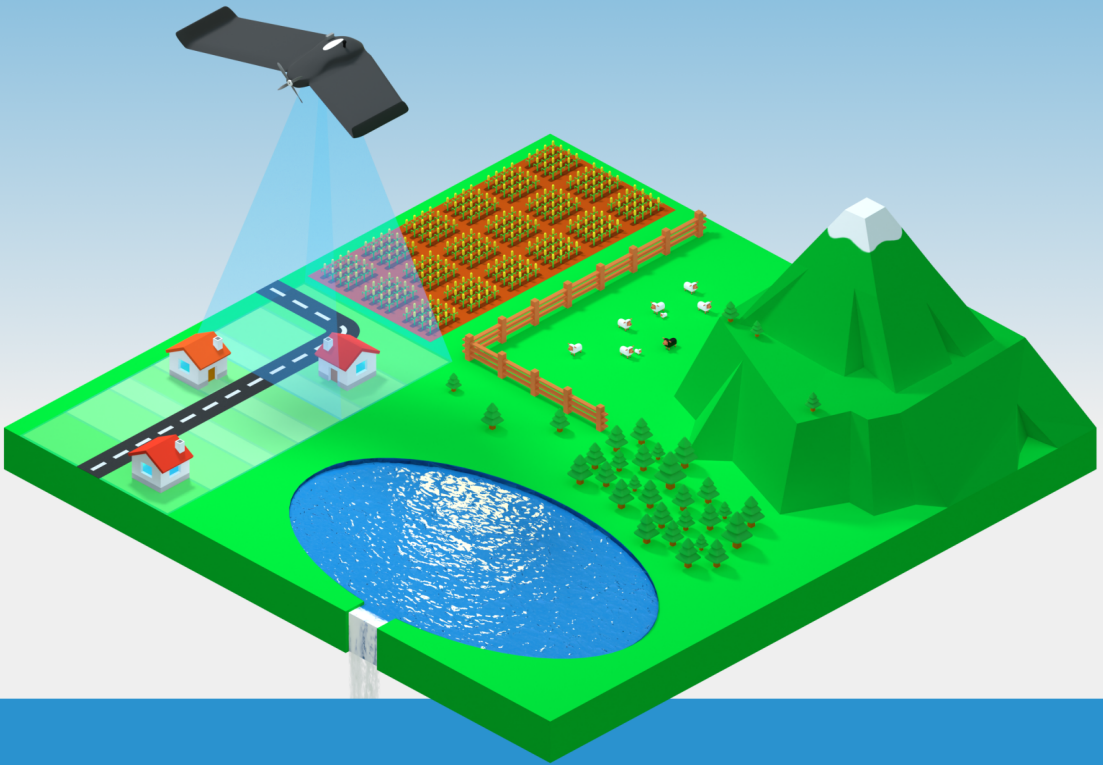


**A Practical Guide to Drone Mapping  
Using Free and Open Source Software**



# **WebODM**

## **The Missing Guide**

Third Edition

Piero Toffanin

# Contents

<b>Copyright</b>	<b>i</b>
<b>Epigraph</b>	<b>ii</b>
<b>Preface</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>vi</b>
<b>I. Introduction</b>	<b>1</b>
Why WebODM? . . . . .	2
What You Can Do With WebODM . . . . .	3
Becoming a Successful User . . . . .	5
<b>II. Getting Started</b>	<b>7</b>
<b>1. The WebODM Ecosystem</b>	<b>8</b>
<b>2. Installing The Software</b>	<b>10</b>
Hardware Requirements . . . . .	11
Installing on Windows . . . . .	12
Installing on macOS . . . . .	13
Installing on Linux . . . . .	16

