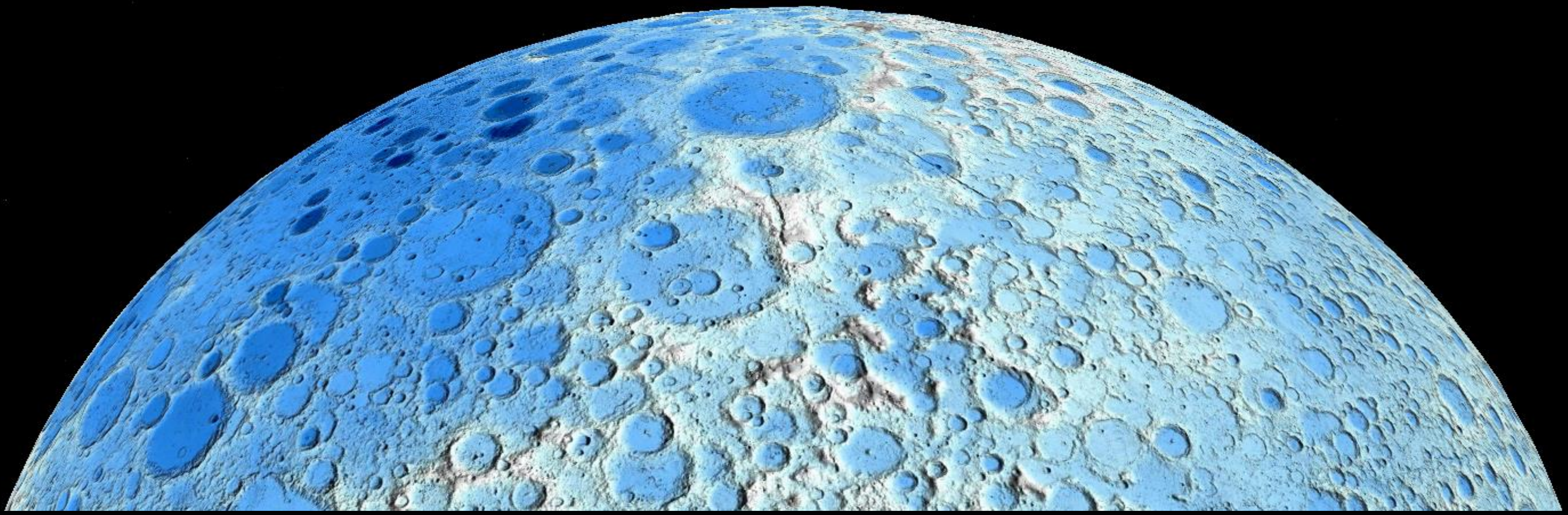
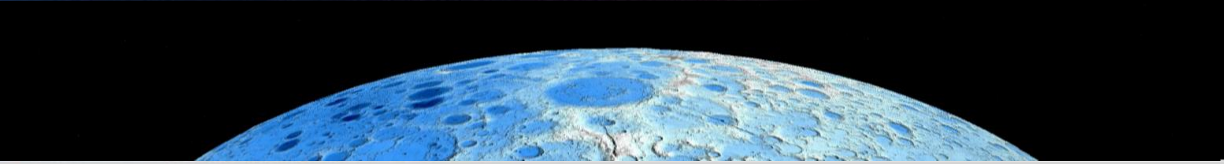


AI based Lunar Data rendering and Visualization in Celestial Mapping System

Graham Mackintosh, Parul Agrawal and Allison Zuniga

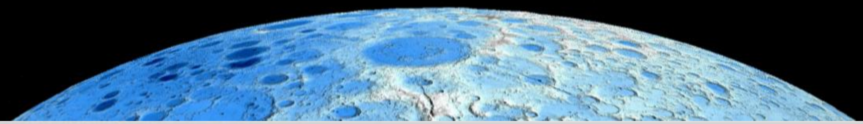
NASA Ames Research Center

May 2024



Outline

- Introduction to CMS
- 3rd party data ingestion (HORUS datasets)
- Functional analysis on ingested datasets
- Subsurface feature visualization capability
- AI enhanced data ingestion pipeline and analytics
- AI assisted geo-referencing and rendering
- AI assisted subsurface data ingestion, geo-referencing and rendering in CMS



Introduction

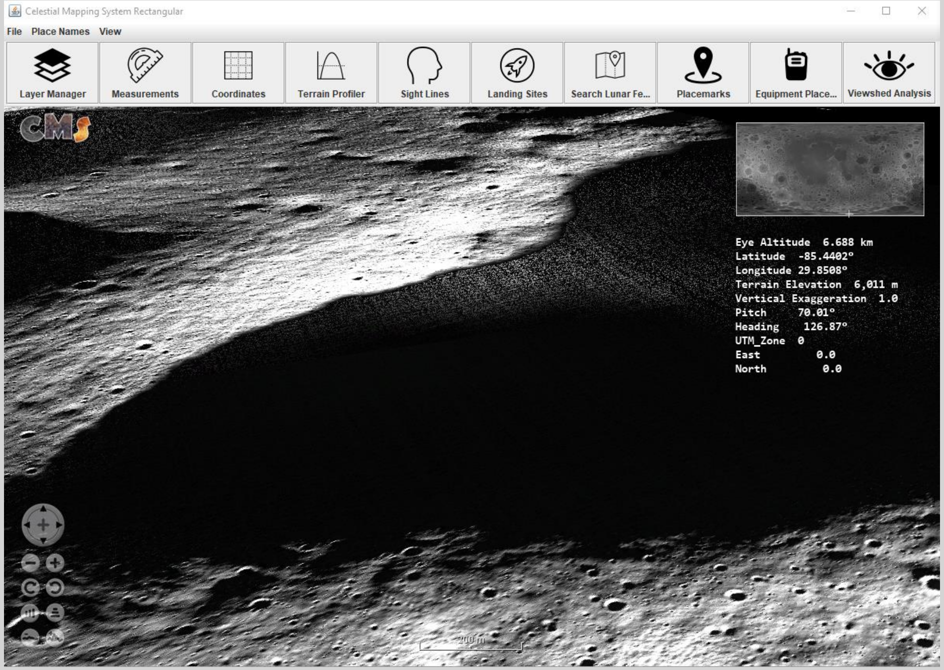
- CMS is a multiplatform application to generate user-interactive virtual 3D globes for celestial bodies within our solar system.
- Various layers are built on top of the virtual globe to provide visualization of high-resolution imagery, enable precise measurements, build extensive analytical capabilities and a broad range of functionalities
- CMS website - https://celestial.arc.nasa.gov/

Key Features

- **3rd party Maps and data ingestion, rendering and visualization**
- 3D Measurement tool kit
- Line of sight analysis
- Equipment placement & Planning
- Data import-export
- 3D COLLADA Models
- Sun angle calculations
- **Subsurface visualization (in development)**

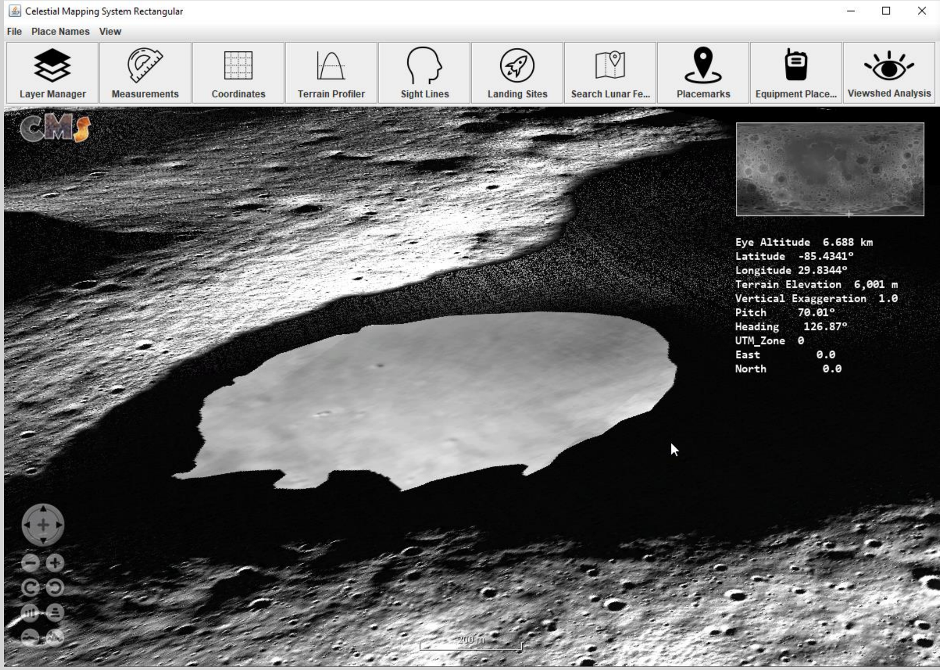
Example of 3rd party data Ingestion - Illumination of PSR by HORUS

Ingestion of super enhanced images in CMS created by Hyper-effective nOise Removal U-net Software [HORUS] * near Nobili Crater - VIPER landing site



```
Eye Altitude  6.688 km
Latitude  -85.4402°
Longitude 29.8508°
Terrain Elevation  6,011 m
Vertical Exaggeration  1.0
Pitch     70.01°
Heading    126.87°
UTM_Zone  0
East          0.0
North          0.0
```

PSR site shown in LROC NAC layer of CMS

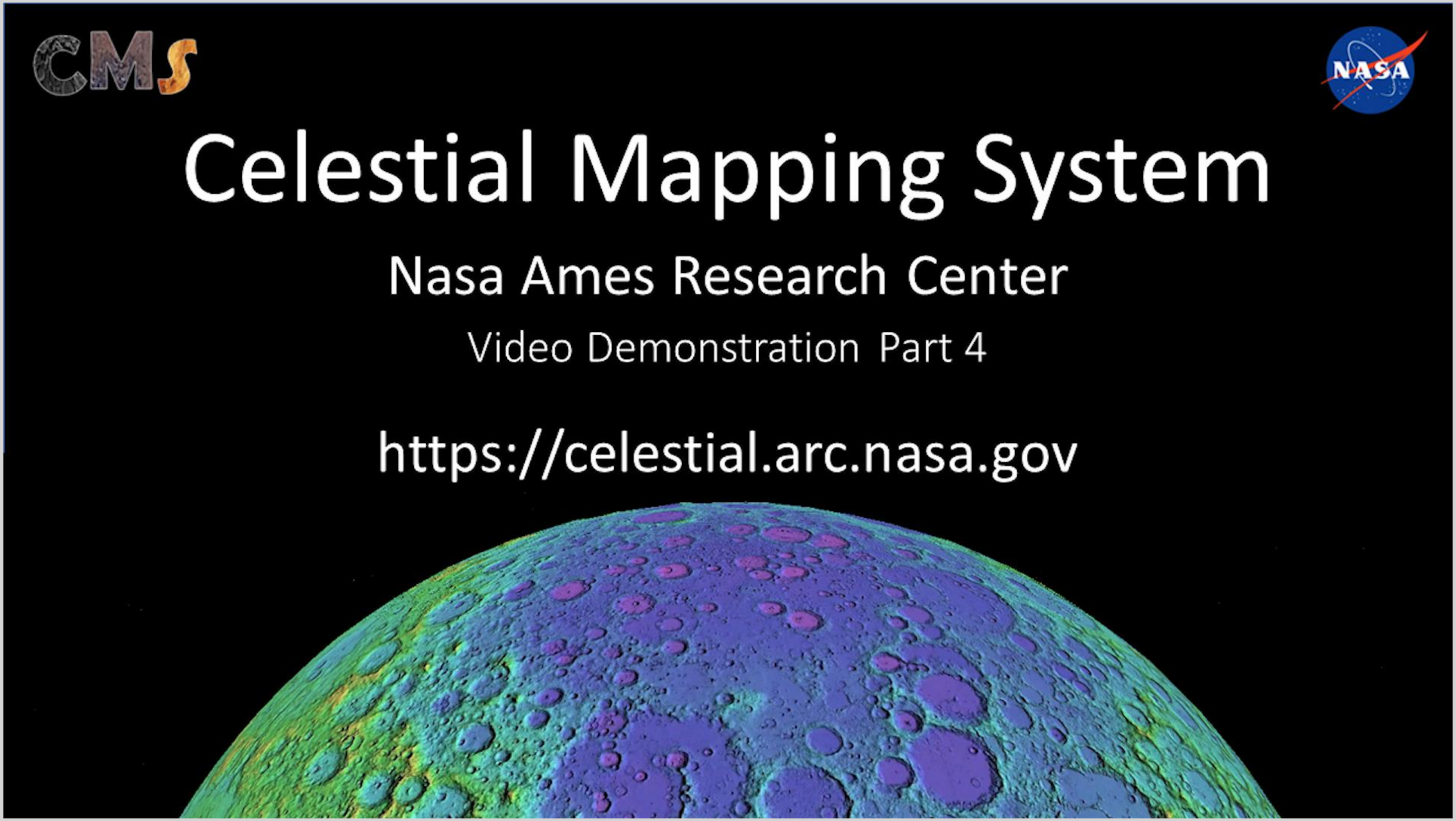


```
Eye Altitude  6.688 km
Latitude  -85.4341°
Longitude 29.8344°
Terrain Elevation  6,001 m
Vertical Exaggeration  1.0
Pitch     70.01°
Heading    126.87°
UTM_Zone  0
East          0.0
North          0.0
```

