Scientific Research Publishing

Journal of Geographic Information System, 2023, 15, 515-523
https://www.scirp.org/journal/jgis
ISSN Online: 2151-1969
ISSN Print: 2151-1950

# Visualization of Vector Tiles on CesiumJS Virtual Globe

**Kang Kuk*, Woo Kyongil, Yun Cholnam, Kim Wonsok**

Institute of Remote Sensing and Geoinformatics, Academy of Science, Pyonyang, The Democratic People's Republic of Korea
Email: *kkuk628@outlook.com, wugis1219@gmail.com, bbanzzakji@gmail.com, jinyx1976@hotmail.com

**How to cite this paper:** Kuk, K., Kyongil, W., Cholnam, Y. and Wonsok, K. (2023) Visualization of Vector Tiles on CesiumJS Virtual Globe. *Journal of Geographic Information System*, **15**, 515-523.
https://doi.org/10.4236/jgis.2023.155025

**Received:** August 16, 2023
**Accepted:** October 13, 2023
**Published:** October 16, 2023



## Abstract

Essentially, CesiumJS—which can be accessed through the link, http://cesiumjs.org, is an open-source JavaScript library for creating virtual globe environment in performance effective, high quality of rendering, precision, and user friendly. It is a wonderful tool for 3D-themed visualizations of earth. CesiumJS has a number of data sources, but none of them supports vector tile format. This article explains how to visualize Mapbox vector tiles in a CesiumJS virtual globe environment. CartoDB/BigQuery hosts vector tiles, and a process for producing vector tiles from massive vector data using the BigQuery tiler of CartoDB has been provided.

## Keywords

Mapbox, 3D Visualization, BigQuery, CartoDB, Virtual Earth

## 1. Introduction

Vector data is used extensively in geographic information systems (GIS) to analyse and administrate virtual landscapes. Methods that enable the integrated visualization of landscape and geospatial vector data are thus necessary. Usually, these data are set up as either raster or vector data. Geometry is represented via vector data as a set of coordinate lists for points, lines, and polygons. Eliminating extra or undesired detail from the output image is a crucial challenge for computer graphics and GIS communities. GIS geometric simplification is one of the more general approaches, which are ways to graphically depict a specific geographic entity or group of geographic entities using various visuals when showing the entities at various map scales [1].

Rendering vector data representing features like road networks, political districts, and rivers is crucial in GIS systems. It is difficult to render vectors in a way that allows them to drape perfectly over topography. Vector data can be ren-

dered on a 3D model using one of two common methods. Both represent the vector data as primitives of geometry or pre-render the vector data to image with style. We contend that vector data need to be handled separately from raster data for various reasons and that the graphics pipeline should render it as distinct geometry. Modern methods create a hurdle since they produce a LOD terrain whose constituent triangles change every frame. Therefore, the vector geometry must likewise change at each frame for the vector data to display accurately on the 3D mesh [2].

A common area of thematic cartography is the 3D visualization of data on a virtual globe. Since the debut of Google Earth, one of the most prevalent methods of releasing such material has been the creation of KML overlays. For several years, the Google Earth plugin made it straightforward to publish these visualizations based on KML to online. Ultimately, the plug-in technology grew antiquated, and Google had discontinued this service by the end of 2016. With the advent of new technologies such as HTML5 and WebGL, a number of JavaScript 3D globe frameworks and APIs were created to make it easier to show 3D interactive globe models on websites without needs of any plugins [2].

