Eurographics Conference on Visualization (EuroVis) 2016
E. Bertini, N. Elmqvist, and T. Wischgoll
(Guest Editors)

Short Paper

Interactive Web-based Visualization for Accessibility Mapping of Transportation Networks

Alexander Schoedon[1], Matthias Trapp[2], Henning Hollburg[3], and Jürgen Döllner[2]

[1] Institute for Geography at the University of Potsdam, [2] Hasso Plattner Institute at the University of Potsdam, [3] Motion Intelligence GmbH

Abstract

Accessibility is a fundamental aspect in transportation, routing, and spare-time activity planning concerning traveling in modern cities. In this context, interactive web-based accessibility-map visualization techniques and systems are important tools for provisioning, exploration, analysis, and assessment of multi-modal and location-based travel time data and routing information. To enable their effective application, such interactive visualization techniques demands for flexible mappings with respect to user-adjustable parameters such as maximum travel times, the types of transportation used, or used color schemes. However, traditional approaches for web-based visualization of accessibility-maps do not allow this degree of parametrization without significant latencies introduced by required data processing and transmission between the routing server and the visualization client. This paper presents a novel web-based visualization technique that allows for efficient client-side mapping and rendering of accessibility data onto transportation networks using WebGL and the OpenGL transmission format. A performance evaluation and comparison shows the superior performance of the approach over alternative implementations.

Categories and Subject Descriptors (according to ACM CCS): E.1.3 [Data]: Data Structures—Graphs and networks H.3.5 [Information Systems]: Online Information Services—Web-based services I.3.6 [Computer Graphics]: Methodology and Techniques—Graphics data structures and data types

1. Introduction

Today, accessibility analysis and visualization is often performed using Desktop GIS (Geographic Information Systems). Such systems exploit the computation power of desktop PC but posses limited applicability to everyday life, due to data availability and access, as well as expert domain knowledge required by a user. With respect to these restrictions, performing server-based accessibility analysis in combination with interactive web-based accessibility map visualization (Fig. 1) has various advantages: (1) its usage is not limited to stationary desktop systems but available on a variety of devices (esp. mobile); (2) potentially massive data sources are not required to be completely transmitted, stored, or managed; and (3) implementations based on web-services and WebGL [JG16] can be easily integrated into existing systems and visualization frameworks. In contrast to existing accessibility-map visualization concepts, this paper focuses on visualizing the travel time data directly on the respective transportation network features, rather than (possibility generalized) polygons [GKD10] or specific graph layouts [KSW*12]. This enables a precise mapping of travel data to the geo-referenced transportation network. However, considering the high geometric complexity (vertices, primitives) introduced by increasing quality of transportation networks [ZZ10], e.g., of massive open data transportation networks (OpenStreetMap (OSM) or General Transit Feed Specification (GTFS)), an implementation of

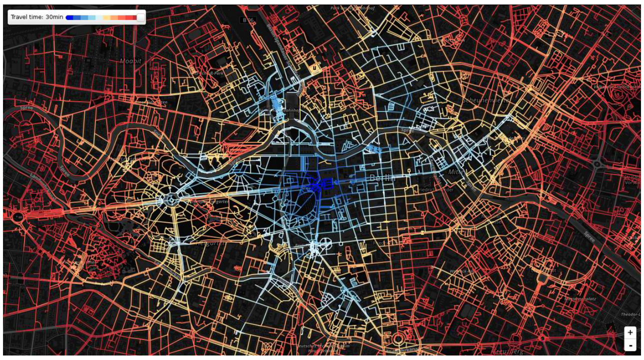


Figure 1: *A high-detailed web-based accessibility-map visualization based on transportation network data. It shows color-mapped travel times (blue: 0 min. to red: 30 min.) for public transportation w.r.t. a single starting point as a 2D map overlay.*

an interactive web-based visualization technique comprises a number of conceptual and technical challenges.

