Computers & Geosciences 85 (2015) 1–9

Contents lists available at ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

MaRGEE: Move and Rotate Google Earth Elements

Mladen M. Dordevic[1], Steven J. Whitmeyer*

Geology and Environmental Science, James Madison University, Harrisonburg, VA 22807, USA

ARTICLE INFO

Article history:
Received 19 February 2015
Received in revised form
15 July 2015
Accepted 6 September 2015
Available online 12 September 2015

Keywords:
Google Earth
KML
JavaScript
Geospatial
Tectonics

ABSTRACT

Google Earth is recognized as a highly effective visualization tool for geospatial information. However, there remain serious limitations that have hindered its acceptance as a tool for research and education in the geosciences. One significant limitation is the inability to translate or rotate geometrical elements on the Google Earth virtual globe. Here we present a new JavaScript web application to “Move and Rotate Google Earth Elements” (MaRGEE). MaRGEE includes tools to simplify, translate, and rotate elements, add intermediate steps to a transposition, and batch process multiple transpositions. The transposition algorithm uses spherical geometry calculations, such as the haversine formula, to accurately reposition groups of points, paths, and polygons on the Google Earth globe without distortion. Due to the imminent deprecation of the Google Earth API and browser plugin, MaRGEE uses a Google Maps interface to facilitate and illustrate the transpositions. However, the inherent spatial distortions that result from the Google Maps Web Mercator projection are not apparent once the transposed elements are saved as a KML file and opened in Google Earth. Potential applications of the MaRGEE toolkit include tectonic reconstructions, the movements of glaciers or thrust sheets, and time-based animations of other large- and small-scale geologic processes.

1. Introduction

In the years since its debut in 2005, the Google Earth virtual globe has become a highly effective visualization tool for geospatial information (Butler, 2006; Bailey and Chen, 2011). The initial wonder and awe of members of the general public at the ease with which they could fly to any place on the globe and view detailed imagery of the ground surface has been supplanted by its ubiquitous use as a geobrowser of Earth imagery and data. By integrating Keyhole MarkUp Language (KML), now an OpenGIS scripting language, within the Google Earth platform the utility of Google Earth was greatly expanded beyond just passive viewing of the modern-day Earth surface (Ballagh et al., 2011). Geoscientists and others have the means to display original data and models within a user-friendly, 3D virtual globe interface. Introduction of the TimeSpan feature further expanded the features of Google Earth to allow geoscientists to incorporate the 4th dimension of time within Google Earth visualizations. In many aspects, Google Earth has become the preeminent way to display 4D geoscientific and environmental information.

Unfortunately, there remain several limitations in Google Earth that have restricted its full potential as a research and educational tool. Many of these limitations have been previously documented in publications that describe workaround solutions (De Paor and Whitmeyer, 2011; Blenkinsop, 2012, Zhu et al., 2014b). Examples include Screen Overlays that must be coded outside of the Google Earth application (e.g. Whitmeyer, 2010; Dordevic, 2012), 3D COLLADA models that can be positioned on the Google Earth terrain, but must be created by an external application such as SketchUp (De Paor and Pinan-Llamas, 2006; De Paor and Williams, 2006; Dordevic et al., 2010; De Paor and Whitmeyer, 2011; Mochales and Blenkinsop, 2014), and an opaque ground surface that makes displaying and visualizing subsurface data and models challenging (De Paor et al., 2008; Whitmeyer and De Paor, 2008).

Perhaps the most significant limitation of Google Earth from a process-focused science perspective is the inability to transpose (translate, rotate) Google Earth elements, such as lines (paths), polygons, or groups of placemarks. This limitation makes it extremely difficult to show dynamic natural processes, as the creator of a visualization has to draft each incremental position of an element individually. Our initial solution to this problem used a command-line PERL script to load a KML file with geometrical components that the visualization creator wanted to translate or rotate (Whitmeyer and Patterson, 2013). Initially our application focused on local or regional transformations, and therefore did not address aerial distortions of polygons or line elements that would occur when translating across multiple degrees of latitude (e.g. the methodology of Zhu et al., 2014a). However, as the focus of our

* Corresponding author. Fax: +1 540 568 8058.
E-mail addresses: dordevmx@jmu.edu (M.M. Dordevic), whitmesj@jmu.edu (S.J. Whitmeyer).