OpenWebGlobe, developed by Swiss researchers, is one such library, as is the WebGL Earth JavaScript API, which began as a bachelor thesis and is now maintained by Klokan Technologies. Although these two programs have their uses, using the CesiumJS API is preferable due to the limitations of these two products. CesiumJS has far more capability, superb documentation, and a hot development community, guaranteeing that most issues are swiftly resolved on programmer forums. This is not the case with the two products mentioned above. Every industry uses CesiumJS, which enables 3D web visualization on any device and supports CAD, web, BIM, point clouds, CityGML, and photogrammetry. These businesses range from real estate to sports management and engineering. But when it became increasingly difficult to render and stream 3D visualizations effectively over time, the CesiumJS community created an open data format called 3D tiles that supports streaming and is tailored for rendering. CesiumJS is widely used as it has the benefits of being cross-device, cross-platform, and dynamic. It enables rendering a lot of objects on both the 3D globe and map without requiring the installation of any browser extensions. One option to use CesiumJS is available: either convert data to the Cesium 3D Tiles format and create the visualization using the CesiumJS library or transmit the data to the Cesium Ion platform, which streams the data and creates the visualization on its own. The development of cameras and geometric objects, as well as the integration of multiple data sources, are all made possible by CesiumJS, the only open-source software that has good user community maintenance.

Tiles can be styled dynamically because they are vectors, which enables a broad variety of map styles on global data. Using Mapbox GL JS or the Mapbox Maps SDKs for Android and iOS, it can dynamically modify the look of map without downloading new tiles [3].

A small vector tile makes it possible to create high-resolution global maps, load them quickly, and cache them effectively. The Mapbox cartographers in charge of the vector tilesets that Mapbox offers put a lot of effort into finding a balance between accuracy and effectiveness [4].

It can quickly re-render the map because all of map data is loaded in the client side, enabling seamless zooming, tilting, and rotation.

One can easily use dynamical querying feature to access map's characteristics. The integration of PostGIS with MapTiler Server is based on the idea of layers. As a result of calling the ST_AsMVTGeom function, each layer is represented by a separate SQL query that returns geometry in tile coordinate space. The ST_AsMVT function can create a binary MapBox Vector Tile by passing this geometry, which MapTiler Server anticipates will be labeled "mvtgeometry", to the function [5].

CartoDB provides unique geographical visualization, analysis, and app creation capabilities directly within Google Cloud's data warehouse. It utilizes geographic data natively with BigQuery while using straightforward SQL to create attractive visualizations. BigQuery can also execute powerful spatial analytics and generate amazing maps from billions of information. This paper presents a technique for visualizing Map-box vector tiles in a 3D virtual environment. The hosting of vector tiles on CartoDB/Bigquery and the process for producing them are detailed.

## 2. Related Work

There are various custom methods used to load vector tiles on CesiumJS. There have only been a few studies on visualizations that employ approaches to vector-based tiling. An overview of current techniques for storing, processing, and displaying data in a geo-graphic environment is provided in this section [6].

A survey on how large data is managed to utilize web technologies was written by Vitolo *et al.* [7]. Data collections expand quickly as computing science develops. New technologies are required to handle such a massive volume of data. He explains how this must be possible regarding analysis methods, labour processes, and dataset interaction. He highlights the current web technologies used to process straightforward datasets in the paper. Additionally, maintaining a relationship to standards maintained by the Open Geospatial Consortium (OGC) while handling more complex data becomes more difficult.

Yue & Jiang ([8]) explores big data in the context of geographic information systems, whereas Vitolo concentrates on big data and web technologies. According to him, it is crucial to support the geospatial domain because GIS software, in particular, needs to be capable of handling big datasets containing geospatial data.

Most interactions between end users and these datasets occur through desktop GIS programs. To the best of our knowledge, more needs to be done to develop web-based apps that can manage huge data and give clients and end users access regardless of the platform.

The most used geographic data storage technology is PostGIS. Other new technology-based alternatives are available, but because of their simplicity and the scarcity of geospatial computer scientists, people frequently remain with the tried-and-true PostGIS database. It is also frequently necessary to use OGC standard data formats, which facilitates the implementation of PostGIS applications [9].