## Contents

Basic Linux Commands . . . . .	18
Hello, WebODM! . . . . .	20
<b>3. Processing Datasets</b>	<b>22</b>
Dataset Size . . . . .	22
File Requirements . . . . .	23
Process Tasks . . . . .	24
Testing Different Task Options . . . . .	28
<b>4. Essential Operations</b>	<b>30</b>
View Results . . . . .	30
Download the Raw Assets . . . . .	32
Share Results With Others . . . . .	33
Read the Quality Report . . . . .	34
View Map Layers Side by Side . . . . .	34
Crop Results . . . . .	35
Make Measurements . . . . .	36
Generate Contours . . . . .	37
Add Media Files . . . . .	37
Import External Assets . . . . .	40
Export Tasks To Another WebODM . . . . .	40
Manage Plugins . . . . .	40
Change The Look and Feel . . . . .	41
Customize the Basemaps . . . . .	41
Create New Users & Groups . . . . .	41
Project Permissions . . . . .	42
Managing Tags . . . . .	43
<b>5. The Processing Pipeline</b>	<b>45</b>
Load Dataset . . . . .	46

## Contents

Structure From Motion . . . . .	46
Multi View Stereo . . . . .	50
Point Filtering . . . . .	51
Meshing . . . . .	51
Texturing . . . . .	54
Georeferencing . . . . .	55
Digital Elevation Model Processing . . . . .	56
Orthophoto Processing . . . . .	58
Report Generation . . . . .	60
Post Processing . . . . .	60
<b>6. Task Options in Depth</b>	<b>61</b>
3d-tiles . . . . .	63
align . . . . .	64
auto-boundary . . . . .	68
auto-boundary-distance . . . . .	68
bg-removal . . . . .	68
boundary . . . . .	70
build-overviews . . . . .	71
camera-lens . . . . .	72
cameras . . . . .	75
cog . . . . .	75
copy-to . . . . .	76
crop . . . . .	76
dem-decimation . . . . .	77
dem-euclidean-map . . . . .	78
dem-gapfill-steps . . . . .	79
dem-resolution . . . . .	81
dsm . . . . .	82

## Contents

dtm . . . . .	82
end-with . . . . .	83
fast-orthophoto . . . . .	84
feature-quality . . . . .	88
feature-type . . . . .	89
force-gps . . . . .	89
gcp . . . . .	90
geo . . . . .	90
gltf . . . . .	91
gps-accuracy . . . . .	91
gps-z-offset . . . . .	92
help . . . . .	92
ignore-gsd . . . . .	93
matcher-neighbors . . . . .	94
matcher-order . . . . .	96
matcher-type . . . . .	96
max-concurrency . . . . .	98
merge . . . . .	98
mesh-octree-depth . . . . .	98
mesh-size . . . . .	101
min-num-features . . . . .	102
no-gpu . . . . .	105
optimize-disk-space . . . . .	105
orthophoto-compression . . . . .	105
orthophoto-cutline . . . . .	106
orthophoto-kmz . . . . .	108
orthophoto-no-tiled . . . . .	108
orthophoto-png . . . . .	109
orthophoto-resolution . . . . .	110

*Contents*

pc-classify . . . . .	110
pc-copc . . . . .	116
pc-csv . . . . .	116
pc-ept . . . . .	117
pc-filter . . . . .	117
pc-las . . . . .	118
pc-quality . . . . .	118
pc-sample . . . . .	121
pc-skip-geometric . . . . .	121
primary-band . . . . .	122
project-path . . . . .	122
radiometric-calibration . . . . .	122
rerun . . . . .	123
rerun-all . . . . .	123
rerun-from . . . . .	123
rolling-shutter . . . . .	125
rolling-shutter-readout . . . . .	125
sfm-algorithm . . . . .	125
sfm-no-partial . . . . .	126
skip-3dmodel . . . . .	126
skip-band-alignment . . . . .	128
skip-orthophoto . . . . .	128
skip-report . . . . .	128
sky-removal . . . . .	128
sm-cluster . . . . .	130
sm-no-align . . . . .	130
smrf-scalar . . . . .	130
smrf-slope . . . . .	130
smrf-threshold . . . . .	131