[1] Now employed at the Incorporated Research Institutions of Seismology, Washington D.C., USA.

http://dx.doi.org/10.1016/j.cageo.2015.09.004
0098-3004/© 2015 Published by Elsevier Ltd.

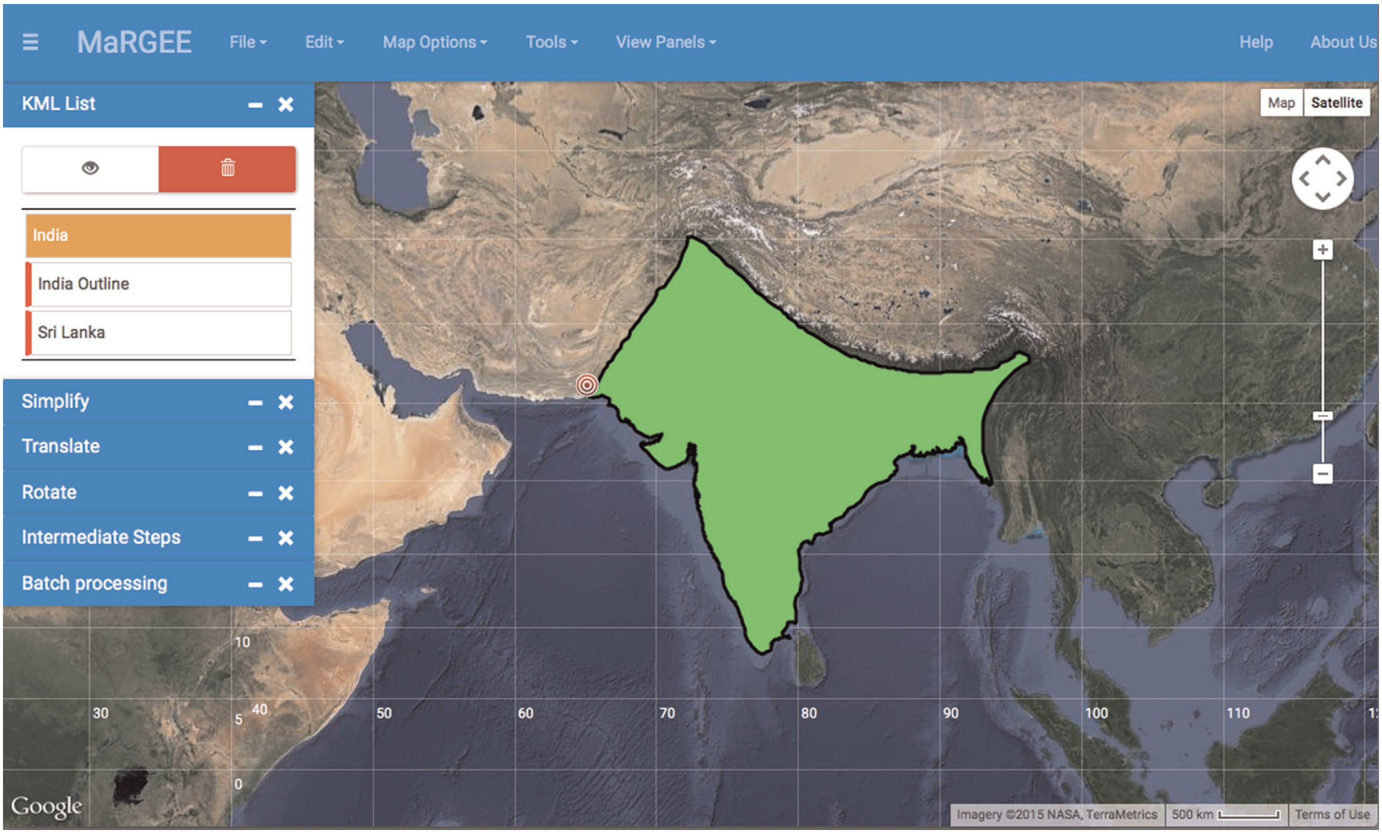

Fig. 1. View of the initial MaRGEE screen after a KML file has been loaded. The file consists of three elements: a filled polygon named India, an outlined polygon named India Outline, and a filled polygon named Sri Lanka. The India polygon is selected, and the India Outline and Sri Lanka polygons have been hidden from view.

Google Earth visualizations expanded to include movements and processes through geologic timescales (thousands to millions of years) it became apparent that our transposition algorithm needed to be based on spherical geometry to preserve the shapes and spatial relationships of Google Earth geometry elements across multiple degrees of latitude and longitude (e.g. De Paor, 1996).

Details of how geometry elements are represented in KML and processed in Google Earth have been documented elsewhere (Wernecke, 2009; Zhu et al., 2014a) and do not warrant reiteration here. In the sections that follow we focus on relevant KML file structures, the importance of using spherical geometry to transpose Google Earth geometry elements, and the algorithms we use to accomplish spherical transpositions. This is followed by a description of the JavaScript web application: "Move and Rotate Google Earth Elements" (MaRGEE): http://geode.net/margee/. A github link to code for the files used in the MaRGEE web application and an example KML file that was produced using MaRGEE and the methods described below are included in Appendix A.

1.1. KML structure

