Dive Reconstruction Pipeline

Michael Nixon

Goals - Original

- Reconstruct scenes filmed while diving
- Demonstrate ORB SLAM capabilities of an underwater drone in Gazebo

Updated

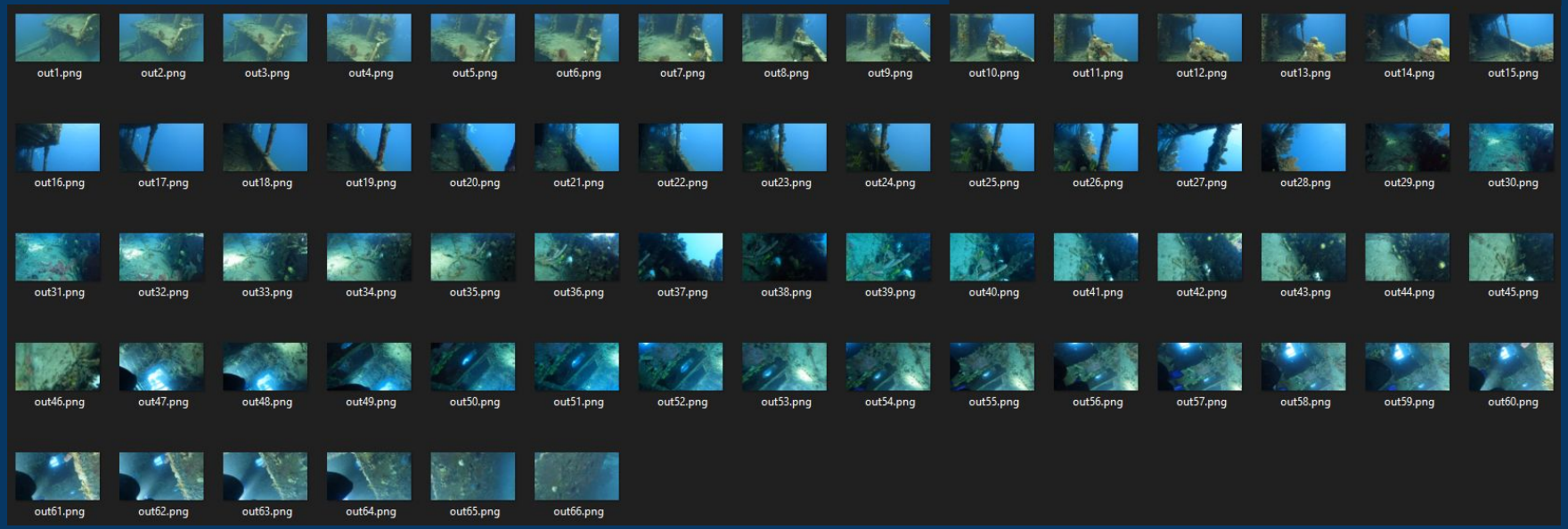
- Reconstruct scenes filmed while diving
- ~~Demonstrate ORB SLAM capabilities of an underwater drone in Gazebo~~
- Implement the Python API for Agisoft to create a reconstruction pipeline for dive footage

Measure of Success: Create an easy to follow and repeatable workflow that produces reasonable 3d models of provided video without intimate knowledge of the programs at work

Methodology - Video Processing

- ffmpeg
  - Split video into series of images
- Metashift Video Import
  - Works for many common video formats (not .MOV)

ffmpeg
- winget install ffmpeg
- ffmpeg -i input.mp4 -vf fps=1 output%d.png



Methodology - Python API

```
>>> import Metashape
>>> doc = Metashape.app.document
>>> doc.open("project.psz")
>>> chunk = doc.chunk
>>> chunk.matchPhotos(downscale=1, generic_preselection=True, reference_
↪preselection=False)
>>> chunk.alignCameras()
>>> chunk.buildDepthMaps(downscale=4, filter=Metashape.AggressiveFiltering)
>>> chunk.buildDenseCloud()
>>> chunk.buildModel(surface_type=Metashape.Arbitrary, interpolation=Metashape.
↪EnabledInterpolation)
>>> chunk.buildUV(mapping=Metashape.GenericMapping)
>>> chunk.buildTexture(blending=Metashape.MosaicBlending, size=4096)
>>> doc.save()
```

```
chunk = Metashape.app.document.addChunk()
chunk.addPhotos(imgFiles)
camera = chunk.cameras[0]
camera.photo.meta["Exif/FocalLength"]
```

```
#image matching and alignment for the active chunk
chunk = Metashape.app.document.chunk
for frame in chunk.frames:
    frame.matchPhotos(downscale=1)
chunk.alignCameras()
```