Illuminated site by using ingested and merged HORUS images within CMS

* Ref: Bickel V.T, et al., 2021 "Peering into lunar permanently shadowed regions with deep learning", Nature Comm 12, 5607

Platform Demo



Celestial Mapping System

Nasa Ames Research Center

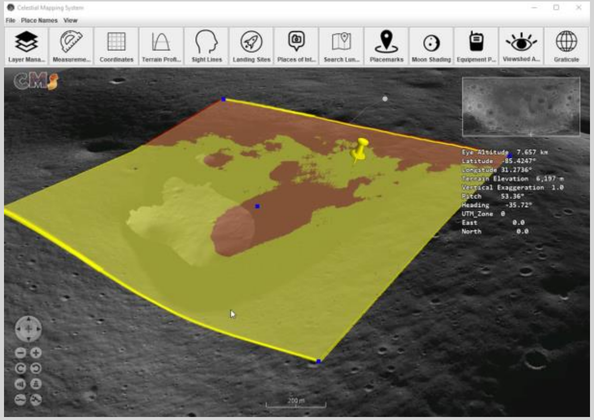
Video Demonstration Part 4

https://celestial.arc.nasa.gov

Functional Analysis on Illuminated PSR

Once the 3rd party data is ingested and rendered in CMS, it can be utilized for various analyses. *

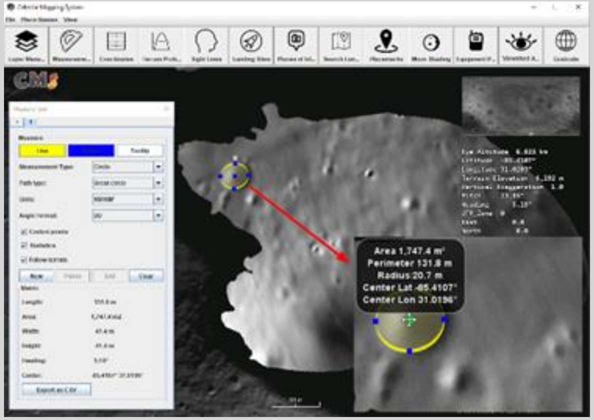
Visibility Analysis



```
Eye Altitude  7.657 km
Latitude  -85.4247°
Longitude 31.2736°
Terrain Elevation  6,197 m
Vertical Exaggeration  1.0
Pitch     53.36°
Heading    -35.72°
UTM_Zone  0
East        0.0
North        0.0
```

Viewshed Analysis with observer location shown by yellow pin

Measurements

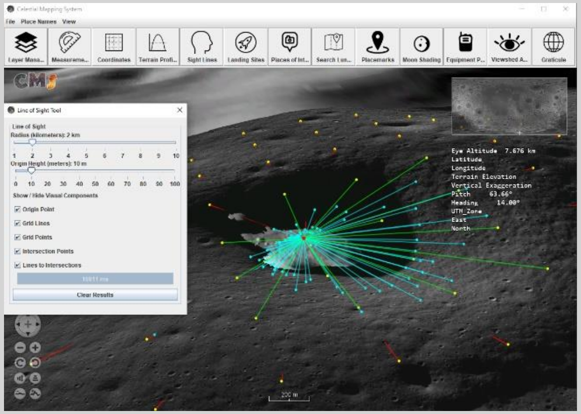


```
Eye Altitude  [illegible] km
Latitude  [illegible]
Longitude [illegible]
Terrain Elevation  6,192 m
Vertical Exaggeration  1.0
Pitch     [illegible]
Heading    [illegible]
UTM_Zone  0
East        0.0
North        0.0
```

Area 1,747.4 m²
Perimeter 131.8 m
Radius 20.7 m
Center Lat -85.4107°
Center Lon 31.0196°

Measurement of a crater inside the PSR by 3D measurement tool

Equipment Placement and Coverage



Line of Sight Tool

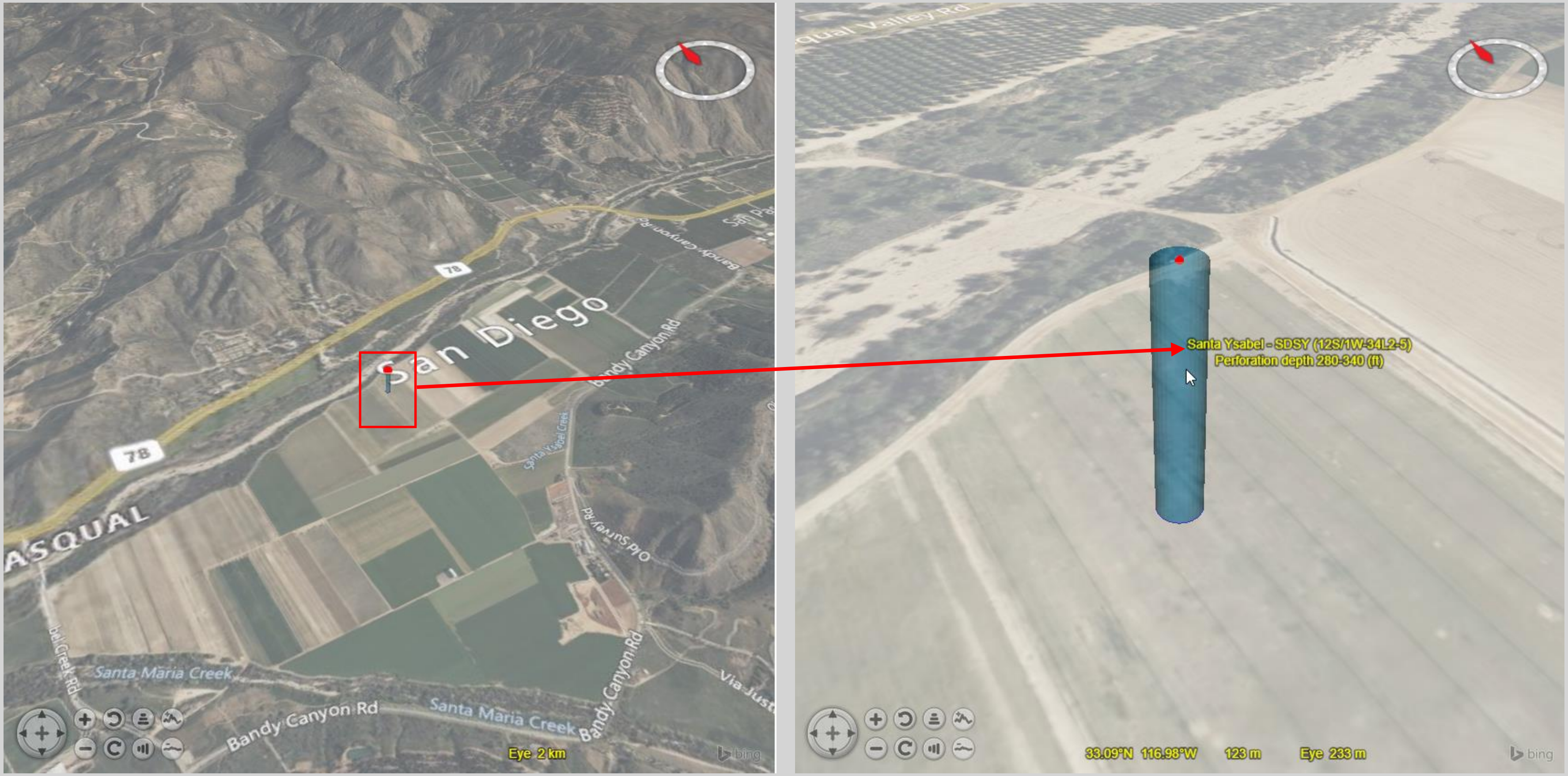
Line of Sight
Radius (kilometers): 2 km
1 2 3 4 5 6 7 8 9 10
Origin Height (meters): 10 m
0 10 20 30 40 50 60 70 80 90 100
Show / Hide Visual Components
☑ Origin Point
☑ Grid Lines
☑ Grid Points
☑ Intersection Points
☑ Lines to Intersections
Clear Results

```
Eye Altitude  7.676 km
Latitude
Longitude
Terrain Elevation
Vertical Exaggeration
Pitch     63.66°
Heading    14.00°
UTM_Zone
East
North
```

Equipment placement and analysis of coverage

*Reference: Agrawal P. et. al. " GLOBAL 3D DATA VISUALIZATION AND ANALYSIS PLATFORM WITH ADVANCED MACHINE LEARNING CAPABILITIES IN SUPPORT OF LUNAR EXPLORATION", 55th LPSC 2024

