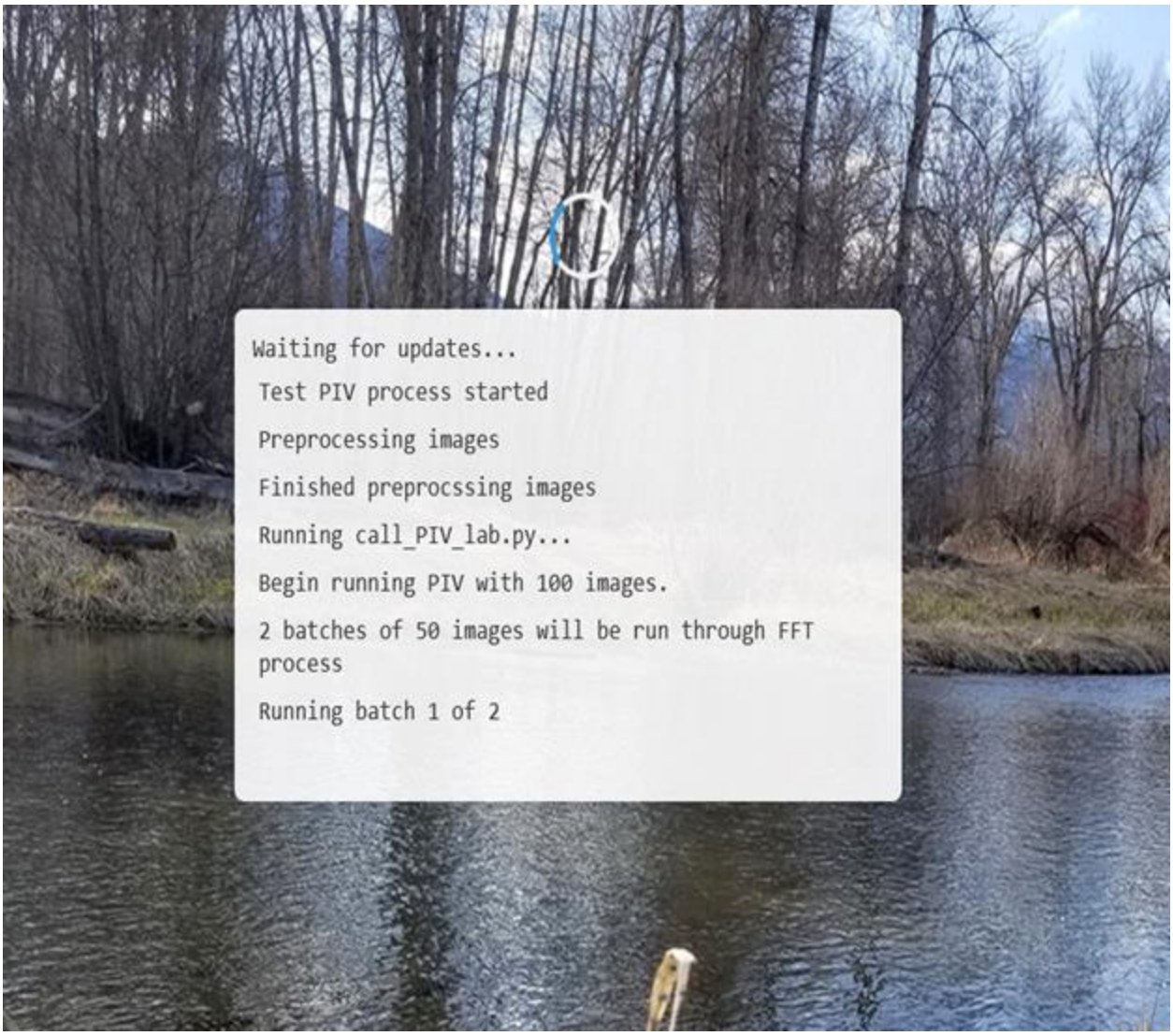

Figure 44. Example of the waiting screen when running a PIV test.

RUN PIV

This button is used for continuous PIV calculations over an extended period. Ensure all parameters are configured correctly before starting.

- A verification page will appear, asking you to confirm and manage test files to optimize disk space.
- You can choose to save test files to a USB or delete them to free up space.
- The system will estimate how long it can run before disk space is full, based on the size of the last test file.
- Once PIV starts, a yellow banner will appear across all pages indicating that PIV is running.

After starting, you will be redirected to a page to view results or return to the main menu.

RESULTS

The Results page allows you to view data stored in the **save_data** folder. By default, it displays the most recent PIV calculation.

- Data are organized by configuration date and run date – see Figure 45.
- Test files include "test" in their filenames and contain a single run.
- Main PIV configurations may include multiple runs with selectable dates.