Even though tiling is a fairly old idea, it is nonetheless cutting-edge. This technology must accommodate future demands. Currently, pre-rendered raster images are used to store tiles. Tile caching is a subject that is frequently brought up while discussing tiling. In addition to server-side caching for raster images, vector tiles, such as non-transmission-optimized GeoJSON vector tiles, can also be cached.

Most approaches concentrate on a single subject rather than combining various concepts. One method involves using simple queries to retrieve data from the back end and compressing files for delivery. However, allowing users to browse over the data does not assist. The servers must be specifically queried for a specific subset of data. Tile based approaches to creating a map are primarily concerned with tiling rather than with trying to compress or cache vector tiles without using much memory at every stage of the process [10].

## 3. Materials and Methods

Data processing for vector data hosting on Carto/Bigquery was done (**Figure 1**).

Occasionally, it is impossible to display data legibly at a particular zoom level. For in-stance, a thick succession of topographic lines would appear as a tangle of objects when seen at a low zoom level. In contrast, low-resolution data might appear excessively coarse at high zoom levels. The Mapbox Uploads API analyses your data and automatically determines the maximum and minimum zoom levels at which tiles should be rendered to prevent both potential concerns. The uploaded image resolution determines raster tile sets' minimum and maximum zoom levels. The tile set will render at more zoom levels if the images have a higher resolution.

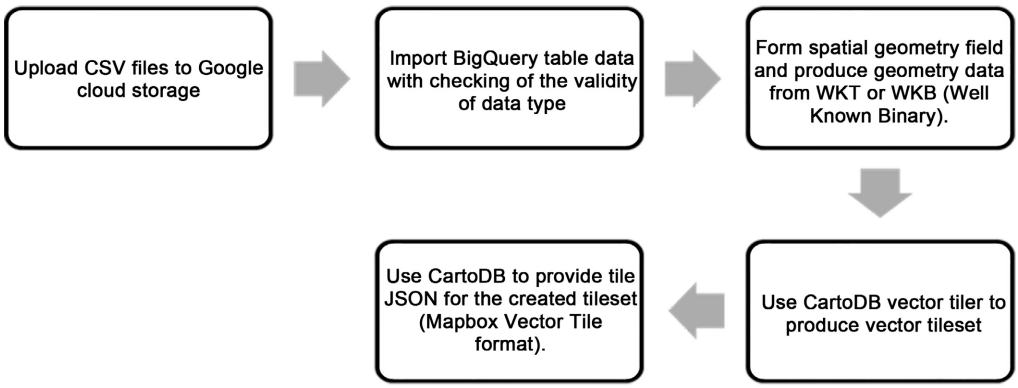


**Figure 1.** Generating Mapbox vector tiles in BigQuery.

Tile imagery was generated with style and applied to CesiumJS globe as an imagery layer using Mapbox Vector Tile Basic JS Renderer. In the newest version (1.2.1), Mapbox throws a warning (Chrome, Mac): Geometry exceeds the allowed extent, reduce your vector tile buffer size. This warning is actually coming from the vector tiles themselves, so the recommended way to avoid it is to set smaller buffers when generating tiles. This limitation and warning aren't new, but at one point, the threshold will be stabilized [11].

Simplifying the map at lower zoom levels lowers complexity in areas where the details are unnoticeable. This simplification increases the map's loading speed.

Sometimes this occurs immediately after loading, at least during map navigation. A maximum amount of data can exist in a single vector tile. By reducing complex vector features during the upload process, we ensure that each tile in your vector tile set falls be-low this maximum size and displays accurately on your maps.

When the zoom level exceeded maximum built zoom, vector data was inserted as a primitive with a custom style or 3D model. Normally vector tiles are built to level 14 as the maximum, and mapping software over-zooms with no loss in quality.

The number of vector objects is significantly lower at high-level zoom than at low-level zoom. Therefore, when it is low level, we render it as an image; when it is high level, we render it as a vector to avoid performance issues. Additionally, it will offer options for the primitive shown in CesiumJS when rendering vectors (Primitives). In the case of point-type vector data, it will be a 2D shape or a 3D model.