## Contents

smrf-window . . . . .	131
split . . . . .	131
split-image-groups . . . . .	131
split-overlap . . . . .	131
texturing-keep-unseen-faces . . . . .	132
texturing-single-material . . . . .	134
texturing-skip-global-seam-leveling . . . . .	134
tiles . . . . .	136
use-3dmesh . . . . .	136
use-exif . . . . .	137
use-fixed-camera-params . . . . .	137
use-hybrid-bundle-adjustment . . . . .	138
version . . . . .	138
video-limit . . . . .	139
video-resolution . . . . .	139
Changing Options and Restarting . . . . .	139
<b>7. Ground Control Points</b>	<b>145</b>
Marking Checkpoints . . . . .	149
Creating a GCP file using POSM GCPi . . . . .	149
Creating a GCP file using GCP Editor Pro . . . . .	154
Using GCP files . . . . .	154
How GCP files work . . . . .	156
Common Mistakes . . . . .	157
<b>8. Geolocation Files</b>	<b>158</b>
Using GEO files . . . . .	161
<b>9. Multispectral Datasets</b>	<b>162</b>
Supported Images . . . . .	162

## Contents

Processing . . . . .	164
Radiometric Calibration Basics . . . . .	167
Viewing Results in WebODM . . . . .	170
Thermal Datasets . . . . .	172
<b>10. Image Masks</b>	<b>174</b>
Manually Creating Image Masks . . . . .	175
<b>11. Rolling Shutter Correction</b>	<b>178</b>
Usage . . . . .	179
Limitations . . . . .	182
<b>12. Camera Calibration</b>	<b>183</b>
<b>13. Report Analysis</b>	<b>188</b>
Dataset Summary . . . . .	188
Processing Summary . . . . .	189
Previews . . . . .	190
Survey Data . . . . .	190
GPS/GCP/3D Errors Details . . . . .	191
Feature Details . . . . .	196
Reconstruction Details . . . . .	198
Track Details . . . . .	201
Camera Models Details . . . . .	201
JSON output . . . . .	203
<b>14. Flying Tips</b>	<b>204</b>
Fly Higher . . . . .	204
Fly on Overcast Days . . . . .	205
Fly Between 10am and 2pm . . . . .	205
Fly at Different Elevations and Capture Multiple Angles . . . . .	205

## Contents

Fly on Calm Days . . . . .	206
Increase Overlap . . . . .	207
Set Drone to Hover While Taking Images . . . . .	207
Check Camera Settings . . . . .	208
<b>III. Advanced Usages</b>	<b>209</b>
<b>15. The Command Line</b>	<b>210</b>
Installation . . . . .	211
Command Line Basics . . . . .	211
Using ODX . . . . .	214
Processed Files Owned By Root . . . . .	215
Add New Processing Nodes to WebODM . . . . .	216
Examine EXIF/XMP Tags . . . . .	217
Further Readings . . . . .	218
<b>16. Docker Essentials</b>	<b>219</b>
Docker Basics . . . . .	219
Managing Containers . . . . .	221
Managing Images . . . . .	224
Managing Volumes . . . . .	225
Docker Compose Basics . . . . .	227
Managing Disk Space . . . . .	229
Changing Entrypoint . . . . .	230
Assigning Names To Containers . . . . .	230
Jumping Into Existing Containers . . . . .	231
<b>17. GPU Processing</b>	<b>232</b>
Windows . . . . .	232