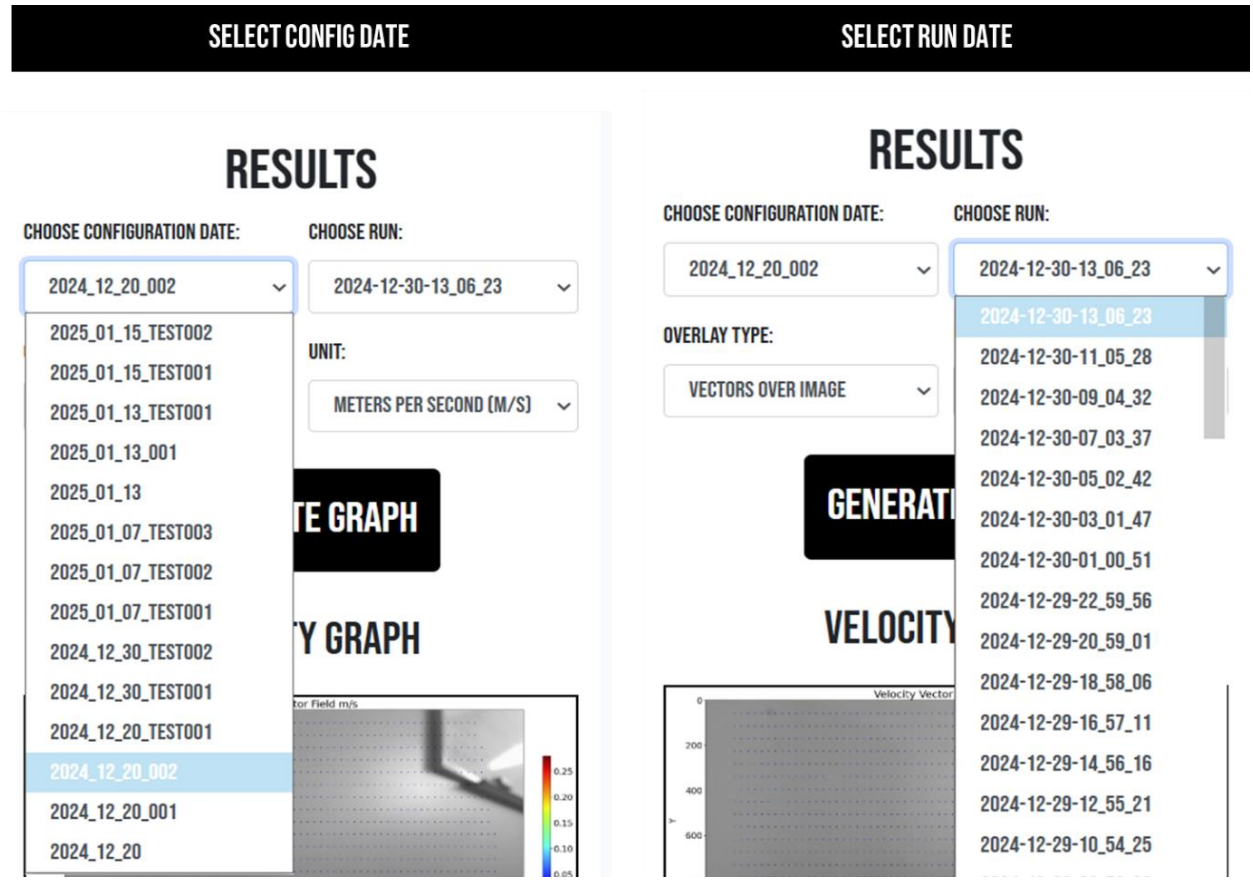

Figure 45. An example of how to select the configuration and PIV run to view its outputs.

You can select the output type and display options:

- **Overlay Type**: Choose between vectors displayed over the river image or over the magnitude color map of the vectors.
- **Unit Type**: Select between meters per second or feet per second.

Once you configure these options, click **GENERATE GRAPH** to display the results. This process may take a few seconds. See Figure 46 for example outputs.