Notes:

- Works for photo series
- Can utilize multispectral cameras
- No mention of video in the documentation

```
# Code for setting up a multispectral camera
doc = Metashape.app.document
chunk = doc.chunk
rgb = ["RGB_0001.JPG", "RGB_0002.JPG", "RGB_0003.JPG"]
nir = ["NIR_0001.JPG", "NIR_0002.JPG", "NIR_0003.JPG"]
images = [[rgb[0], nir[0]], [rgb[1], nir[1]], [rgb[2], nir[2]]]
chunk.addPhotos(images, Metashape.MultiplaneLayout)
```

Legally requires a Pro License for Metashape

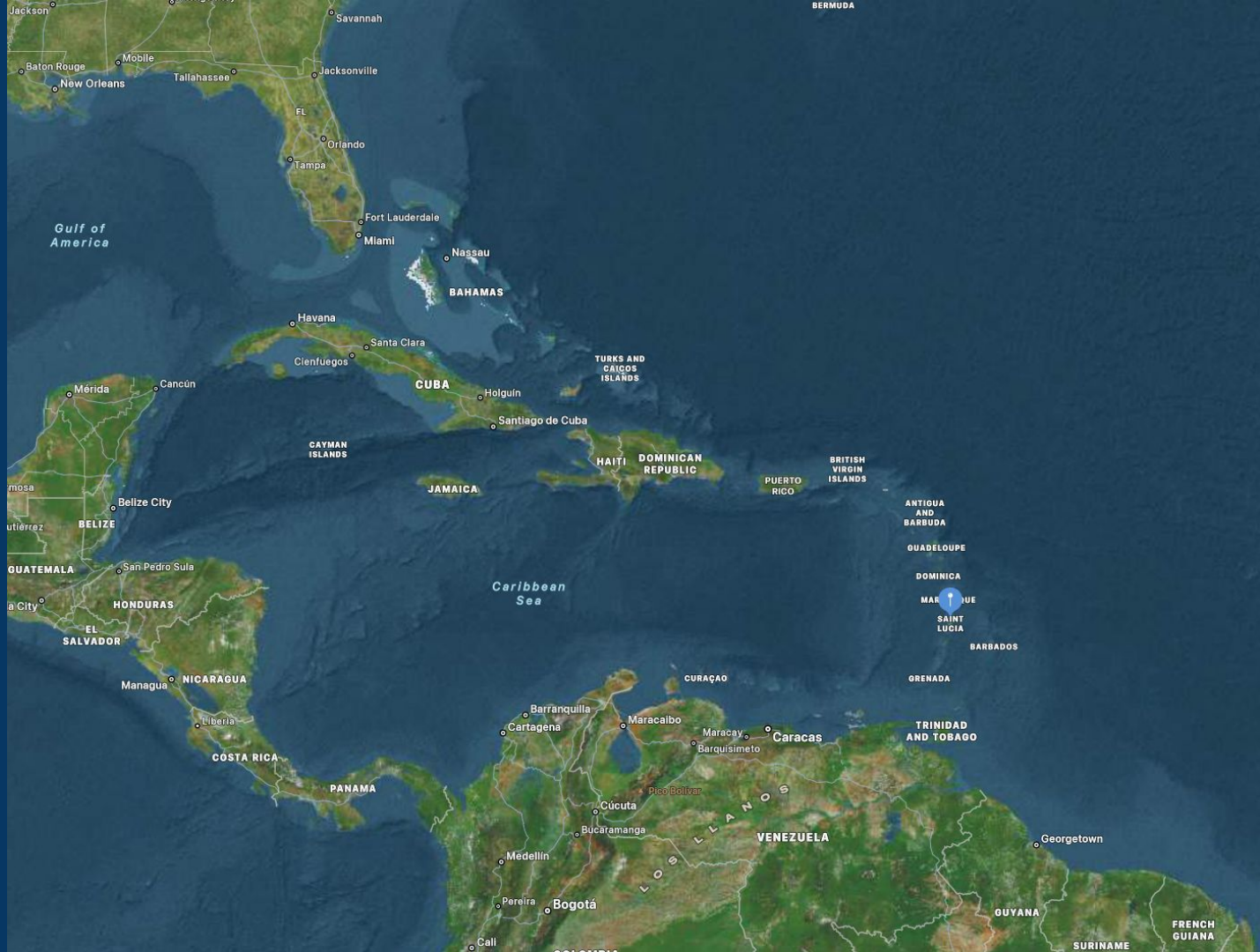
Metashape Python Reference

Film Used - The Main Wreck

The Second Bit of Wreck

Filmed in St Lucia

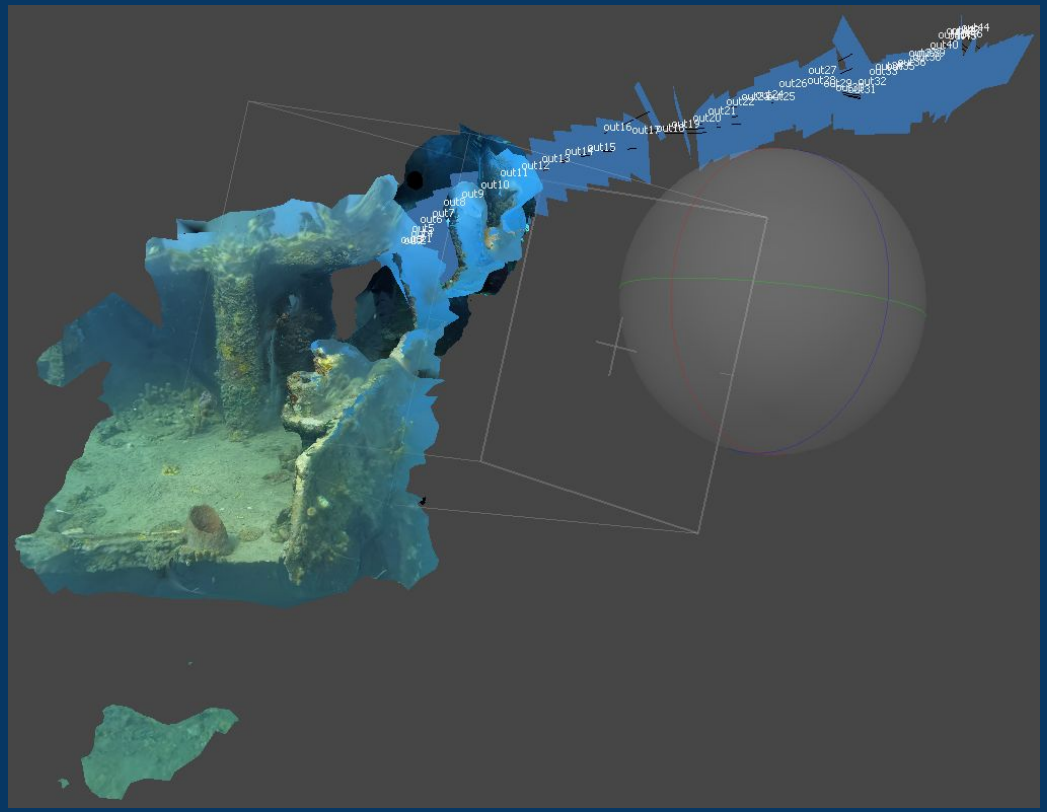
Island Country in the Caribbean



