Technical Memorandum

Date:	December 9, 2022
To:	Lauren Mork - Upper Deschutes Watershed Council
From:	Joe Rudolph and Luke Russell - Wolf Water Resources
Project:	Whychus Creek Monitoring - 2022
Subject:	Whychus Creek UAS Survey

This technical memo and the work it summarizes were funded by a grant from Portland General Electric's Habitat Support Program.

Introduction

Wolf Water Resources (W2r) conducted an unmanned aerial system (UAS) survey on July 18th through 20th, 2022 along Whychus Creek Stage 0 restoration study areas near Sisters, Oregon (44.323361° N, -121.507105° W). The purpose of this assessment was to document site conditions and produce useful datasets for Stage 0 monitoring applications. The aerial survey effort was done in conjunction with a robust monitoring effort led by Upper Deschutes Watershed Council (UDWC) and supports the velocity mapping work being completed by Brandon Overstreet, Ph.D. (United States Geological Survey; USGS).

The project study area includes two contiguous reaches along Whychus Creek: Phase 1 of restoration, at Whychus Canyon Preserve (Phase 1; Reach 4), and Phase 2a of restoration, at Rimrock (Phase 2a), and the lateral expanse of their respective active floodplains. For both reaches, visible (red green blue; RGB) and multispectral imagery were collected. Additional video collection was completed at Phase 2a to support a velocity mapping analysis. This memo provides an outline of the survey and photogrammetry processing methodology, as well as metadata for the UAS photogrammetry products. A brief description of suitable data usage and limitations are also discussed.

Methods

The reach boundaries and floodplain extents were predetermined prior to acquiring imagery and were generally developed based on flight areas from previous acquisitions. An existing network of ground control points (GCPs) were used throughout Reach 4 and supplemented within Phase 2a, as

some GCPs were displaced during summer 2021 restoration construction. Flights were conducted by Joe Rudolph (Remote Pilot, W2r) with the assistance of Lauren Mork, UDWC.

Visible Imagery Collection and Processing

A Phantom 4 Pro and 3-band (RGB) 1'' CMOS (20 MP) sensor was used to acquire visible band imagery. All flights were flown at approximately 300 ft above ground level (AGL) with 70 percent frontal and 60 percent side overlap between images. An average ground sampling distance (GSD) of 1 inch was achieved.

Images were processed in the Pix4D desktop environment where photogrammetric techniques were employed to align photos for ortho-mosaicking and three-dimensional reconstruct. The photogrammetric workflow involved initial image matching, GCP calibration, and raster product development. A high-resolution orthomosaic, digital surface model (DSM), and digital terrain model (DTM) were derived from the input imagery and GCP information. Appendix A provides a detailed summary of data processing and the quality of the orthomosaic, DSM, and DTM.

Multispectral Imagery Collection and Processing

A Phantom 4 Pro Multispectral and 5-band (2 MP) sensors were used to acquire multispectral imagery. Bands collected include Red, Green, Blue, Red-edge, and Near-Infrared. All flights were flown at approximately 300 ft above ground level (AGL) with 70 percent frontal and 60 percent side overlap between images. An average ground sampling distance (GSD) of 1.6 inches was achieved.

Images were processed in the Pix4D desktop environment where photogrammetric techniques were employed to align photos for ortho-mosaicking. Within Pix4D an orthomosaic was produced for each band. Bands were imported into ArcGIS and combined to generate a composite orthomosaic containing all five bands. The photogrammetric workflow involved initial image matching and raster product creation. A high-resolution orthomosaic, DSM, and DTM were derived from the input image. The composite imagery was georeferenced to the RGB imagery in ArcGIS.

Velocity Video Plots

A Phantom 4 Pro and 3-band (RGB) 1'' CMOS (20 MP) sensor was used to acquire 45-second videos in Phase 2a at intervals determined in the field by Brandon Overstreet, Ph.D., USGS. In conjunction with the video plots, velocity measurements were collected using a SonTek FlowTracker2 acoustic Doppler velocimeter along four transects within the velocity mapping area of interest. Transects were established by placing 1 x 1-foot floor tiles at the endpoints of each transect. A survey tape was stretched between the targets and velocity measurements were referenced as the distance from the river right transect endpoint. Five to ten velocity stations were measured along each transect at non-uniform spacing dictated by the wood and sediment bars along the transect. Velocity was measured at 0.6 total depth and each measurement was averaged over 40 seconds. Targets were set for five transects but only four were measured due to time constraints. Velocity analysis for Phase 2a using this data is described in the accompanying technical memo "Mapping continuous spatial heterogeneity in stream velocity using image velocimetry from unoccupied aerial systems (UAS), Whychus Creek, Oregon, July 2022" (UDWC 2023).

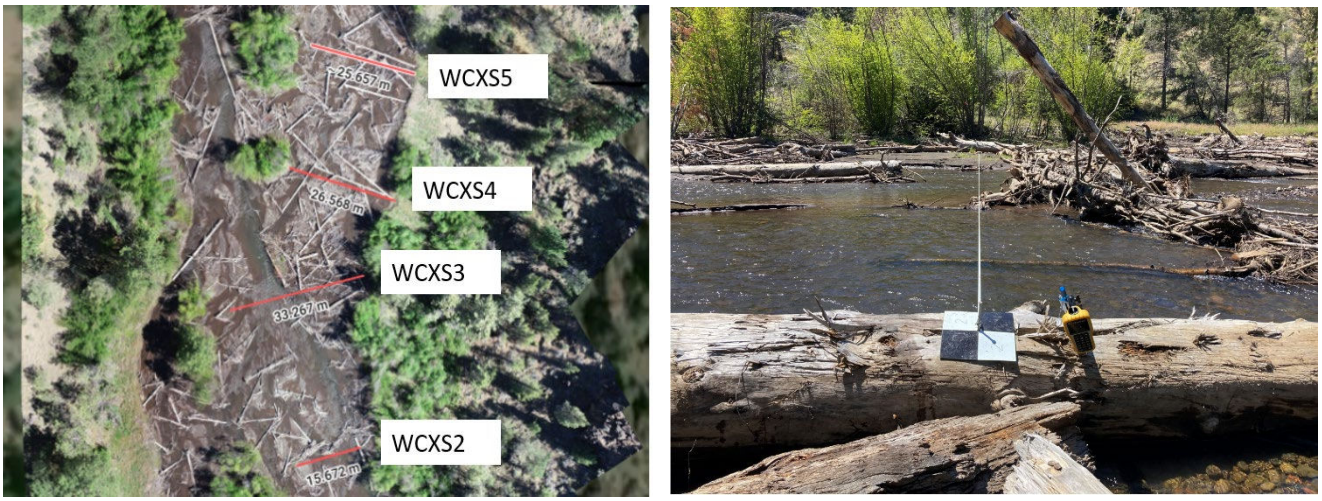


Figure 1: Velocity locations and placements (Credit: Brandon Overstreet, Ph.D.)

Land Cover Classification

To classify land cover features, an automated classification process was developed using multispectral imagery and a photogrammetry-derived digital surface model (DSM). The classification process was completed using open-source python code and employing the rasterio, scipy, numpy, and geopandas libraries. A simplified workflow is shown in Figure 2, and the full source code is attached as Appendix B.

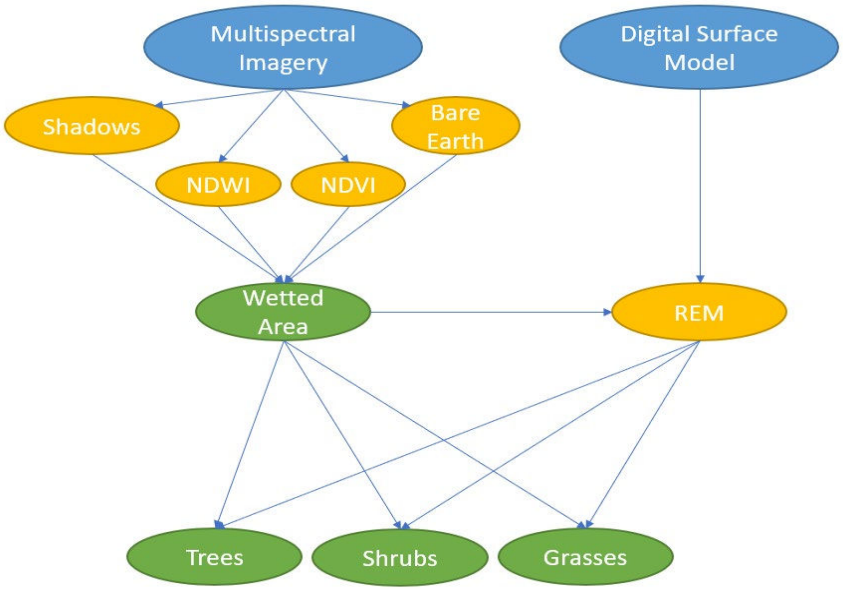


Figure 2: Simplified workflow diagram for classification of the wetted area and vegetation type. Blue bubbles indicate input data, yellow bubbles represent intermediate steps, and green bubbles are the final classification outputs.

In addition to multispectral imagery and DSM, the python script requires 11 input parameters from the user (Table 1). These inputs can be modified to adjust to specific site morphology and imagery conditions, however, maintaining consistency of parameter variables between sites or years of imagery will yield more comparable results.

In a recently disturbed floodplain wood can be identified with reasonable accuracy. Under recently disturbed conditions, large wood is unobscured by vegetation and is neatly contrasted by inundated areas. On the east side of the Cascades, conditions favorable to automated wood classification will

likely persist for the first five years following restoration. This period is also the most active for wood redistribution before vegetation establishes and becomes the significant driver of hydrogeomorphic processes. In a densely vegetated floodplain wood is increasingly difficult to isolate due to the similar spectral signature of wood and bare earth and to being obscured by vegetation. For this reason, it was not possible to consistently map large wood, and therefore it was omitted from the classification scheme for Phase 1/Reach 4 where restoration had been implemented (with associated disturbance) six years prior, in 2016. Higher spectral resolution imagery (either more bands or hyperspectral) is likely the single best way to increase the accuracy of wood detection through remote sensing.

Table 1: Description of the input variables for the classification workflow and values chosen for each site. Reach 4 was split into three equal sections for computational efficiency.

Variable Name	Description	Phase 2A	R4 Upstream	R4 Middle	R4 Downstream
be_threshold	If the reflectance of each band exceeds this threshold, then a pixel is considered bare earth.	17000	15000	15000	15000
shadow_threshold	If the reflectance of each band is less than this threshold, then a pixel is considered shadow.	7000	7000	7000	7000
elev_threshold	Initial elevation to filter out wetted area. Best to choose the maximum floodplain elevation.	2605	2712	2708	2695
NDWI_threshold	Threshold to identify water pixels using NDWI.	0	0	0	0
NDVI_threshold	Threshold to identify vegetated pixels using NDVI.	0	0	0	0
wetted_area_min_size	Minimum size of wetted area to include. Helpful to omit single pixel polygons and other tiny one-off areas that meet the wetted area criteria.	25	25	50	50
wetted_area_buffer	Buffer size to close holes and gaps in the wetted area. Note that this expands the wetted area unless set to zero. (Unit is feet)	1	1	1	1
largest_poly_only	Selects the largest wetted area polygon. This only works if the true wetted area is captured in a single polygon.	TRUE	FALSE	FALSE	FALSE
valley_bottom_buffer	Buffer size to close holes and gaps in the valley bottom. Expands the valley bottom by 1/10 the buffer size.	100	100	50	50
max_elev_grass	Maximum relative elevation of herbaceous.	2	2	2	2
max_elev_shrub	Maximum relative elevation of shrub/scrub.	5	5	5	5
In_channel_wood	True/false of whether or not to classify wood within the wetted area.	True	False	False	False

The first step in the classification process is to create a series of intermediate outputs from the multispectral imagery. The normalized difference water index [NDWI] alone is insufficient to delineate the wetted area. High and low reflectance pixels (bare earth and shadows) are often classified as water when using the NDWI. Off-nadir pixels near the edge of the flight path may also be identified as water. We masked the NDWI to remove shadows, bare earth, vegetation, and areas exceeding the floodplain elevation to achieve a more reliable delineation of wetted area.

The next step in the classification process focused on vegetation, which relied on a relative elevation model [REM] created by detrending the DSM with a plane fit to random points within the wetted area. This REM is necessary for identifying vegetation types by their canopy height. The REM is also used to delineate the valley bottom. The uplands were masked out to focus on the classification of the floodplain vegetation communities. Classification in the uplands is less accurate due to topographic influence, varying relative elevation of the ground surface to the water surface, and off-nadir skew at the edge of the imagery.

Pixels that remain uncategorized are classified as "unclassified". This category includes bare earth, but also some large wood and snags. Careful interpretation of this category is required if drawing conclusions from these classification results. Pixels in the uplands are set to no data.

