3D Data Formats: A Field Guide for Geospatial Folks

Emma Krantz

Photo by Michael Liao on Unsplash

3D Geospatial Data Formats Field Guide

https://github.com/TerriaJS/3D-Geospatial-Data-Formats-Field-Guide

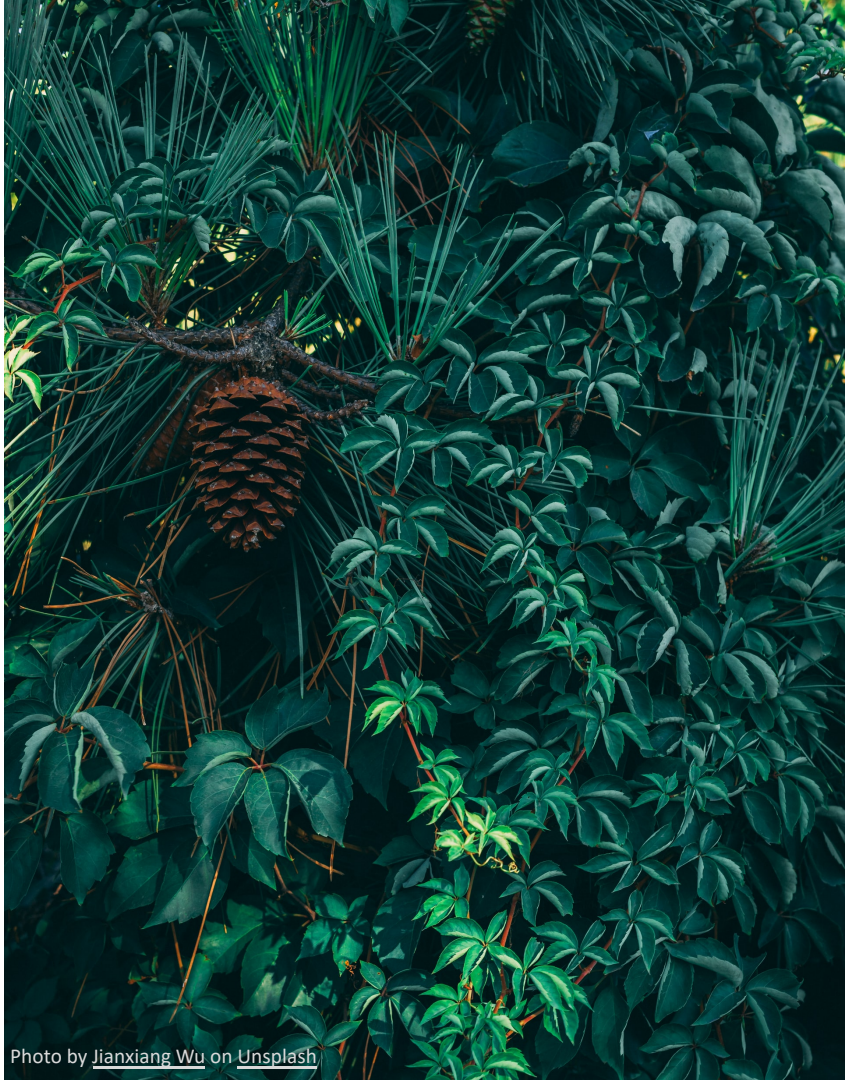


Photo by Jianxiang Wu on Unsplash

TerriaJS

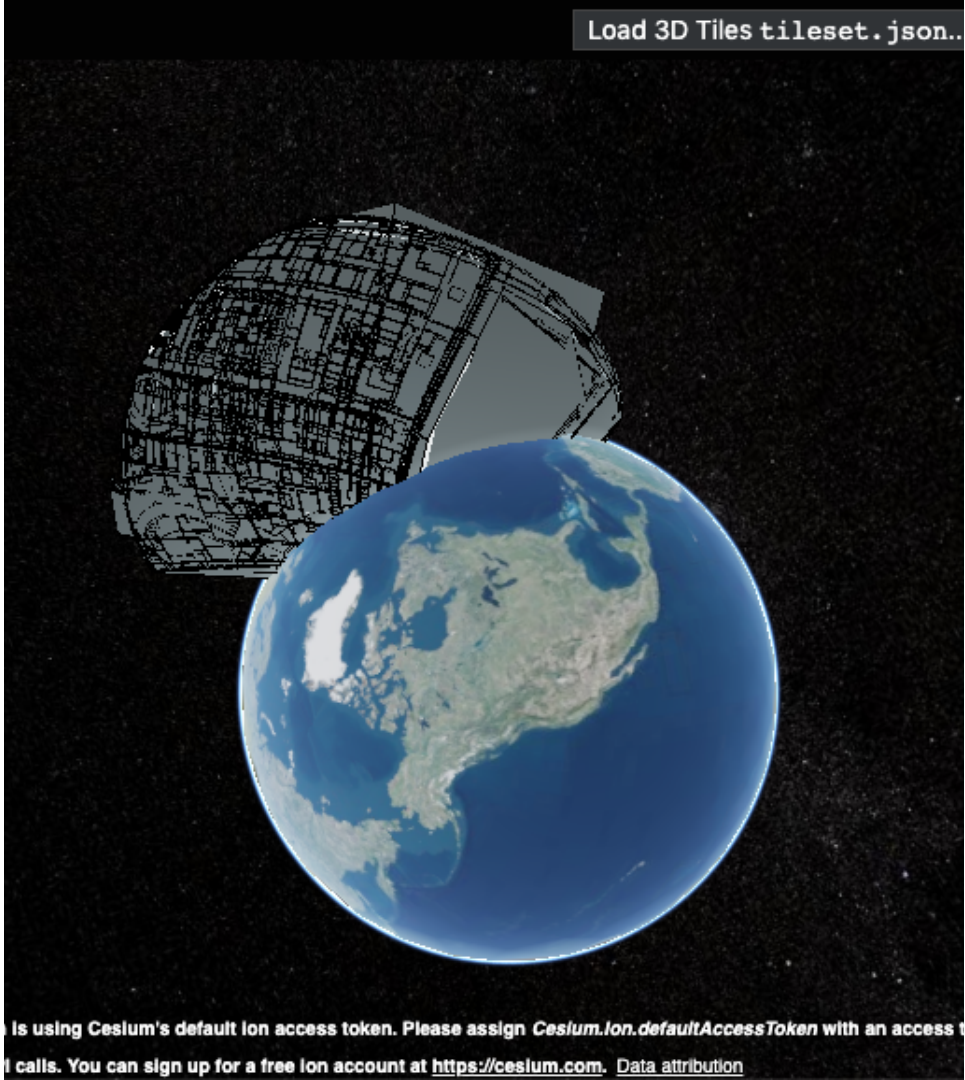
- An open-source framework for web-based geospatial catalogue explorers
- Cesium-based
- Along with NSW Department of Customer Service's Spatial Services, we built the NSW Digital Twin
- And the QLD Digital Twin with the Department of Natural Resources, Mines and Energy

CesiumJS