Real-time rendering transportation networks as scenery for data visualization in web-based applications is a performance critical task depending on the geometric complexity of the network and associated travel times. For example, an OSM dataset of the Berlin region comprises approx. $9 \cdot 10^5$ edges (Oct. 2015). Using tradi-

DOI: 10.2312/eurovisshort.20161165

tional visualization approaches using web browsers either faces users with a predefined, static filtering and mapping (raster data) or a notable computation-intense rendering process (vector data). These two fundamental approaches covering filtering, mapping, and rendering web-based maps are widely established and have proven to be effective, but exhibit drawbacks:

Raster Formats: Data transmitted in pre-rendered raster data formats does not require any client-side processing prior to rendering, can be compressed as well as cached [ESR06], and is used by major web-mapping services such as Google or Bing maps. However, one major disadvantage is the lack for client-side filtering and mapping without requesting a complete map tile reload.

Vector Formats: In contrast thereto, geodata transmitted using vector formats enable client-side filtering, mapping, and rendering. This client-side processing however introduces a major performance impact: both the data processing and rendering are usually performed on CPU using JavaScript (JS) algorithms. Recent approaches do support hardware-accelerated rendering (GPU) but lack functionality for client side vector data processing [Gaf12].

Thus, both approach changes in filtering (e.g., selecting a travel time threshold) or mapping (e.g., color mapping, line styles etc.) would result in a complete data re-transmission, loading, and processing. To summarize, an interactive visualization technique for web-based accessibility maps should adhere to the following requirements and challenges: it should support a web-based, hardware-accelerated implementation using WebGL [Par12] (R1); a standardized and compact data representation that allows for decoupling network geometry from temporal data to reduce data transmission and updates (R2); as well as enable interactive client-side filtering, mapping, and rendering for visual feedback (R3).

This paper proposes a new approach for web-based visualization of accessibility maps based on transportation network data. This geometry-based approach uses vector data (lines) stored and transmitted using a new standardized glTF file format, which reduces performance-critical computations in the visualization client (i.e., coordinate transformations) and thus facilitates real-time rendering with high run-time performance yielding low client response times. To summarize, this work makes the following contributions: (1) it presents a concept to decouple the visualization geometry and data items for interactive web-based accessibility maps based on WebGL [JG16], and (2) it demonstrates the effectiveness of this approach by a comparative performance evaluation of different implementation variants.

Related work comprises basically accessibility map visualization, web-based visualization frameworks, and the rendering of transportation networks using GPUs. Glander et al. present an accessibility map visualization technique with a focus on polygon-based approaches [GKD10], and raster-based distance transforms [MG10] which both lack precision in display offered by our approach. In [YLT*15], a web-based system for visualization of multi-modal accessibility for multiple land-uses is presented. However, the visualization technique does not focus on the specifics of transportation network representations. Altmaier et al. (2003) was among the first to outline issues in web-based geovisualization applications [AK03]. In [BS07], challenges, requirements, and concepts of client-based browsing of spatial data on the World Wide Web are discussed. The presented system is based on a Java-Applet and does not exploit modern web technologies for rendering complex spatial data. Vaaraniemi et al. (2011) as well as Trapp et al. (2015) develop and evaluate approaches for transport network visualization utilizing modern graphics hardware [VTW11, TSD15]. However, the presented rendering techniques can currently not be implemented using technologies for browser-based rendering and do no cover specifics of data representation and formats.

2. Accessibility-Maps for Transportation Networks

This section briefly discusses design decisions on web-mapping frameworks, data formats, and required geographic projections. To evaluate the concepts based on real-world data sets, we choose to rely on the Route360-JS° API for server-side computing of routing data. Although there are alternatives with advanced WebGL-integrations available (e.g., OpenLayers 3, Cesium), this work focuses on the web-mapping framework Leaflet-JS for interoperability with the Route360°-JS API. However, the glTF-based approach is not specific to Leaflet and can be integrated into other web-mapping frameworks.