The Cup

Filmed in
Tempe, AZ

Results - Main Wreck

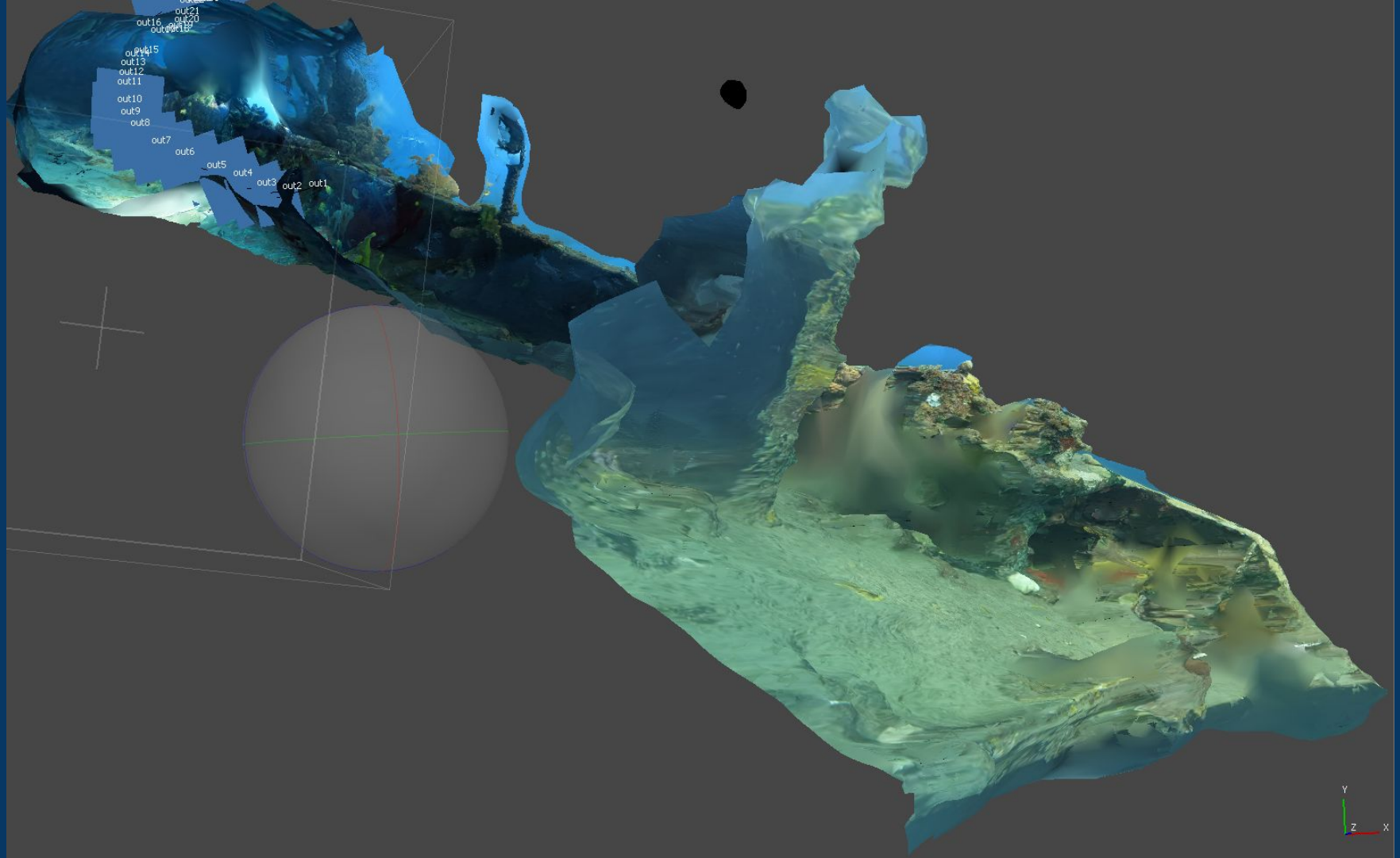


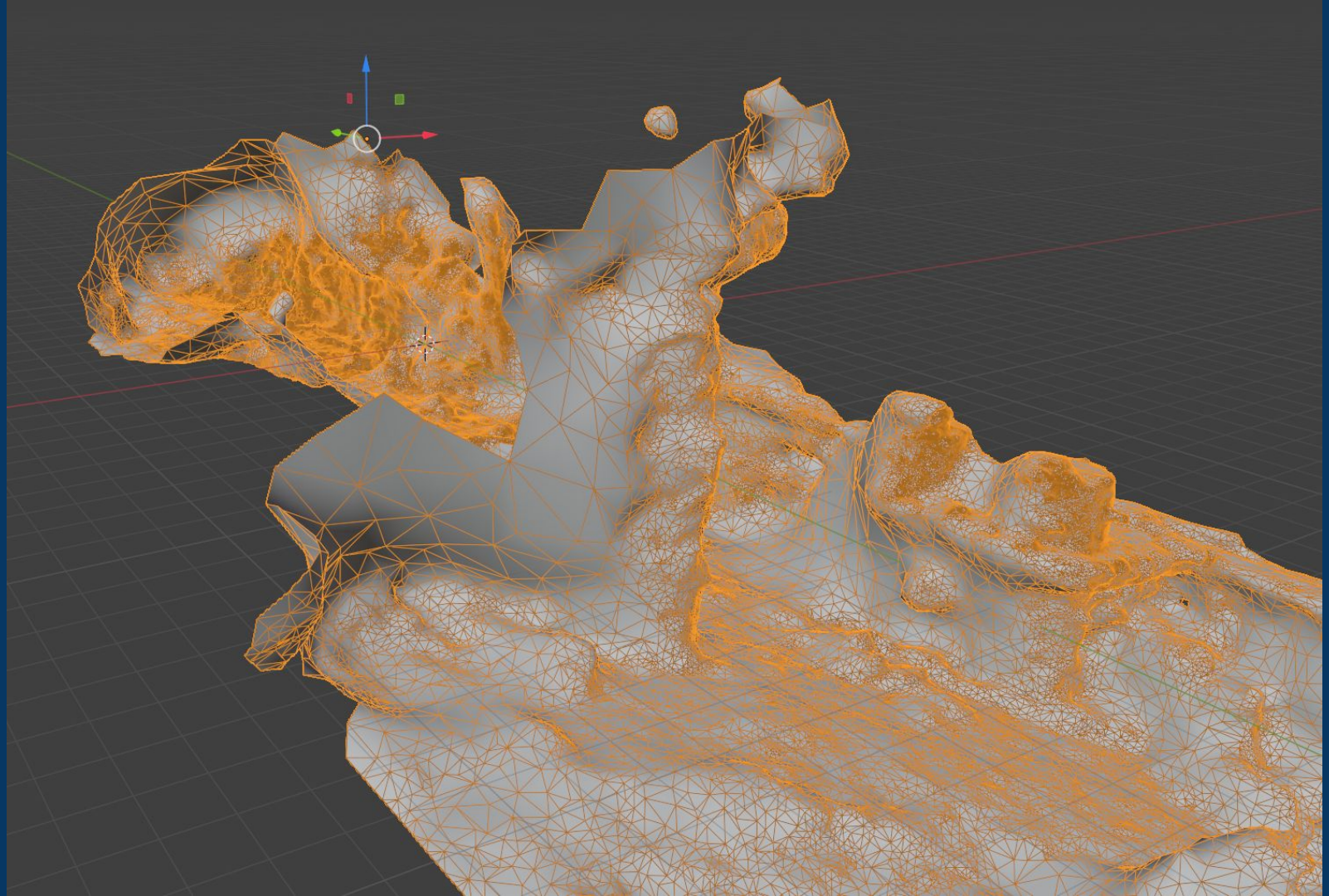
Stats:

- 46/66 images aligned
- 23016 tie points
- Point Cloud of 330k pts
- 3D Model 502k faces