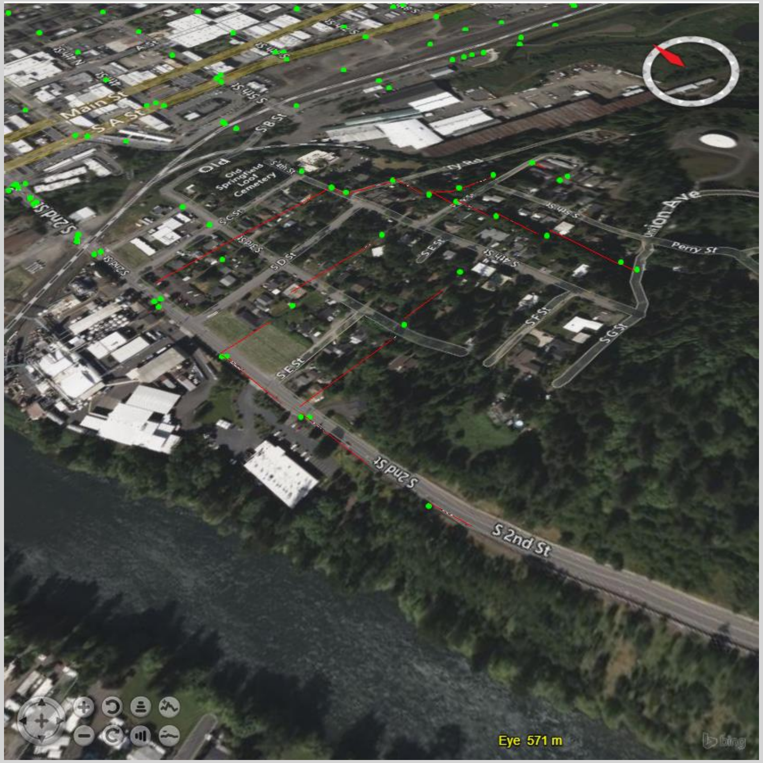
Subsurface Capabilities



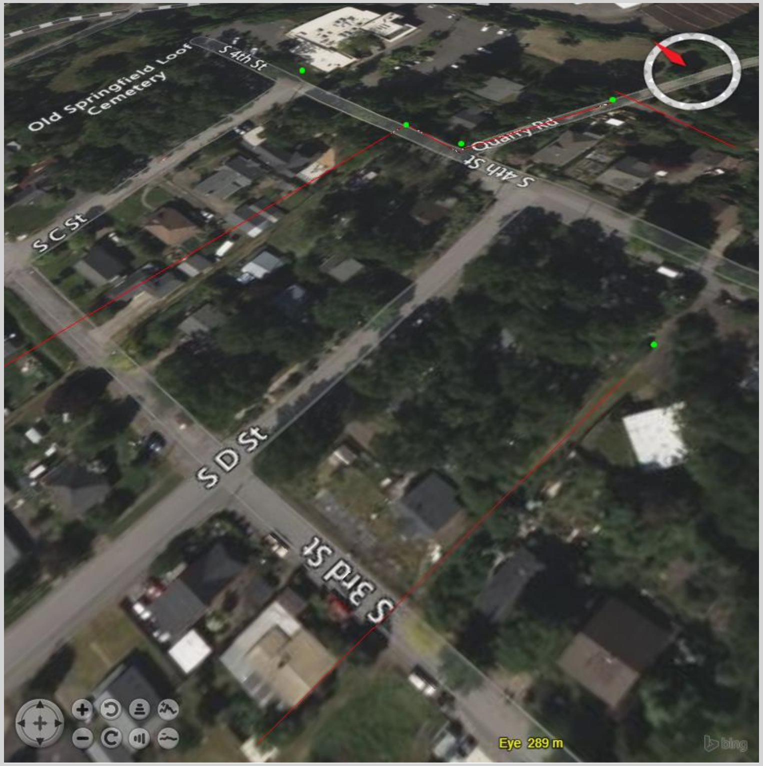
Area of Interest

Subsurface 3D object

Subsurface feature (water pipes)



Overview of region



Water Pipes (red) and Manhole Covers (green)

Potential Lunar Subsurface Features in CMS

- Seismic data rendering
- LiDAR data rendering and visualization of Lunar lava tubes
- 3D representation of Lunar lava tube cave entrance

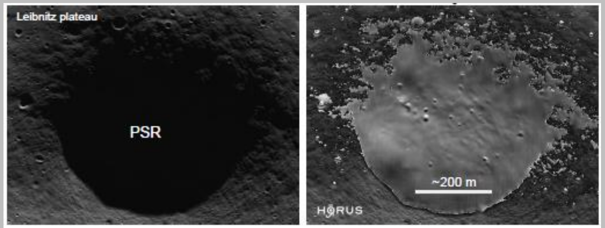
AI ENHANCED DATA PIPELINE and ANALYTICS

Use AI to amplify the CMS differentiators:

1. **Local-to-global**: AI enhancements that span all scales of geography and datasets.
2. **“Data Open”** : Rapid data import pipeline, robust layer management.
3. **Digging into subsurface**: there is a whole new Moon waiting for us!
4. **Intelligent Analytics**: Assisting in the hunt for subsurface resources

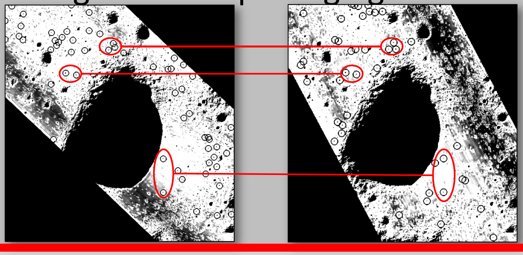
Raw Data Augmentation: ✓

- Super resolution
- Signal-noise enhancement



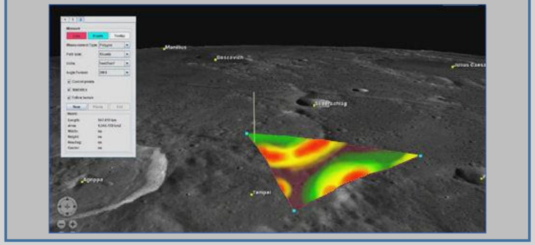
Accelerated data pipeline:

- Automated georectification and mosa
- Intelligent overlap merging



AI Analytics:

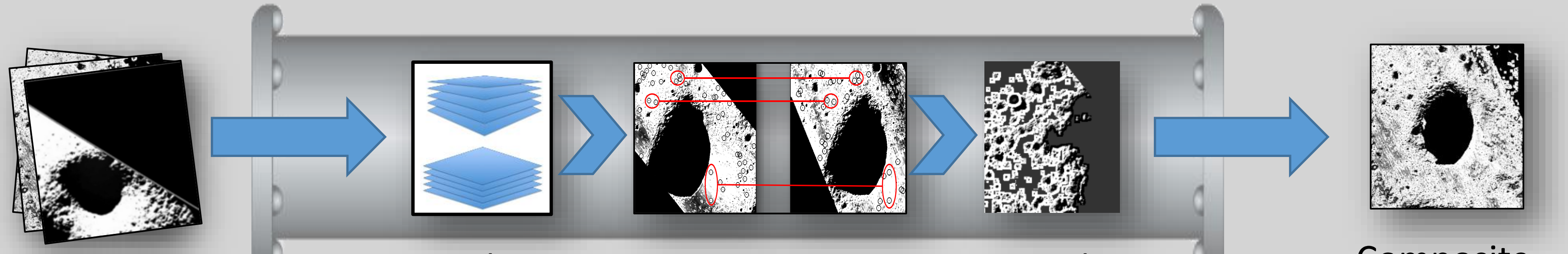
- Sub-surface layers
- Multi-layer similarity search



AI Data Pipeline Process

- **Immediate Goal**: Validate a process capable of rapid and automated ingest of 1000s of south pole images enhanced by HORUS

- **Future Goal**: Extend this capability to automate the ingest of many datasets, including subsurface.



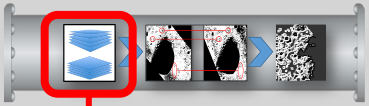
Raw image data

Rank & stack

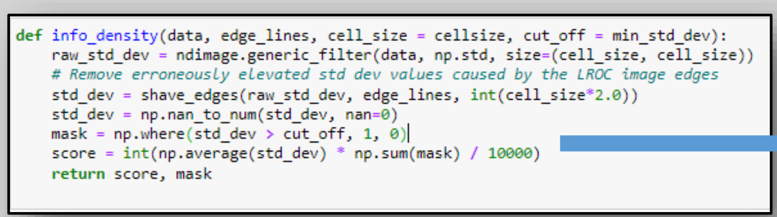
Group & Georectify

Mask, Merge & Mosaic

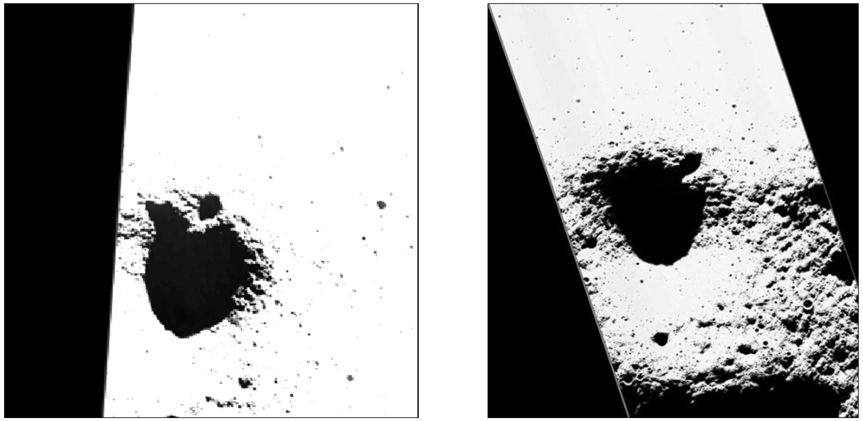
Composite image into CMS



Step 1) Rank and stack using statistical measures of information density.

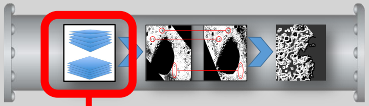


```
def info_density(data, edge_lines, cell_size = cellsize, cut_off = min_std_dev):
    raw_std_dev = ndimage.generic_filter(data, np.std, size=(cell_size, cell_size))
    # Remove erroneously elevated std dev values caused by the LROC image edges
    std_dev = shave_edges(raw_std_dev, edge_lines, int(cell_size*2.0))
    std_dev = np.nan_to_num(std_dev, nan=0)
    mask = np.where(std_dev > cut_off, 1, 0)
    score = int(np.average(std_dev) * np.sum(mask) / 10000)
    return score, mask
```

