May 15, 2019

Visualizing Large Geographic Datasets

Sascha Brawer

Visualizing Large Geographic Datasets

Sascha Brawer

Abstract

Disclosed is a system and method for visualizing large geographic datasets that focuses on prominent features while still showing the spatial distribution of all data. The visualization can be efficiently implemented with standardized application protocols that are commonly used in Geographic Information Systems such as web maps.

Keywords

- Geographic Information Systems (GIS)
- cartography
- web maps
- focus plus context
- data visualization
- information visualization

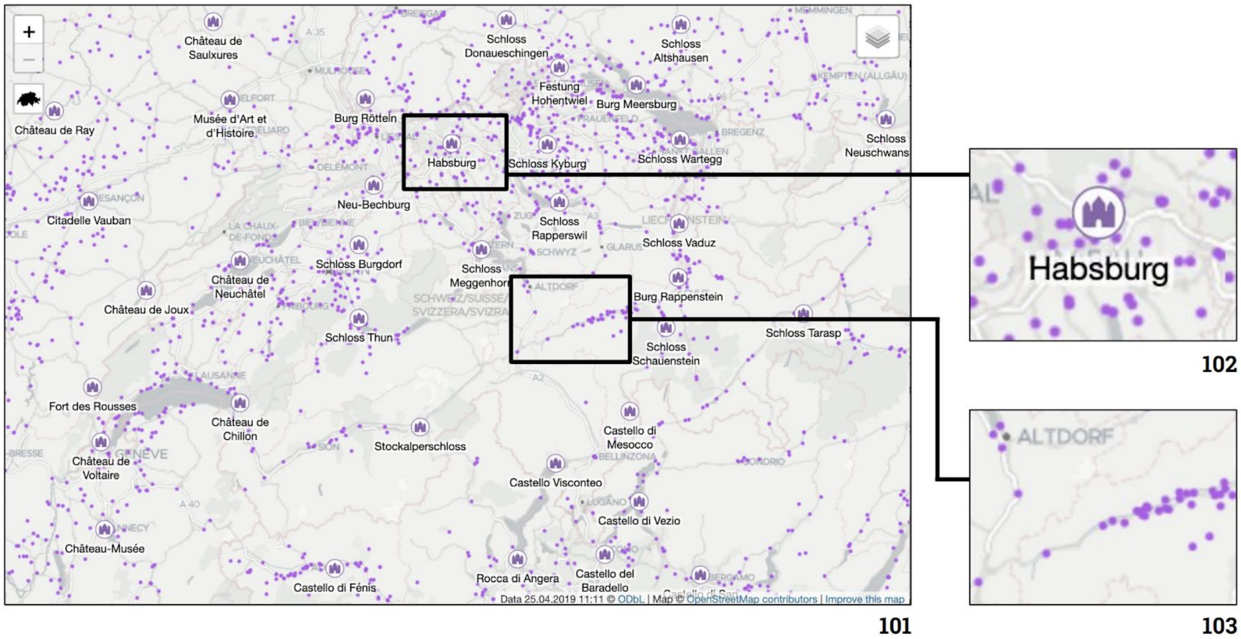

Fig. 1: Example application of the disclosed system and method: An interactive thematic map of Swiss castles (101). The visualization focuses on a small number of important features (102) while still depicting the spatial distribution of all castles, including minor ruins, as contextual background (103).

Background

When Geographic Information Systems such as web maps visualize large datasets, finding a compromise between focus and coverage is challenging. Typical maps either select a few prominent features, or they try to show everything at once. In the former case, humans can quickly focus on the selected features but at the cost of omitting meaningful background data. For example, being able to glance at the spatial distribution of minor ruins in remote alpine valleys is interesting because it tells about historic lines of conflict, even though each single ruin may not be particularly remarkable. On the other hand, when a map tries to visualize everything at once, it becomes hard to focus on key features in a sea of unprioritized data.

Disclosed System and Method

The disclosed visualization system and method merges two layers into one. The **foreground** uses a relevance function to display only a few important features of interest; the **background** depicts the spatial distribution of the entire dataset.