The exemplary transportation network used in this paper is a part of an OSM data set of Berlin. It comprises streets, footways, and data of the transportation infrastructure network. Based on this network a single-source shortest-path accessibility analysis [Mey01] is performed using the Route360°-JS API. It results in the travel times required by foot, bicycle, car, or public transportation from a single starting point to all remaining nodes of the network. The geometry, topology, and accessibility data (represented on a per-node basis) is stored in a database which is used for data conversion to the file formats of our visualization techniques.

For handling data-intense client/server communication, it is important to evaluate options on data exchange formats. Both, Coughlin and Trevett motivate why a standardized data format close to hardware devices for applications on the web are required [Cou14, Tre12]. We considered the following file formats for comparison: (1) COLLADA this digital asset exchange format (`.dae`) has been established for modeling purposes and exchange of geometry and scene descriptions [BF08]; (2) the Geography Markup Language (`.gml`) is based on a XML grammar standardized by the Open Geospatial Consortium (OGC) to represent geographical features [GML07]; (3) GeoJSON (`.json`) a JavaScript object notation that is extended by geographic features comprising geometries and its properties [BDD*08]; and (4) the OpenGL transmission format (`.gltf`) was recently released by Khronos Group and is designed to be a file format close to the requirements of the rendering hardware [CP15]. Table 1 (next page) compares the file formats with respect to their memory footprint and client-side processing requirements. The former is important to evaluate the required bandwidth, while the processing is the aforementioned bottleneck in performance of transforming geographic data into GPU array-buffers.

Concerning memory consumptions, both glTF and COLLADA perform above average. The comparison uses the Cesium Milk

Format	Memory (Byte)	Client Processing
`.dae.gz`	54,949	required, decompress
`.gltf.gz`	58,791	decompress only
`.json.gz`	67,693	required, decompress
`.glb`	89,168	not required
`.gltf`	173,597	not required
`.gml.gz`	211,137	required, decompress
`.dae`	212,788	required
`.json`	824,790	required
`.gml`	2,698,953	required

Table 1: *Comparison and rating (dark background means not suitable) of data formats with respect to the amount of memory required for server-to-client data transmission and client-side processing prior to rendering. The data set used for comparison comprises 1840 vertices and 3624 faces (no texture data included).*

Truck (provided by Analytical Graphics Inc.). As a result, the gzipped version of glTF (`.gltf.gz`) is more compact than a binary version (`.glb`). Due to high memory footprints, traditional geodata formats (esp. JSON and GML) are not suitable for our approach (cf. R2). Further, regarding the client-side processing requirements prior to rendering, the OpenGL transfer format allows to store array buffers which eliminates any JavaScript processing except requesting and reading the data. Due to this, and by respecting R1 and R3, the glTF file format is considered to be the best choice.

To minimize the client-side data processing and to reduce transformation operations, it's important to reduce re-projections and avoid spherical units (e.g., degree, latitude/longitude). Leaflet-JS uses the EPSG:4326 standard projection, which is the world geodetic system 1984 (WGS84) and uses a latitude/longitude coordinate format, thus all programming interfaces of Leaflet return values in degree. To remove the need for computing-intense spherical transformation and to simplify coordinates, all geographic references are projected to a normalized range $[0,1]$ with it's origin in the north-west corner. This moves as close as we can get to hardware-coordinates and conveys device-independence.

3. Browser-based Implementation Variants

This section covers Leaflet-JS integration details and limitations specific to different prototypes of web-based implementations for transportation networks used as scenery for an interactive accessibility-map visualization technique. Subsequently, the performance of the following approaches for Leaflet-JS plug-ins are evaluated (Sec. 3.3):

Leaflet line rendering (JavaScript + GeoJSON) uses software rasterization for rendering. The interactive rendering capability of this approach is limited to small travel time ranges.

Canvas overlay (WebGL + GeoJSON) uses a WebGL integration in Leaflet-JS and renders GeoJSON data representations. It is limited w.r.t. client-side data processing (Sec. 3.1).

glTF tiles (WebGL + glTF) is based on a geometry buffer tiling service that explicitly utilizes the glTF data file format (Sec. 3.2).