## Contents

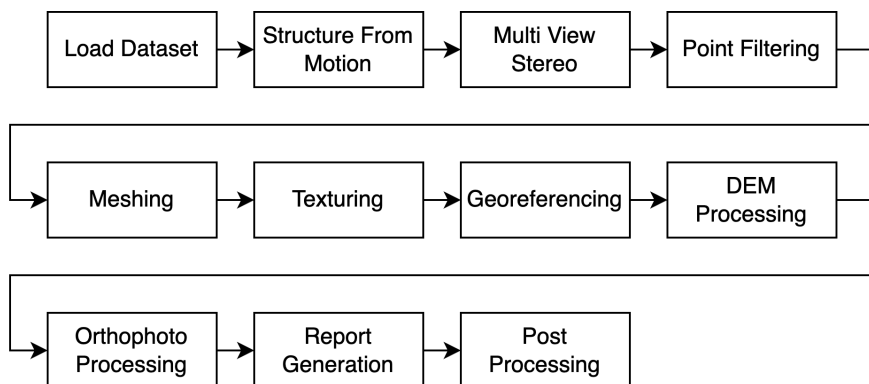
Linux . . . . .	233
Notes on GPU usage . . . . .	234
<b>18. Processing Large Datasets</b>	<b>235</b>
Split-Merge Options . . . . .	237
Local Split-Merge . . . . .	239
Distributed Split-Merge . . . . .	241
Using Image Groups and GCPs . . . . .	245
Limitations . . . . .	246
<b>19. The NodeODX API</b>	<b>248</b>
Launching a NodeODX Instance . . . . .	249
NodeODX Configuration . . . . .	250
Using the API with cURL . . . . .	252
Remove a Task . . . . .	254
API Specification . . . . .	255
Definitions . . . . .	275
Exercises . . . . .	276
<b>20. Automated Processing With Python</b>	<b>278</b>
Getting Started . . . . .	279
Example 1: Hello NodeODX . . . . .	280
Example 2: Process Datasets . . . . .	281
Concluding Remarks . . . . .	284
API Reference . . . . .	284
<b>Glossary</b>	<b>292</b>
<b>About The Author</b>	<b>296</b>

## 5. The Processing Pipeline

“Any sufficiently advanced technology is indistinguishable from magic.” -  
*Arthur C. Clarke*

If this was the first time you’ve used a software like WebODM, or if you can remember the first time you used it, it’s likely that at some point you had a **woah!** moment: how could a computer take simple 2D images and turn them into georeferenced mosaics, 3D models and point clouds?

Going from images to 3D models and orthophotos is a process best visualized as a series of incremental steps. Each step relies on the work of previous steps.



WebODM’s processing pipeline

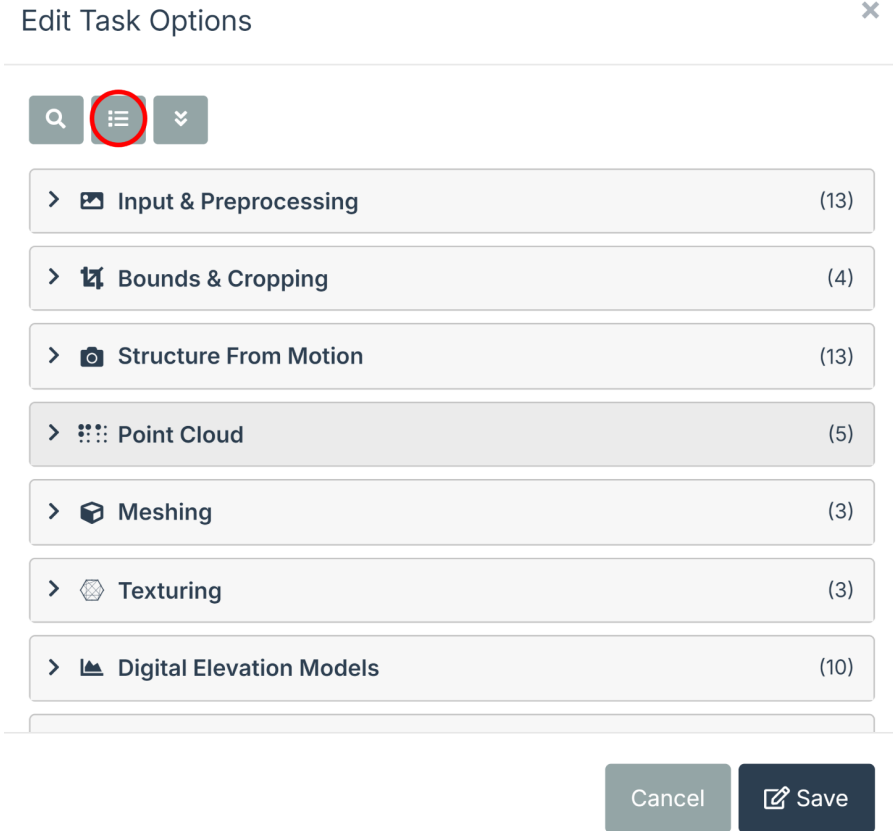
In this chapter we will explore an overview of the pipeline. We will not cover