Generally speaking, this notice can be disregarded if the end-user is not experiencing any other problems. The correct tiles on the maximum zoom of the tile set must be chosen when the current reference zoom is greater than the maximum zoom of the vector tile set. When the source's maximum zoom level is set, Mapbox-Gl will be instructed to keep zooming after that point rather than requesting the source at all zoom levels.

The speed at which vector tiles are generated is slow or unavailable because it may interfere with the display of other layers depending on the source of the data or the number of vector tiles in a layer at a given zoom level. In other words, the process may not be able to produce enough vector tiles as needed.

**Figure 2** shows the process shows the overall process from the tiling of vector data to the visualization on the CesiumJS virtual globe.

## 4. Results

In terms of thematic content, the prospects of 3D visualization are limitless. There are many more visual variations than in the context of 2D visualization. It is different from saying that it has to overload the scene with data rather, displaying too many items on the globe or employing too many visual objects at the

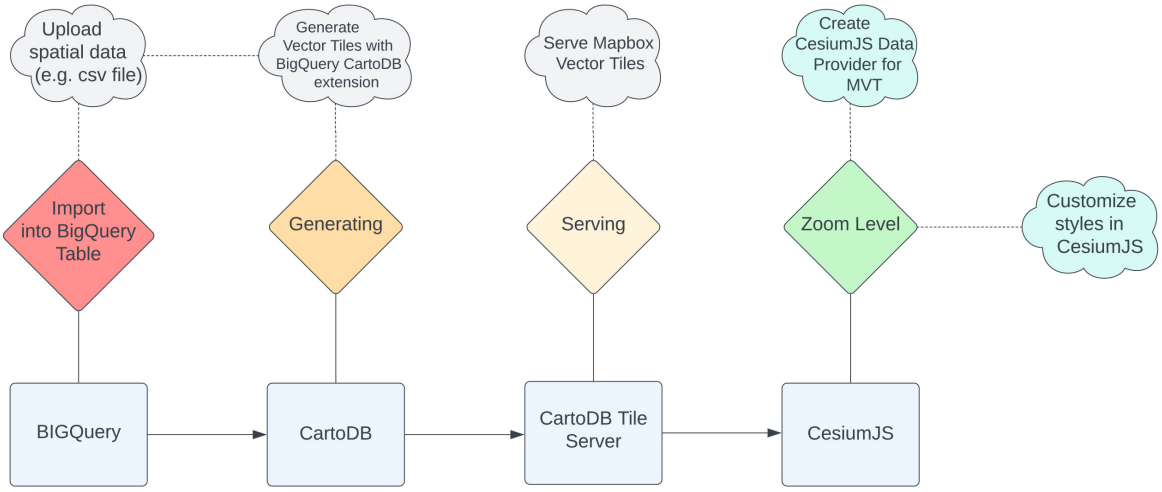


**Figure 2.** Work flow to render vector tiles in CesiumJS.

same time may lead to data loss. Most theme globes, however, only employ one of the three most prevalent depiction methods: choropleths, 3D prism maps, and graded 3D symbols. To integrate an interactive virtual globe into an HTML website using CesiumJS API, it can be implemented without having to write a lot of code. This is detailed in the CesiumJS API documentation, particularly in the instructional section. After we have a virtual globe on our website, we can add various sorts of themed visualizations to it in various ways based on the dataset format.

The findings imply that the suggested CartoDB vector tiler, as opposed to the conventional vector database method, can assist us in more effectively visualizing vector tiles on CesiumJS Web Globe. A vector tile at a lower level (large scale) can be requested and seen on a 3D terrain surface, making these benefits even more obvious.

The selection of appropriate objects from vector tile for a certain zoom level is another generalization aspect. There are two different ways in the terms of the custom BigQuery query developed to implement this aspect. The first is using a field to keep zoom level which the data shall be rendered. The field can be type of Integer representing a zoom level or string type to classify vector objects.