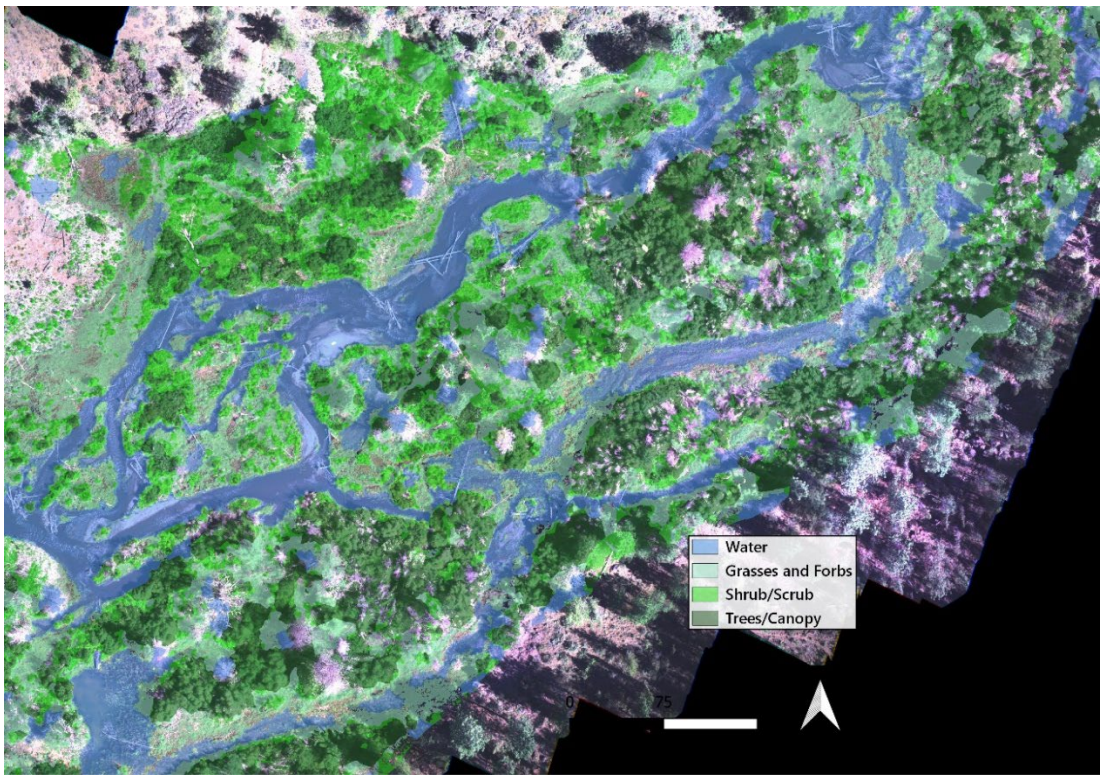


Figure 3: Excerpt of the classification scheme for the downstream end of Reach 4.

Table 2: Classification results by total area and percent of floodplain.

	Total Area (Acres)					Percent of Floodplain Area				
Reach	**Wetted Area**	**Herbaceous**	**Shrub/ Scrub**	**Trees/ Canopy**	**Bare Earth or Unclassified**	**Wetted Area**	**Herbaceous**	**Shrub/ Scrub**	**Trees/ Canopy**	**Bare Earth or Unclassified**
R4 US	3.2	3.2	2.7	2.4	4.0	20%	21%	17%	15%	26%
R4 Middle	2.7	3.1	2.3	4.1	2.3	19%	21%	16%	28%	16%
R4 DS	3.0	3.5	1.3	3.0	2.3	22%	27%	10%	23%	17%
R4 Totals	**8.9**	**9.8**	**6.3**	**9.5**	**8.6**	*totals not applicable to metric*				
Phase 2A	7.4	2.3	2.5	7.2	10.1	25%	8%	8%	25%	34%

Table 2 shows the distribution of classified features. A relatively high percentage of Phase 2a is classified as bare earth, due to dense wood accumulations and growing sediment bars in this reach confusing the algorithm (Figure 4). Changing the NDWI and NDVI threshold parameters as well as lowering the minimum water area parameter may increase the accuracy of wetted area classification in this unique reach, however, additional time is needed to complete a more detailed sensitivity analysis.

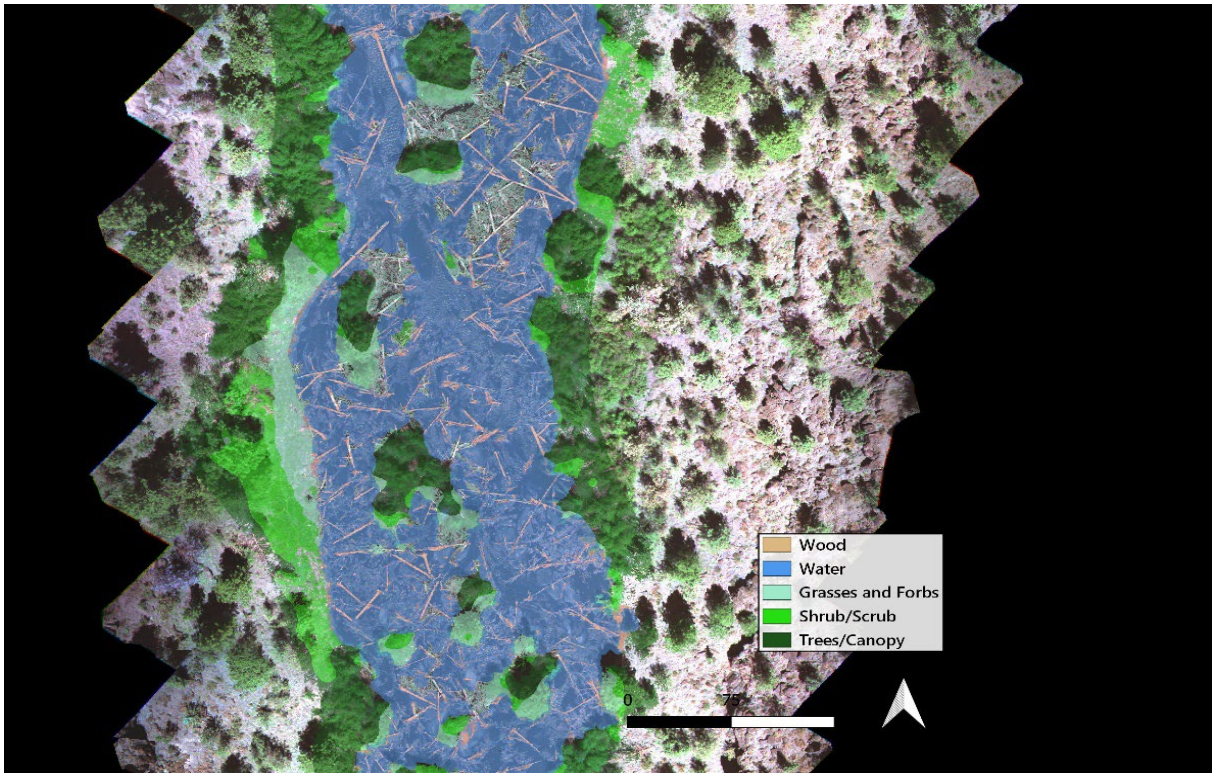


Figure 4: Excerpt of the classification scheme for the Phase 2A reach. Classification errors due to large wood accumulations and sediment bar growth are visible here.

Validation of the classification results are shown in Table 3. Ten points per category were manually identified from the multispectral imagery and DSM and compared to the classification. The bare earth classification has low accuracy due to the spectral similarity between bare earth and grasses. The shrub category also has a low accuracy likely due to the smaller range of acceptable relative elevations. Adjusting the relative elevation ranges of the different vegetation types may yield more accurate and/or desirable results.

Other known issues include shadows along the valley wall being identified as water, dense wood accumulations and vegetation obscuring the wetted area, and inaccuracies in how surfaces (e.g.

snags, shape of trees, turbulent water surfaces) are represented in the DSM. These issues all influence the overall results and should be considered when making interpretations of the data. The automated workflow is not perfect, and is accurate on the order of hundreds of square feet. Most importantly, it allows for fast, objective comparisons of different imagery sets over a variety of floodplain morphologies.

Table 3: Validation and accuracy results of the classification scheme.

Category	ID	Accuracy	Notes
Bare Earth	1	40%	Includes unclassified pixels
Water	2	70%	
Herbaceous	3	80%	
Shrubs	4	40%	Higher inaccuracy due to narrow relative elevation band
Trees	5	80%	

Tree Species Identification

While showing great potential, the tree species identification process requires refinement. The workflow and results are presented here, but they are not considered accurate or final. This process uses discrete object detection and supervised deep-learning with multispectral imagery to identify individual species; tree species selected for analysis from 2022 imagery included Ponderosa pine, Juniper, Alder, Cottonwood, and Willow. There are two major steps in the workflow: first to identify all tree crowns, and second to classify each tree crown into one of the five selected species.

First, to identify all tree crowns in the project area, the DeepForest python package for training and predicting individual tree crowns from airborne RGB imagery was used. The package contains a prebuilt model trained on data from the National Ecological Observatory Network. The output of the DeepForest model is a layer of bounding boxes around each tree crown. Passing the resulting tree crowns through a DSM filter would help remove incorrectly identified tree crowns, a problem most common in areas with a high density of large wood, but this was not done due to budget constraints. The model is also sensitive to imagery resolution and scale of desired features, making it difficult to capture all trees (large and small) in one pass. Multiple runs of the model at different scales could be combined in future efforts to create a more robust dataset of tree crowns.

To categorize each tree crown into tree species, a custom convolutional neural network (CNN) model was developed. This model was trained using tree survey data from the project site collected in June and July, 2022. Using the bounding boxes output from the DeepForest model, an image was extracted for each tree crown. These images were used as inputs into the CNN, which classified each tree crown as a species. The final result is a feature class of bounding boxes for each tree crown assigned as a given tree species (Figure 5).

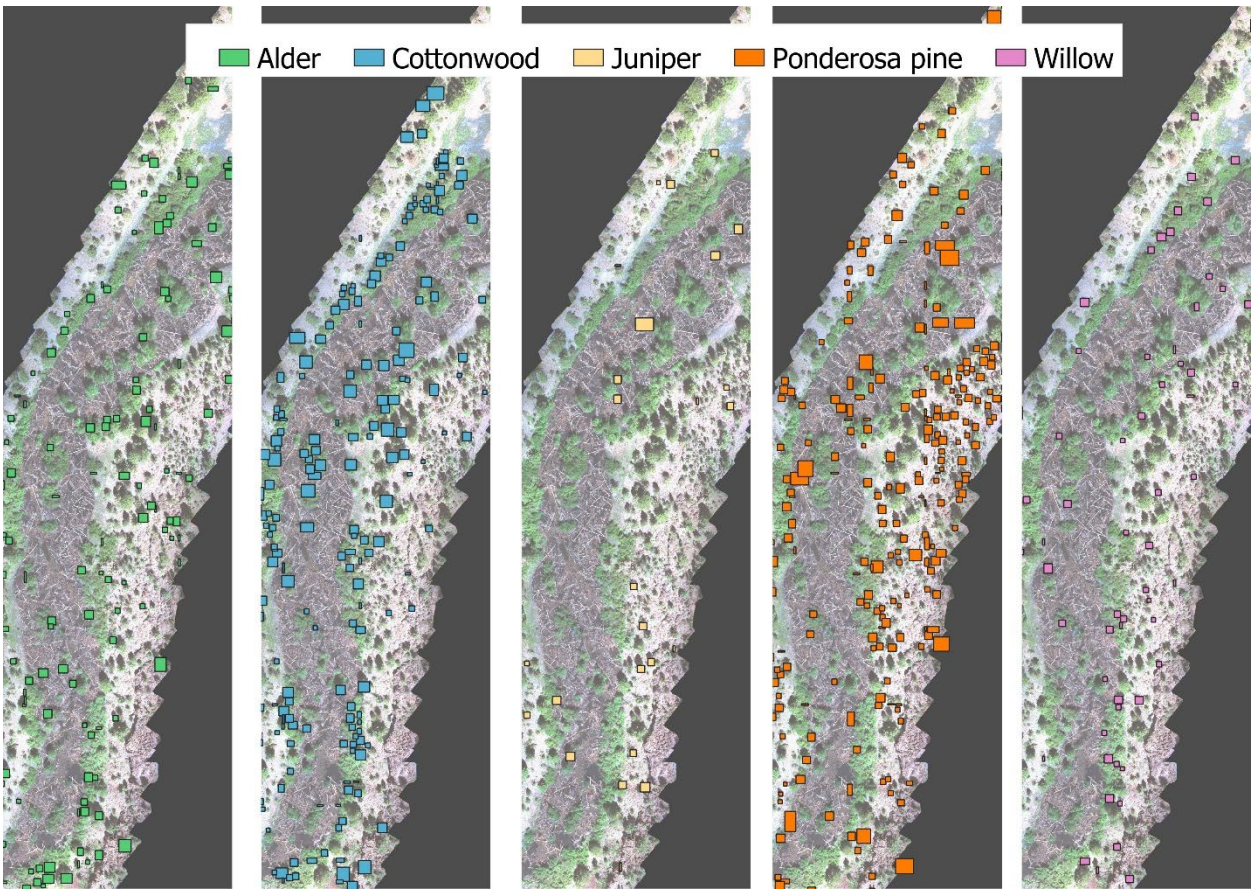


Figure 5: Tree species identification results. Each bounding box represents one tree.

The existing model exhibits 90% accuracy with the training dataset (n=32) and 60% accuracy with the validation dataset (n=10). While the initial results are promising, the model is likely over-fit (overrepresents or misidentifies target species) to the relatively small training/validation datasets. Further calibration and validation of the model with different imagery and tree survey data will increase the overall accuracy and robustness.

Table 4. Effectiveness Monitoring Using Photogrammetric Remote Sensing Techniques

Effectiveness Monitoring Table provided by Lauren Mork (UDWC) and adapted by W2r (12/2022)

Feature	**Metric**	**Remote Sensing**	**Refinements**
Wood	% Wood area or volume per acre process space	Map large wood distribution for up to five years following restoration. Divide the site into equal areas or by geomorphic unit.	Determine an approach to map wood in well-established (>5 years) sites.
Vegetation	% Woody (tree and shrub) riparian vegetation area per acre process space	Classify multispectral imagery and refine using REM to determine vegetation distribution. Compute percent covers based on the total project area or on a per-area basis to evaluate spatial-temporal trends.	Incorporate LiDAR to develop a more refined REM and refine vegetation classification groupings.
Beaver dams	Number of beaver dams	Manually identify and count the number of occurrences and spatial distribution through time.	n/a
Surface water	% Wetted area per acre process space	Classification from multispectral imagery and calculation area.	Diminishing accuracy as vegetation obscures visible wetted area. LiDAR may substitute as the frequency of observations reduces over time.
Channel dynamism	% area new and abandoned active channel per time or flow event interval?	Channel mapping over time from imagery. Establish transects to measure the number of channel intersections (for Phase 1 have used groundwater well cross-sections).	n/a
Topographic roughness (on floodplain and within active channels)	Wood, vegetation, unvegetated bars, and microtopography	Derive from classified imagery to evaluate surface roughness and REM to identify micro-topography.	LiDAR relative to REM will make this analysis more robust.
Depth	Range and median; mapped	It is possible to evaluate depth by spectral signature.	Green LiDAR derived REM would make this analysis more reliable; possible future analysis using 2022 green LiDAR.

Next Steps

In addition to the refinements and suggestions noted in Table 4, the following next steps can be achieved with existing or additional data.