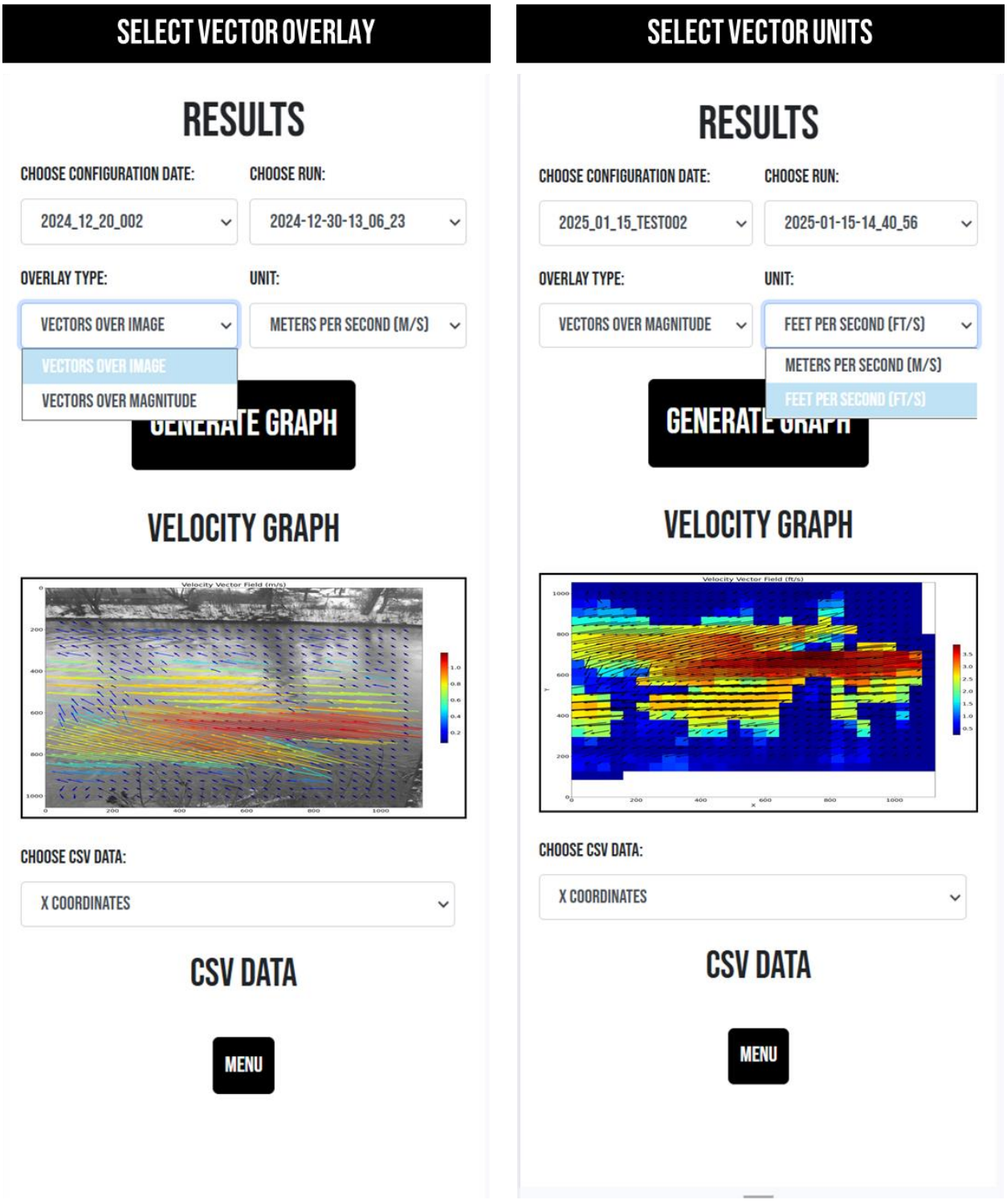

Figure 46. Example of choosing the vector overlay and units you would like displayed.

To view the actual numeric values of the output PIV vectors, select the desired components, and the CSV file will be displayed below the graph – see Figure 47 for an example.