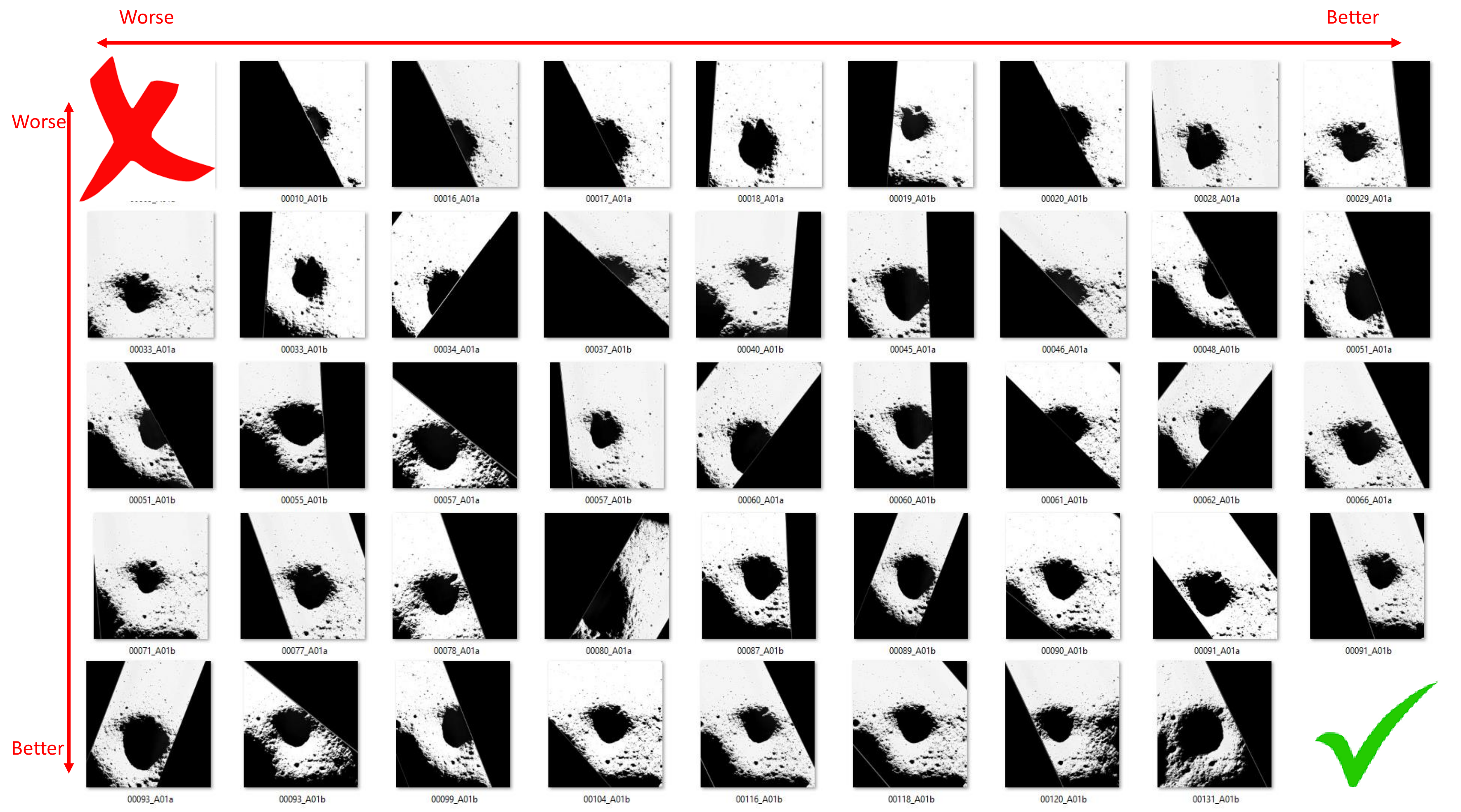


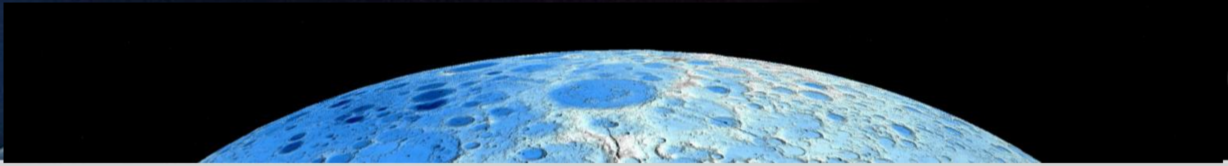
Score: 9 ✗

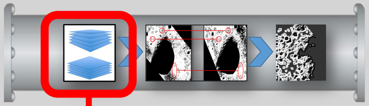
Score: 211 ✓



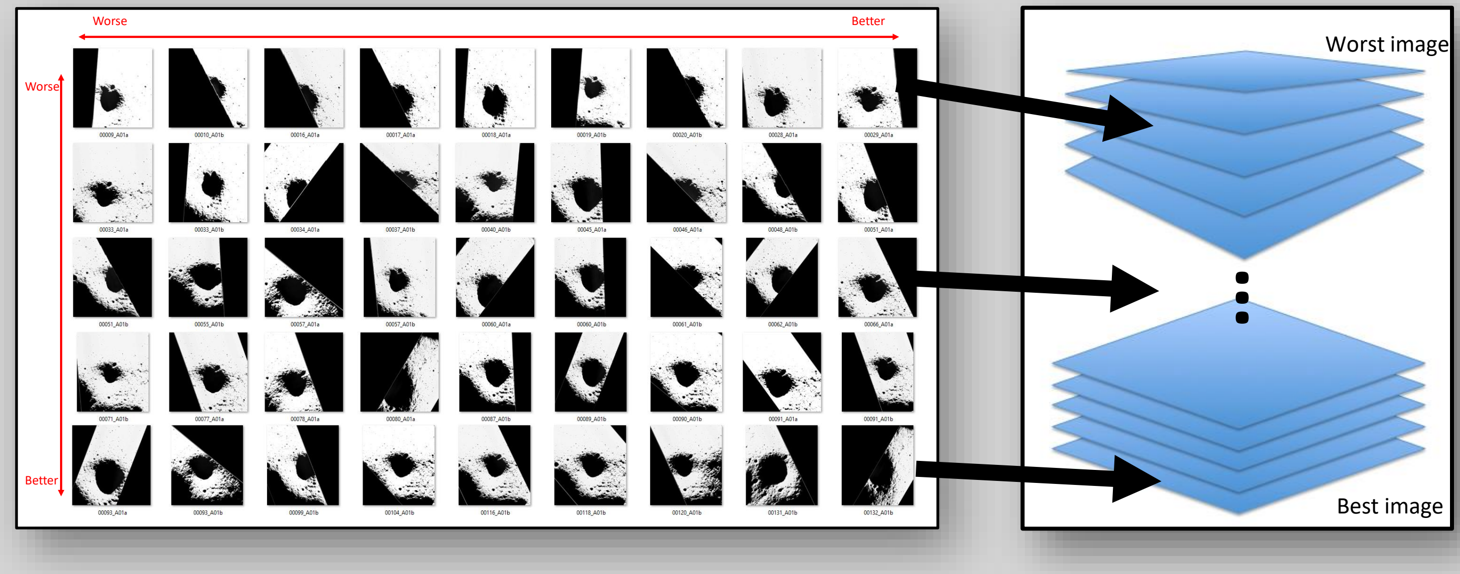
Step 1) Rank and stack using statistical measures of information density.





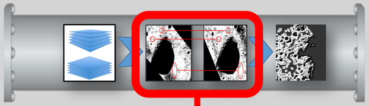


Step 1) Rank and stack using statistical measures of information density.

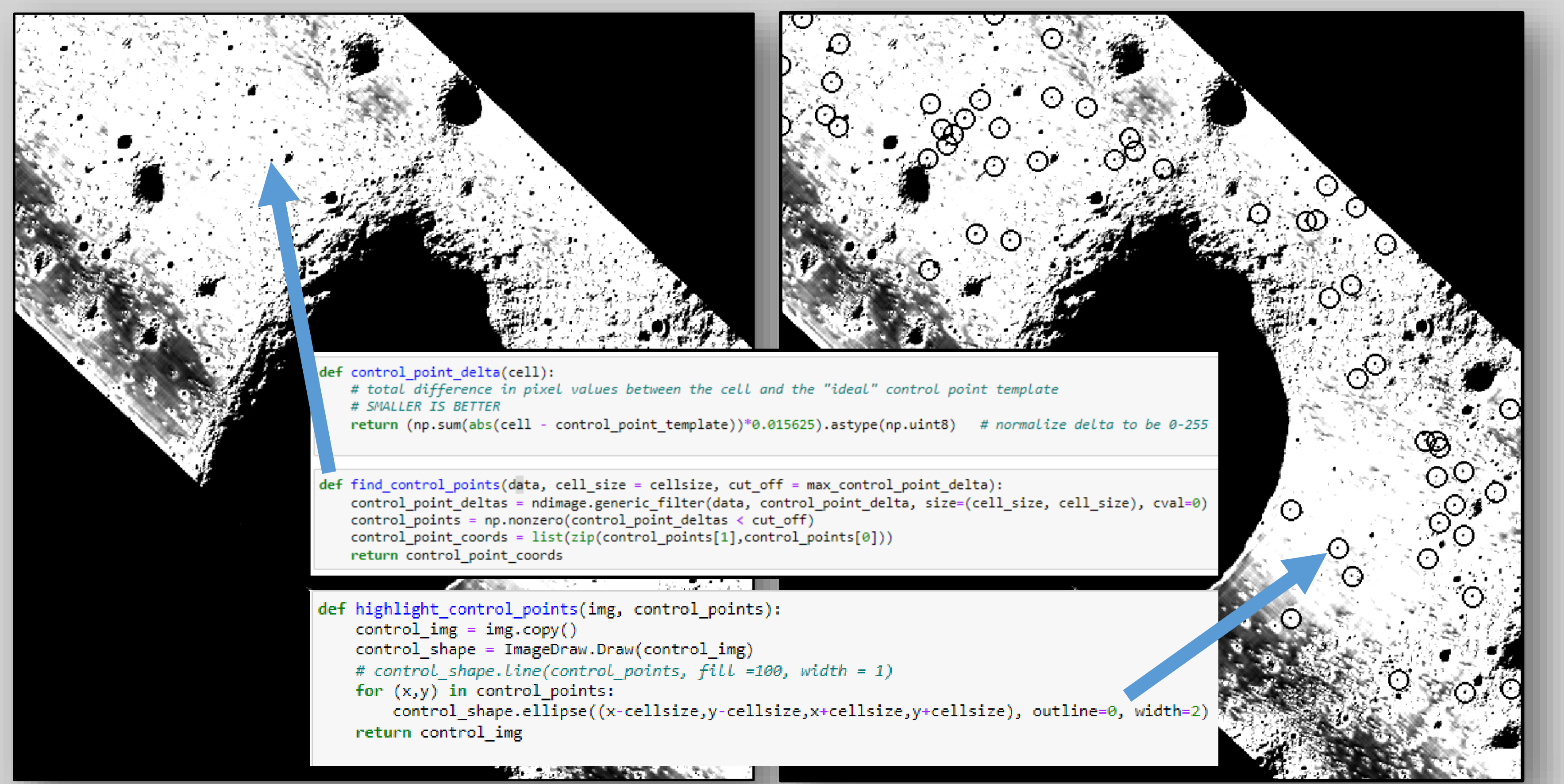


Stack images for merging based on their image density score.

Highest score becomes the background, lower scores are merged onto the aggregate image that has been generated below.



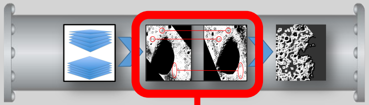
Step 2) Georectify: Find control features in each image