- Sensitivity testing for input parameters including wetted area in Reach Phase 2a to inform the degree to which changing the input parameter changes the result, e.g. what is the effect of a wetted area size of 1 ft v. 5 ft on the final calculated area? Some parameters are likely more sensitive than others.
- Incorporate a LiDAR derived REM as the basis for vegetation community classification.
- Incorporate LiDAR to develop a refined REM and vegetation classification results.
- Complete a ground-based habitat survey characterizing vegetation communities. Survey points should be collected in the center of the community or a survey grade polygon of a substantial portion of the community. This data will help refine the classification and validation process.
- Continue to refine NDVI and NDWI thresholds to refine wetted area and large wood identification and mapping. Apply established thresholds to other reaches and imagery with different lighting conditions to evaluate the potential for automation across all reaches or other eastern Oregon project sites.
- The shrubs class could be reclassified as woody vegetation and included in the tree category. This would likely improve the overall accuracy of the two classes.
- Process previous years of multispectral imagery to generate year-over-year comparisons.

Suggestions for data processing and analysis

- Rather than classifying different vegetation types and worrying about the exact cutoff between trees and shrubs and grasses, use DSM elevation distribution to track changes in floodplain connectivity and vegetation growth.
- For larger file sizes (>1 GB) it may be necessary to split the imagery into smaller extents for computational efficiency.
- As vegetation establishes and on-the-ground features become obscured, more costly LiDAR should be acquired at less frequent intervals to continue monitoring.

Web Visualization:

The various raster products can be viewed in the Pix4D web interface using the links provided below:

Rim Rock Phase 2a RBG Imagery

Canyon Reach 4 RGB Imagery

Data Download Links:

Surveys 2022

https://w2r.egnyte.com/fl/FI4PdpGaia/Surveys_2022_

Imagery_Outputs_2022

https://w2r.egnyte.com/fl/Wu7wSFzyGA/Imagery_Outputs_2022_

Appendix A: Whychus Phases 1 & 2a Imagery Processing Report

Appendix B: Python Script and Workflow

Quality Report

Generated with PIX4Dmapper version 4.7.5

Important: Click on the different icons for:

Help to analyze the results in the Quality Report

Additional information about the sections

Click here for additional tips to analyze the Quality Report

Summary

Project	Whychus_R4
Processed	2022-10-28 15:29:29
Camera Model Name(s)	FC6310_8.8_5472x3648 (RGB)
Average Ground Sampling Distance (GSD)	2.06 cm / 0.81 in
Area Covered	0.277 km^2 / 27.6665 ha / 0.11 sq. mi. / 68.4008 acres

Quality Check

Images	median of 75674 keypoints per image	✓
Dataset	326 out of 327 images calibrated (99%), all images enabled	✓
Camera Optimization	0.11% relative difference between initial and optimized internal camera parameters	✓
Matching	median of 5903.71 matches per calibrated image	✓
Georeferencing	yes, 4 GCPs (4 3D), mean RMS error = 0.012 ft	✓

Preview

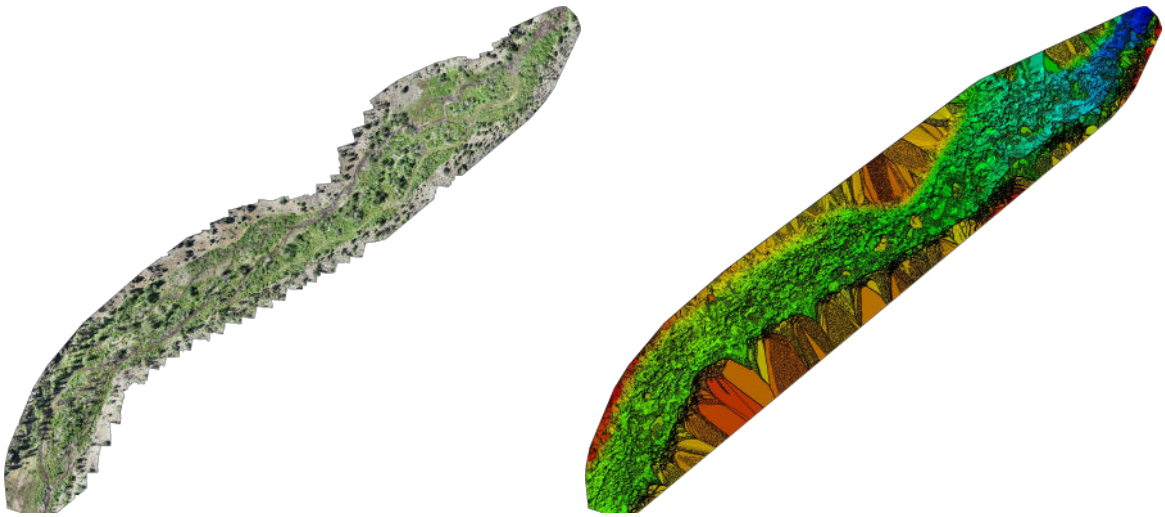


Figure 1: Orthomosaic and the corresponding sparse Digital Surface Model (DSM) before densification.

Calibration Details

Number of Calibrated Images	326 out of 327
Number of Geolocated Images	327 out of 327

Initial Image Positions

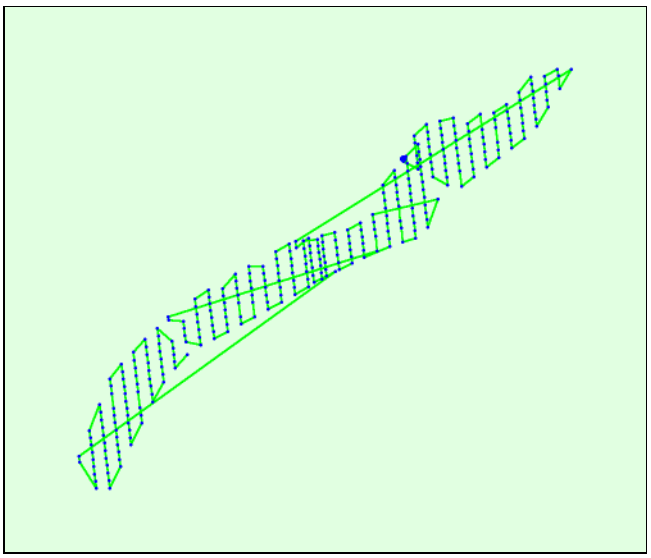


Figure 2: Top view of the initial image position. The green line follows the position of the images in time starting from the large blue dot.

Computed Image/GCPs/Manual Tie Points Positions

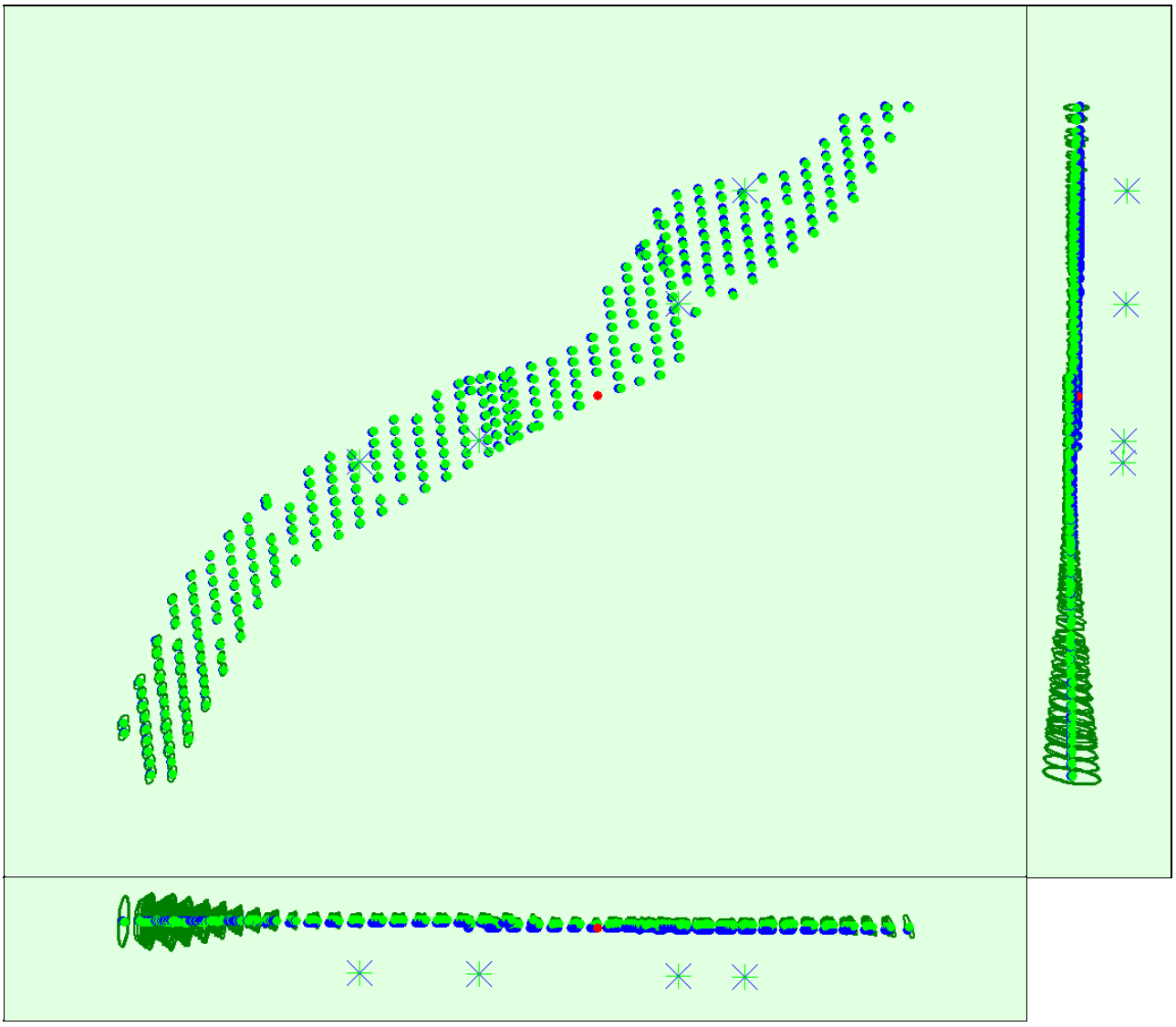


Figure 3: Offset between initial (blue dots) and computed (green dots) image positions as well as the offset between the GCPs initial positions (blue crosses) and

their computed positions (green crosses) in the top-view (XY plane), front-view (XZ plane), and side-view (YZ plane). Red dots indicate disabled or uncalibrated images. Dark green ellipses indicate the absolute position uncertainty of the bundle block adjustment result.

Absolute camera position and orientation uncertainties

	X [ft]	Y [ft]	Z [ft]	Omega [degree]	Phi [degree]	Kappa [degree]
Mean	0.127	0.173	0.420	0.034	0.026	0.006
Sigma	0.053	0.082	0.267	0.012	0.010	0.003

Overlap

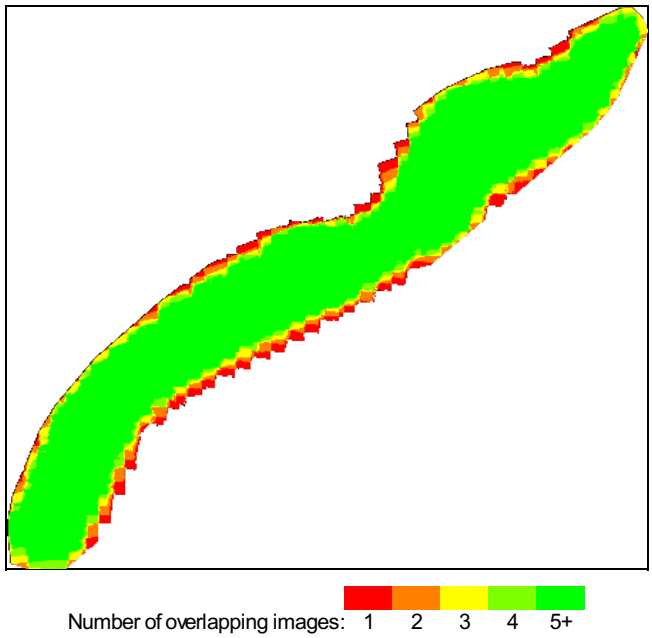


Figure 4: Number of overlapping images computed for each pixel of the orthomosaic.
Red and yellow areas indicate low overlap for which poor results may be generated. Green areas indicate an overlap of over 5 images for every pixel. Good quality results will be generated as long as the number of keypoint matches is also sufficient for these areas (see Figure 5 for keypoint matches).

Bundle Block Adjustment Details

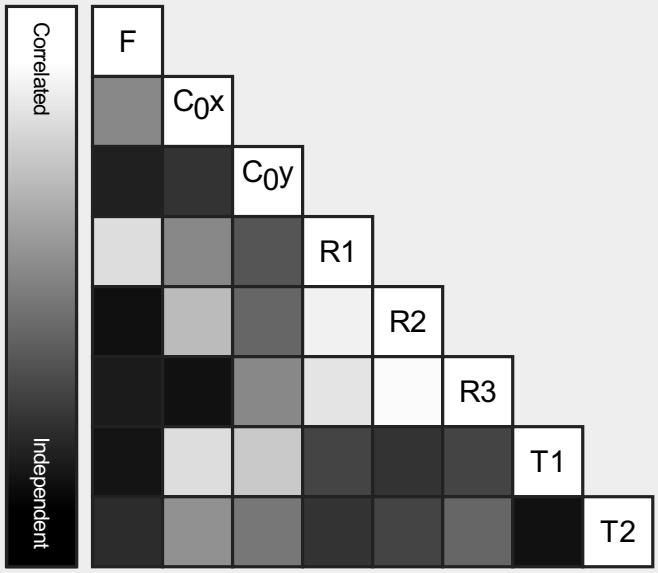
Number of 2D Keypoint Observations for Bundle Block Adjustment	2174841
Number of 3D Points for Bundle Block Adjustment	972569
Mean Reprojection Error [pixels]	0.181

Internal Camera Parameters

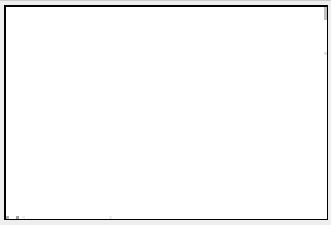
FC6310_8.8_5472x3648 (RGB). Sensor Dimensions: 12.833 [mm] x 8.556 [mm]

EXIF ID: FC6310_8.8_5472x3648

	Focal Length	Principal Point x	Principal Point y	R1	R2	R3	T1	T2
Initial Values	3668.759 [pixel] 8.604 [mm]	2736.001 [pixel] 6.417 [mm]	1823.999 [pixel] 4.278 [mm]	0.003	-0.008	0.008	-0.000	0.000
Optimized Values	3672.991 [pixel] 8.614 [mm]	2753.986 [pixel] 6.459 [mm]	1821.530 [pixel] 4.272 [mm]	0.004	-0.007	0.007	-0.001	0.002
Uncertainties (Sigma)	3.742 [pixel] 0.009 [mm]	0.162 [pixel] 0.000 [mm]	0.105 [pixel] 0.000 [mm]	0.000	0.000	0.000	0.000	0.000



The correlation between camera internal parameters determined by the bundle adjustment. White indicates a full correlation between the parameters, ie. any change in one can be fully compensated by the other. Black indicates that the parameter is completely independent, and is not affected by other parameters.