Vector data can be rendered in three dimensions using a geometry-based or texture-based method after the vector tiles have been parsed. Elevation interpolation techniques have been employed to convert the vector tiles into three dimensional coordinates in the geometry-based method. The vector data can be rendered on a terrain surface using texture-based technique. The vector tiles should be pre-rendered into tile images before mapping them onto a 3D terrain surface with texture mapping techniques to when using a texture-based approach to view the vector tiles.

User configured map styles can be merged into the pre-rendered tile image when a texture-based technique is used to render the vector tiles. Although there are no significant technical issues in the geometry-based approach to render

these basic symbols, the user experience is negatively impacted by the low rendering efficiency, especially when working with global vector map data. The result of texture-based rendering of huge amount of vector data is shown at low zoom level in Figure 3.

The next method is geometry-based approach, where point data have been rendered as 3D object, in relation to the zoom level, controls visibility of them (Figure 4). One of its advantages is that vector tiles carry the data we need to visualize on them so we can rapidly access them using APIs. The properties attribute of a vector tile is where data are kept. We can develop more adaptable data-driven visualizations by gaining access to tile attributes.

The CesiumJS web globe is a very user-friendly platform, making it advantageous to utilize it to immediately process the vector tiles because the user can readily visualize the data. Utilizing the CesiumJS web globe has the drawback of not analyzing massive amounts of data. A GIS program like ArcGIS or QGIS has

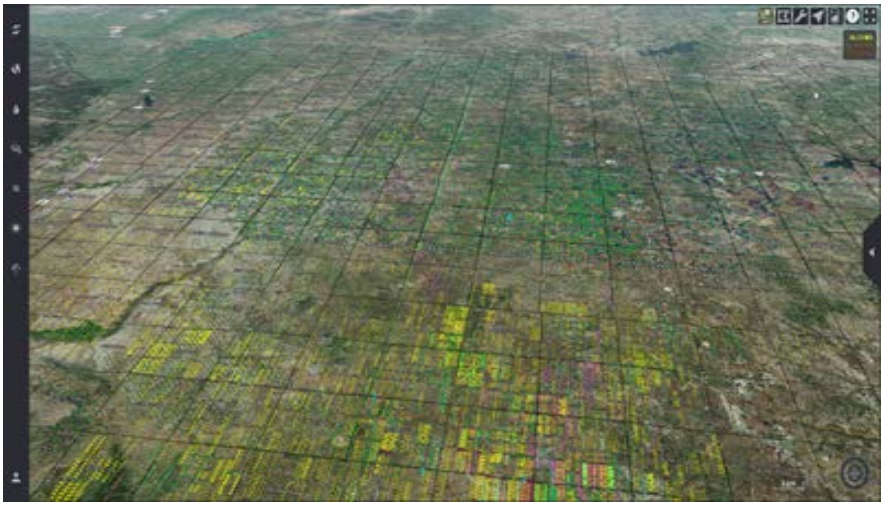

Figure 3. Texture based rendering of vector tile.