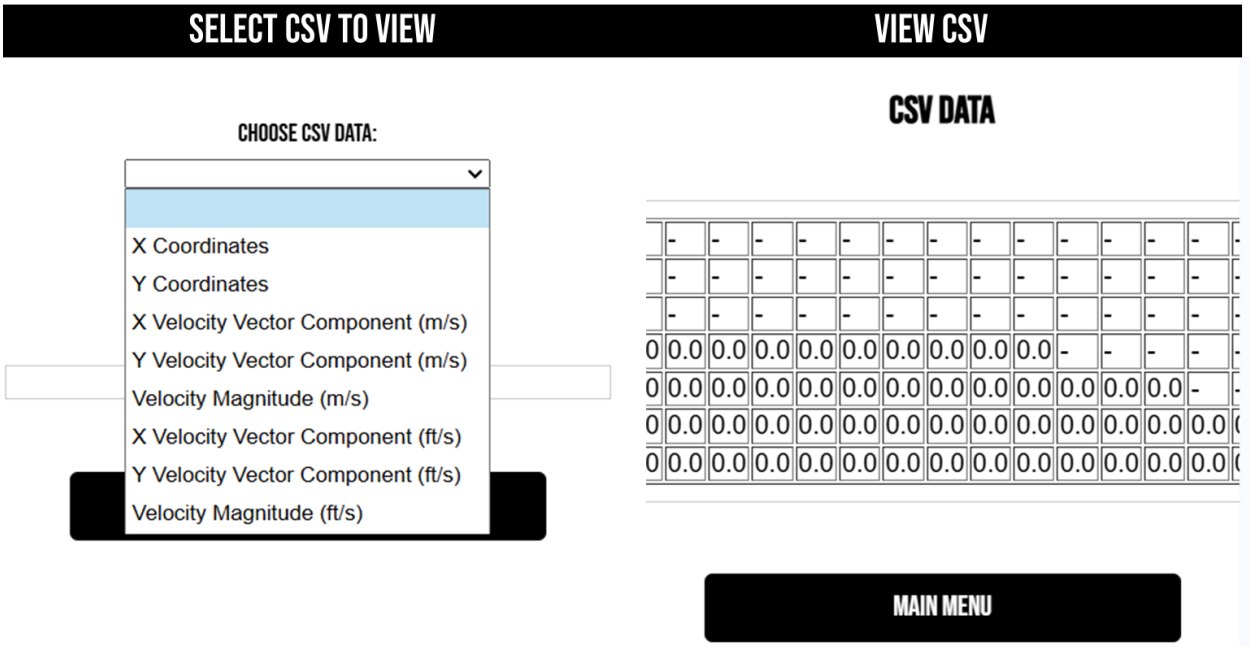

Figure 47. Example showing display of PIV data in CSV format.

OFFSITE PROCESSING

When you choose to save data from the SPEC onto a USB using the web app's save functionality, a directory named Offsite_Processing is included. This directory contains a script called processVideo.py, which enables users to process videos from a PIV run offsite. This functionality provides flexibility to reevaluate data if needed.

The script does not perform the full PIV calculation but instead focuses on transforming the images. It can undistort the images and apply homography transformations based on user-defined trapezoid points. The output can then be used as input to other image velocimetry algorithms. Additionally, users can select the frame extraction rate from the video, with a maximum frame rate of 30 Hz.

To use the script, simply place the video and configuration file into the Offsite_Processing directory. The script will then extract and transform the frames accordingly. If you wish to adjust the frame rate, you can do so by modifying the frameInterval variable in the configuration file. The value of frameInterval represents the time (in seconds) between each frame. For example, to extract frames at 10 frames per second (10 Hz), set the value to 1/10. Figure 48 depicts the post-processing workflow.

Figure 48. Offsite processing directory process block diagram.

DISCLAIMERS

PIV ACCURACY REQUIREMENTS

For PIV to produce accurate velocity fields, specific conditions must be met:

1. **Distinct Features:**
    - Distinct features or textures that move with the flow of the river need to be present for the algorithm to pick them up.
2. **Minimal Wind and Waves**:
    - High wind or significant waves disrupt the water surface, leading to inaccuracies in the PIV calculation.
    - The algorithm relies on tracking particles consistently across consecutive frames, and surface disturbances can result in erroneous outputs.
3. **Water Clarity**:
    - If the water is too clear, the algorithm may mistakenly track objects or features on the riverbed, skewing the velocity calculations.

Maintaining optimal environmental conditions ensures the PIV algorithm can effectively track surface particles and provide reliable results.

HOMOGRAPHY AND ITS IMPACT

Homography transforms the camera's perspective by stretching the pixels at the top of the image to align with the dimensions of the bottom of the trapezoid. However, this transformation has implications:

1. **Pixel Stretching and Resampling**:

- As the trapezoid's sides angle inward, the top pixels become increasingly stretched and resampled.
- This distortion affects the pixel size, particularly near the top of the image.

2. **Impact on Velocity Accuracy**:
   - Pixels closer to the top of the image might deviate further from the actual velocity due to the stretching effect.
   - Beyond a certain angle threshold, the pixel distortion becomes significant enough to compromise the accuracy of the velocity outputs.

This threshold is currently under evaluation to determine the optimal angle range for accurate results.

NIGHTTIME PIV

Currently, there is no system in place to illuminate the river during the night hours. This is also under evaluation.

REFERENCES

[1] C.J Legleiter, "TRiVIA - Toolbox for River Velocimetry using Images from Aircraft," U.S. Geological software release, September 2024. [Online]. Available: https://doi.org/10.5066/P9AD3VT3..

[2] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, Cambrige: Cambridge Univeristy Press, 2024.

[3] Keystone. [Online]. Available: https://us.xgimi.com/blogs/projectors-101/why-the-screen-shadowy-after-keystone-correction.

[4] C. J. Legleiter and P. J. Kinzel, "An enhanced and expanded Toolbox for River Velocimetry using Images from Aircraft (TRiVIA)," *River Research and Applications,* vol. 40, no. 8, pp. 1602-1612, 2024.

[5] C. J. Legleiter and P. J. Kinzel, "The Toolbox for River Velocimetry using Images from Aircraft (TRiVIA)," *River Research and Applications,* vol. 39, no. 8, pp. 1457-1468, 2023.