- The 3D engine that Terria uses to render things
- Web-based, open source
- Flexible, designed to be used in many different applications
- Great support for terrain, showing datasets on terrain
- Terria adds support for additional data formats, a workbench, a catalogue explorer and more.

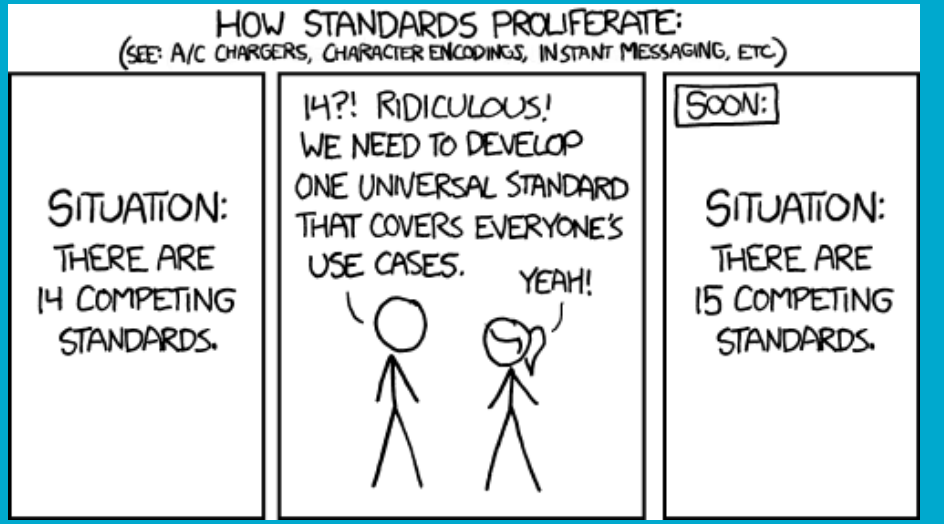
3D is hard

- 3D geometry is way harder than 2D geometry
- Major open source tools are primarily designed for 2D
  - QGIS
  - GDAL
  - GeoServer
- Significantly less standardization for 3D data formats
  - 3D OGC specifications exist, but they're much less widely adopted and supported