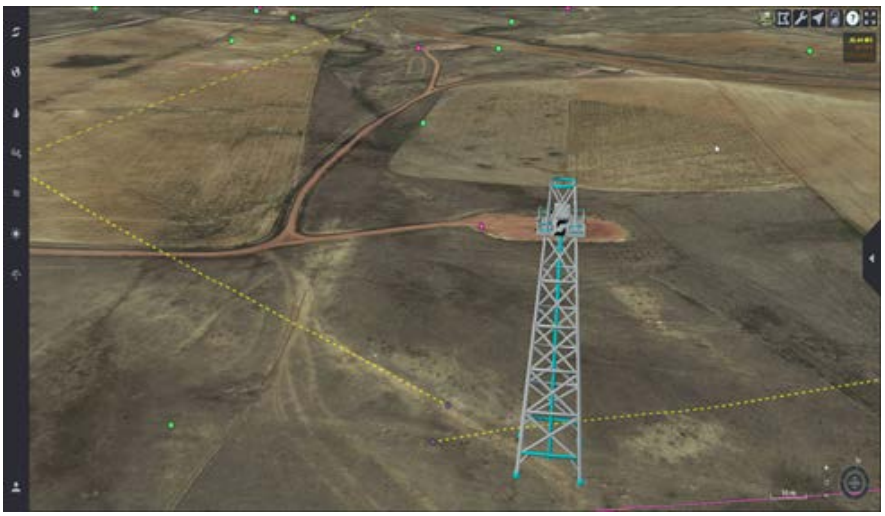

Figure 4. 3D model of vector tile rendered on virtual globe.

the benefit of handling massive volumes of data. The drawback of utilizing GIS software is that it requires more training to use than the CesiumJS online globe and is not as user-friendly.

## 5. Conclusions

Vector tiles are a potent tool for data visualization on a map, in conclusion. They offer a scalable and effective technique to render massive amounts of data. To depict any form of data, including points, lines, and polygons, utilize vector tiles. Vector tiles enable the production of unique data layers that may be applied to various data displays. Heatmaps, choropleths, and other data-driven visualizations can be made using vector tiles. A potent platform for displaying vector tiles is CesiumJS. It offers numerous features and choices for personalizing visualizations. CesiumJS is a good framework for doing this, and vector tiles are a useful method to visualize data on a map.

Information is displayed on maps using vector tiles, a type of map data. Points, lines, and polygons recorded in a vector tile format make up vector tiles. Using vector tiles, maps can represent a range of data, including highways, structures, and geographic features. CesiumJS makes it possible to create and share vector tiles. Web maps, smartphone apps, and Augmented Reality (AR) applications can all benefit from using Cesium to produce vector tiles.

## Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

## References

[1] De Beukelaar, I.T.Y. (2018) Cartographic Implications of Vector Tile Technology. Master's Thesis. Utrecht University, Utrecht.

[2] Zhou, M. (2016) A Virtual Globe-Based Vector Data Model: Quaternary Quadrangle Vector Tile Model. *International Journal of Digital Earth*, **9**, 230-251. https://doi.org/10.1080/17538947.2015.1016558

[3] mapbox (2019) Mapbox Documentation. https://docs.mapbox.com/

[4] mapbox (2019) Mapbox Vector Tile Specification. https://github.com/mapbox/vector-tile-spec

[5] CARTO (2021) CARTO BIGQUERY TILER. https://carto.com/bigquery-tiler/

[6] CesiumJS (2021) CESIUMJS. https://cesium.com/platform/cesiumjs/

[7] Vitolo, C. (2015) Web Technologies for Environmental Big Data. *Environmental Modelling & Software*, **63**, 185-198. https://doi.org/10.1016/j.envsoft.2014.10.007

[8] Yue, P. and Jiang, L.C. (2014) Biggis: How Big Data Can Shape Next-Generation GIS. *The Third International Conference on Agro-Geoinformatics*, Beijing, 11-14 August 2014, 1-6. https://doi.org/10.1109/Agro-Geoinformatics.2014.6910649

[9] Ingensand, J. (2016) Implementation of Tiled Vector Services: A Case Study. In Sdw@ giscience: 19. https://api.semanticscholar.org/CorpusID:973025

[10] Murshed, S.M., Al-Hyari, A.M., Wendel, J. and Ansart, L. (2018) Design and Im-

plementation of a 4D Web Application for Analytical Visualization of Smart City Applications. *International Journal of Geo-Information*, **7**, Article 276. https://doi.org/10.3390/ijgi7070276

[11] Mapbox (2019) Vector Tiles Standards. https://docs.mapbox.com/data/tilesets/guides/vector-tiles-standards/