The number of Automatic Tie Points (ATPs) per pixel, averaged over all images of the camera model, is color coded between black and white. White indicates that, on average, more than 16 ATPs have been extracted at the pixel location. Black indicates that, on average, 0 ATPs have been extracted at the pixel location. Click on the image to the see the average direction and magnitude of the re-projection error for each pixel. Note that the vectors are scaled for better visualization. The scale bar indicates the magnitude of 1 pixel error.

2D Keypoints Table

	Number of 2D Keypoints per Image	Number of Matched 2D Keypoints per Image
Median	75674	5904
Min	67206	686
Max	86614	18200
Mean	75217	6671

3D Points from 2D Keypoint Matches

	Number of 3D Points Observed
In 2 Images	832179
In 3 Images	94477
In 4 Images	25916
In 5 Images	9760
In 6 Images	4538
In 7 Images	2502
In 8 Images	1398
In 9 Images	773
In 10 Images	428
In 11 Images	251
In 12 Images	147
In 13 Images	85
In 14 Images	41
In 15 Images	31
In 16 Images	14
In 17 Images	11
In 18 Images	8
In 19 Images	2
In 20 Images	5
In 21 Images	3

2D Keypoint Matches

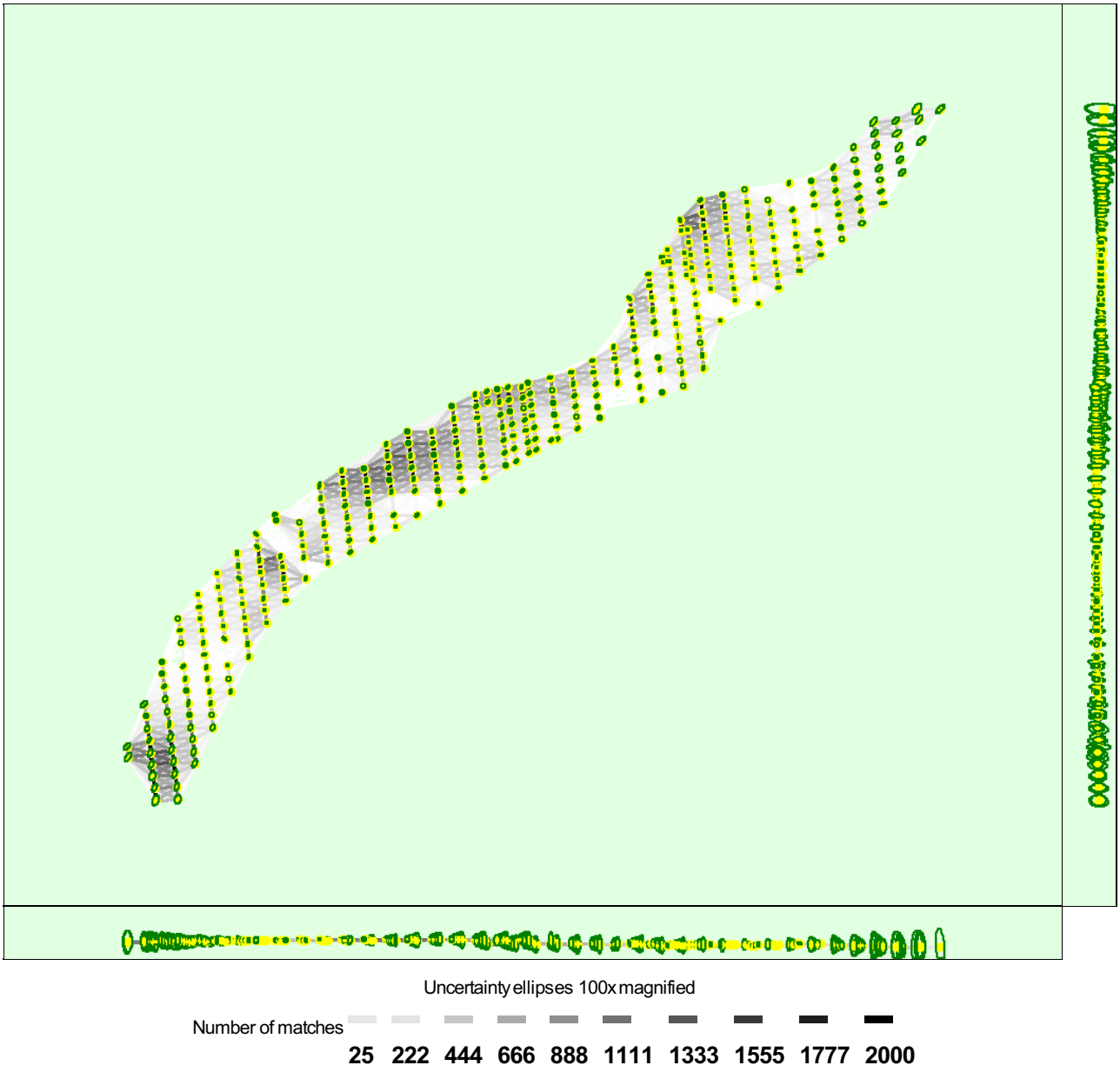


Figure 5: Computed image positions with links between matched images. The darkness of the links indicates the number of matched 2D keypoints between the images. Bright links indicate weak links and require manual tie points or more images. Dark green ellipses indicate the relative camera position uncertainty of the bundle block adjustment result.

Relative camera position and orientation uncertainties

	X [ft]	Y [ft]	Z [ft]	Omega [degree]	Phi [degree]	Kappa [degree]
Mean	0.095	0.104	0.249	0.032	0.027	0.007
Sigma	0.030	0.037	0.140	0.010	0.012	0.002

Geolocation Details

Ground Control Points

GCP Name	Accuracy XY/Z [ft]	Error X [ft]	Error Y [ft]	Error Z [ft]	Projection Error [pixel]	Verified/Marked
1 (3D)	0.020/ 0.020	-0.003	-0.001	-0.011	0.410	13 / 13
2 (3D)	0.020/ 0.020	0.011	-0.004	0.020	0.852	9 / 9

3 (3D)	0.020/ 0.020	-0.010	0.002	-0.019	0.399	8 / 8
4 (3D)	0.020/ 0.020	0.014	0.006	0.037	0.140	3 / 3
Mean [ft]		0.002804	0.000619	0.006866		
Sigma [ft]		0.009929	0.003525	0.022720		
RMS Error [ft]		0.010317	0.003579	0.023735		

Localisation accuracy per GCP and mean errors in the three coordinate directions. The last column counts the number of calibrated images where the GCP has been automatically verified vs. manually marked.

Absolute Geolocation Variance

Min Error [ft]	Max Error [ft]	Geolocation Error X [%]	Geolocation Error Y [%]	Geolocation Error Z [%]
-	-49.21	0.00	0.00	0.00
-49.21	-39.37	0.00	0.00	0.00
-39.37	-29.53	0.00	0.00	0.00
-29.53	-19.69	0.00	0.00	0.00
-19.69	-9.84	0.00	0.00	20.86
-9.84	0.00	48.77	63.50	34.97
0.00	9.84	51.23	21.47	19.94
9.84	19.69	0.00	15.03	20.25
19.69	29.53	0.00	0.00	3.99
29.53	39.37	0.00	0.00	0.00
39.37	49.21	0.00	0.00	0.00
49.21	-	0.00	0.00	0.00
Mean [ft]		-7.439229	2.145222	-12.732587
Sigma [ft]		2.080329	6.800447	10.591411
RMS Error [ft]		7.724629	7.130782	16.561907

Min Error and Max Error represent geolocation error intervals between -1.5 and 1.5 times the maximum accuracy of all the images. Columns X, Y, Z show the percentage of images with geolocation errors within the predefined error intervals. The geolocation error is the difference between the initial and computed image positions. Note that the image geolocation errors do not correspond to the accuracy of the observed 3D points.

Geolocation Bias	X	Y	Z
Translation [ft]	-7.439229	2.145222	-12.732587

Bias between image initial and computed geolocation given in output coordinate system.

Relative Geolocation Variance

Relative Geolocation Error	Images X [%]	Images Y [%]	Images Z [%]
[-1.00, 1.00]	100.00	100.00	100.00
[-2.00, 2.00]	100.00	100.00	100.00
[-3.00, 3.00]	100.00	100.00	100.00
Mean of Geolocation Accuracy [ft]	16.404199	16.404199	32.808399
Sigma of Geolocation Accuracy [ft]	0.000001	0.000001	0.000001

Images X, Y, Z represent the percentage of images with a relative geolocation error in X, Y, Z.

Geolocation Orientational Variance	RMS [degree]
Omega	0.556
Phi	0.754
Kappa	13.583

Geolocation RMS error of the orientation angles given by the difference between the initial and computed image orientation angles.

Initial Processing Details

System Information

Hardware	CPU: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz RAM: 32GB GPU: Intel(R) UHD Graphics 630 (Driver: 30.0.101.1122), NVIDIA Quadro P1000 (Driver: 30.0.15.1329)
Operating System	Windows 10 Pro, 64-bit

Coordinate Systems

Image Coordinate System	WGS 84 (EGM96 Geoid)
Ground Control Point (GCP) Coordinate System	NAD83(2011) / Oregon South (ft) (EGM96 Geoid)
Output Coordinate System	NAD83(2011) / Oregon South (ft) (EGM96 Geoid)

Processing Options

Detected Template	3D Maps
Keypoints Image Scale	Full, Image Scale: 1
Advanced: Matching Image Pairs	Aerial Grid or Corridor
Advanced: Matching Strategy	Use Geometrically Verified Matching: no
Advanced: Keypoint Extraction	Targeted Number of Keypoints: Automatic
Advanced: Calibration	Calibration Method: Standard Internal Parameters Optimization: All External Parameters Optimization: All Rematch: Auto, yes

Point Cloud Densification details

Processing Options

Image Scale	multiscale, 1/2 (Half image size, Default)
Point Density	Optimal
Minimum Number of Matches	3
3D Textured Mesh Generation	yes
3D Textured Mesh Settings:	Resolution: Medium Resolution (default) Color Balancing: no
LOD	Generated: no
Advanced: 3D Textured Mesh Settings	Sample Density Divider: 1
Advanced: Image Groups	group1
Advanced: Use Processing Area	yes
Advanced: Use Annotations	yes
Time for Point Cloud Densification	40m:27s
Time for Point Cloud Classification	11m:45s
Time for 3D Textured Mesh Generation	09m:06s

Results

Number of Generated Tiles	4
Number of 3D Densified Points	35244700
Average Density (per ft^3)	7.08

DSM, Orthomosaic and Index Details

Processing Options

DSM and Orthomosaic Resolution	1 x GSD (2.06 [cm/pixel])
DSM Filters	Noise Filtering: yes Surface Smoothing: yes, Type: Medium
Raster DSM	Generated: yes Method: Inverse Distance Weighting Merge Tiles: yes
Orthomosaic	Generated: yes Merge Tiles: yes GeoTIFF Without Transparency: no Google Maps Tiles and KML: no
Raster DTM	Generated: yes Merge Tiles: yes
DTM Resolution	5 x GSD (2.06 [cm/pixel])
Time for DSM Generation	32m:13s
Time for Orthomosaic Generation	01h:00m:59s
Time for DTM Generation	01h:25m:06s
Time for Contour Lines Generation	00s
Time for Reflectance Map Generation	00s
Time for Index Map Generation	00s

Quality Report

Generated with PIX4Dmapper version 4.7.5

Important: Click on the different icons for:

Help to analyze the results in the Quality Report

Additional information about the sections

Click here for additional tips to analyze the Quality Report

Summary

Project	Whychus_P2
Processed	2022-10-27 15:01:31
Camera Model Name(s)	FC6310_8.8_5472x3648 (RGB)
Average Ground Sampling Distance (GSD)	2.09 cm / 0.82 in
Area Covered	0.090 km^2 / 9.0111 ha / 0.03 sq. mi. / 22.2785 acres

Quality Check

Images	median of 71028 keypoints per image	✓
Dataset	117 out of 117 images calibrated (100%), all images enabled	✓
Camera Optimization	1.16% relative difference between initial and optimized internal camera parameters	✓
Matching	median of 13541.2 matches per calibrated image	✓
Georeferencing	yes, 3 GCPs (3 3D), mean RMS error = 0.109 ft	✓

Preview

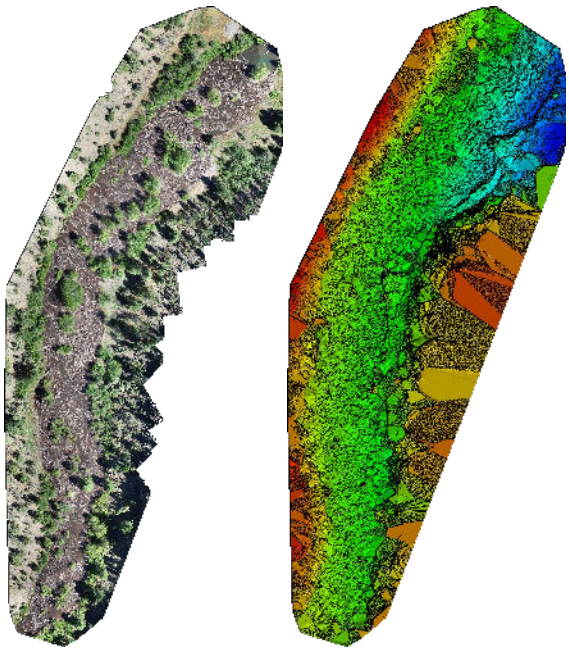


Figure 1: Orthomosaic and the corresponding sparse Digital Surface Model (DSM) before densification.