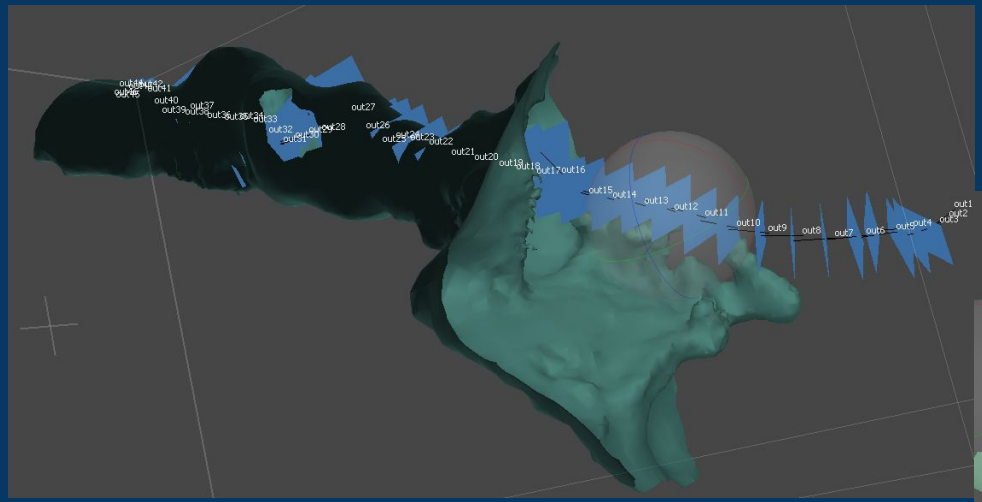
What Worked

- 1fps
- Sequential image preselection
- Model Generation from Depth Maps

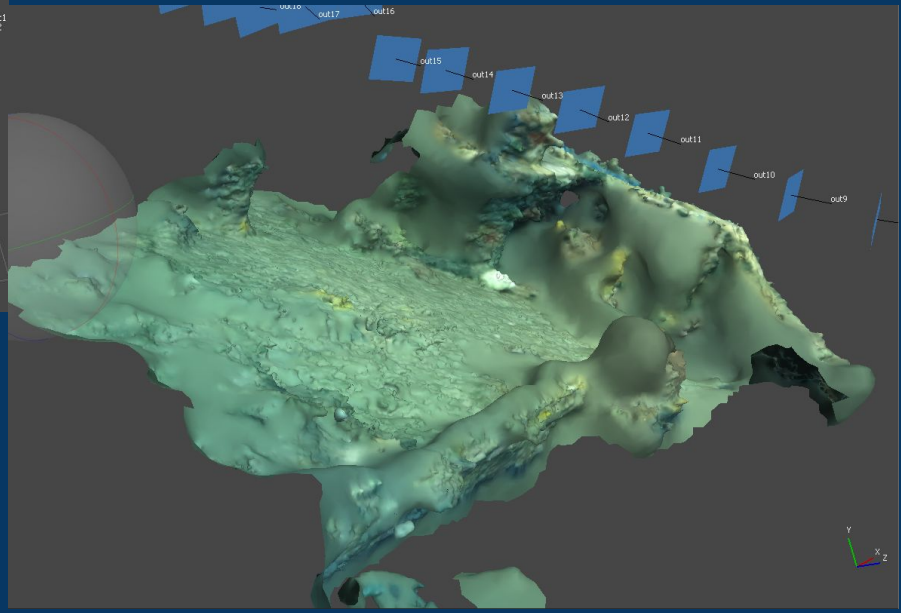

out21
out20
out16
out15
out14
out13
out12
out11
out10
out9
out8
out7
out6
out5
out4
out3
out2
out1
Y
Z
X



Results - Other Model Building Methods



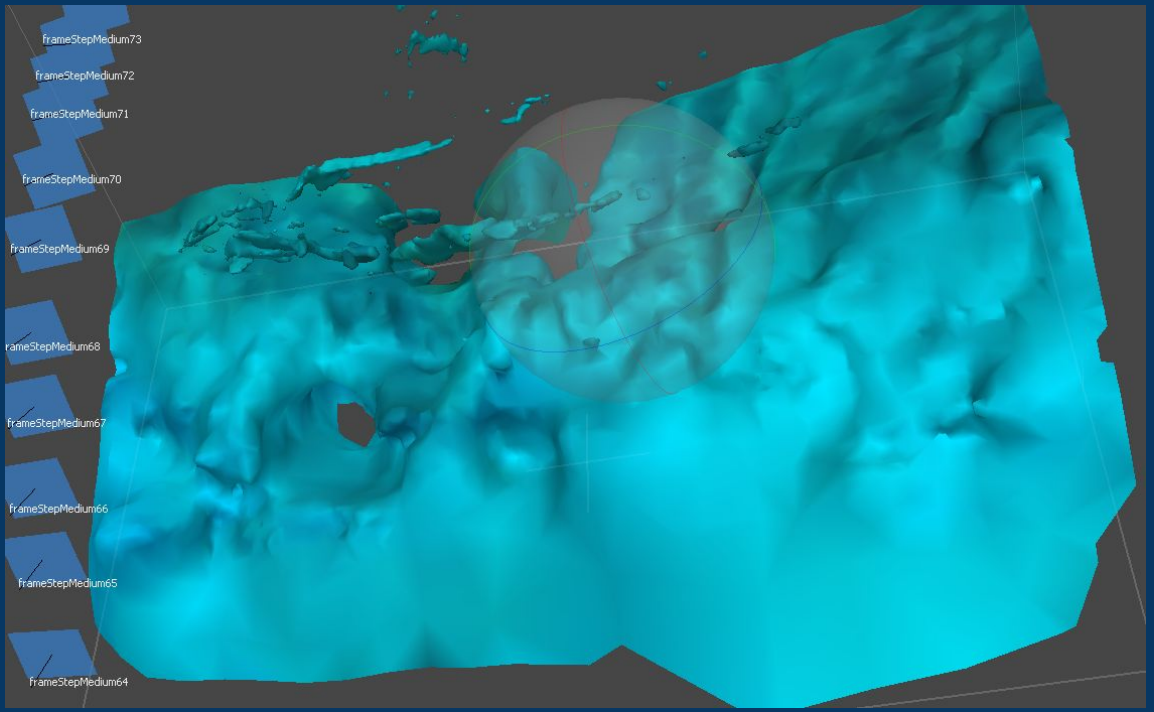
Tie Points (left)
Point Cloud (below)