MaRGEE functions by loading a KML or standard KMZ (zipped KML package) file that contains the Google Earth geometry elements (points, paths, polygons) to be transposed or edited (Fig. 1). The primitive KML geometry types (Google, 2013 a; Fig. 2) that MaRGEE can handle are KmlPoint, KmlLineString (path) and KmlPolygon (polygon), as these proved to be most relevant in global-scale transpositions of Google Earth elements.

Paths are rendered in Google Earth as segments of a great circle. However, polygons are rendered in Google Earth by connecting individual coordinates, such that lines of constant bearing result in visually 'deformed' polygons near the poles (Fig. 3). By changing the KmlAltitudeMode in the polygon style element to something

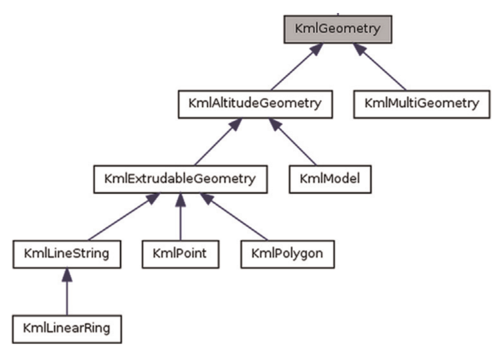

Fig. 2. Schematic of geometry types in KML file structure.

other than "Clamped…" polygons are rendered in the same manner as Paths. The same effect can be achieved by changing the polygon fill mode to "outline": KmlPolyStyle.setFill(0).

Other geometry types (KmlModel, KmlGroundOverlay, KmlMultiGeometry) were excluded from our implementation. KmlModel is used to display COLLADA models in Google Earth. It is anchored with a single coordinate (KmlLocation), scale (KmlScale) and model orientation (KmlOrientation). KmlGroundOverlay is positioned using KmlLatLonBox and is not particularly useful for rotation as it is bounded by lines of constant bearing. KmlMulti-Geometry could be useful, as it represents all of the above as a collection of primitive geometries. However, it cannot be created directly in the Google Earth application, which significantly diminishes its usability in creating polygons and other KML elements.

MaRGEE can read and write to KML and KMZ filetypes (Google, 2013b). A KMZ file is a Zip archive that contains a KML document