Calibration Details

Number of Calibrated Images	117 out of 117
Number of Geolocated Images	117 out of 117

Initial Image Positions

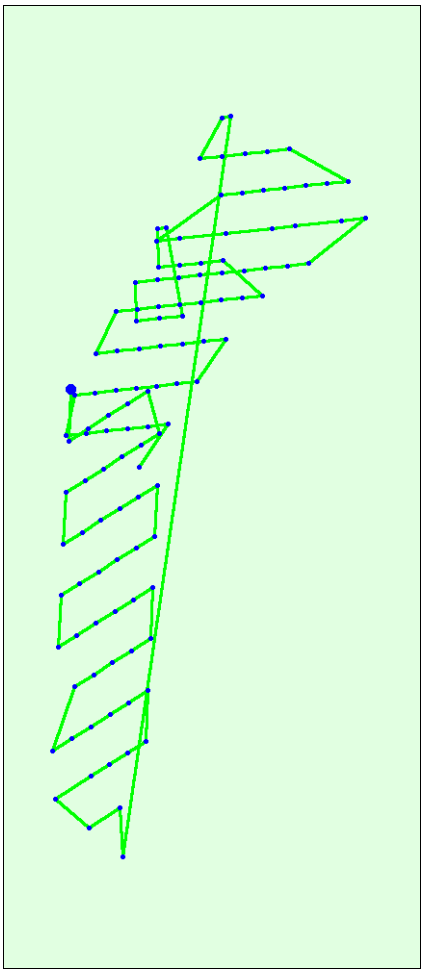
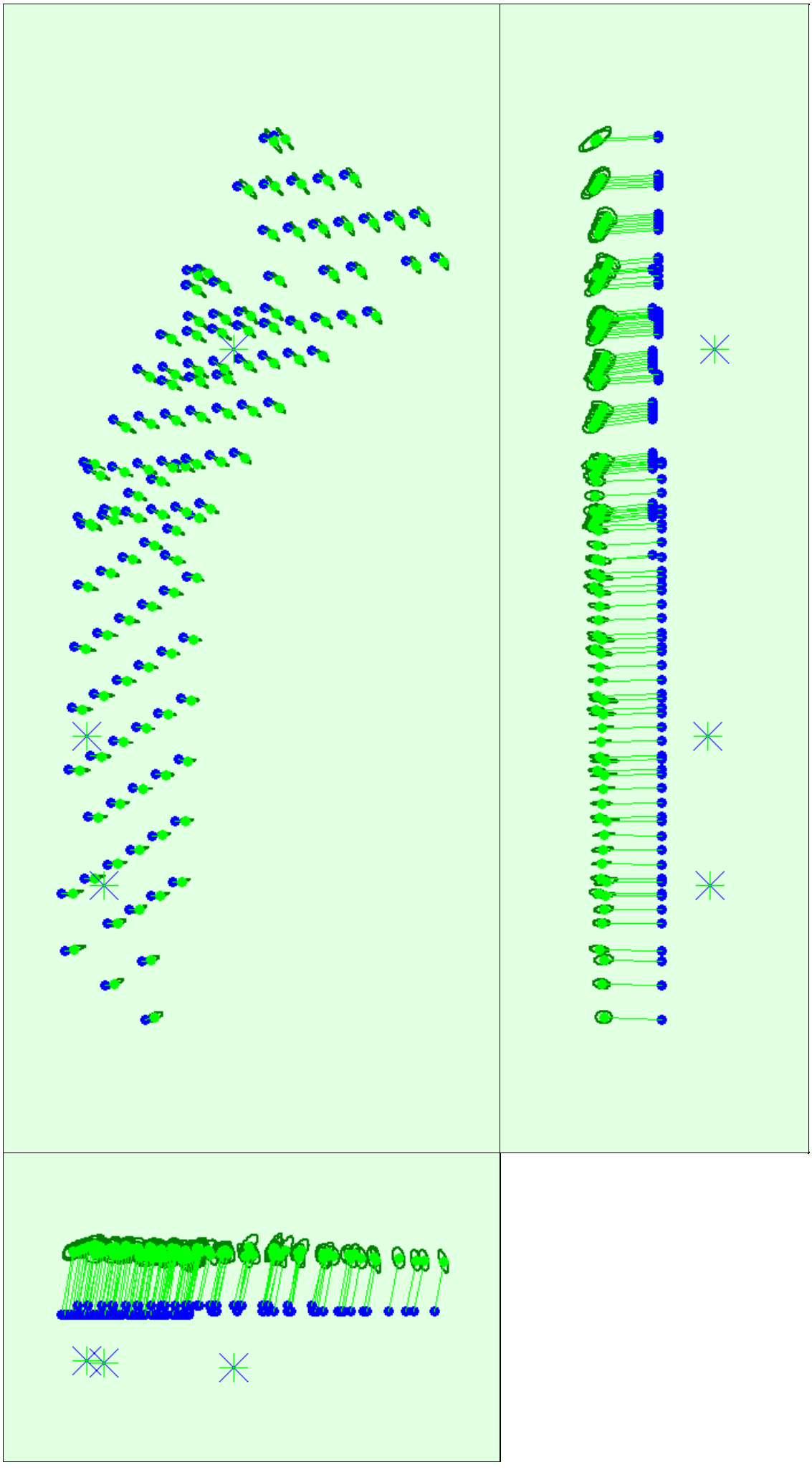


Figure 2: Top view of the initial image position. The green line follows the position of the images in time starting from the large blue dot.

Computed Image/GCPs/Manual Tie Points Positions



Uncertainty ellipses 10x magnified

Figure 3: Offset between initial (blue dots) and computed (green dots) image positions as well as the offset between the GCPs initial positions (blue crosses) and their computed positions (green crosses) in the top-view (XY plane), front-view (XZ plane), and side-view (YZ plane). Dark green ellipses indicate the absolute position uncertainty of the bundle block adjustment result.

Absolute camera position and orientation uncertainties

	X [ft]	Y [ft]	Z [ft]	Omega [degree]	Phi [degree]	Kappa [degree]
Mean	1.858	1.139	2.191	0.396	0.302	0.036
Sigma	0.285	0.594	0.282	0.104	0.132	0.017

Overlap

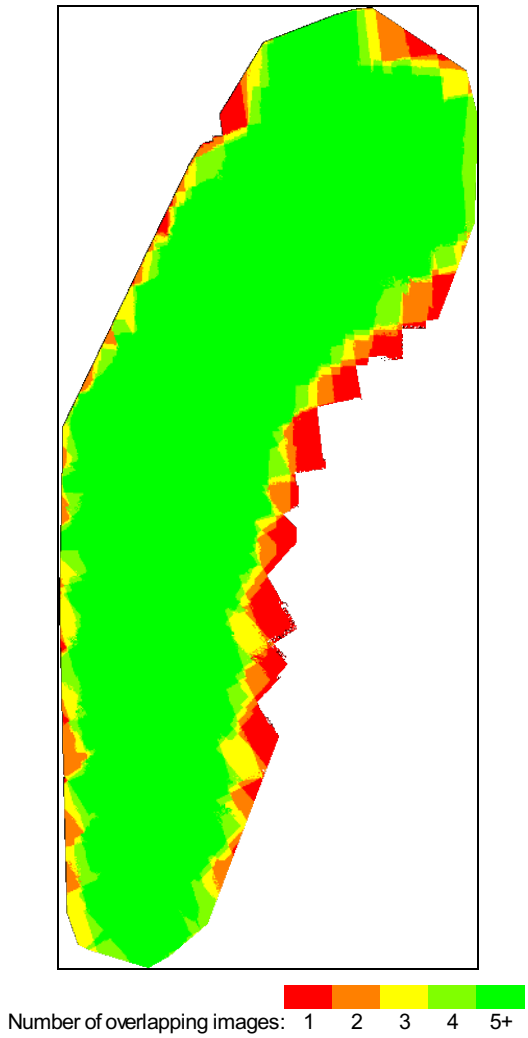


**Figure 4: Number of overlapping images computed for each pixel of the orthomosaic.
Red and yellow areas indicate low overlap for which poor results may be generated. Green areas indicate an overlap of over 5 images for every pixel. Good quality results will be generated as long as the number of keypoint matches is also sufficient for these areas (see Figure 5 for keypoint matches).**

Bundle Block Adjustment Details

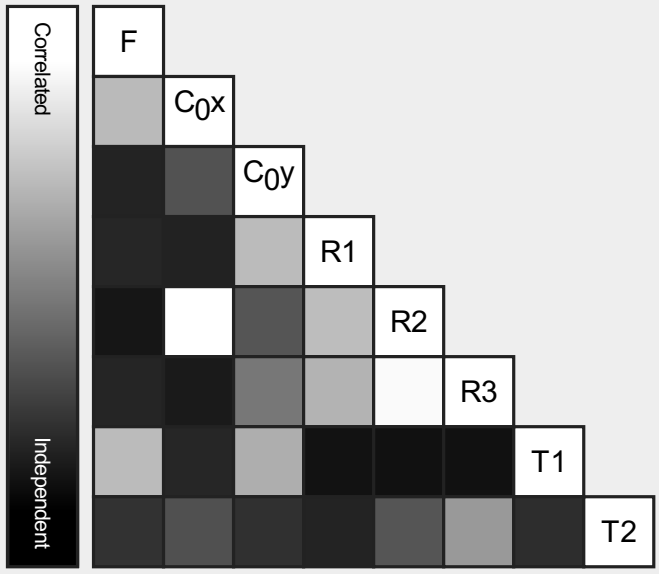
Number of 2D Keypoint Observations for Bundle Block Adjustment	1529959
Number of 3D Points for Bundle Block Adjustment	625951
Mean Reprojection Error [pixels]	0.188

Internal Camera Parameters

FC6310_8.8_5472x3648 (RGB). Sensor Dimensions: 12.833 [mm] x 8.556 [mm]

EXIF ID: FC6310_8.8_5472x3648

	Focal Length	Principal Point x	Principal Point y	R1	R2	R3	T1	T2
Initial Values	3668.759 [pixel] 8.604 [mm]	2736.001 [pixel] 6.417 [mm]	1823.999 [pixel] 4.278 [mm]	0.003	-0.008	0.008	-0.000	0.000
Optimized Values	3626.098 [pixel] 8.504 [mm]	2761.292 [pixel] 6.476 [mm]	1824.625 [pixel] 4.279 [mm]	0.001	-0.007	0.007	-0.001	0.002
Uncertainties (Sigma)	28.373 [pixel] 0.067 [mm]	1.584 [pixel] 0.004 [mm]	0.752 [pixel] 0.002 [mm]	0.001	0.003	0.002	0.000	0.000



The correlation between camera internal parameters determined by the bundle adjustment. White indicates a full correlation between the parameters, ie. any change in one can be fully compensated by the other. Black indicates that the parameter is completely independent, and is not affected by other parameters.

The number of Automatic Tie Points (ATPs) per pixel, averaged over all images of the camera model, is color coded between black and white. White indicates that, on average, more than 16 ATPs have been extracted at the pixel location. Black indicates that, on average, 0 ATPs have been extracted at the pixel location. Click on the image to the see the average direction and magnitude of the re-projection error for each pixel. Note that the vectors are scaled for better visualization. The scale bar indicates the magnitude of 1 pixel error.

2D Keypoints Table

	Number of 2D Keypoints per Image	Number of Matched 2D Keypoints per Image
Median	71028	13541
Min	50981	609
Max	79863	21338
Mean	70132	13077

3D Points from 2D Keypoint Matches

	Number of 3D Points Observed
In 2 Images	478904
In 3 Images	87501
In 4 Images	30318
In 5 Images	12781
In 6 Images	6761
In 7 Images	3851
In 8 Images	2201
In 9 Images	1287
In 10 Images	796
In 11 Images	544
In 12 Images	365

In 13 Images	253
In 14 Images	125
In 15 Images	91
In 16 Images	68
In 17 Images	36
In 18 Images	31
In 19 Images	15
In 20 Images	8
In 21 Images	8
In 22 Images	3
In 23 Images	4