Tie Points
- 23k faces
- 11k vertices

Point Cloud
- 498k faces
- 250k vertices

Results - The Second Bit of Wreck (Medium Time Step)



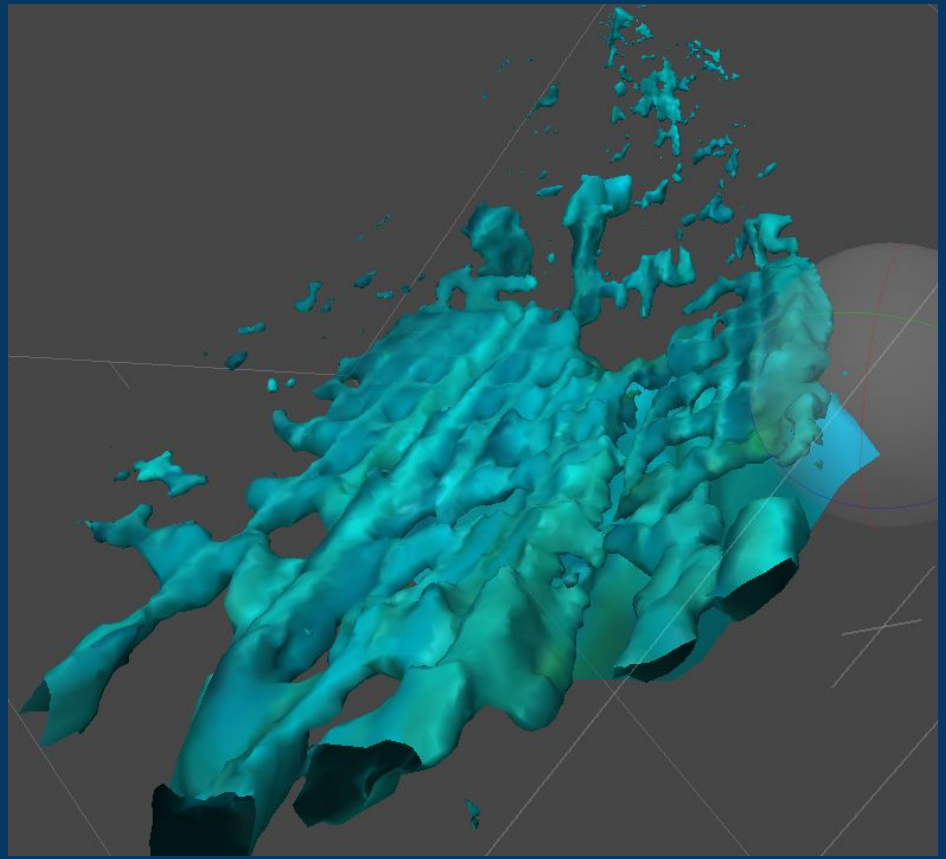
Stats:

- 74k tie points
- 72/125 images aligned
- 3D Model 25k faces

What Worked:

- Nothing really :(
- Model does not look like any of the images

Results - The Second Bit of Wreck (Large Time Step)