## 6. Task Options in Depth

There are several steps involved in the data processing pipeline. Each step has several adjustable settings that influence the output. The software exposes a subset of these available knobs through various options. When creating a task, a user can choose to tweak one or more options to change the behavior of the pipeline.

Options by default are displayed by category, but they can also be displayed alphabetically. To do that, click the list view button from the top right side of the dialog.

## 6. Task Options in Depth



Tuning options is more art than science. That's mostly because the best options for certain datasets do not automatically transfer to others. As a general rule, one should start with the defaults, which work fairly well for most datasets and apply tweaks as needed.

A quick note about large language models (LLMs) is warranted here. Many people are turning to these tools for help in picking good processing settings. I

## 6. Task Options in Depth

must warn that as of 2026 LLMs are *pretty bad* at giving advice when it comes to tuning WebODM options, while sounding very confident in doing so<sup>1</sup>.

This chapter is about understanding in detail what each option does. By the end of the chapter you'll be able to quickly improve your results, explain why certain models turn out the way they do and know what to tweak if the results don't turn out the way you want.

A few of these options might be hidden in WebODM and available only from ODX. This is because sometimes the option does not make sense in the context of the graphic interface workflow, or it's simply not supported.

When there is some math to explain, I write the formulas using Python because it's easier than math notation and can be typed on a computer. You can copy/paste the code on a website such as [online-python.com](https://online-python.com) and follow along even if you don't know Python.

Feel free to jump around and use this chapter as a reference. As the software gets better, some of these options might disappear from future versions and new ones might be introduced. The list below is taken from the software as of April 28th 2026. In alphabetical order:

### 3d-tiles

OGC 3D Tiles<sup>2</sup> are a format specification for visualization and interaction with 3D geospatial content. These files can be displayed with software such as the open source virtual globe engine Cesium<sup>3</sup>. ODX has support for generating point clouds and textured 3D models in 3D Tiles format by turning on this option (and doesn't rely on third party services to do so).

---

<sup>1</sup>The bs machines: [thebullshitmachines.com](https://thebullshitmachines.com)

<sup>2</sup>OGC 3D Tiles: [ogc.org/standard/3dtiles/](https://ogc.org/standard/3dtiles/)

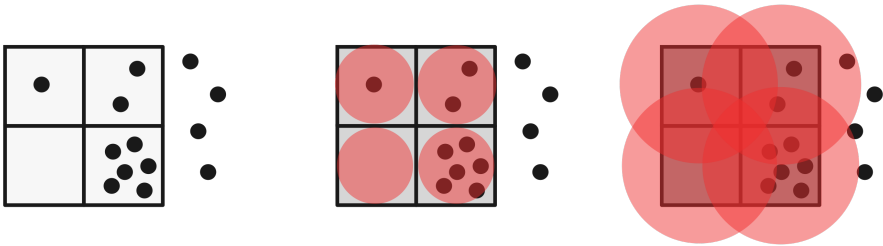
<sup>3</sup>Cesium: [github.com/CesiumGS/cesium](https://github.com/CesiumGS/cesium)

## 6. Task Options in Depth

Euclidean map results are stored in the *odm\_dem* directory.