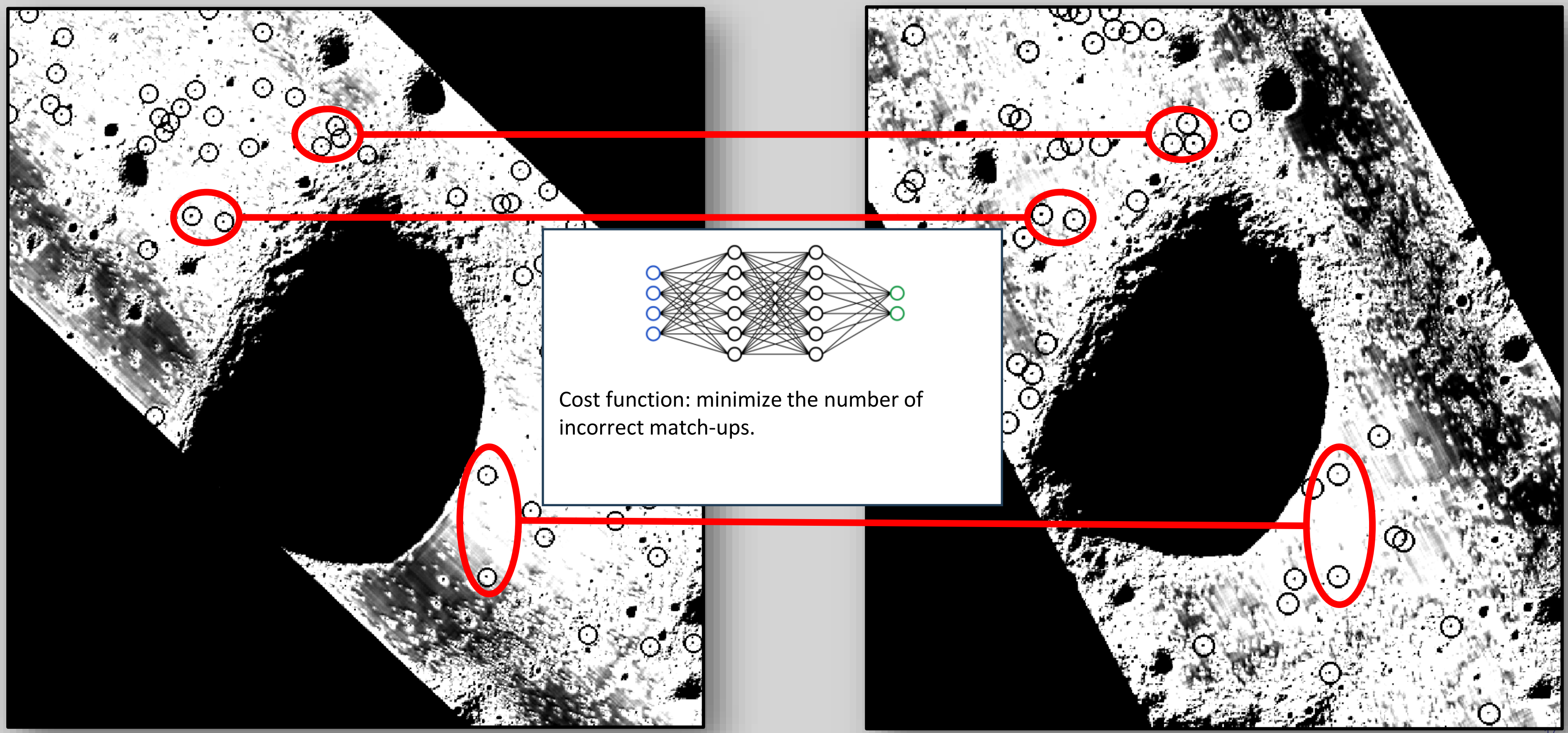
```
def control_point_delta(cell):
    # total difference in pixel values between the cell and the "ideal" control point template
    # SMALLER IS BETTER
    return (np.sum(abs(cell - control_point_template))*0.015625).astype(np.uint8)   # normalize delta to be 0-255
```

```
def find_control_points(data, cell_size = cellsize, cut_off = max_control_point_delta):
    control_point_deltas = ndimage.generic_filter(data, control_point_delta, size=(cell_size, cell_size), cval=0)
    control_points = np.nonzero(control_point_deltas < cut_off)
    control_point_coords = list(zip(control_points[1],control_points[0]))
    return control_point_coords
```

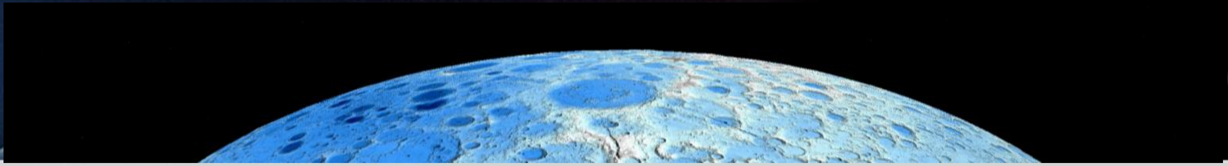
```
def highlight_control_points(img, control_points):
    control_img = img.copy()
    control_shape = ImageDraw.Draw(control_img)
    # control_shape.line(control_points, fill =100, width = 1)
    for (x,y) in control_points:
        control_shape.ellipse((x-cellsize,y-cellsize,x+cellsize,y+cellsize), outline=0, width=2)
    return control_img
```

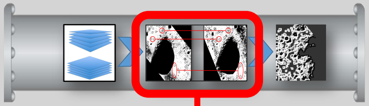


Step 2) Georectify: AI to find matching control points



Cost function: minimize the number of incorrect match-ups.





Step 2) Georectify: AI to find matching control points

Synthetic training data to the rescue

SYNTHETIC TRAINING DATA
Random number generation for list of “image #1” synthetic control point coordinates

Synthetic #1

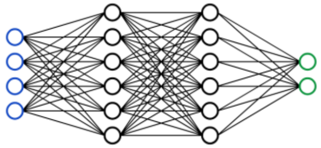
[(0, 35), (136, 49), (212, [illegible]0), (142, 54), (166, 69), (283, 74), (206, 78), (278, 87), (85, 108), (59, 190), (102, 198), (0, 210), (88, 226), (125, 316), (0, 344), (146, 365), (276, 375), (248, 391), (282, 392)]

Synthetic #2

[(194, 0), (233, 0), (286, 4), (50, 23), (157, 38), (277, 39), (79, 44), (129, 75), (267, 79), (125, 84), (160, 88), (139, 97), (149, 102), (355, 103), (144, 111), (275, 114), (330, 121), (292, 124), (224, 125), (452, 126), (185, 130), (388, 133), (469, 178), (490, 182), (220, 188), (133, 189), (200, 195), (309, 521), (406, 530), (506, 550), (399, 552), (291, 564), (285, 570), (480, 571), (422, 619), (381, 630), (455, 639), (547, 667), (547, 685), (553, 712)]

Use simple Cartesian functions to randomly rotate, translate and dilate the first list of control points into a second list; then randomly add/remove other points and scramble the order of the points in each list.

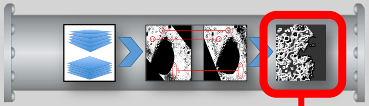
Do this millions of times to create a massive fully labelled dataset for training and testing.



Cost function: minimize the number of incorrect match-ups.

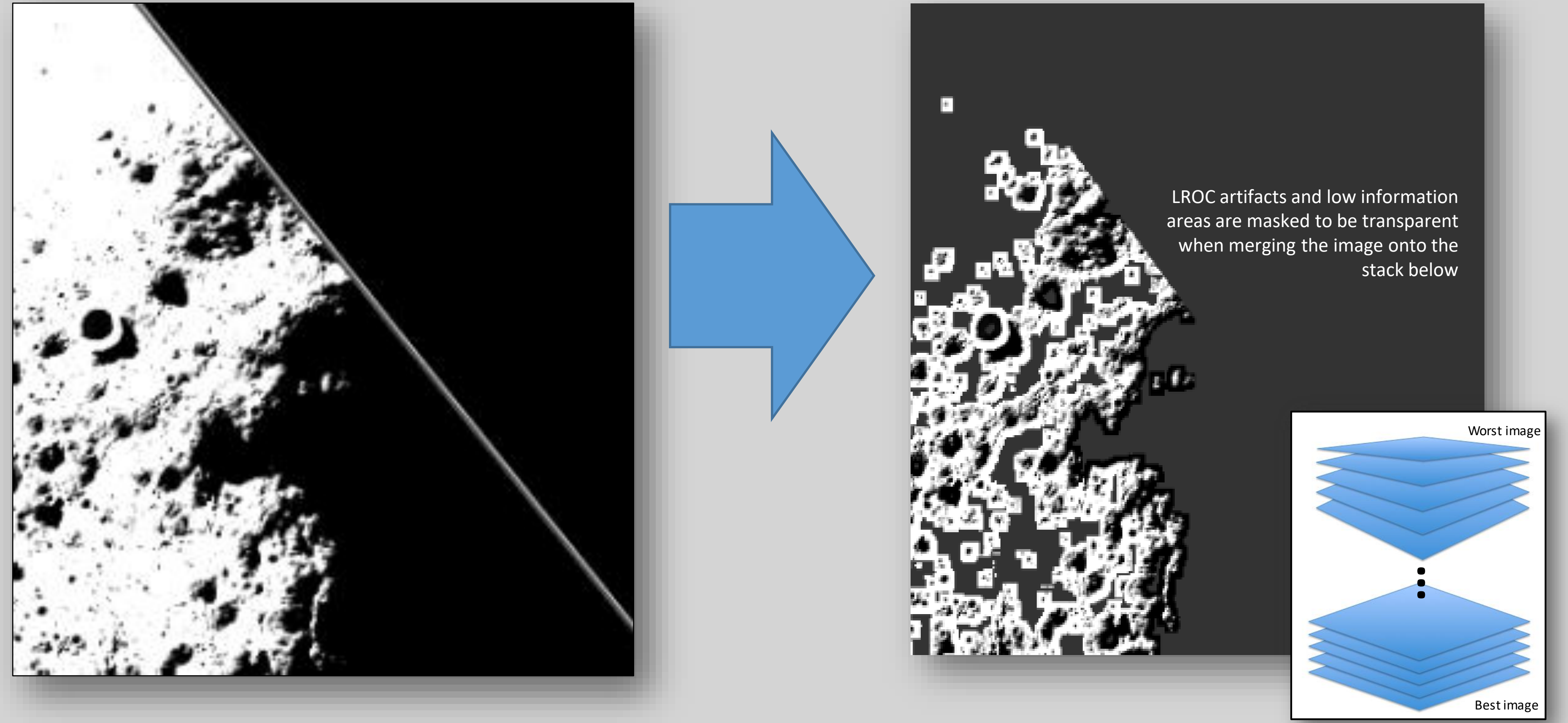
[0, 0, 6, 0, 0, 21, 0, 0, 3, 0, 0, 0, 0, 5, 46, 0, 0, 0, 0, 0, 17, 0]

This is synthetic data, so we know that this is the correct answer, which is used to train the model.

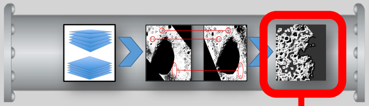


Step 3) Intelligent Mask & Merge for Optimal Information Gain

Mask the areas of the new image that are of a lower info density than the aggregate image already built up below it

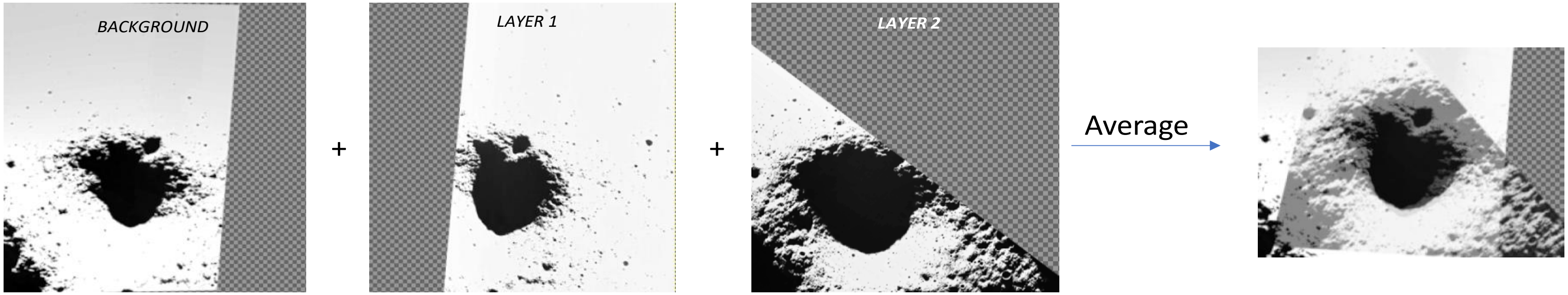


LROC artifacts and low information areas are masked to be transparent when merging the image onto the stack below