2D Keypoint Matches

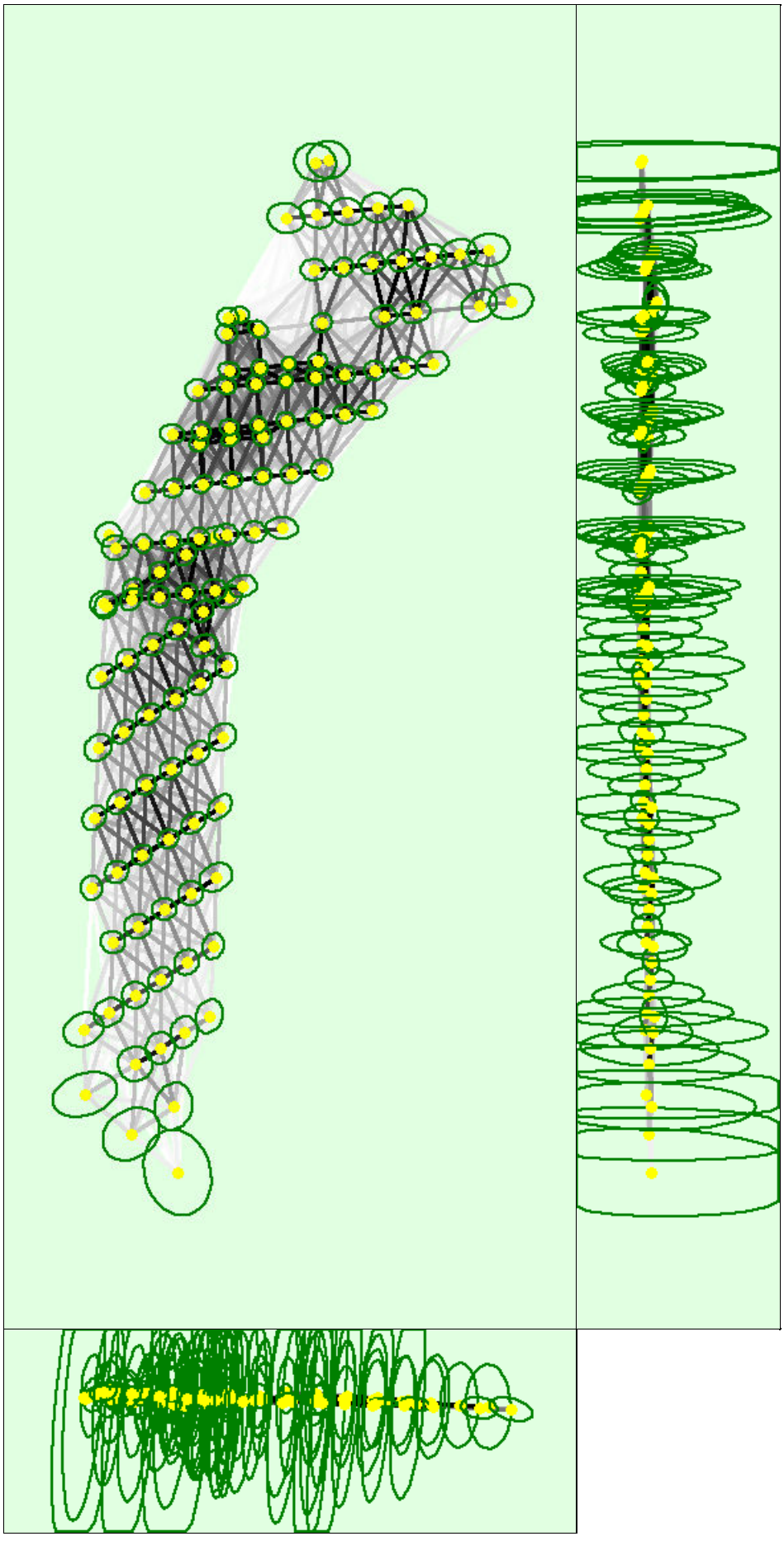


Figure 5: Computed image positions with links between matched images. The darkness of the links indicates the number of matched 2D keypoints between the images. Bright links indicate weak links and require manual tie points or more images. Dark green ellipses indicate the relative camera position uncertainty of the bundle block adjustment result.

Relative camera position and orientation uncertainties

	X [ft]	Y [ft]	Z [ft]	Omega [degree]	Phi [degree]	Kappa [degree]
Mean	0.252	0.233	0.974	0.383	0.299	0.038
Sigma	0.083	0.085	0.766	0.146	0.171	0.017

Geolocation Details

Ground Control Points

GCP Name	Accuracy XY/Z [ft]	Error X [ft]	Error Y [ft]	Error Z [ft]	Projection Error [pixel]	Verified/Marked
1 (3D)	0.020/ 0.020	0.177	-0.180	0.045	0.921	10 / 10
2 (3D)	0.020/ 0.020	-0.169	0.124	-0.100	0.544	10 / 10
4 (3D)	0.020/ 0.020	-0.008	0.018	0.011	0.881	21 / 24
Mean [ft]		0.000042	-0.012409	-0.014861		
Sigma [ft]		0.141474	0.125869	0.061923		
RMS Error [ft]		0.141474	0.126479	0.063681		

Localisation accuracy per GCP and mean errors in the three coordinate directions. The last column counts the number of calibrated images where the GCP has been automatically verified vs. manually marked.

Absolute Geolocation Variance

Min Error [ft]	Max Error [ft]	Geolocation Error X [%]	Geolocation Error Y [%]	Geolocation Error Z [%]
-	-49.21	0.00	0.00	0.00
-49.21	-39.37	0.00	0.00	0.00
-39.37	-29.53	0.00	0.00	0.00
-29.53	-19.69	0.00	0.00	3.42
-19.69	-9.84	0.00	3.42	15.38
-9.84	0.00	46.15	44.44	29.06
0.00	9.84	53.85	52.14	31.62
9.84	19.69	0.00	0.00	17.95
19.69	29.53	0.00	0.00	2.56
29.53	39.37	0.00	0.00	0.00
39.37	49.21	0.00	0.00	0.00
49.21	-	0.00	0.00	0.00
Mean [ft]		-23.420459	8.095307	-127.700187
Sigma [ft]		1.748429	5.322391	10.845620
RMS Error [ft]		23.485632	9.688232	128.159920

Min Error and Max Error represent geolocation error intervals between -1.5 and 1.5 times the maximum accuracy of all the images. Columns X, Y, Z show the percentage of images with geolocation errors within the predefined error intervals. The geolocation error is the difference between the initial and computed image positions. Note that the image geolocation errors do not correspond to the accuracy of the observed 3D points.

Geolocation Bias	X	Y	Z
Translation [ft]	-23.420459	8.095307	-127.700187

Bias between image initial and computed geolocation given in output coordinate system.

Relative Geolocation Variance

Relative Geolocation Error	Images X [%]	Images Y [%]	Images Z [%]
[-1.00, 1.00]	100.00	100.00	100.00
[-2.00, 2.00]	100.00	100.00	100.00
[-3.00, 3.00]	100.00	100.00	100.00
Mean of Geolocation Accuracy [ft]	16.404199	16.404199	32.808399
Sigma of Geolocation Accuracy [ft]	0.000001	0.000001	0.000002

Images X, Y, Z represent the percentage of images with a relative geolocation error in X, Y, Z.

Geolocation Orientational Variance	RMS [degree]
Omega	1.894
Phi	3.821
Kappa	8.413

Geolocation RMS error of the orientation angles given by the difference between the initial and computed image orientation angles.

Initial Processing Details

System Information

Hardware	CPU: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz RAM: 32GB GPU: Intel(R) UHD Graphics 630 (Driver: 30.0.101.1122), NVIDIA Quadro P1000 (Driver: 30.0.15.1329)
Operating System	Windows 10 Pro, 64-bit

Coordinate Systems

Image Coordinate System	WGS 84 (EGM96 Geoid)
Ground Control Point (GCP) Coordinate System	NAD83(2011) / Oregon South (ft) (EGM96 Geoid)
Output Coordinate System	NAD83(2011) / Oregon South (ft) (EGM96 Geoid)

Processing Options

Detected Template	No Template Available
Keypoints Image Scale	Full, Image Scale: 1
Advanced: Matching Image Pairs	Aerial Grid or Corridor
Advanced: Matching Strategy	Use Geometrically Verified Matching: no
Advanced: Keypoint Extraction	Targeted Number of Keypoints: Automatic
Advanced: Calibration	Calibration Method: Standard Internal Parameters Optimization: All External Parameters Optimization: All Rematch: Auto, yes

Point Cloud Densification details

Processing Options

Image Scale	multiscale, 1/2 (Half image size, Default)
Point Density	Optimal
Minimum Number of Matches	3
3D Textured Mesh Generation	yes
3D Textured Mesh Settings:	Resolution: Medium Resolution (default) Color Balancing: no
LOD	Generated: no
Advanced: 3D Textured Mesh Settings	Sample Density Divider: 1

Advanced: Image Groups	group1
Advanced: Use Processing Area	yes
Advanced: Use Annotations	yes
Time for Point Cloud Densification	16m:07s
Time for Point Cloud Classification	05m:49s
Time for 3D Textured Mesh Generation	08m:01s

Results

Number of Generated Tiles	1
Number of 3D Densified Points	13665027
Average Density (per ft^3)	8.33

DSM, Orthomosaic and Index Details

Processing Options

DSM and Orthomosaic Resolution	1 x GSD (2.09 [cm/pixel])
DSM Filters	Noise Filtering: yes Surface Smoothing: yes, Type: Medium
Raster DSM	Generated: yes Method: Inverse Distance Weighting Merge Tiles: yes
Orthomosaic	Generated: yes Merge Tiles: yes GeoTIFF Without Transparency: no Google Maps Tiles and KML: no
Raster DTM	Generated: yes Merge Tiles: yes
DTM Resolution	5 x GSD (2.09 [cm/pixel])
Time for DSM Generation	09m:18s
Time for Orthomosaic Generation	21m:04s
Time for DTM Generation	18m:51s
Time for Contour Lines Generation	00s
Time for Reflectance Map Generation	00s
Time for Index Map Generation	00s

First lets define all of the input parameters required for this to work!

In [139…

```
##### INPUTS #####
imagery_file_path = r'path/to.your/multispectral/imagery.tif'
dsm_file_path = r'path/to/your/digital/surface/model.tif'
be_threshold = 17000
shadow_threshold = 7000
elev_threshold = 2605
wetted_area_min_size = 1
wetted_area_buffer = 1
largest_poly_only = True
valley_bottom_buffer = 100
rel_elev_grass = 2
rel_elev_shrub = 5
in_channel_wood = True
```

Import all of the python modules we will be using. Rasterio and numpy are for images and arrays (raster data). Pandas, geopandas and shapely are for vector geometries. Matplotlib is for viewing and plotting the data.

In [140…

```
import rasterio as rio
from rasterio import features
from rasterio.mask import mask
import numpy as np

import pandas as pd
import geopandas as gpd
from shapely.geometry import Polygon,Point,shape

import matplotlib.pyplot as plt
```

Then we can read all of the data from our UAS imagery. We can create an array for all bands and a separate array for each band.

In [141…

```
# read raster bands as arrays
with rio.open(imagery_file_path) as src:
    all_bands = src.read().astype(float)
    r = all_bands[0,:,:]
    g = all_bands[1,:,:]
    b = all_bands[2,:,:]
    nir = all_bands[3,:,:]
    re = all_bands[4,:,:]

    transform = src.transform
    crs = src.crs
    profile = src.profile.copy()
    og_profile = src.profile.copy()
```

Now lets begin with some simple classification techniques to identify the wetted area. For classifying water, we will use NDWI. We can filter the NDWI by areas classified as vegetation using the NDVI and areas of bare earth using a high reflectance threshold across all bands. We can also use the DSM generated from drone imagery to exclude the uplands.

In [142…

```
be = (all_bands>be_threshold).all(axis=0)
shadows = (all_bands<shadow_threshold).all(axis=0)


ndwi = (g - nir)/(g + nir)

ndvi = (nir - r)/(nir+r)

with rio.open(dsm_file_path) as src:
    dsm = src.read(1)

dsm = np.where(dsm==0,np.nan,dsm)
```

```
C:\Users\lrussell\AppData\Local\Temp\ipykernel_12868\1193667893.py:5: RuntimeWarning: in
valid value encountered in divide
  ndwi = (g - nir)/(g + nir)
C:\Users\lrussell\AppData\Local\Temp\ipykernel_12868\1193667893.py:7: RuntimeWarning: in
valid value encountered in divide
  ndvi = (nir - r)/(nir+r)
```

The final NDWI looks for cells where the NDWI is greater than 0, the NDVI is less than zero, the DSM is below 2712 feet, and the cell is not marked as bare earth.

In [143…

```
ndwi = np.where((ndwi>0) & (dsm<elev_threshold) & (ndvi<0) & (~shadows) & (~be),1,0)
```

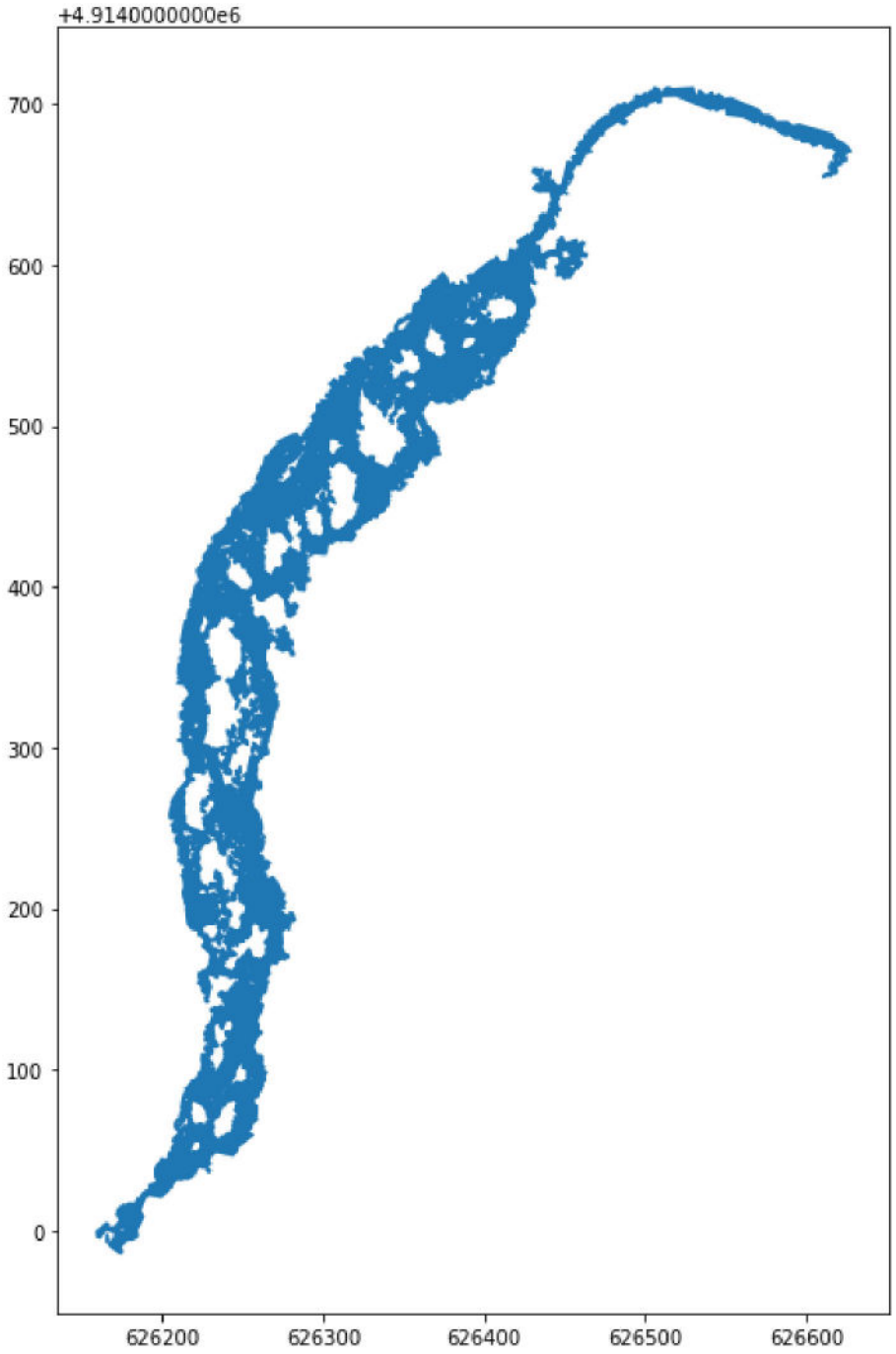
From this, we can extract the geometry of the wetted area. We can filter out tiny areas and single pixels for computational efficiency. Once we have all the polygons we can simplify the geometry (again for computational efficiency). Finally, we buffer and un-buffer the joined polygons. This last step is a neat trick to remove many of the small holes in the wetted area and merges areas that were separated by small gaps in the NDWI classification. This also fixes any topology errors of self-intersecting polygons.