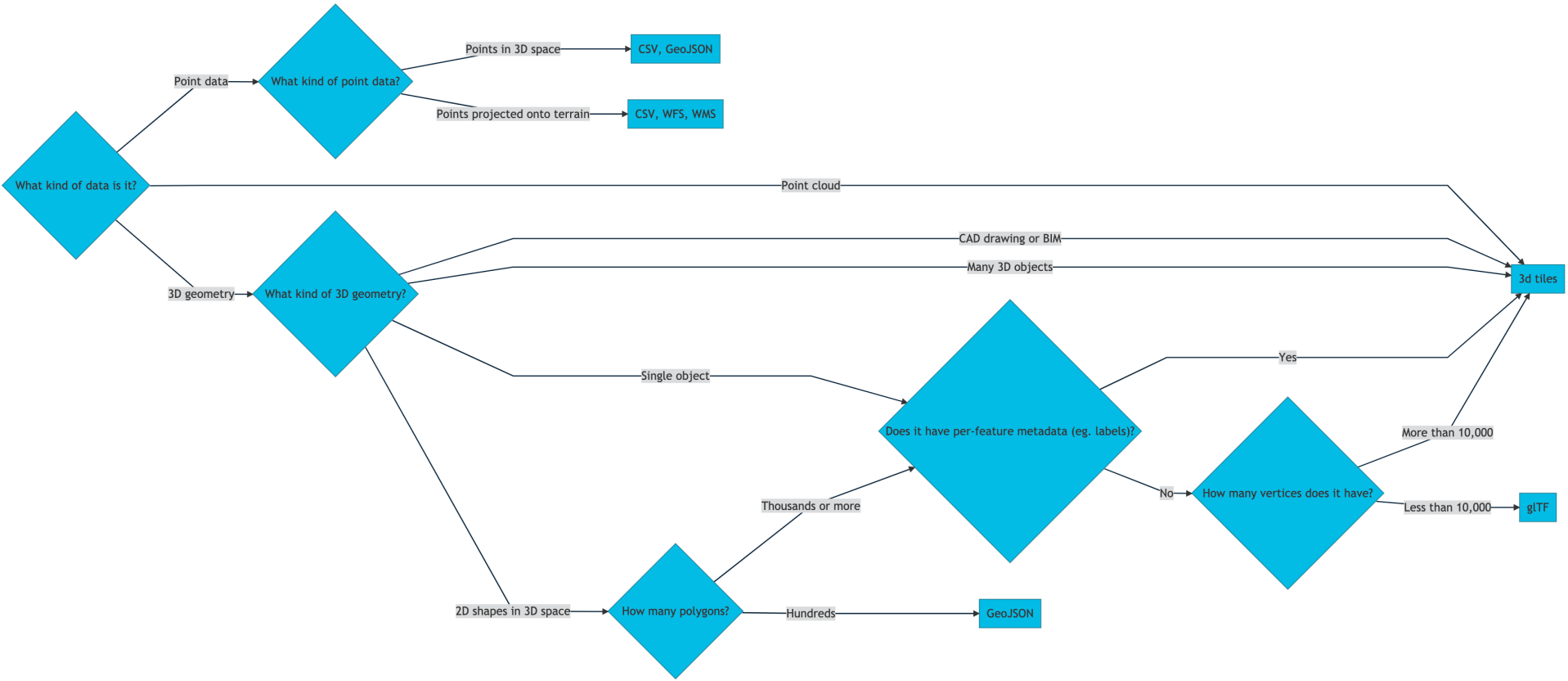
Formats, formats, formats!

- 4 categories of formats:
  - Modelling
    - glTF, COLLADA, obj, fbx…
  - CAD
    - rvt, IFC, dwg, 3ds, stl, dxf…
  - Point cloud
    - las, laz…
  - Geospatial
    - GeoJSON
    - DEM
    - i3s, 3d tiles
    - Geodatabases
    - …