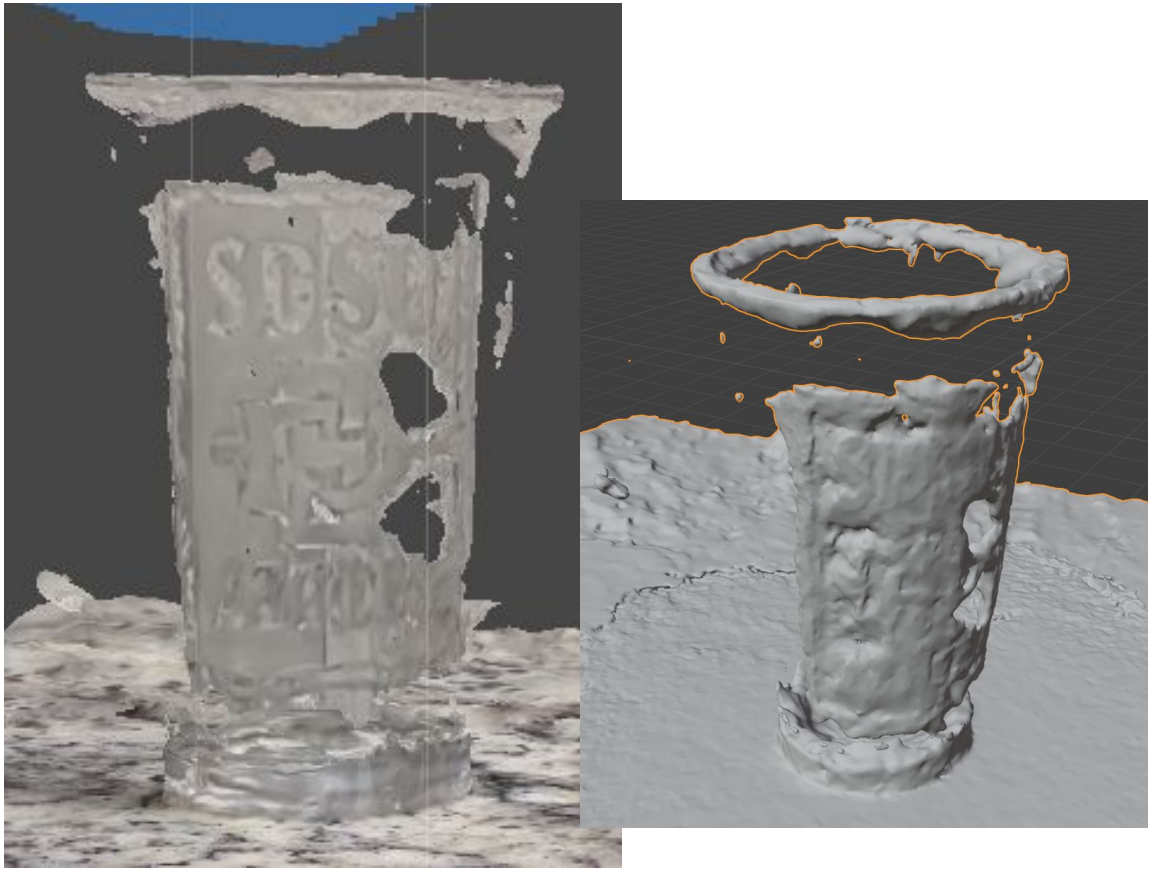
Stats:

- 81k tie points
- 59/59 images aligned
- 3D Model 49k faces

What Worked:

- Improved on model clarity
- Model loosely resembles the source images

The Cup (Small Time Step)



Stats:

- 419/419 Photos Aligned
- 272k Tie Points
- 3D Model 503k faces
- Point Cloud 7.24M points

What Didn't Work:

- Model Scale
- Detecting clear glass

Fun Fact: scale is unknown, in this case the model is nearly 7m tall

Underwater vs Above Water

Best Underwater Model

- 46/66 images aligned
- 23016 tie points
- Point Cloud of 330k pts
- 3D Model 502k faces

Best Above Water Model:

- 419/419 Photos Aligned
- 272k Tie Points
- 3D Model 503k faces
- Point Cloud 7.24M points

Evaluation

Measure of Success: Create an easy to follow and repeatable workflow that produces reasonable 3d models of provided video without intimate knowledge of the programs at work

Easy to follow

Repeatable

Reasonable results

No Intimate knowledge necessary

Conclusions

- ffmpeg
- Agisoft API
- Photos
  - Number
  - Quality
  - Position
- Panning around an object vs through/over
- Clear or Translucent Objects
- Metashape Presets
  - Image Preselection
  - Tie points

Questions?

Thank You!

Meta Analysis

What doing?	Time (rounded to .25 hrs)	helpful to end product	Total
Project ideas/research	6	yes	126.75
installing gazebo	1.5	no	Wasted
installing gazebo classic	0.5	no	28.75
fiddling with gazebo robot models	2	no	
orb slam research and integration	16.25	no	
installing and understanding ffmpeg	1	yes	
film processing	1	yes	
agisoft install	0.5	yes	
understanding agisoft	2	yes	
cup filming	1	yes	
importing and aligning photos in agisoft	23.25	yes	
building models	27.75	yes	
texture building	6.25	yes	
point cloud generation	4	yes	
learning python api	5	yes	
python implementation *legally*	3	no	
jerryrigging the pirated license	3.75	yes	
blender imports for viewing	2.25	yes	
messing w blender render	5.5	no	
troubleshooting	14.25	yes	