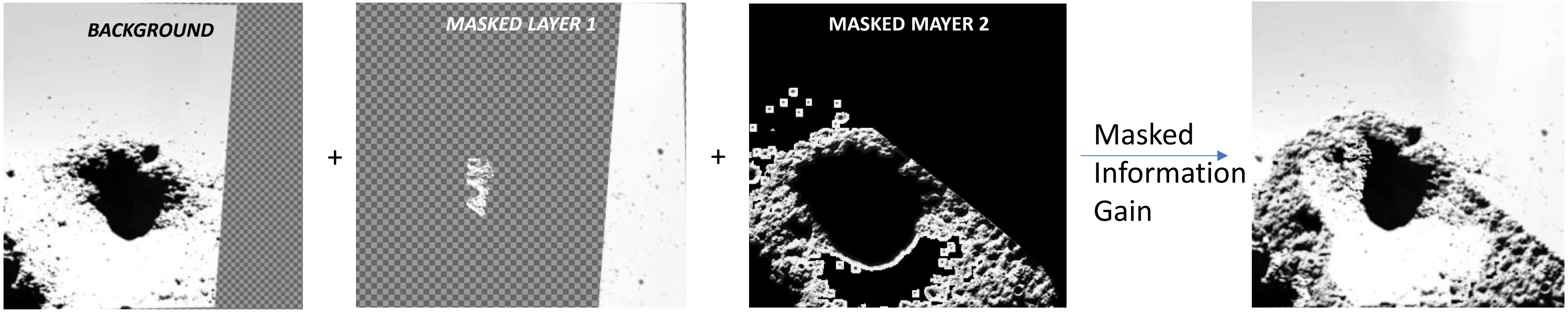


Step 3) Intelligent Mask & Merge for Optimal Information Gain

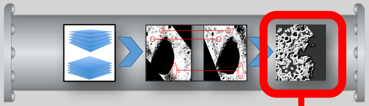
Pixel Averaging vs. Masked Addition



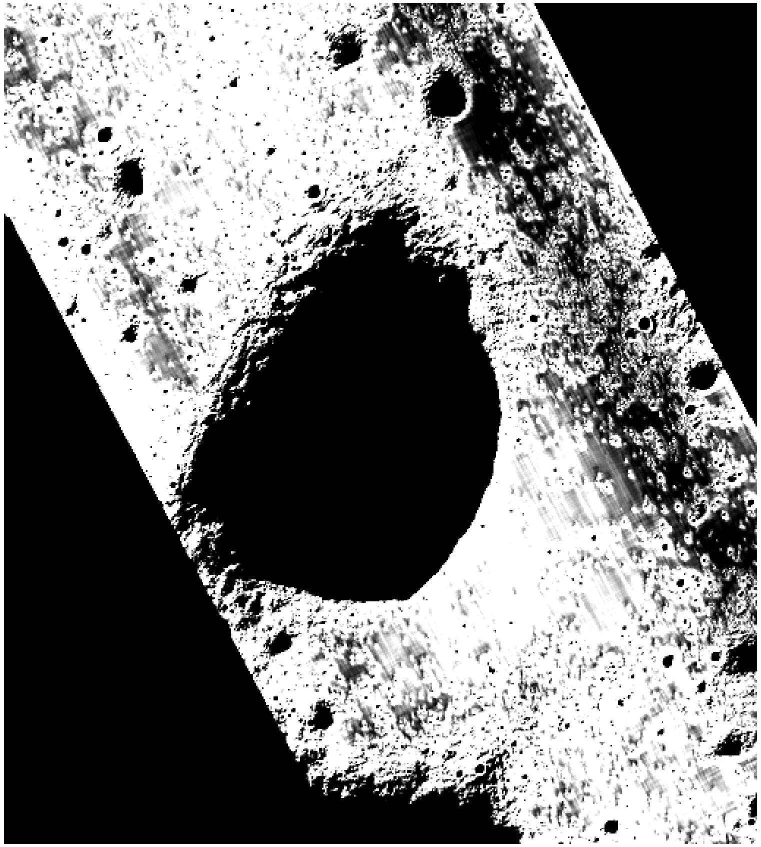
Average



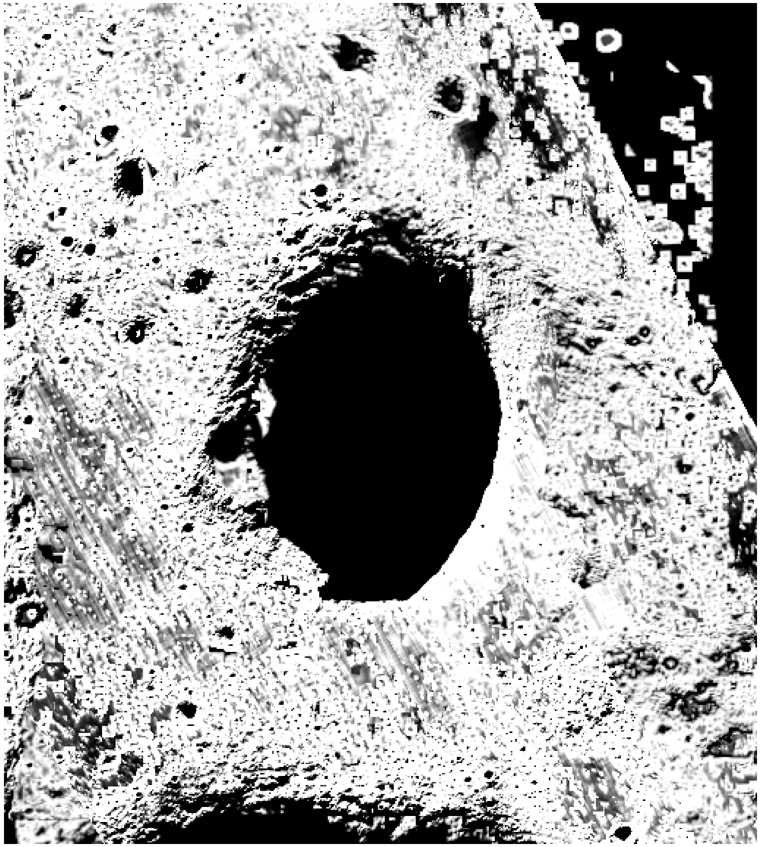
Masked Information Gain



Step 3) Intelligent Mask & Merge for Optimal Information Gain



Highest scoring image is selected as the background

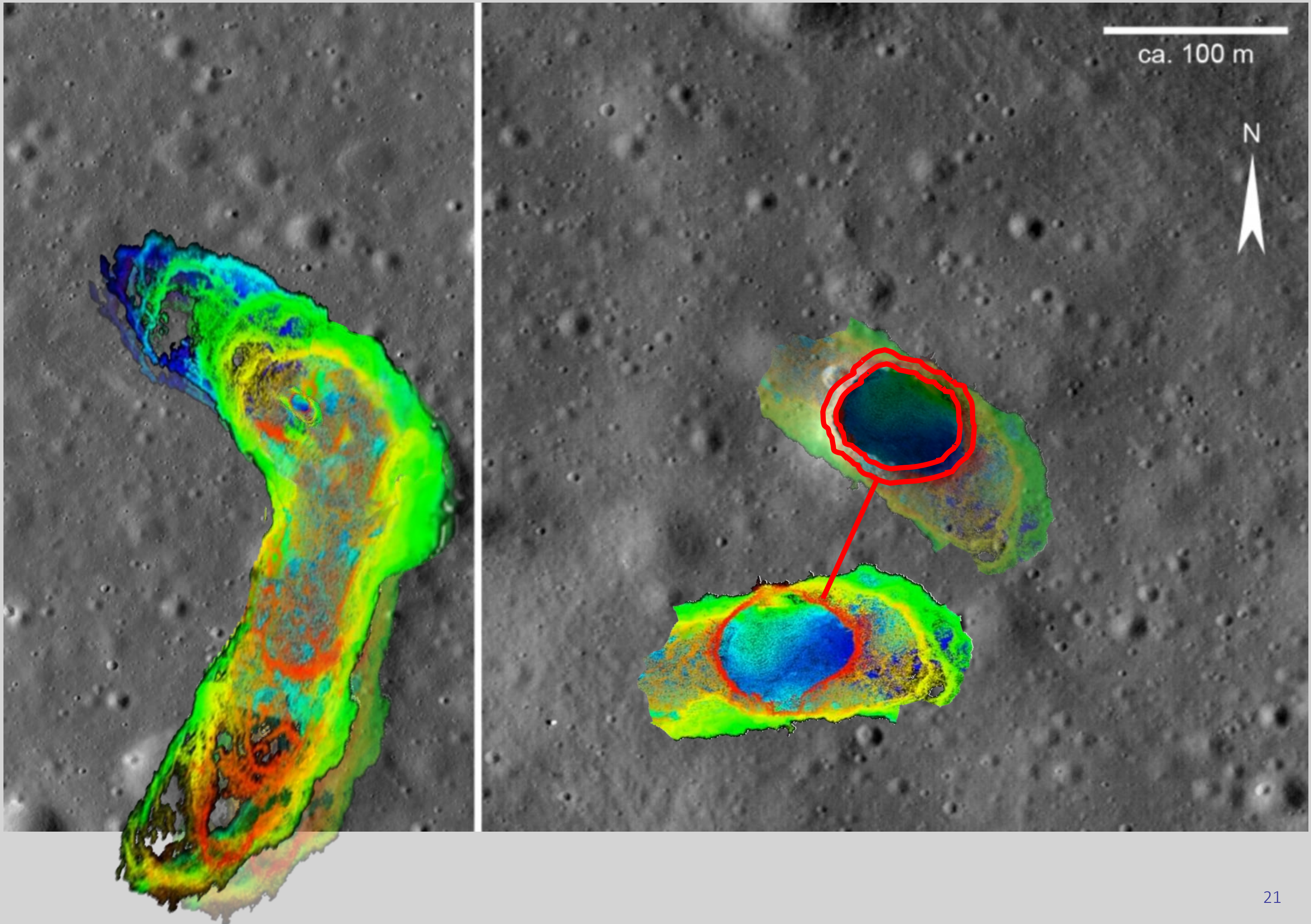


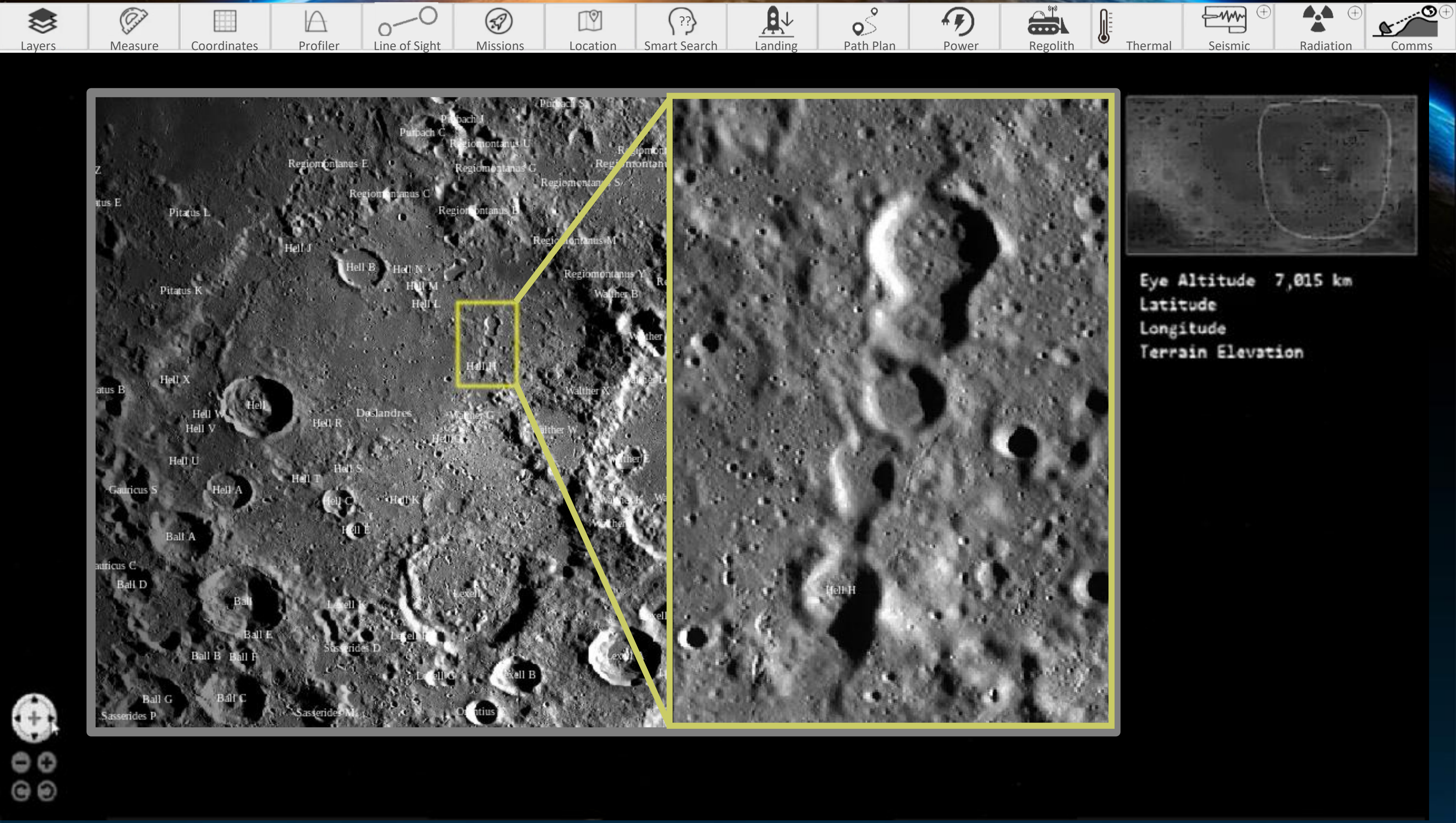
Addition of 7 layers that are masked based on where each layer would add (vs. subtract) information

Looking ahead: Expand the CMS automated ingest pipeline to include a wide range of surface and subsurface data

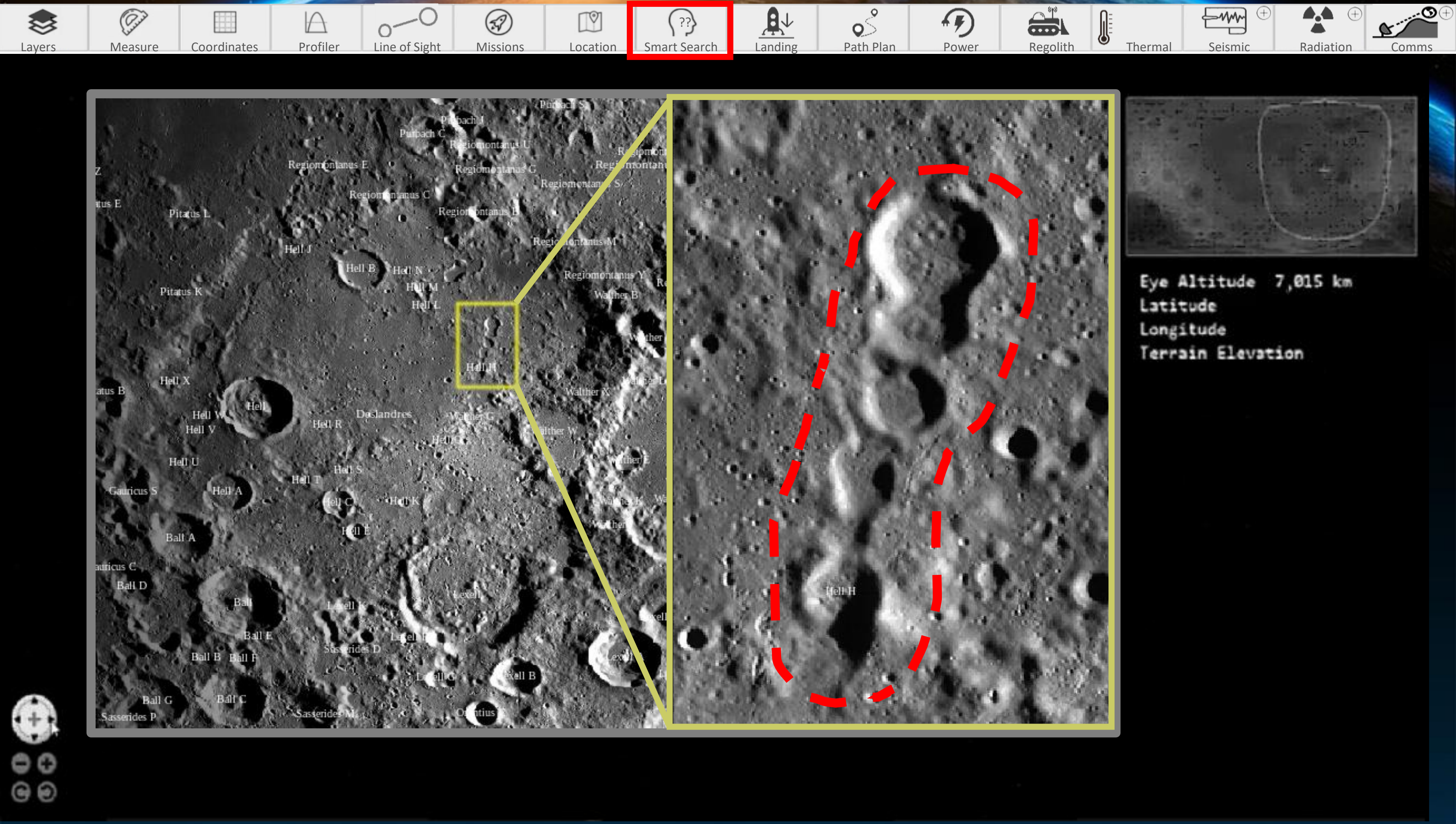
Digging Down: AI Assisted Pipeline for Subsurface Data

- Use local surface features (e.g. rim outline) to georectify subsurface data.
- Use large scale features to fine tune georectification





Eye Altitude 7,015 km
Latitude
Longitude
Terrain Elevation



Eye Altitude 7,015 km

Latitude

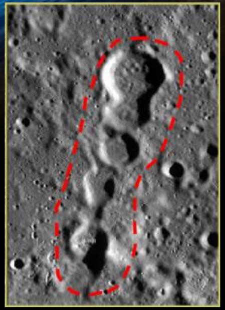
Longitude

Terrain Elevation

Layers | Measure | Coordinates | Profiler | Line of Sight | Missions | Location | Smart Search | Landing | Path Plan | Power | Regolith | Thermal | Seismic | Radiation | Comms

Smart Search

Change



Find Similar

Copernicus Catena

Davy Catena

Mendeleev Catena

Giordano Bruno Catena

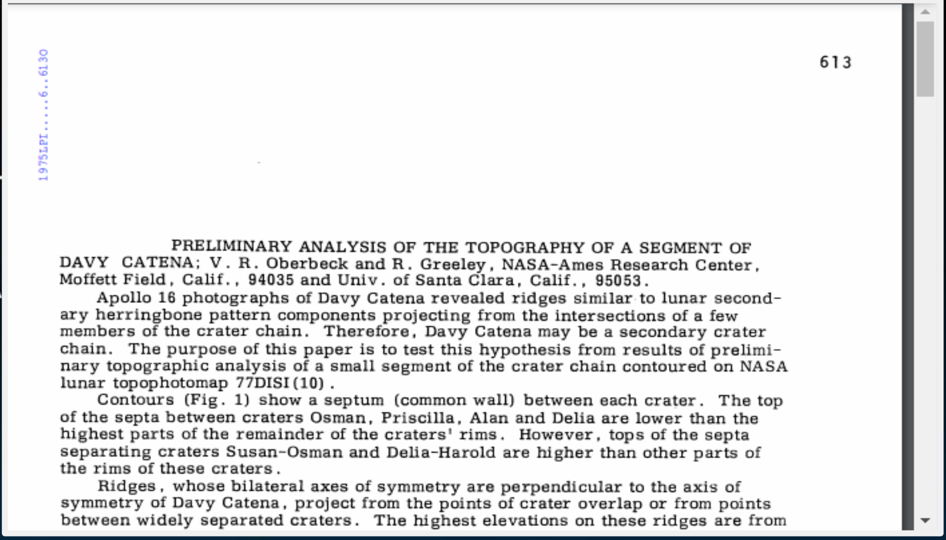
Mare Orientale Catena

Tadpole Catena

More

Related research

- PDF *Preliminary Analysis of the Topography of A Segment of Davy Catena* VR Oberbeck, R Greeley - Lunar and Planetary Science ..., 1975 - adsabs.harvard.edu