### dem-gapfill-steps

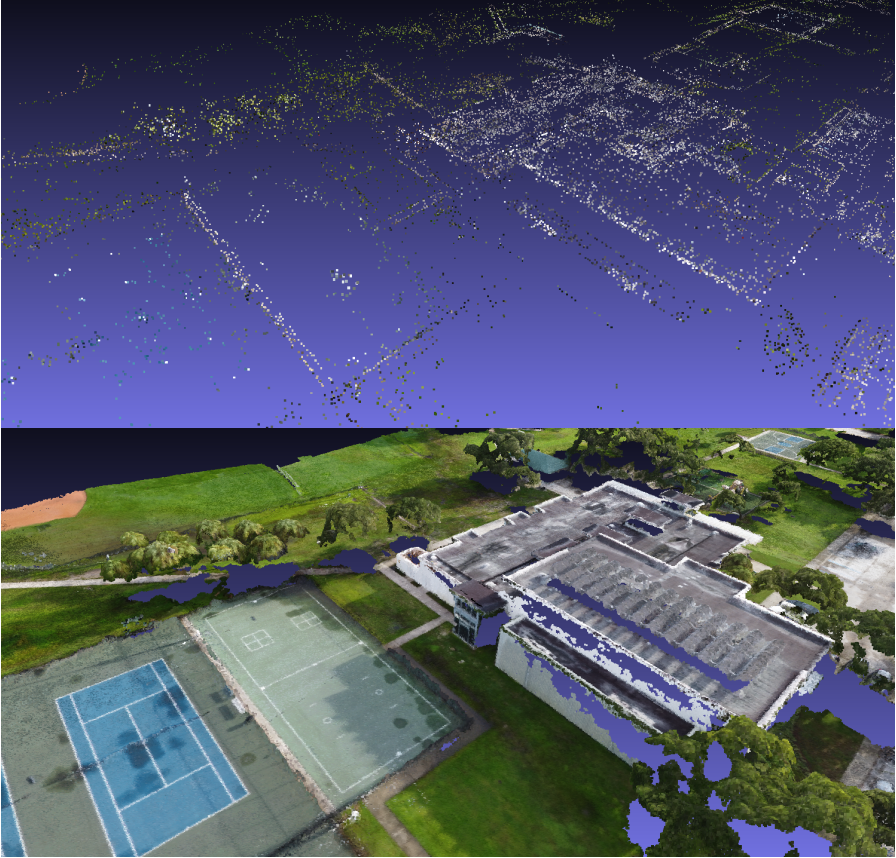
The process of going from point cloud to DEMs is not as straightforward as it may seem. Since DEMs are *rasters* (images), they have *cells* (pixels). Each cell should have a value. Depending on the resolution of the raster, certain cells may have zero, one or more points that fall into it. Every cell needs a value, even if no points fall directly into it, otherwise there will be empty areas (gaps) in the DEM! One way to overcome this is to use a radius around each cell. Every point that falls within the radius is considered part of the cell.



Pixels and points (left), radius of 0.5 (middle) and radius of 1 (right)

But how big should the radius be? If too small, as in the 0.5 radius example above, some cells might remain empty. If too big, there will be too much smoothing and accuracy will suffer. Since different point clouds have varying degrees of density, one solution is to compute multiple DEMs with different radiuses and stack them.

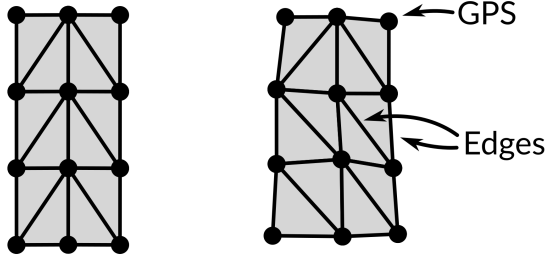
## 6. Task Options in Depth



Sparse (top) vs. dense (bottom) point cloud outputs

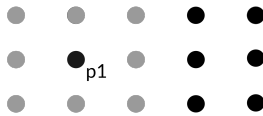
Both point clouds can be used to generate a mesh. However, it's better to have more points, as meshes can be created with more details. In the dense point cloud screenshot above, the building on the right side of the scene is well defined, but it's poorly represented in the sparse point cloud. Buildings are especially difficult to model without a dense point cloud, so this option tends to yield poor results in urban areas. For flat areas such as farmlands, however,

## 6. Task Options in Depth



Initial graph (left) and graph with randomly moved positions and new edges (right). Every edge indicates an image pair

