STRUGGLE WITH WEBGL TO RENDER VECTOR DATA AND CLOUD OPTIMIZED GEOTIFF

How to manage massive amounts of vector and raster data using WebGL and do not die trying

MANUEL JESÚS MORILLO JIMÉNEZ
GUADALTEL
manueljmorillo@guadaltel.com
JOSÉ ENRIQUE SORIANO SEVILLA
GUADALTEL
jenriquesoriano@guadaltel.com

ABSTRACT: Geospatial information and its treatment has evolved along time from the centralization and publication of data in a single repository via standards such as WMS to the service of the information as it is to be processed by browsers via WFS. This evolution, together with an improvement in the technical capabilities of the devices and communication infrastructure, has led mapping clients to become more autonomous when it comes to managing geospatial information. Geospatial data has gone from being managed on the server to being served directly to the mapping client to be exploited by it.

However, the WFS protocol has some shortcomings in terms of performance when it comes to the format in which to serve the information, giving way to more optimal formats for the service of vector information such as .pbf. This format allows the transmission of large amounts of data to the local browser client. This information, increasingly larger, requires the use of specific rendering libraries such as WebGL. WebGL provides a plus of capacity at the time of managing and rendering the data in the client mapping, making optimal performance of the capacities of the devices.

To this end, the libraries analysed have been tested with real cases of massive spatial information management that requires a significant effort to render due to the volume of variables and data used in the spatial data analysed. The present work shows a state of the art of the existing WebGL libraries and a real test field on which the data have been tested, showing the results obtained and the most optimal solution.

Similarly to the vector data, the spatial information of images has evolved through different formats until reaching its optimization for the service and its exploitation through the Cloud Optimized GeoTIFF (COG) format. This format allows the obtaining of the imagery information by means of techniques similar to streaming from a standard request.

The following frameworks have been considered for the representation of large amounts of data:

- OpenLayers
- Mapbox GL js
- Deck GL

- kepler.gl

Resulting on the tests executed to represent large amount of data, Mapbox GL has revealed as the more flexible tools in terms of performance and capabilities.

KEYWORDS: OpenLayers, deck gl, mapbox gl, kepler gl, Cloud Optimized Tiff, COG