- PDF Wichman, et al. "*The Davy Crater Chain: Implications for tidal disruption in the Earth-Moon System and elsewhere*." Geophysical research letters 22.5 (2016): 583-586.
- PDF Melosh, H. J., and E. A. Whitaker. "*Crater Chains on the Moon: Records of Comets Split by the Earth's Tides?*." Lunar and Planetary Science Conference, Vol. 25. 2014.
- PDF Eppler, Dean B., and Grant Heiken. "*Lunar crater chains of non-impact origin*." Proceedings of the Sixth Lunar Science Conference: Houston, Texas, March 17-21, 2014.
- PDF Kling, C. L., et al. "Field-Based Assessment of Pit Crater Chains." Lunar and Planetary Science Conference. No. 2132. 2019.
- PDF Wilhelm, Thorsten, et al. "Unsupervised Learning of Scene Categories on the Lunar Surface." VISIGRAPP (5: VISAPP). 2019.



613

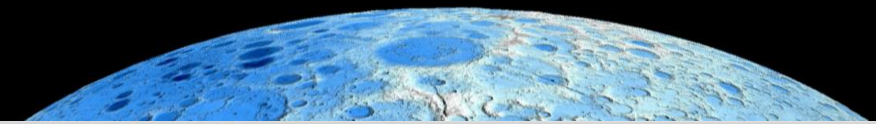
1975LPI.....6..613O

PRELIMINARY ANALYSIS OF THE TOPOGRAPHY OF A SEGMENT OF DAVY CATENA; V. R. Oberbeck and R. Greeley, NASA-Ames Research Center, Moffett Field, Calif., 94035 and Univ. of Santa Clara, Calif., 95053.

Apollo 16 photographs of Davy Catena revealed ridges similar to lunar secondary herringbone pattern components projecting from the intersections of a few members of the crater chain. Therefore, Davy Catena may be a secondary crater chain. The purpose of this paper is to test this hypothesis from results of preliminary topographic analysis of a small segment of the crater chain contoured on NASA lunar topophotomap 77DISI (10).

Contours (Fig. 1) show a septum (common wall) between each crater. The top of the septa between craters Osman, Priscilla, Alan and Delia are lower than the highest parts of the remainder of the craters' rims. However, tops of the septa separating craters Susan-Osman and Delia-Harold are higher than other parts of the rims of these craters.