During preliminary tests, the native line rendering shows limitations with respect to client-side data processing and rendering performance (above 5 min. travel time). Therefore, we choose not to

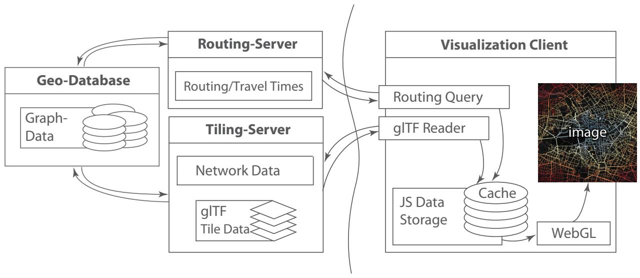


Figure 3: *Conceptual overview of the glTF-based approach.*

persuade this approach and thus do not elaborate on its implementation within this paper.

3.1. GeoJSON-based Visualization Client

Leaflet-JS provides a 2D canvas on a HTML DOM element that is used for basic web-mapping tools such as raster-based tiles or simple vector items. An extended *canvas overlay* class provides Leaflet-JS with a 3D overlay that handles re-drawing of the canvas and enables basic WebGL context integration. On each map interaction, a draw call is issued by the overlay. This function processes the underlying GeoJSON and (1) extracts all geographic features, (2) marks each feature visible, if its travel time is below a user-selected threshold, and (3) converts each visible feature to typed array buffers (vertices and colors), including coordinate mapping.

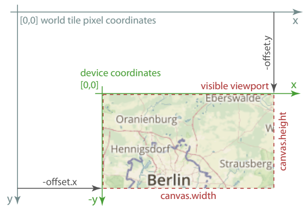
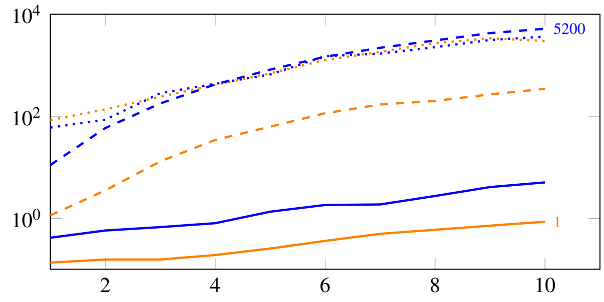


Figure 2: *Viewport coordinates.*

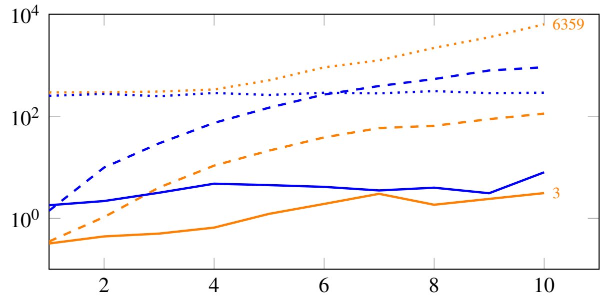
To display geographic coordinates in Mercator projection (EPSG:3857) on a GPU-rendered map, two major steps have to be performed. First, the geographic projection of each coordinate has to be transformed into a normalized projection by translating the coordinate origin to the top-left corner by the distance of half an equator. Further, the units will be scaled from 40.075.016 meters to a normalized range of $[0,1]$. Finally, this projection has to be transformed to screen space coordinates of the current device viewport. The device coordinate origin will be translated to the top-left corner of the visible viewport. The geometry view will be scaled using the current zoom level scale factor and visible canvas size. The top-left corner of the map can be retrieved from Leaflet-JS and used as an offset for the model view to translate to the displayed geometries (Fig. 2). This implementation approach shows two performance bottlenecks: (1) the conversion from geodata representation to GPU array-buffers and (2) coordinate transformations for rendering (cf. Sec. 3.3).