To **prioritize geographic features**, any partial ordering function can be used. For example, a ranking of castles could be derived by analyzing Wikimedia page view logs to count how often people look at the Wikipedia pages for a castle; or by ranking the web sites of tourist attractions by analyzing the World Wide Web's link structure with an algorithm such as PageRank; or by analyzing access logs of an online database about tourist attractions; or by analyzing location records in a database of location tracks; or by analyzing scholarly publications to count how often historians have published about any given castle; or by building frequency statistics about how often people have taken pictures of a site; or from a variety of other sources. Some types of features carry attributes suitable for ranking, such as the altitude of mountain peaks. A ranking function may depend on the dataset, and it will typically incorporate multiple signals.

For **efficient implementation**, it makes sense to sort all features in the dataset in decreasing priority order, so that the most prominent feature gets sorted first. The sorting can be done in advance, for example as a pre-processing step that runs within an indexing pipeline for a distributed geographic information system; or it can be done during server start-up; or it can be done on demand, such as when the dataset is accessed for the first time.

For the **foreground layer**, the client side of a distributed geographic information system such as a web map requests the top features ("limit") for the area of interest ("bbox"). Users often pan their viewport by a small amount; for better latency and to prevent disruptive visualization changes, it makes sense for the client to request a larger area of interest ("bbox") than the current viewport, such as twice or thrice the current viewport size. The server responds by returning the requested features in order of decreasing priority. Clients can request the top features through existing, standardized network protocols such as the Open Geospatial Consortium's Web Feature Service (WFS); or via custom protocols that are implementation-specific.

Next, we **prune the prominent features** to prevent visual clutter. The pruning can be done on either the server or the client side. It works by processing features in order of decreasing priority. Before a feature gets allowed into the foreground layer, we check whether it spatially intersects any other features that are already part of the foreground. If the current feature does not overlap with anything already shown, it gets added to the foreground layer. Otherwise, in case of overlap, the current feature gets dropped/pruned while leaving the current collection unchanged. As long as features are processed in order of decreasing priority, visual conflicts get resolved by dropping the less prominent features. The test for spatial overlap can be efficiently implemented with common spatial structures such as R-trees. Because the features are processed in order of decreasing priority, the overlap test may be conservative (finding intersection among features that actually do not overlap); this allows using faster but less accurate geometric algorithms.

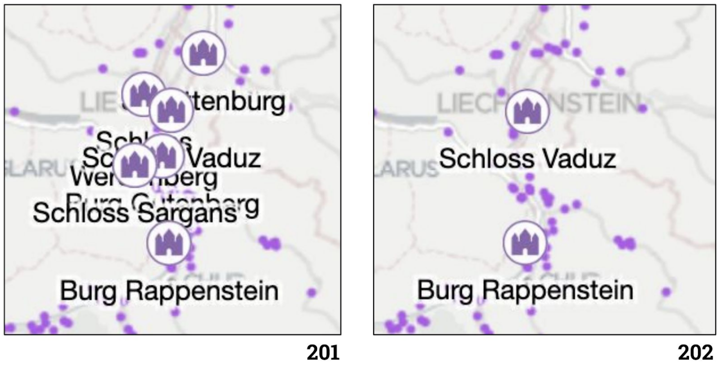

Fig. 2: Without pruning, the map's foreground layer may contain visual clutter (201).
A pruning step makes the map look cleaner, and it becomes easier to understand (202).

For the **background layer**, it makes sense for the server component to render all geographic features into a raster map, in particular when the visualized dataset is very large. The raster map typically gets split into a set of smaller image tiles of a manageable size. The background layer can be pre-rendered in advance for the entire dataset; or it can be rendered on demand, at the time when a client requests a particular tile. To improve serving performance and user latency, image tiles can be cached. Clients can request the image tiles through existing, standardized network protocols such as the Open Geospatial Consortium's Web Map Tile Service (WMTS); or via custom protocols that are implementation-specific. If the visualization of all features becomes too dense, it can make sense to apply traditional techniques of cartographic generalization. Because the background layer can be pre-computed, it is possible to apply computationally expensive techniques of cartographic visualization.