In [144…

```
# extract all areas classified as water with area greater than 50 sqft
polys = []
vals = []
for shapedict, value in features.shapes(ndwi,transform=transform):
    polys.append(shape(shapedict))
    vals.append(value)
wetted_area = gpd.GeoDataFrame(vals,columns=['vals'],geometry=polys,crs=crs)
wetted_area = wetted_area[wetted_area.vals==1]
wetted_area = wetted_area[wetted_area.geometry.area>wetted_area_min_size]
wetted_area.geometry = wetted_area.geometry.simplify(1)
wetted_area.geometry = wetted_area.buffer(wetted_area_buffer)
wetted_area = wetted_area.dissolve().explode()
if largest_poly_only:
    wetted_area = wetted_area[wetted_area.geometry.area == wetted_area.geometry.area.ma

wa_mask = features.geometry_mask(wetted_area.geometry.values,r.shape,transform,invert=T

fig,ax = plt.subplots(1,1,figsize=(16,12))
wetted_area.plot(ax=ax)
```

Out[144…

```
<AxesSubplot:>
```



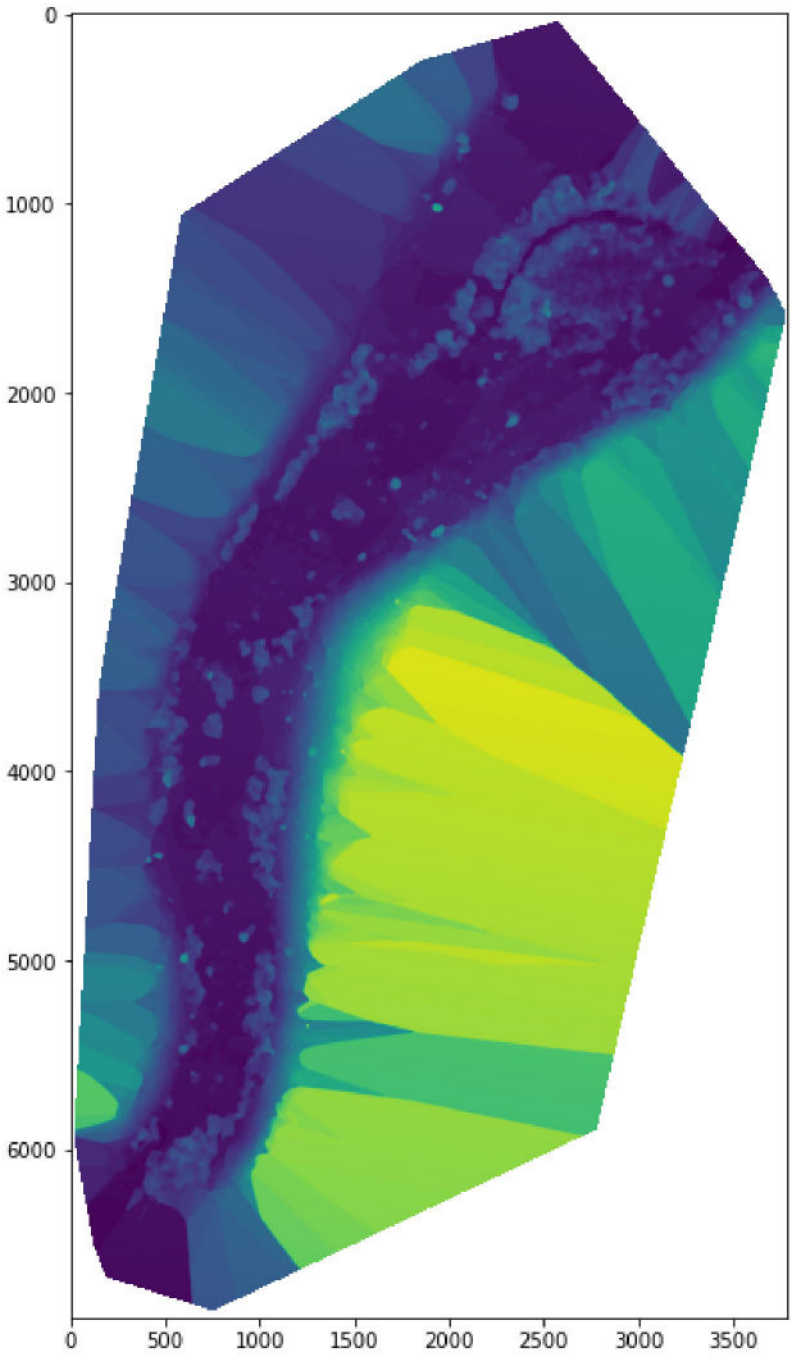
Now lets sample points wihin the wetted area to create a REM from the DSM collected by the UAS. We will use this to differentiate between different vegetation classes. Tracking the distribution of relative elevations may be a useful monitoring metric all on it's own (see the histogram below)!

In [145…

```
def random_point_in_shp(shp):
    within = False
    while not within:
        x = np.random.uniform(shp.bounds[0], shp.bounds[2])
        y = np.random.uniform(shp.bounds[1], shp.bounds[3])
        within = shp.contains(Point(x, y))

    r,c = rio.transform.rowcol(transform,x,y)
    elev = dsm[r,c]
    return [elev,r,c]
```

```
points = pd.DataFrame([],columns=['elev','r','c'])
for i in range(25):
    points.loc[len(points)] = random_point_in_shp(wetted_area.geometry.unary_union)
points['temp'] = 1
points = points.astype(float)

ci,ri = np.meshgrid(np.arange(dsm.shape[1]), np.arange(dsm.shape[0]))

y = points.elev
X = points[['r','c','temp']]
fit = np.linalg.lstsq(X, y, rcond=None)[0]
plane = fit[0]*ri + fit[1]*ci + fit[2]
rem = dsm - plane

dsm_mask = np.where(dsm==9999,1,0)
fig,ax = plt.subplots(1,1,figsize=(16,12))
ax.imshow(np.where(dsm_mask,np.nan,rem))
```

Out[145… <matplotlib.image.AxesImage at 0x2c32d76d640>



Let's extract the valley bottom from the REM to remove the uplands from the image. We aren't as interested in changes there and the varying elevation messes with the REM thresholds used to define vegetation classes. This time we will use the buffer trick but with a buffer distance large enough to remove all holes (from canopied areas) in the valley bottom delineation.

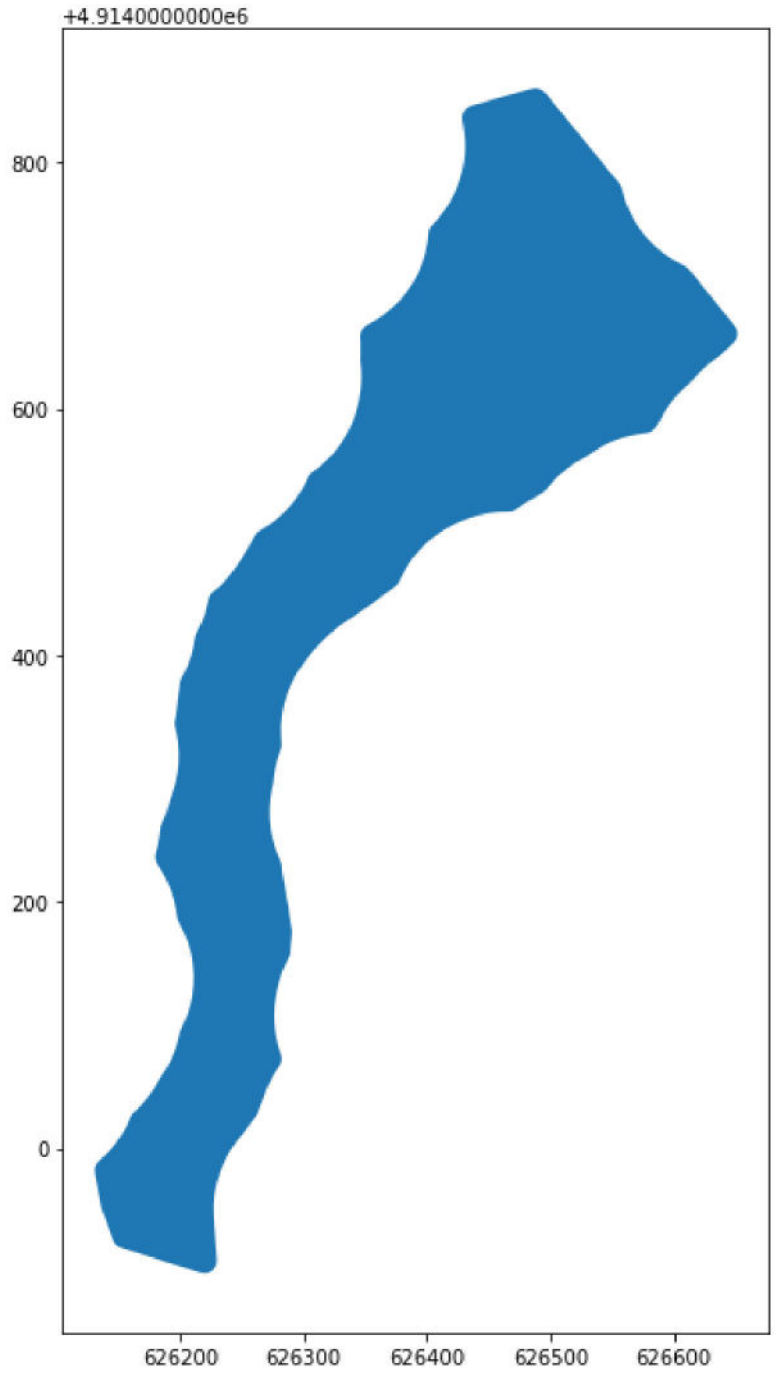
In [146…

```
vb = np.where(rem<2,1,0)

polys=[]
for shapedict, value in features.shapes(vb,transform=transform):
    if value == 1:
        polys.append(shape(shapedict).buffer(0))

vb = gpd.GeoDataFrame([],geometry=polys,crs=crs)
vb = gpd.GeoDataFrame([],geometry=[vb.geometry.unary_union],crs=crs)
vb.geometry = vb.geometry.simplify(1)
```

```
vb.geometry = vb.geometry.buffer(valley_bottom_buffer).buffer(valley_bottom_buffer*-0.9

fig,ax = plt.subplots(1,1,figsize=(16,12))
vb.plot(ax=ax)
```

Out[146…

```
<AxesSubplot:>
```



We can create a mask from the valley bottom polygon an clip all our rasters!

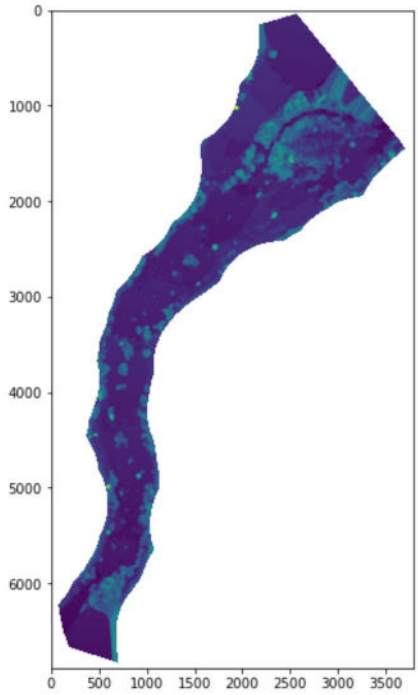
In [147…

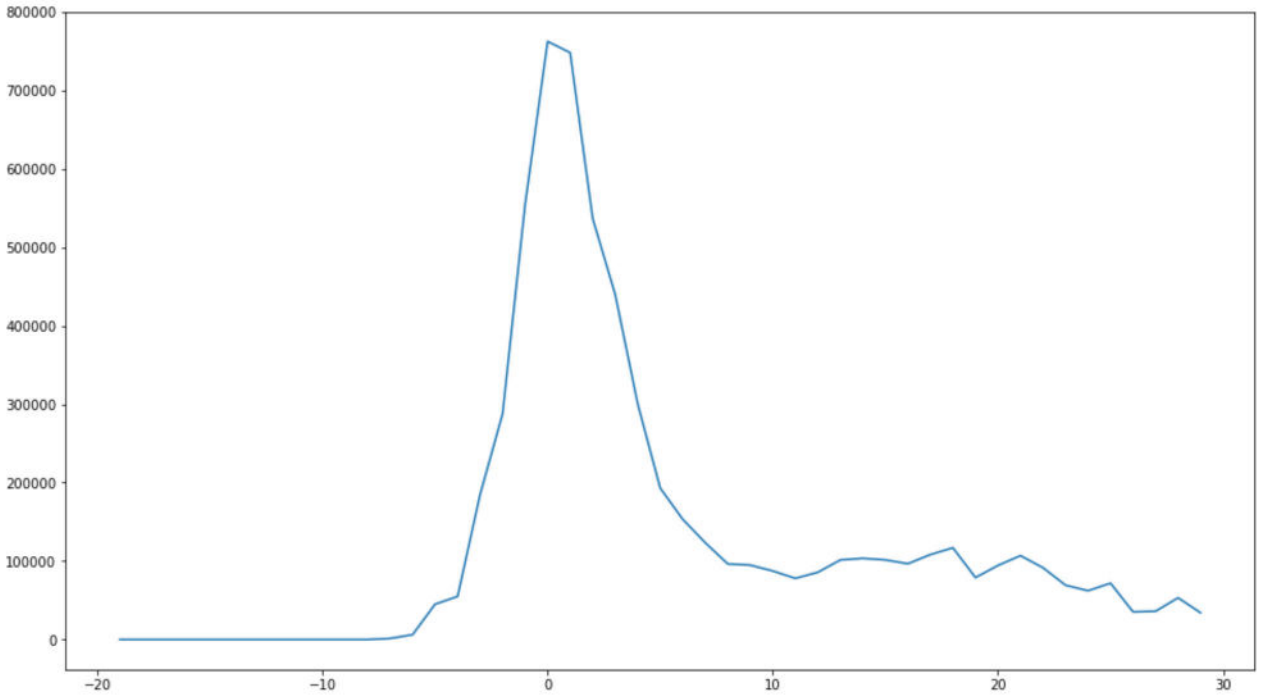
```
vb_mask = features.geometry_mask(vb.geometry.values,dsm.shape,transform,invert=True)

r = np.where(vb_mask,r,np.nan)
g = np.where(vb_mask,g,np.nan)
b = np.where(vb_mask,b,np.nan)
nir = np.where(vb_mask,nir,np.nan)
re = np.where(vb_mask,re,np.nan)
```

```
rem = np.where(vb_mask,rem,np.nan)

fig,ax = plt.subplots(2,1,figsize=(16,20))
ax[0].imshow(rem)
hist = np.histogram(rem[~np.isnan(rem)],bins=np.arange(-20,30,1))
ax[1].plot(hist[1][1:],hist[0])
```