3.2. glTF-based Visualization Client

This approach eliminates client-side data processing and represents a new geometry-tiling approach based on glTF and following the common approach to partition map data into portions of similar size. Figure 3 shows an conceptual overview of our approach: (1) the input transportation network will be pre-processed to glTF-tiles that are stored on a dedicated tiling server. In this step all coordinate transformations required for rendering are performed and a correspondence to the routing graph is established; (2) the visualization client queries the geometry of the tiles according to the



(a) Mapping



(b) Rendering

Figure 4: *Run-time performance for data mapping and rendering with respect to different travel times (1-10 min.), browser (blue: Chrome, orange: Firefox), and implementation (dot: lines rendering, dash: canvas overlay, solid: glTF-tiles) in milliseconds on a logarithmic scale.*

current viewport and request travel times respectively by using the graph correspondence; (3) for visualization, the network tiles are cached and directly used for mapping and rendering.

Table 2 highlights the advantages of glTF-tiling. It supports client-side mapping (e.g., color mapping and lines styles) at run-time on according to user inputs without retransmission of data, similar to vector tiles. The data processing is transferred from client to server-side, similar to raster-tiling approaches. This results in high run-time performance with low latencies and the resulting geometries can be cached client-side. The rendering is performed on dedicated GPU using WebGL and thus ensures real-time performance for geometrical complex geodata sets.

3.3. Performance Evaluation and Comparison

The tests are conducted using a Lenovo Thinkpad X230 equipped with an Intel Core i7-3520M CPU running at 2.90GHz (Quad-core), 16GB DDR3-RAM @ 1600MHz (SODIMM), and an integrated Intel HD Graphics 4000 with 256MB RAM (shared). The implementation variants are evaluated using Firefox 44.0 and Chrome 48.0 browsers running on an Arch Linux kernel. The tests are performed under the following conditions: mapping and rendering run-time are measured using the same zoom factor and center location (complete data set is visible) using a full-screen viewport resolution of 1366×768 pixels.

Table 3 shows an overview of the test data used for performance measurement. The data is a subset of the inner city of Berlin and comprises car travel times from a single starting point (cf. Fig.1). Based on the maximum travel time set by the user (required for an accessibility-map visualization), the geometric complexity varies with respect to the number of vertices and lines to be displayed. Figure 4 shows an overview of the performance comparison results (more precise results are presented in the supplemental material).

Tiling Approach	Processing	Mapping	Rendering
Raster	Server	static	Server
Vector	Client/CPU	dynamic	Client/GPU
glTF	Server	dynamic	Client/GPU

Table 2: *Comparison and rating (dark background means not suitable) of tiling approaches with respect to the requirements of data processing, mapping, and rendering.*

	Travel time in minutes				
Geometry	1	2	4	8	10
Vertices	36	162	1,901	14,874	25,821
Edges	20	88	1,006	7,950	13,829

Table 3: *Varying geometric complexity of the test data set.*

These indicate superior run-time performance of the glTF approach over the native line rendering and GeoJSON implementation for mapping and rendering. Further, they show an increased scalability of the glTF approach w.r.t. the geometric complexity of the network and significant differences in mapping and WebGL 1.0 rendering performance for the different web browsers tested.

4. Conclusions and Future Work

This paper presents a new approach for efficient rendering and mapping of interactive web-based accessibility-maps. The presented tiling approach based on glTF file format reduces the amount of data processing operations required by the client, increases run-time performance for efficient rendering, and simultaneously reduces the amount of transmitted data for visualization. With respect to rendering, the work compares this technique to two alternatives, showing a significant performance increase for massive data sets. The presented results enable the development of new interactive techniques for web-based transportation network visualization systems and tools.

Based on these results, future work focuses on developing a glTF processing back-end with separated tiling and routing server logic. The presented approach supports the decoupling of network geometry and accessibility data, which further reduces the amount of travel data transmission during updates. Further, a complete working client/server infrastructure is evaluated regarding it's overall performances to compare the results with traditional raster or vector approaches.

Acknowledgments

This work was funded by the Federal Ministry of Education and Research (BMBF), Germany within the InnoProfile Transfer research group "4DnD-Vis" (`www.4dndvis.de`) and "MOBIE" (`www.mobie-project.de`).