Ridges, whose bilateral axes of symmetry are perpendicular to the axis of symmetry of Davy Catena, project from the points of crater overlap or from points between widely separated craters. The highest elevations on these ridges are from



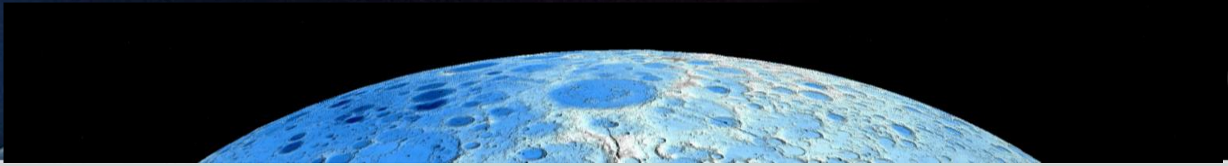
Summary

- CMS is a user-interactive environment to visualize and analyze data on celestial bodies, with a current focus on the Moon to support NASA’s mission priorities.
- CMS has an open architecture, allowing 3rd party integration of maps, rendering engines and specialized analytics.
- CMS is also “*data open*” with advanced data import/export and robust data layer management.
- AI capabilities are being integrated into CMS to automate the data ingest process for a wide range of datatypes, including future subsurface constructs.
- Support for subsurface data layers, such as lava tubes and natural resource deposits, are in development.

Questions ??

parul.agrawal-1@nasa.gov
allison.f.zuniga@nasa.gov
graham.mackintosh@nasa.gov

Supporting Material



Step 3) Georectify: AI To find matching control points

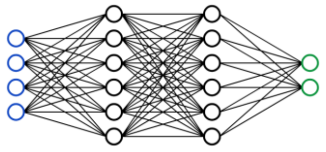
Don’t use the image… train the neural net using control point coordinates.

Image #1

[(0, 35), (136, 49), (212, 50), (142, 54), (166, 69), (283, 74), (206, 78), (299, 84), (278, 87), (85, 108), (59, 190), (102, 198), (0, 210), (83, 226), (101, 292), (125, 316), (0, 344), (146, 365), (276, 375), (248, 391), (282, 392)]

Image #2

[(194, 0), (233, 0), (280, 4), (50, 23), (157, 38), (277, 39), (79, 44), (129, 75), (267, 79), (125, 84), (160, 88), (139, 97), (149, 102), (355, 103), (144, 111), (275, 114), (330, 121), (292, 124), (224, 125), (452, 126), (185, 130), (388, 133), (469, 178), (490, 182), (220, 188), (133, 189), (200, 195), (148, 198), (139, 200), (128, 221), (231, 232), (175, 298), (174, 311), (445, 352), (575, 403), (446, 432), (488, 466), (500, 491), (282, 498), (309, 521), (406, 530), (506, 550), (399, 552), (291, 564), (285, 570), (480, 571), (446, 572), (543, 572), (540, 582), (340, 584), (640, 590), (503, 598), (422, 619), (381, 630), (455, 639), (547, 667), (547, 685), (553, 712)]

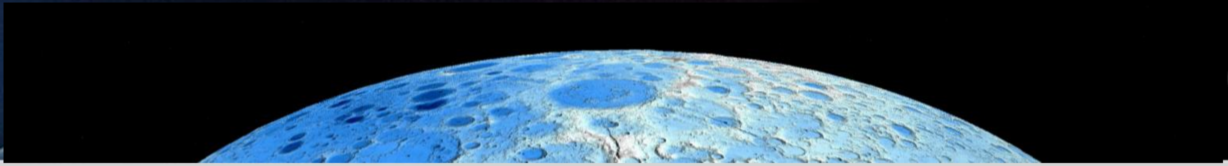


Trained by minimizing the number of incorrect match-ups.

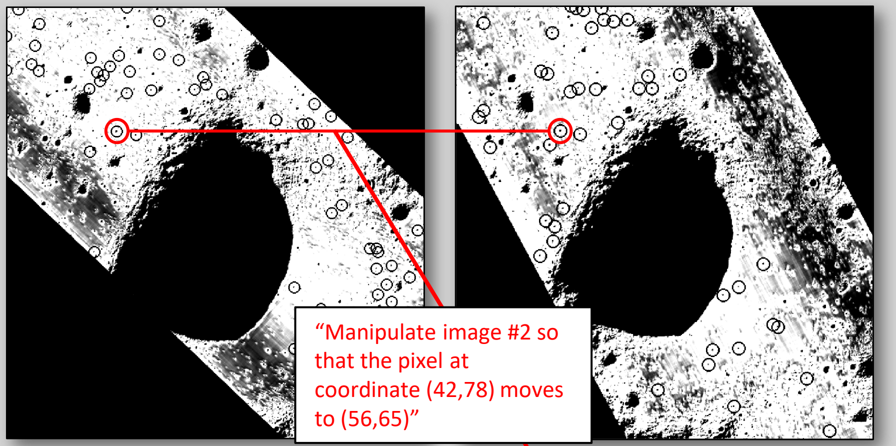
Output is a vector of matching control points. Zero = no match.

[0, 0, 9, 0, 0, 26, 0, 0, 0, 0, 39, 0, 0, 5, 46, 0, 0, 0, 0, 8, 14, 0, 0, 0, 0, 17, 0]

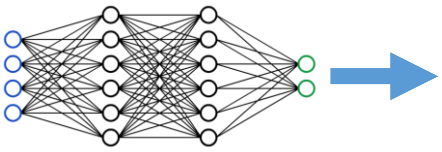
3rd control point in image #1
is the same as
9th control point in image #2



Step 4) Image Transformations to Achieve Pixel-perfect Overlap of All Control Points

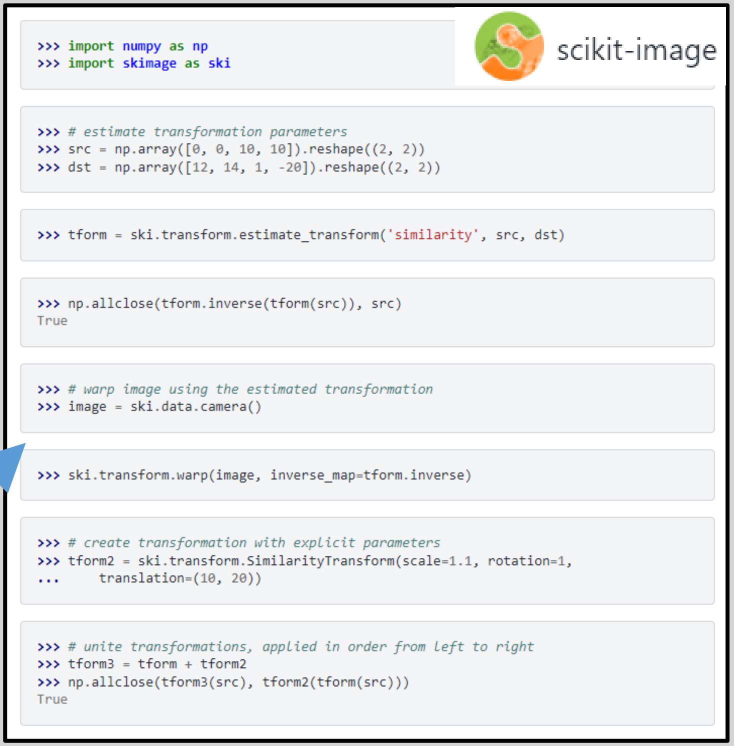


"Manipulate image #2 so that the pixel at coordinate (42,78) moves to (56,65)"



[[(56,65), (42,78)],
[(17,41), (8,33)],
[(204,178), (195,182)],
[(217, 45), (197, 52)]]

List of matching control point pairs

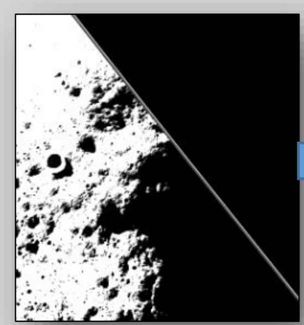


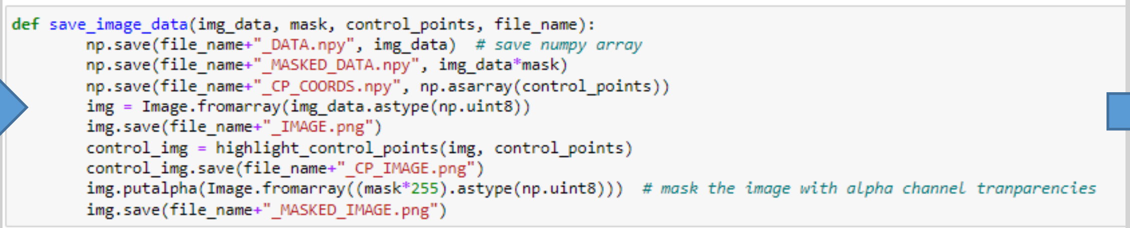
```
>>> import numpy as np
>>> import skimage as ski

>>> # estimate transformation parameters
>>> src = np.array([0, 0, 10, 10]).reshape((2, 2))
>>> dst = np.array([12, 14, 1, -20]).reshape((2, 2))

>>> tform = ski.transform.estimate_transform('similarity', src, dst)

>>> np.allclose(tform.inverse(tform(src)), src)
True

>>> # warp image using the estimated transformation
>>> image = ski.data.camera()

>>> ski.transform.warp(image, inverse_map=tform.inverse)

>>> # create transformation with explicit parameters
>>> tform2 = ski.transform.SimilarityTransform(scale=1.1, rotation=1,
...     translation=(10, 20))

>>> # unite transformations, applied in order from left to right
>>> tform3 = tform + tform2
>>> np.allclose(tform3(src), tform2(tform(src)))
True
```

Many potential open-source libraries (e.g. scikit)

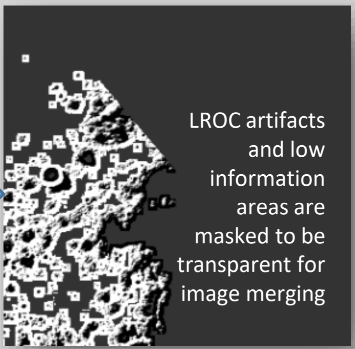
Step 5) Intelligent Merging of Overlaps for Optimal Information Gain

Mask areas of the image being merged in that are of a lower info density than the aggregate image built up below it

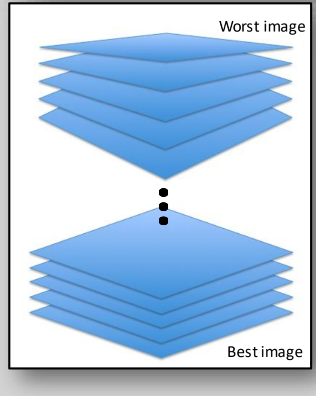




```
def save_image_data(img_data, mask, control_points, file_name):
        np.save(file_name+"_DATA.npy", img_data)  # save numpy array
        np.save(file_name+"_MASKED_DATA.npy", img_data*mask)
        np.save(file_name+"_CP_COORDS.npy", np.asarray(control_points))
        img = Image.fromarray(img_data.astype(np.uint8))
        img.save(file_name+"_IMAGE.png")
        control_img = highlight_control_points(img, control_points)
        control_img.save(file_name+"_CP_IMAGE.png")
        img.putalpha(Image.fromarray((mask*255).astype(np.uint8)))  # mask the image with alpha channel tranparencies
        img.save(file_name+"_MASKED_IMAGE.png")
```



LROC artifacts and low information areas are masked to be transparent for image merging


Worst image
Best image