ODX also supports a different method to perform preemptive matching by considering only the nearest neighbors of each image instead of using a graph. It is enabled by setting this option. The illustration below shows the result of setting this option to 8:



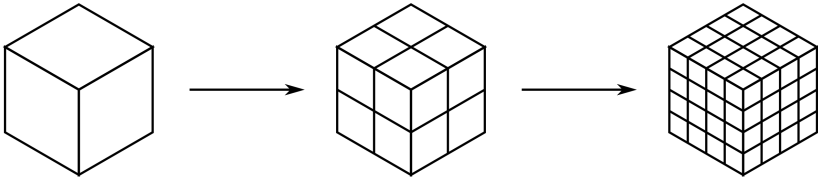
Dots represent approximate image locations, extracted from EXIF tags. When the `matcher-neighbors` is set to 8, only the 8 nearest neighbors (highlighted in gray) are considered for matching with image `p1`

This option can sometimes be beneficial for speeding up processing by reducing the number of matching pairs. If no GPS information is available, this option is disabled and all image pairs are considered, unless `matcher-order` is specified.

## 6. Task Options in Depth

The details of the algorithm are fascinating, but probably outside the scope of this book.

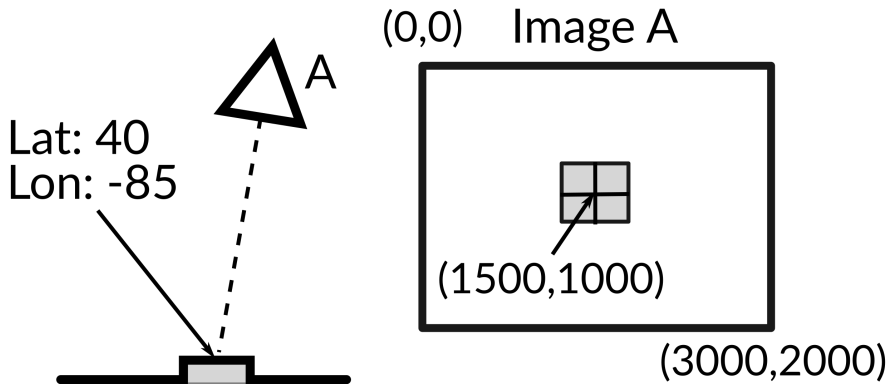
To understand how this option affects the output, it helps to visually understand the concept of an octree. First, octree means *eight-tree* (okta is *eight* in Greek). Why eight? Because at each level (or *depth*) of the tree, each box (or *node or branch*) of the tree is divided in eight parts. At the first level there's only one branch. At the second level there's 8. At the third there's 64 and so forth.



An octree with depth 1, 2 and 3

Lower depths in an octree allow finer details to be captured.

## 7. Ground Control Points



A GCP marker is photographed by camera A to produce Image A. In a second step, the pixel location of the marker (1500,1000) from Image A can be manually tagged with its real world coordinates (latitude 40, longitude -85)

Using ground control points can increase the georeferencing accuracy of a reconstruction, since measurements of static (non-moving) objects using a high precision GPS are often better than those obtained from the GPS of moving UAVs.

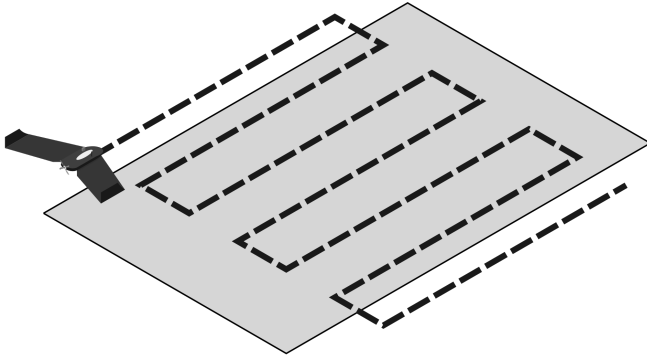
The ideal number of ground control points ranges between 5 to 8, placed evenly across the area to be flown. Adding more than 8 ground control points does not necessarily result in increased accuracy.