References

[AK03] ALTMAIER A., KOLBE T. H.: Applications and solutions for interoperable 3D geo-visualization. In *Proceedings of the photogrammetric week* (2003), pp. 251–267. 2

[BDD*08] BUTLER H., DALY M., DOYLE A., GILLIES S., SCHAUB T., SCHMIDT C.: *The GeoJSON Format Specification*. Geographic JSON WG, Jun 2008. 2

[BF08] BARNES M., FINCH E. L.: *COLLADA Digital Asset Schema Release 1.4.1 Specification*, 2 ed. The Khronos Group Inc., Sony Computer Entertainment Inc., Mar 2008. 2

[BS07] BRABEC F., SAMET H.: Client-based spatial browsing on the world wide web. *Internet Computing, IEEE 11*, 1 (Jan 2007), 52–59. 2

[Cou14] COUGHLIN B.: 3D for the modern web - Declarative 3D and glTF. *GMU CS-752* (2014). 2

[CP15] COZZI P., PARISI T.: *GL Transmission Format*, Dec 2015. 2

[ESR06] ESRI®: *Comparing Vector and Raster Mapping for Internet Applications*. ESRI 380 New York St., Redlands, CA 92373-8100 USA, aug 2006. 2

[Gaf12] GAFFURI J.: Toward web mapping with vector data. In *GIScience* (2012), Xiao N., Kwan M.-P., Goodchild M. F., Shekhar S., (Eds.), vol. 7478 of *Lecture Notes in Computer Science*, Springer, pp. 87–101. 2

[GKD10] GLANDER T., KRAMER M., DÖLLNER J.: Accessibility maps for the visualization of quality of mobility in public transport. In *Kartographische Nachrichten 60. Jahrgang* (2010). 1, 2

[GML07] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, GENEVA, SWITZERLAND: *ISO 19136 (GML 3.2.1)*, 2007. 2

[JG16] JACKSON D., GILBERT J.: *WebGL Specification*, 11 ed. The Khronos Group Inc., Jan 2016. 1, 2

[KSW*12] KRAUSE J., SPICKER M., WÃŰRTELER L., SCHAEFER M., ZHANG L., STROBELT H.: Interactive Visualization for Real-time Public Transport Journey Planning. In *Proceedings of SIGRAD 2012 – Interactive Visual Analysis of Data* (2012), pp. 95 – 98. 1

[Mey01] MEYER U.: Single-source shortest-paths on arbitrary directed graphs in linear average-case time. In *In Proc. 12th ACM-SIAM Symposium on Discrete Algorithms* (2001), ACM Press, pp. 797–806. 2

[MG10] MÜLLER M., GLANDER T.: Distance transformations for accessibility mapping in the public transport domain: A performance assessment. In *Proceedings of GIScience* (2010). 2

[Par12] PARISI T.: *WebGL: Up and Running*, 1st ed. O'Reilly Media, Inc., 2012. 2

[Tre12] TREVETT N.: 3D transmission format. In *Seventh Augmented Reality Standards Community Meeting Talks* (2012). 2

[TSD15] TRAPP M., SEMMO A., DÖLLNER J.: Interactive rendering and stylization of transportation networks using distance fields. In *GRAPP 2015 - Proceedings of the 10th International Conference on Computer Graphics Theory and Applications, Berlin, Germany, 11-14 March, 2015.* (2015), pp. 207–219. 2

[VTW11] VAARANIEMI M., TREIB M., WESTERMANN R.: High-quality cartographic roads on high-resolution DEMs. *Journal of WSCG 19*, 2 (2011). 2

[YLT*15] YIN S., LI M., TILAHUN N., FORBES A., JOHNSON A.: Understanding transportation accessibility of metropolitan chicago through interactive visualization. In *The First International Workshop on Smart Cities and Urban Analytics (UrbanGIS)* (2015), ACM. 2

[ZZ10] ZIELSTRA D., ZIPF A.: Quantitative studies on the data quality of openstreetmap in germany. In *Sixth International Conference GIScience 2010*. Zuerich, Switzerland, 2010. 1