https://xkcd.com/927/

Converting data for Terria



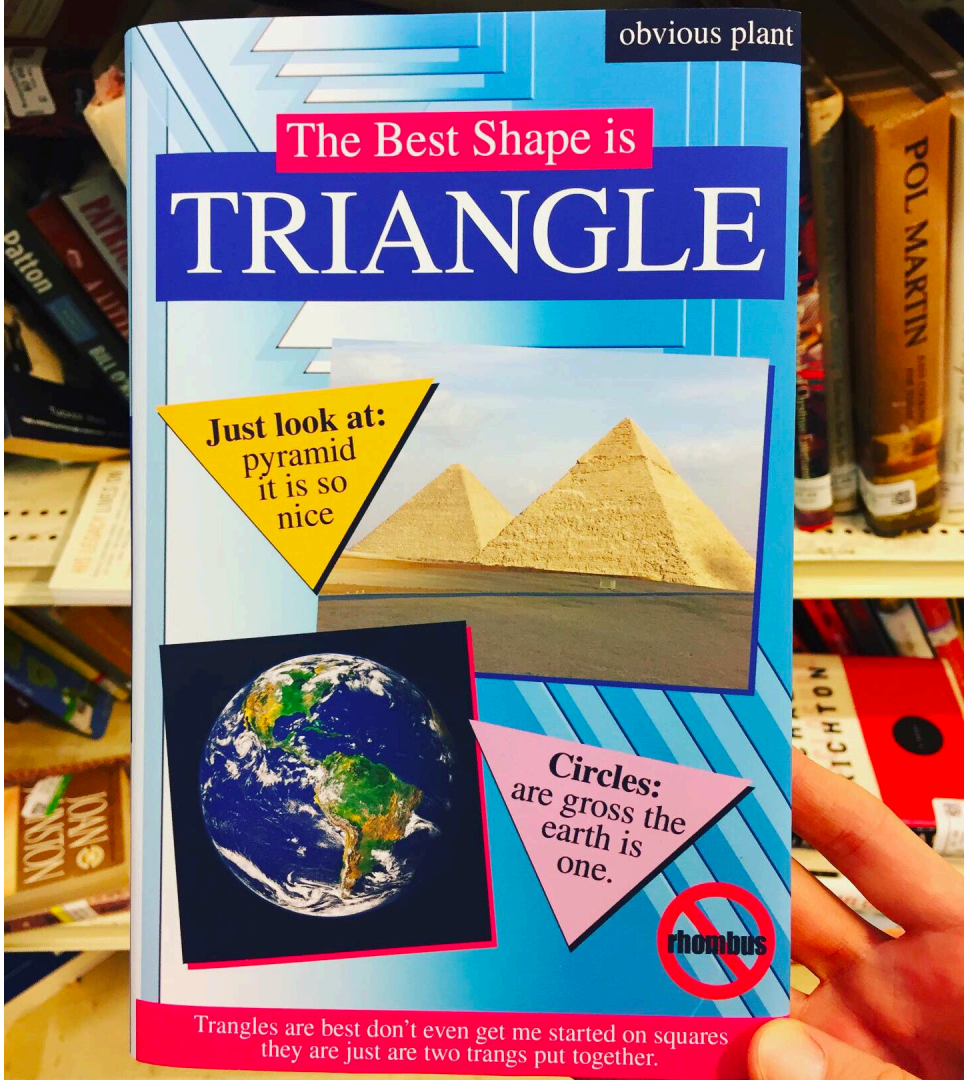
Things to consider when picking a format

1. What does the visualization platform support?
2. What information does the format encode?
3. Performance
4. Open source vs proprietary format

Photo by Milan Seitler on Unsplash

3D model performance

- The closer your source data is to the raw vertex data that gets passed to your graphics card, the better.
- glTF is great for this because it's a JSON description of the layout of the buffer and then the actual buffer
- i3s and 3d tiles both use this kind of representation
- Otherwise, some kind of triangulation has to happen— eg. for GeoJSON



3d Tiles

- OGC specification
- Similar to (but not interoperable with) Esri's i3s/slpk format.
- Core principle: for large datasets you don't have to load everything at once
- Splits a 3D model into tiles, storing each part of the model in multiple levels of detail.
- You can load a model that was originally 100s of GBs in a web browser!
- Supports more than just 3D models eg. point clouds

My favourite formats

- For 3D models: glTF
- For big 3D models: 3d tiles
- For CAD: IFC
- Fallback: GeoJSON

But I don’t get to choose the source format!

- Trying to convert between 3D data formats sucks.
- We had to write our own converters
  - i3s-to-3d tiles
  - BIM Chicken – ifc-to-glTF

https://thedottyhouse.co.uk/product/with-deepest-sympathy-card-2/

Some handy converters

- For converting from IFC: ifcopenshell.org
- For point clouds: pdal.io ❤️
- For converting to glTF:
  - KhronosGroup/COLLADA2GLTF
  - Blender's glTF exporter
  - CesiumGS/obj2gltf
- Not open source, but great: Cesium Ion, for turning 3d models into 3d tiles.

Georeferencing

- I wrote a tutorial blog post on georeferencing models for Cesium: https://medium.com/terria/georeferencing-3d-models-for-cesium-7ccf609ee2ef
- For portability and maximum precision, keep models in local ENU or similar local cartesian coordinates, then transform to mapping platform coordinates on load.

Photo by Capturing the human heart. on Unsplash

Get in touch!

Data61
Emma Krantz
Data Engineer

emma.krantz@csiro.au
terria.io
Forum: https://github.com/TerriaJS/terriajs/discussions

https://github.com/TerriaJS/3D-Geospatial-Data-Formats-Field-Guide