If the same marker is visible from multiple images, it should be tagged multiple times for each image. Ideally each marker should be tagged at least 3 times. Another way to think of it is to capture each marker on at least 3 images. This is so that the marker's location can be triangulated during computation.

Ground control points can be used by providing an additional text file along with the input images. The file follows a simple format:

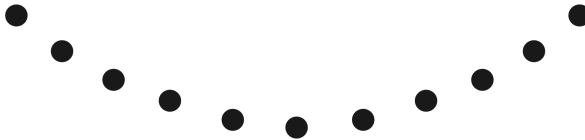
The first line indicates the spatial reference system (SRS) of the world

## 12. Camera Calibration



Typical flight path from mission planning software. Not great for self-calibration

This doesn't mean a person should never fly this pattern. It just means that when flying this pattern, people need to be aware that the internal camera parameters' estimates will not be as good. Inaccurate parameters lead to an improper camera lens model, which over large areas typically results in a *doming* effect.



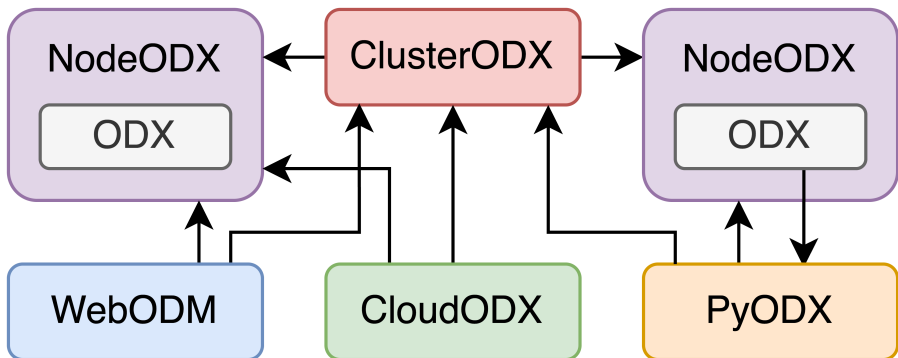
Point cloud exhibiting doming. The terrain appears arched instead of straight

Doming is best cured by following best practices while collecting aerial imagery: flying at different elevations (maximize scale variation) and varying angles.

## 19. The NodeODX API

ODX is a processing engine and WebODM is a friendly user interface. NodeODX<sup>1</sup> was historically built to allow WebODM to communicate with ODX over a network. Today NodeODX has expanded its role and is the glue that binds together many WebODM projects. Each project understands the API that NodeODX defines. When we say *NodeODX* we are referring to the reference implementation of the NodeODX API available at [github.com/WebODM/NodeODX](https://github.com/WebODM/NodeODX).

At its core, the API defines ways to easily create new tasks, manage such tasks (cancel, delete, restart), download results and query status information.



Many WebODM projects use the NodeODX API to communicate with each other

---

<sup>1</sup>Node is a reference to Node.js, the language NodeODX is written in

# Glossary

**2.5D Model:** A model where elevation is simply *extruded* from the ground plane and thus is not a true 3D model.

**Artifacts:** undesired alterations generated as the result of a digital process.

**API:** Application Programming Interface. A set of functions allowing the creation of applications that access the features or data of another application.

**Bundle Adjustment:** a refinement step during the Structure From Motion process that improves the location of cameras, the 3D points of the scene and the camera parameters.

**CloudODX:** A command line tool to process aerial imagery in the cloud.

**ClusterODX:** A NodeODX API compatible autoscalable load balancer and task tracker for connecting multiple NodeODX nodes under a single network address.

**CRS:** Coordinate Reference System. A CRS is a coordinate-based system used to locate geographical entities.

**CSV:** Comma Separated Value is a textual file format where fields are typically separated by commas or some other character such as a space or a tab.

**cURL:** a software providing a library and command-line tool for transferring data using many protocols.

**DEM:** Digital Elevation Model (either a DSM or a DTM).