Out[147…

```
[<matplotlib.lines.Line2D at 0x2c34bb07910>]
```



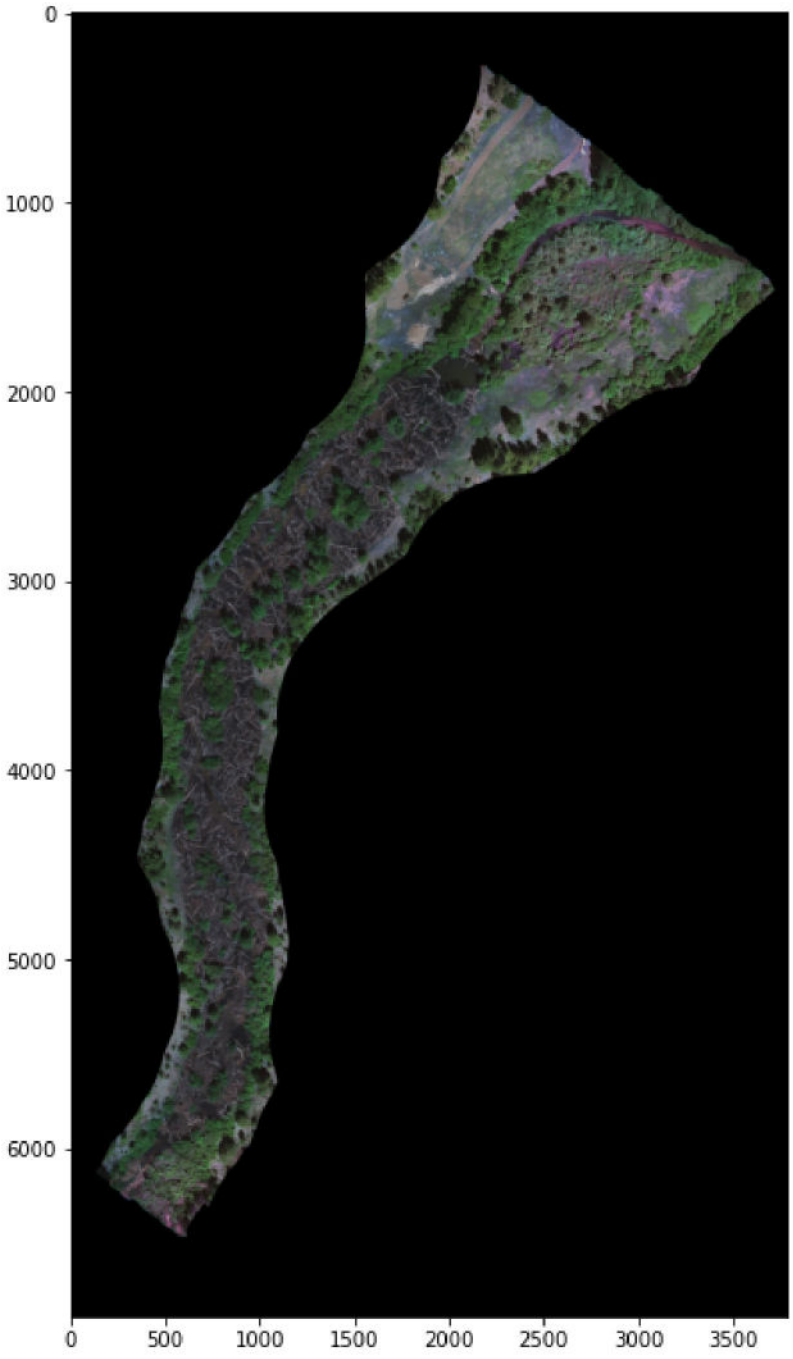


In [148…

```
fig,ax = plt.subplots(1,1,figsize=(16,12))
ax.imshow((np.stack([r,g,b],axis=-1)//256).astype(int))
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or
[0..255] for integers).
```

Out[148… <matplotlib.image.AxesImage at 0x2c31ef9bf70>



Now let's move on to vegetation. We can look at three commonly used vegetation indices to see how we might combine all three. We can see that the NDVI and MSAVI are similar, although MSAVI shows more contrast. The EVI shows more variation and noise, but isolates canopied area best. I decided to use the REM over the EVI for identifying canopied areas due to the presence of noise in the EVI and wide range of values making it difficult to consistently set meaningful threshold.

In [149…

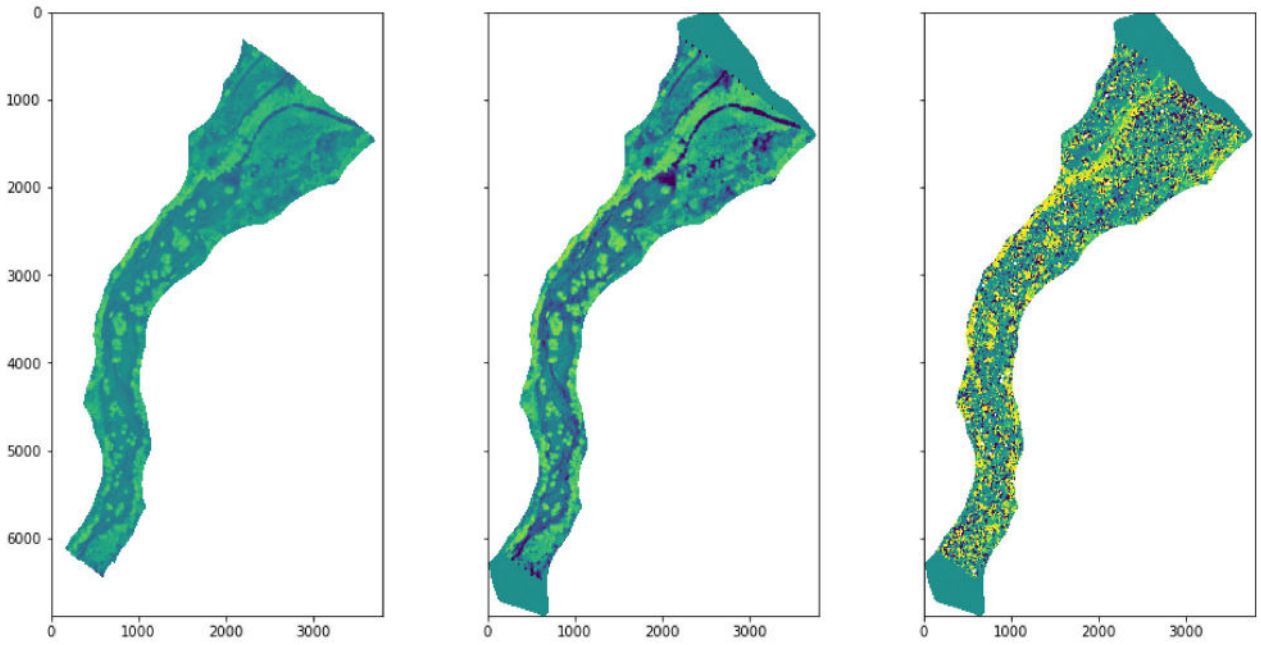
```
ndvi = (nir - r)/(nir+r)

evi = 2.5*((nir - r)/(nir + 6*r - 7.5*b + 1))
evi = np.where(np.isinf(evi),np.nan,evi)
```

```
msavi = 0.5*((2*nir+1) - np.sqrt((2*nir + 1)**2 - 8*(nir-r)))

fig,ax = plt.subplots(1,3,sharex=True,sharey=True,figsize=(16,8))
ax[0].imshow(ndvi,vmin=-1,vmax=1)
ax[1].imshow(msavi,vmin=-1,vmax=1)
ax[2].imshow(evi,vmin=-2.5,vmax=2.5)
```

```
C:\Users\lrussell\AppData\Local\Temp\ipykernel_12868\3158117950.py:1: RuntimeWarning: in
valid value encountered in divide
  ndvi = (nir - r)/(nir+r)
C:\Users\lrussell\AppData\Local\Temp\ipykernel_12868\3158117950.py:3: RuntimeWarning: di
vide by zero encountered in divide
  evi = 2.5*((nir - r)/(nir + 6*r - 7.5*b + 1))
```

Out[149…

```
<matplotlib.image.AxesImage at 0x2c31aeaafa0>
```

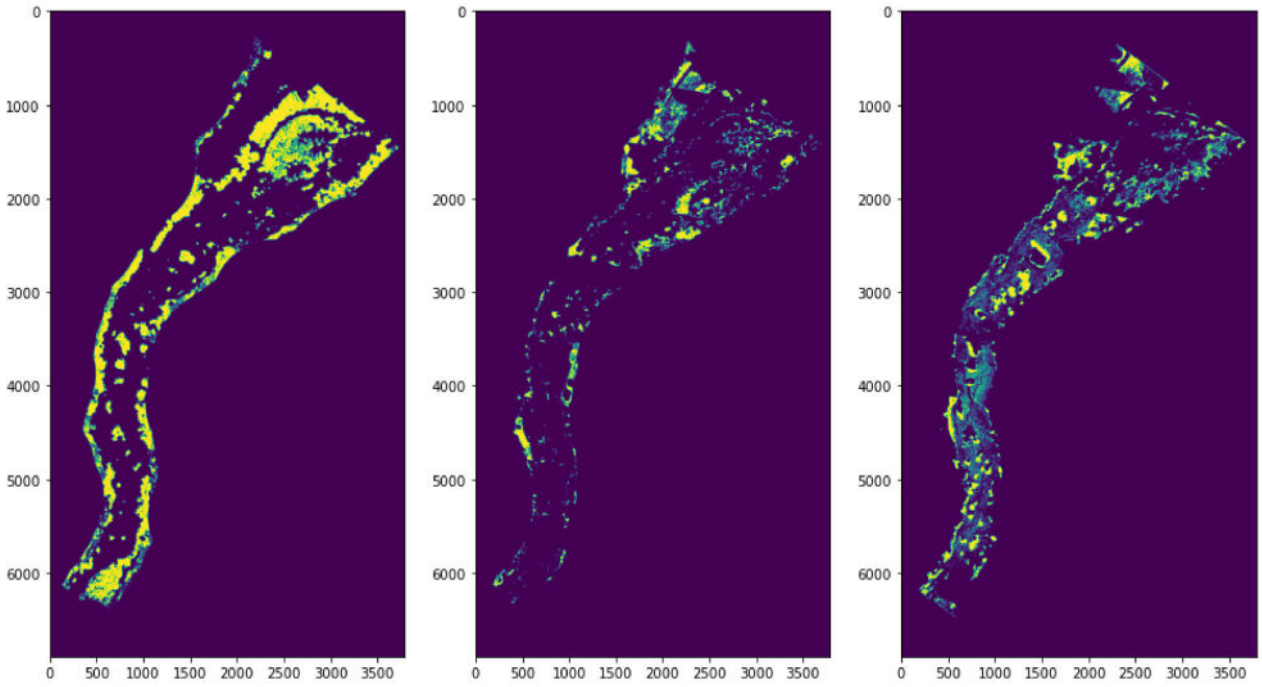


In [150…

```
canopy = ((ndvi>0) & (msavi>0) & (rem>5) & (vb_mask))
shrub = ((ndvi>0) & (msavi>0) & (rem<5) & (rem>2) & (vb_mask))
grasses = ((ndvi>0) & (msavi>0) & (rem<2) & (vb_mask))

fig,ax = plt.subplots(1,3,figsize=(16,12))
ax[0].imshow(canopy)
ax[1].imshow(shrub)
ax[2].imshow(grasses)
```

Out[150…

```
<matplotlib.image.AxesImage at 0x2c302212a90>
```



In [151…

```
final = np.full(dsm.shape,np.nan)
final = np.where((canopy==1) & vb_mask,5,final)
final = np.where((shrub==1) & vb_mask,4,final)
final = np.where((grasses==1) & vb_mask,3,final)
final = np.where((wa_mask) & vb_mask,2,final)
final = np.where(vb_mask & shadows & np.isnan(final), 0, final)
final = np.where(np.isnan(final),0,final)
```

In [152…

```
if in_channel_wood:
    final = np.where(wa_mask & be,1,final)
```

In [153…

```
final_path = imagery_file_path.split('.')[0]+'_classified.tif'

profile['count'] = 1
profile['nodata'] = 0
with rio.open(final_path,'w',**profile) as dst:
    dst.write(final,indexes=1)
```

In [ ]: