Spatial Text Visualization Using Automatic Typographic Maps

Shehzad Afzal, Ross Maciejewski, *Member, IEEE*, Yun Jang, *Member, IEEE*,
Niklas Elmqvist, *Member, IEEE*, and David S. Ebert, *Fellow, IEEE*

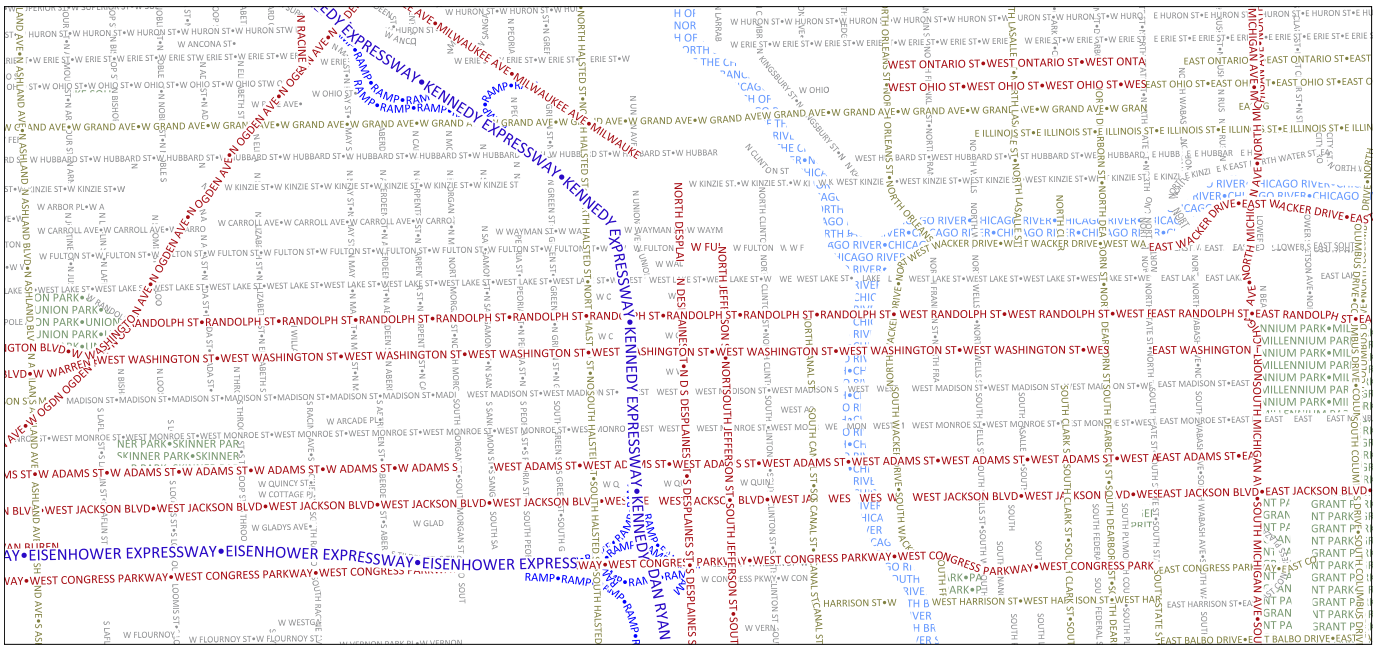


Fig. 1. Typographic map for Chicago, IL built using our automatic visualization technique and geographical data from OpenStreetMap. Colors are used to signify different entity types based on a palette used by Axis Maps in commercially sold Typographic Maps.

Abstract—We present a method for automatically building typographic maps that merge text and spatial data into a visual representation where text alone forms the graphical features. We further show how to use this approach to visualize spatial data such as traffic density, crime rate, or demographic data. The technique accepts a vector representation of a geographic map and spatializes the textual labels in the space onto polylines and polygons based on user-defined visual attributes and constraints. Our sample implementation runs as a Web service, spatializing shape files from the OpenStreetMap project into typographic maps for any region.

Index Terms—Geovisualization, spatial data, text visualization, label placement.

1 Introduction

Textual labels are an intrinsic part of most practical visualization techniques, but they typically play a supporting role in annotating visual entities such as lines, points, and shapes of different color, texture, and weight that make up the visual representation. However, a recent development in the design and infographics community has been visual representations consisting entirely of text [7] (also known as *calligrams* [27]); here, the labels themselves become the sole graphical features. One such interesting technique is Typographic Maps [1], introduced by the cartography company Axis Maps, where entire city maps are rendered only using the names of streets, highways, parks, waterways, and monuments that together make up the spatial features of the city. The result is a detailed and highly aesthetic geographic map made entirely of the geographical labels (i.e., type) themselves.

However, Typographic Maps, while aesthetically pleasing, are manually created by human mapmakers in a lengthy and tedious process [14]. As a result, Typographic Maps exist only for a handful of North American cities, and most cities in the world will likely never have Typographic Maps made for them. Furthermore, the current Typographic Map approach is purely aesthetic, and does not exploit the full potential of visualizing data using text that has been the hallmark of the information visualization field since its inception [34, 45].

Accordingly, we present a technique for automatically generating a typographic map of any geographic region within seconds (Figure 1). Mimicking the practices of the human mapmakers, our technique wraps text along paths to create lines, and fills polygons with text to create shapes. Furthermore, an automatic technique opens the door to conveying additional data using the spatialized text. We show how to combine our typographic maps technique with spatial datasets to create thema-typographic maps where the individual characters in the map are scaled, colored, or highlighted according to the underlying spatial data. Examples of where this could be useful include conveying crime rate in a city, traffic information on a road network, or demographic data in a geographic region.

- *Shehzad Afzal, Niklas Elmqvist, and David S. Ebert are with Purdue University in West Lafayette, IN, USA. E-mail: {safzal, elm, ebertd}@purdue.edu.*
- *Ross Maciejewski is with Arizona State University in Tempe, AZ, USA. E-mail: rmacieje@asu.edu.*
- *Yun Jang is with Sejong University in Seoul, South Korea. E-mail: jangy@sejong.edu.*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

The motivation for our work first and foremost comes from the high visual aesthetics of Typographic Maps—Axis Maps sells poster-sized versions of their maps that are in high demand—which is balanced by the high cost of creating them. An automatic typographic map technique will alleviate the production cost and would make such maps available for any geographic region in the world. The thema-typographic map concept makes the technique potentially useful for data visualization. However, our interest goes beyond this mere automatization process and into the domain of *visual asceticism*: there is something profoundly compelling about a visual representation made up entirely of labels, where the form of the data is also its semantics. The discipline of information visualization is often concerned with providing visual representations that allow people to interpret symbols as a quantity. In the case of cartography, both road names and road lines are symbols that need to be interpreted with respect to their geographic location. By combining the road line directly with the name, we are able to facilitate this interpretation, achieving a combination of symbolic and graphical aspects into a single hybrid representation.

We have implemented our automatic typographic maps technique as a web service that accepts requests for a particular position and region of the world and returns an SVG [43] file representing the typographic map for that region. Our implementation uses OpenStreetMap [29] and spatializes text onto the graphical features of the map. It can also generate thema-typographic maps if the user provides a spatial dataset with the map request. We compare the output of this implementation with a San Francisco map created by Axis Maps. We also show an example of using our thema-typographic map for West Lafayette, IN where high amounts of crime in an area scales up the characters.

The remainder of this article is structured as follows: we begin by reviewing the literature on text visualization, spatializing text, cartography, and label placement. We present our basic technique for automatically generating typographic maps from a geospatial dataset, and then show how these maps can be turned into thema-typographic maps. We describe our implementation and show some examples and close with our conclusions and plans for future work.

2 Background

A *map* is a visual representation of a physical space depicting the relationship between the spatial elements of that space. Cartography, the study and practice of making maps, has been around since the cradle of civilization and has long concerned itself with how to best design these visual representations to present the most important and relevant features in clear, understandable, and actionable ways [15, 30, 36].

Below we review the relationship between cartography and more recent efforts from the geographic information science and visualization domains. We then present exciting innovations in infographics and graphic design on the creative use of typography—an intrinsically cartographic consideration—as visual form. We draw parallels to the field of text visualization and then show how these components can all be combined into typographic maps that convey shape, scale, and data in a single visual representation.

2.1 Cartography and Geovisualization

Scale has been long recognized as one of the most important visual variables to the human perceptual system [3, 4], and humans possess a high degree of spatiocognitive abilities that make it possible, even easy, for us to navigate in geographic space [35]. These facts all form the basic cognitive platform upon which maps and mapmaking are based. However, where traditional maps are static, the new fields of geographic information science (GIScience) and geovisualization deal with intrinsically interactive visual representations [36], commonly using *geographic information systems* (GIS) [23].

Outside of the GIScience domain, spatial and geographic datasets are a common data source for our own visualization field. Shneiderman includes 2D geographic maps as one of seven core data types for information visualization [34], and much recent work has explored the intersection of cartography and visualization (e.g., [10, 19, 47]).

Labels are a key feature of maps [36], static and interactive alike, and thus the fields of cartography and geovisualization have long studied the combination of textual labels on graphical features for making geographical maps. Seminal work [3, 4, 15] and handbooks on cartography [26, 30, 36] tend to include long lists of guidelines on appropriate label placement in different situations, as well as the use of typographic conventions to convey spatial information. These guidelines are also being transferred into automatic labeling algorithms in the digital and GIS domains, with work such as that by Kakoulis and Tollis [17] studying general feature labeling, and that by Kameda and Imai [18] looking at optimal label placement on points and curves.

Many of the label placement strategies outlined in the cited work above focus on conveying information about the spatial features on the map using the label as well. For example, labels may sometimes be curved or meandering [15], and typographic conventions also play a role, such as using italics for names of water features [36]. However, beyond these uses, labels are seldom used in cartography to convey abstract information beyond the spatial features themselves.

2.2 Combining Text and Shape

Labels generally play a supporting role in visual representations, but a radical new idea that is gaining traction in both academic and design communities is to generate graphics where the textual labels alone form the visual features: in other words, the labels become the image. This is actually a form of calligraphy and is known as a *calligram*.

From the academic side, perhaps the most well-known example is the extended graph labels technique proposed by Wong et al. [46]. Designed for node-link diagrams, this technique dispenses with traditional lines connecting nodes in the graph, and instead uses the textual label, curved and repeated as necessary, as the de-facto visual link. The WordBridge technique [20] builds on this idea, but takes it a step further by transforming both links and nodes into full-fledged word clouds (edge and node clouds, respectively). However, both of these use standard graph layout algorithms for the spatial position of nodes.

Chevalier and Diamond [7] review the design and infographic side of the text-as-shape paradigm in a recent miniature survey on the synergy between text analysis and fine art. One of their examples in particular, word cloud portraits, is especially relevant to our work because it also utilizes language to convey a visual form. The specific pieces discussed in the paper are Roscover's word cloud portraits for President Obama [32] and Steve Jobs [31]. However, the spatial layout in these portraits are designed primarily with aesthetics in mind.

Recent work by Maharik et al. [27] propose a method for creating *digital micrograms*, which are calligrams using small-scale and readable text. The smooth vector field method used in their work is potentially useful for our ideas, but their paper focuses on aesthetics, as opposed to the representational maps we study here. The key difference between our technique and that of Maharik et al. is that our input consists of a series of network nodes defining the road topology and theirs consists of a segmented image. Furthermore, our technique is faster (on the order of seconds instead of minutes) and is directly suited to geographical visualization where segments of lines and regions can be scaled based on thematic variables.

2.3 Text Visualization

Text visualization uses interactive visual representations to show information about documents beyond their actual text, and has been a prominent focus in information visualization since the field was established [34, 45]. Text is ubiquitous in our everyday life, and text visualization provides a lightweight and low-barrier approach to seeing a different perspective on this type of data.

Perhaps the most common text visualization technique today is the *word cloud* (or tag cloud) [2, 41], where terms are scaled proportionally to their relative frequency and placed on a visual space in some specific order. Popularized by the social media website Flickr in 2002 [41] (although there exist several examples of their use prior to this), word clouds are now in use by thousands of Internet websites.

However, despite their popularity, word clouds are plagued by a number of problems, such as awarding undue attention to long words, difficulties in comparing term size, and layouts that do not promote

(a) Aligning a character to a path. (b) Aligning (and repeating) a label to a path.

Fig. 2. Fitting text to a path. We use the path normal at each insertion point for a character to derive its orientation.

visual search [41]. The Wordle technique [42] proposes a more space-efficient randomized layout algorithm that results in highly aesthetic clouds. The improved ManiWordle technique [21] allows the user to directly control the cloud layout. A competing approach uses a circular rather than a square layout [25], and yet another allows for layouts optimized around a point or a line [20]. SparkClouds [24] add sparklines to individual terms in a word cloud to show their trend over time, but do not change the term layout. In general, none of these examples use the spatial position of terms for anything other than aesthetics.

Some more recent cloud-based techniques are starting to better utilize the position variable. Parallel tag clouds [9] use one spatial axis to display the temporal attributes of terms in a document corpus. TIARA [33, 44] combines a tag cloud with a trend graph to show changes over time. Clustered word clouds [8, 13] attempt to place commonly co-occurring terms close to each other. TagMaps [48] draw word clouds on top of geographical features, such as for conveying not just the content but also the locality of the keywords. In all of these techniques and systems, not only the size of the word but also its position is important for understanding the visualization.

2.4 Spatializing Text for Geovisualization

Finally, although our work arose from a need to spatialize text in several geospatial applications, we were heavily inspired by the recent Typographic Maps [1] published online by the cartography design firm Axis Maps. These maps, just like our typographic maps, use the names of streets, highways, parks, and city blocks to form a geographical map of the city itself. The maps have garnered much attention and sales on the Internet, and many visitors to the Axis Maps website are requesting maps be created for their cities. However, creating a Typographic Map is a manual and painstaking process performed by a human mapmaker, where creating a single map may take several weeks [14]. As a result, Typographic Maps currently exist only for a select few North American cities, and new maps appear only rarely.

The difference between the original Typographic Maps and our visualization technique for generating typographic maps is that our approach is fully automated using geographic information from OpenStreetMap. While we have made efforts to replicate many of the design practices used by the mapmaker, our automated technique naturally lacks the truly creative and aesthetic touch afforded by a human designer making the map. However, with access to an automatic algorithm, we take the next step by using Typographics Maps to visualize data using a technique inspired by proportional symbol maps, where symbols on the map are used to convey a thematic variable.

3 Typographic Maps

Typographic maps are spatial visualizations where the graphical features making up the visual representation consist **only** of text of different size, rotation, and graphical properties. Each text object is arranged so that it conveys not only the *semantics* of the spatial data (i.e., the label), but also its *shape*. Thus, the visualization utilizes spatial position effectively by placing the labels in the area they belong.

Our focus in this paper is on geographic maps, but here we describe the automatic typographic maps technique in general terms. More specifically, in the text below, we will discuss the abstract input data expected by the technique as well as methods for spatializing text onto polylines and onto regions of space. This discussion sets the stage for our implementation, presented later.

3.1 Data Model

The input data for typographic maps is an abstract 2D shape representation consisting of graphical *layers* and *objects*. Each layer has a name and represents a particular class of graphical objects in the overall representation. For example, for a geographical map, layers are entity types such as highway, ramp, street, block, park, etc. All graphical objects in the shape representation belong to exactly one layer.

As part of the text spatialization process, the user is asked to assign visual attributes, such as font size, color, and weight, to each layer to guide the output. For example, a highway could be assigned a larger font with a black color, while a smaller city street would have a smaller size and use a light gray so that it is not as visually prominent.

The graphical objects in the shape representation can typically be regarded either as 1D paths (e.g., polylines) or 2D regions (e.g., filled polygons). Each graphical object also has a label; for a geographic map, the label of a street would be its name. If a graphical object lacks a label, we can use the name of the layer (often the object type) it belongs to as a label. This label will form the text that will be drawn repeatedly in lieu of the graphical rendering of the path or region itself.

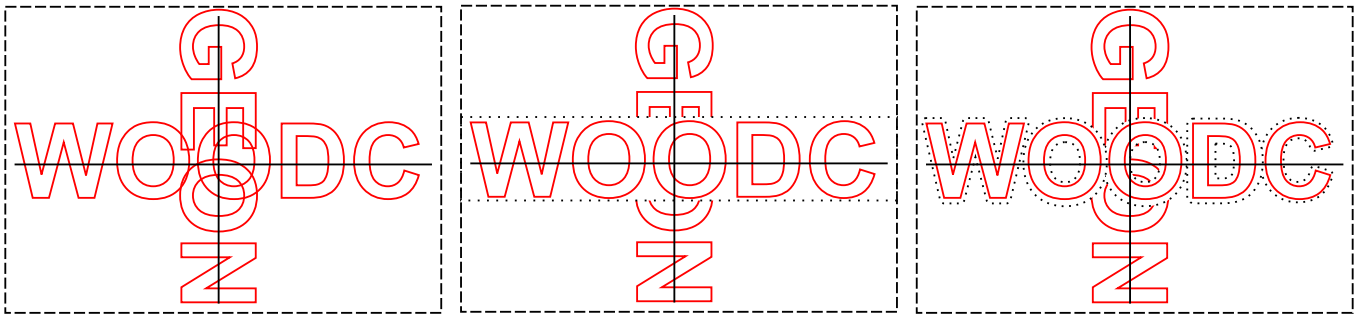
At this point in time, the user may want to define simple text transformation rules to optimize the visual output. For example, because labels will need to be repeated for larger graphical objects, the user may want to specify a separator that will be interleaved with multiple instances of the label. Some layer types may want to use a particular separating character; for example, for an interstate layer (a divided highway) on a geographic map, the separating character may be a road shield. In addition, the whitespace character may be problematic because it causes a visual discontinuity in the output, so the user may want to replace it with something else like an asterisk or dash. Finally, for certain domains, it may make sense to abbreviate or shorten the labels, such as writing “St” instead of “Street”, “Ave” instead of “Avenue”, and “Blvd” instead of “Boulevard” for a city map.

Finally, all shape representations use a layer ordering that governs which layers should be prioritized (i.e., on top of other layers) for spatial interference. In the example of geographical maps, an elevated highway in a city should clearly be prioritized over the city streets crossing under it, or a street winding through a park should be rendered on top of the graphical region representing the park.

3.2 Paths as Text

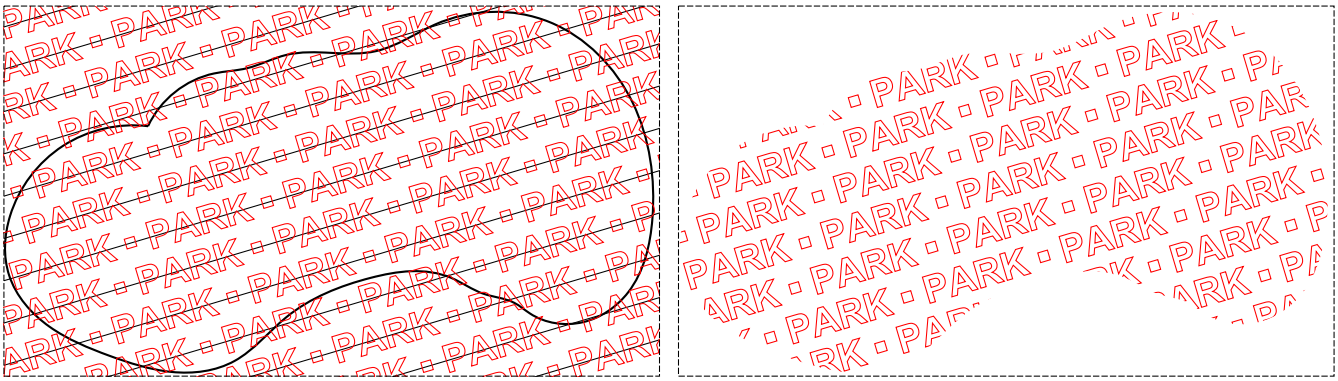
Rendering a path using text amounts to fitting the text to the path and repeating it for the duration of the path’s length (Figure 2). Fitting a textual label, in turn, equates to iteratively placing each graphical character of the label on the next position centered on the path and rotating the character to align with the path normal at that position (Figure 2(a)). We choose not to warp the graphical character, but instead merely use a 2D rigid-body rotation to maximize readability of the label. This avoids graphical artifacts arising from paths with high curvature at the potential expense of suboptimal curve fit.

There are a number of additional points to consider in order to achieve the look of a continuous path using fitted text. First, the width of the line will clearly now be controlled by the font size—the larger the font, the thicker the line—as well as type face—different font types



(a) Clutter from label overlap. (b) Character mask (invisible). (c) Character halo (invisible).

Fig. 3. Reducing clutter from label overlap by adding (b) white masks and (c) halos to the background. Note that the dotted lines in (b) and (c) are added for illustrative purposes, and are invisible in a real implementation.



(a) Filling the region's bounding box with oriented text. (b) Clipping text using the region path.

Fig. 4. Filling and clipping a region with text. Note that we use the same orientation for the whole region, and we also introduce a single-character offset for each line to mitigate character repetition effects across lines.

have different graphical appearances. This naturally places a lower bound on how small a path can be drawn so that the label forming it is still legible. Second, different font types can be used to achieve particular graphical and cartographic effects; for example, in current cartographic practice, a reverse oblique type face is often used to communicate water, such as rivers, lakes, or ponds [15, 36]. Furthermore, to achieve a uniform width, it is often best to use uppercase versions of each word to avoid the lowercase parts of a label taking less vertical space. Finally, as noted above, we introduce a separating character for interleaving repeated instances of a label, and sometimes we may even want to utilize a special whitespace character to avoid the visual discontinuities caused by an empty space in a label.

As mentioned above, our typographic maps technique tracks layer priorities to manage the correct order for overlapping layers, but we may need to do additional work to avoid visual clutter in these situations. Such clutter arises, for example, when two paths intersect, causing the situation shown in Figure 3(a). A simple solution is to add a white rectangle as a background (known in cartography as a *mask* [36]) behind each character as illustrated in Figure 3(b). A more advanced solution would be to add a so-called *halo* [36] (or *null halo*) behind the label; this is illustrated in Figure 3(c) (the use of halos for enhancing depth perception is also prominent in illustrative visualizations in 3D [11, 16, 37]). Another approach may be to use an outer stroke (i.e., an outline) in the background color (typically white). All solutions have strengths and weaknesses—for example, halos are common in some cartographic designs, but may introduce more clutter than the more regular appearance of masks. The Axis Map designs tend to use masks, presumably for this reason.

Finally, another important issue for maximizing the legibility of our labels is to consider its orientation. Cartography has many guidelines for text orientation that we may consider [15, 36]. For example, text is obviously easiest to read when it runs from left to right and right side up. For paths with mostly vertical components, Byrne [6] demonstrated that marquee placement of text was outperformed by rotated horizontal text and that reading the rotated text from bottom to top or top to bottom had no impact. We employ rotated horizontal text reading from bottom top. Our typographic maps technique tries to enforce these orientation rules. In some cases, paths radically change orientation throughout their existence; consider, for example, a beltway circling a city. This makes it difficult to find an optimal orientation. For these situations, we split the path into segments depending on the predominant direction and orient the text on a per-segment basis.

The original Typographic Maps are all manually designed, and it is interesting to see that not all roads are labeled with consistent text orientation (compare "San Bruno Ave" and "James Lick Freeway" on the right side Figure 7(a)). Switching orientaton, particularly for adjacent roads, may be a way to better distinguish between roads and increase label readability. Adding such design guidelines to the layout algorithm is left for future work, however.

3.3 Regions as Text

We render a region as text simply by filling the interior of the region with the text, repeated as necessary (Figure 4). Because regions are two-dimensional areas, fitting the text to the area needs a radically different approach than for paths. Our solution takes the bounding box of the closed path representing the region and fills it with straight lines of the textual label, repeated as necessary (Figure 4(a)). The lines all have the same orientation, and an increasing character offset is used between adjacent lines to avoid unseemly visual artifacts arising from the tiling of the label. Finally, as Figure 4(b) shows, we use the closed path of the region itself as a clip path. However, in sticking with the text-as-shape paradigm, we do not draw the border itself.

Issues may arise when rendering two regions adjacent to each other, in particular if they share the same label. The best option in this case may be to use different visual attributes—such as font weight, size, or color—between the adjacent regions, but this may not always be an option (in particular if the regions belong to the same class, so they have the same color). In these situations, we vary the orientation of the lines in the adjacent regions, causing a visual discontinuity between the regions that will be perceived as a border (Figure 5). Additionally, regions could be segmented and transformed to text using the algorithm presented by Maharik et al. [27].

Fig. 5. Varying text orientation to distinguish between adjacent regions; useful if the regions have no other distinguishing graphical attributes.

An advanced rendering option that we have not yet explored is to use paths other than straight lines inside each region. For example, in Typographic Maps [1], water areas typically use a wavy pattern, presumably to better communicate the region type. In the future, we foresee extending this to other layer types or for additional spatial data, such as conveying contour lines on mountainous or hilly areas, prevailing currents in an ocean, or wind direction on a meteorological map.

4 Thema-Typographic Maps

Thematic maps are geographic maps where a geospatial variable is visually encoded on the map [36], and have been used to visualize various demographical, political, and economical data for various regions in the world. For example, John Snow used a thematic map to identify a contaminated pump during a cholera outbreak in London in 1854 [40]. Examples of such maps include chloropleth maps [36] (coloring spatial aggregations), cartograms [12, 39] (distorting space and distance), and proportional symbol maps [36] (scaling map symbols).

Fig. 6. Scaling individual characters on a path to convey a thematic variable (such as traffic) in the underlying spatial dataset.

We introduce the concept of *thema-typographic maps* based on our automatic typographic maps technique. The basic idea is simple: instead of merely repeating the same textual label along a path or inside a region as described above, we use the font attributes—typically size, but color or intensity is possible—on a per-character level to convey the value of a statistical variable at each character's spatial location. Because the mapping operates merely on font attributes and does not affect the characters themselves, the semantics (i.e., the labels) of the typographic map is preserved. Figure 6 illustrates this simple concept for a path rendered using text; the idea is similar for regions. Note how the varying character size creates the impression of a band that is wide at the edges and that shrinks in thickness in the middle.

Because our thema-typographic map scale (or color) the elements of the map on a per-element level, we see many interesting applications for this technique. For example, Figure 8 shows a thema-typographic map generated using our approach where roads are scaled proportional to crime in the area. This yields a visual mapping where areas with a high degree of crime are made visually larger, and those with less crime are smaller in size. Other uses include demographics, political, and traffic data being overlaid on a geographical map.

5 Implementation

We have implemented the automatic typographic maps technique as a web service called TypoMap that uses OpenStreetMap [29] as a geographic database. Given a map query with the bounding box in longitude and latitude, our web service returns an SVG [43] file representing the typographic map of that region. This file can then be viewed in a modern web browser or vector editor, or printed and viewed off-line.

5.1 Overview

Our web service implementation consists of three simple steps: (1) retrieving map data in XML format for a requested area on the globe from the OpenStreetMap XAPI service; (2) cleaning, filtering, and merging paths and regions in the data and storing it into an internal 2D vector database; and (3) rendering the vector regions and paths in the database as text using the SVG format, that is returned to the caller.

5.2 Data Source: OpenStreetMap

TypoMap uses the XAPI interface to perform read-only map queries to the OpenStreetMap (henceforth, OSM) web service. The return value for such map queries is an XML file in the OSM data format that we proceed to parse and use to populate an internal vector database.

OSM data is optimized for both rendering and routing, so the data is generally of high quality and well-suited for rendering as a typographic map. However, the path data is often redundant and somewhat irregular. Our XML parser performs low-level cleaning and filtering to ensure good results: bike trails and footpaths are omitted, and segments that are smaller than can reasonably be represented on the typographic map (small water features in particular) are also filtered. We then use a higher-level filter component where the resulting data is optimized for rendering as a typographic map based mostly on aesthetics; this involves steps such as (a) connecting separate paths, which occurs in OSM when a road changes name; (b) combining the divided lanes of a highway into the same path; and (c) managing path priority, such as weaving paths or giving horizontal paths priority over vertical.

OSM by default uses a slightly modified version of the spherical Mercator projection (also known as the Google Mercator projection since it was first introduced by Google Maps in 2005), and we adopt the same projection for mapping the longitude and latitude of map features onto the flat canvas of our SVG file output.

Furthermore, all map elements in the OSM data format contain freeform but standardized tags on map features. We use a subset of these standardized tags as the *layers* in our model with a predefined priority order as well as graphical attributes for font, color, and size.

5.3 Typographic Maps in SVG

Rendering typographic maps proceeds one layer at a time. Each layer in the data extracted from the data source has a specific font, color, and size (configurable upon invoking the web service). We render the map simply by rendering each layer at a time, with the lowest priority layer first (meaning that it will end up behind all other layers).

Both paths (polylines) and regions (i.e., the polyline defining the region border) in the TypoMap vector database are represented using the SVG `<path>` construct, defined once in the SVG file but given an identifier so that it can be used multiple times. However, depending on whether we are rendering a path or a region, we proceed differently:

- **Paths:** We first render the white background mask using a simple polyline with the appropriate stroke thickness (depending on the size of the characters in the layer). We motivate our choice of a mask by the fact that this is the solution the Axis Maps designs use. We then render the text using the `<textPath>` element, which wraps the specified text along the path. Because the SVG default is to use the path as a baseline (i.e., the text is placed on

(a) Original Typographic Map of San Francisco, CA.

(b) Automatic typographic map of San Francisco, CA.

Fig. 7. Comparison between a manually created Typographic Map (left) and an automatic typographic map (right) of the same area of downtown San Francisco. The automatic one was generated as SVG in less than one second and then rendered to a same-resolution bitmap for faithful comparison. The original Typographic Map on the left is used with permission from Axis Maps.

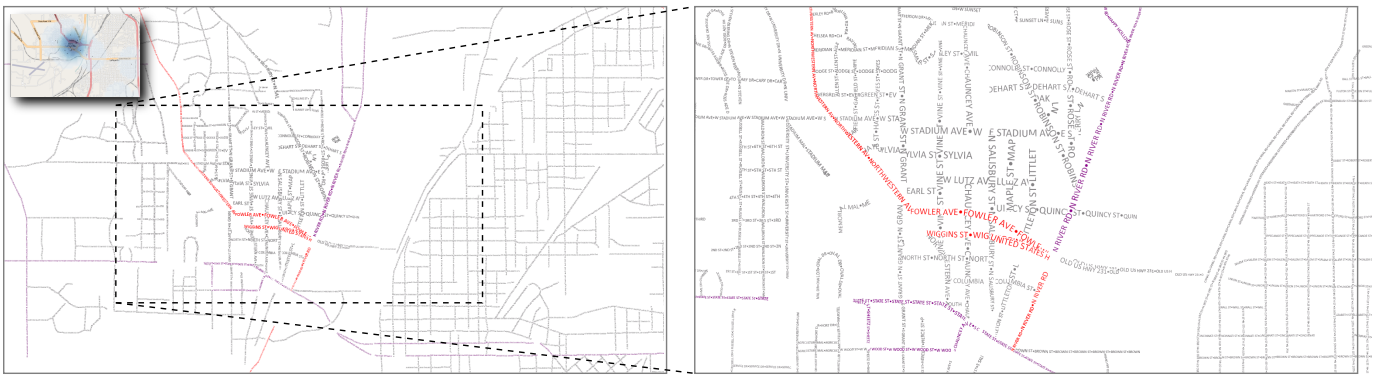


Fig. 8. Thema-typographic for West Lafayette, IN where the statistical variable being visualized is crime rate. larger text means more crime. The right picture shows a zoomed-in area around the Purdue campus, and the inset (upper left) shows the KDE map for the entire region.

the line), we must offset the text 50% of its height so that it is centered on the line instead.

Our implementation tries to orient text according to the cartographic rules outlined earlier; however, due to the arbitrary ordering of vertices and the merging of paths in the OSM database, our layout does not always succeed in enforcing them.

- **Regions:** We first fill the region outline with white to mask out any background features. We then set up clipping using the `<clipPath>` element with the region outline as the argument. This causes any subsequent rendering to be masked against the region outline. After that, we use the bounding box of the region to emit text on evenly spaces lined at the same (Figure 4), randomly chosen, angle. A more advanced implementation may find the primary axis of the region to use as an orientation, or may choose to use map coloring to ensure that adjacent regions get assigned one of four fixed orientations instead.

Our implementation uses Cascading Style Sheets (CSS) classes for the different map layers, which allows the viewer to change the graphical appearance of the map, if not the geometry itself. This is useful when conforming to the strict typography guidelines that cartography stipulates on font family, weight, case, color, and intensity [30, 36].

5.4 Thema-Typographic Maps in SVG

Rendering our thematic extension means slightly more complex SVG output. Since we are resizing individual characters, we must internally keep track of where different characters in a text fall in the 2D space of the map. This means that we must render the text on the path ourselves since the SVG `<textPath>` command gives no way for the designer to know where on the path individual characters will appear.

To achieve this, we use an internal low-fidelity renderer that, given a specific font and size, steps through characters in the text being rendered and uses the letter width and spacing to calculate the exact 2D location on the path where the character will be placed. Using this location, we look up the mapping variable in the spatial dataset and use its value to scale the size of that character. Individual characters in a text object can be modified in this way using the `<tspan>` element.

Our previous strategy of drawing a single white line on the path with the appropriate stroke width as a mask will now fail since the mask will have to be of varying width to accommodate the varying character size. Instead, our current implementation simply sets the stroke width to be the average of the minimum and maximum font size for a path. Unfortunately, this does result in some situations where there may be visual overlap. A better solution that we plan to implement in the future is to generate a white closed polygon as a mask using the envelope of the character bounding boxes along the path.

5.5 Implementation and Performance

TypoMap is implemented in C# and uses only the standard Microsoft .NET libraries. The core component is a collection of 2D vector shapes that is used as the internal geometric representation of the map to render. This also allows us to fully modularize the input and output components: The OSM XML input parser creates the vector representation, and the SVG renderer uses the vector representation for output. We are able to accommodate different input or output formats simply by exchanging these modules. For example, we could use replace the OSM parser with an image processing component that reads an image and generates vector paths and text to achieve output similar to Roscover's word portraits of Barack Obama [32] and Steve Jobs [31].

Rendering performance is on the order of 2-3 seconds (not counting network transfer time) even for large bounding boxes (the OSM XAPI allows for retrieving regions up to 100 square degrees). Generating a thema-typographic map takes on the order of 10 seconds because of the need to internally render the characters on the paths, but performance is still easily within acceptable parameters for a web service.

6 Results

Figure 7 shows a side-by-side comparison between a Typographic Map for San Francisco provided by Axis Maps, and our automatically generated typographic map for the same region. Since we only have access to a rasterized version of the original, we have rendered our SVG file as a raster image for comparison. We have also chosen font faces, colors, and sizes to be similar to those in the original.

Comparing the two maps side-by-side like this, it is clear that there are differences in their appearance. In particular, the mapmakers who created the original Typographic Map have generalized the map in some places to better convey the spatial features—for example, small roads running parallel to a highway have often been manually removed, whereas our automatic typographic map makes no such distinction. Nevertheless, it should be noted that the original Typographic Map on the left was created by a mapmaker working full-time for two weeks, whereas our automatically generated version on the right required only 2-3 seconds to generate. Since our web service generates standardized SVG that could be imported directly into the Adobe Illustrator tool the Axis Maps designers employ, perhaps one use of our technique is as an initial rough rendition of a Typographic Map that a mapmaker can use as a starting point for further refinement.

The thema-typographic map in Figure 8 shows the spatial layout of West Lafayette, IN where crime rate has been visualized onto the map. We define character size to be proportional to the amount of crime in that region, which causes crime-stricken (and presumably dangerous) areas to become large, and safer areas to become small. In this way, the map intuitively conveys a feeling for the rate of crime in the city.

Finally, Figure 9 shows a large-scale typographic map of the downtown area of Seattle, WA where the surrounding Puget Sound and Lake Washington have been polygon-filled with their names. Figure 10 shows an inset around the conference hotel for VisWeek 2012. This map required on the order of 5 seconds to generate.

In creating these images, a number of the generation issues are due to topology issues associated with the OpenStreetMap data. Roads may start and stop and can be renamed or reclassified from node to node. These issues can result in poor textual overlays for streets. Our future work will look at ways of intelligently merging such problematic typology regions in order to produce more stylistic results. Also, depending on the length of the line segment and the length of the label, some map features may not be able to be labeled with the full name. Along with the labeling issue, such small streets can often cause clutter within the image. Future work will explore means of removing small line segments for clutter reduction as well as prioritizing smaller streets so that if overlap occurs, the algorithm will first ensure that at least one iteration of the street name is visible.

We sent the example images of our automatic typographic map technique in this paper to Axis Maps asking for their feedback. The representatives from Axis Maps thought the results were "fantastic" and "very impressive", especially for the depth sorting of the streets and the type casing our technique does. The person did point out two weaknesses: (1) that additional and more aggressive road generalization is needed, and (2) that the polygon fills should be oriented at 45° to set them off from the street grid. We have since addressed both of these comments: our path generalizer can be fine-tuned to control the amount of generalization, and text in the polygon fills can be rotated in any orientation. He also asked whether they could get access to the program themselves at Axis Maps. We are currently preparing a demonstration release for their use.

7 Design Implications: Drawing with Text

While we have focused on geographic maps in this paper, the general concept of *drawing with text* can be applied to virtually any spatial representation. As evidenced by our literature review, combining text and shape has already been explored both within the academic community—such as in the extended graph labels by Wong et al. [46]—as well as within the art, infographic, and design communities—such as in Roscover's word cloud portraits [31, 32]. Our emphasis here is on how to automatically replicate the methodology and design aesthetic of human mapmakers, but it is worthwhile to consider the utility of drawing with text in a broader context.

Visual representations consisting entirely of text are visually interesting and may contain more information than the corresponding representation consisting only of paths and filled regions. As we have seen in our work, this allows us to convey not just the spatial extents of a visual feature, but also something about the semantics of that feature. There is something profoundly compelling about such a visual representation, and it approaches what we would like to call *visual asceticism* or *visual minimalism*.

Of course, by the same token, as data density goes up, so does the visual complexity. A complex visual representation can be difficult to understand and may impose a high cognitive load on the viewer. Therefore, a graphic consisting entirely of text may be counterproductive for situations when the user needs to make quick and accurate decisions. Consider a police officer trying to find the location of a call on an electronic map—the officer's performance may be degraded with a typographic map due to the extra distractions of all streets being drawn as text. On the other hand, here is an opportunity for our typographic maps technique: because our approach is automated, we can seamlessly switch between typographic and standard 2D vector graphics. Continuing our example with the police officer, we may choose to—for lack of a better word—*textualize* the street around the location of the call, but draw all other streets as vector graphics. The resulting map is preattentively distinct and may even be construed as being **less** visually complex than drawing **both** street and street label. Clearly the lesson here is that drawing with text is a powerful yet potentially confounding method, and should be used with caution.

The above example also touches upon another issue that we have not yet discussed in this paper: how do you interact with a spatialized text visualization? Because our typographic map technique produces SVG files as output, standard navigation operations such as panning and zooming are easy to perform. Furthermore, since the wrapped labels remain text in the SVG document, it is even possible to select and copy portions of the streets or regions on the map. However, additional and more complex interactions are certainly possible, even if we have not explicitly studied them in this paper. Examples include searching for entities on the map, highlighting spatial features (perhaps as a result of searching), or switching between drawing with standard vector graphics and with text inside a magic lens [5]. And what about embedding driving directions to a particular destination in the graphical path to the destination on top of the map itself?

The design space of drawing with text goes beyond interaction. Color is one dimension that we merely use to distinguish between different categories of features (similar to the Typographic Maps of Axis Maps), but there exist many other possibilities as well. For example, we discuss mapping the thematic variable for our thema-typographic maps onto a color scale for characters instead of their size. Another design dimension is supporting different languages and even character sets when drawing with text in general, and for our typographic maps

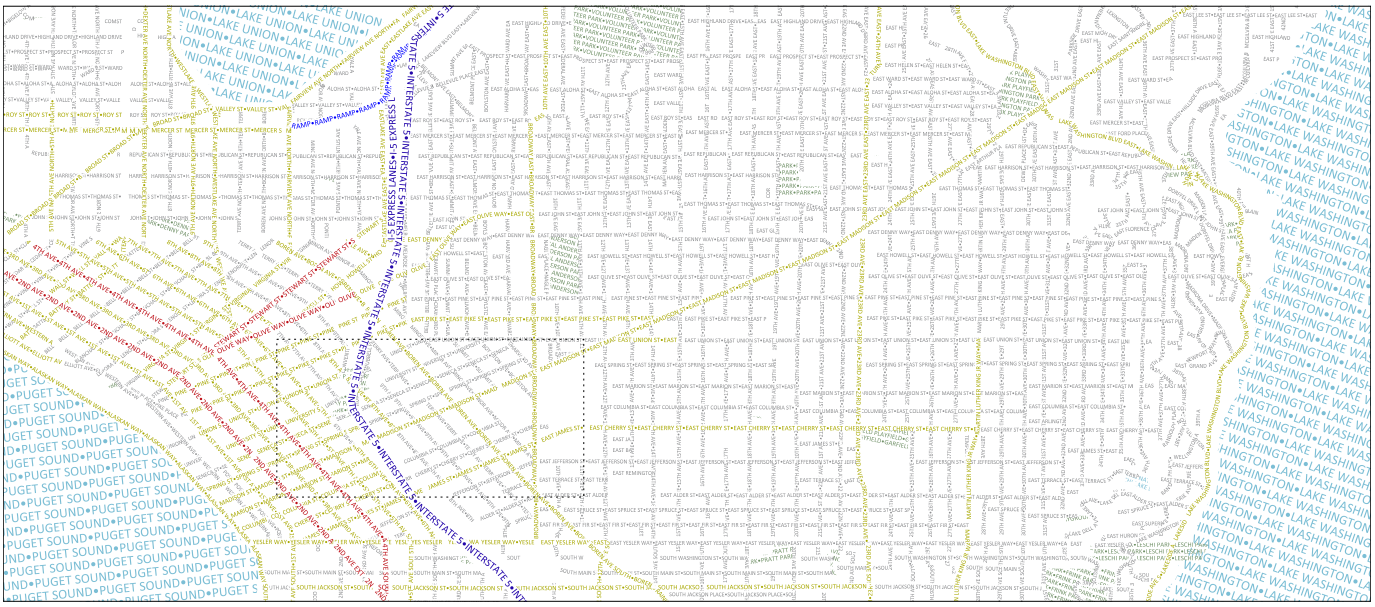


Fig. 9. Automatic typographic map of the downtown area around the VisWeek 2012 hotel in Seattle, WA. This image was generated using our TypoMap web service in less than 1 second based on OpenStreetMap data. The SVG can be viewed (and printed) using a modern web browser.

technique in particular. Maps are clearly relevant for a global audience, but we have yet to study how to support this for our technique.

This is also the place to discuss an oft-overlooked aspect of visualization that the concept of drawing with text exhibits: aesthetics with regards to beautiful imagery [22, 28]. As evidenced by the mainstream success of text visualizations such as tag clouds [2, 41], Wordle [42], and TagMaps [48], the notion of visualizing textual data seems to be inherently compelling to a general audience. Taking the step to use the text itself to represent graphical features has garnered even more attention [7]: Roscover's word cloud portrait of President Obama made the cover of *Time* magazine, and Axis Maps' Typographic Maps [1] are in high demand and are sold commercially on the company's website. In a recent paper [28], Vande Moere and Purchase argue for adopting commercial and artistic design practice to information visualization, and we think that the concept of drawing with text is an excellent compromise between these principles of aesthetics and the core need to accurately convey information using graphics.

Of course, aesthetically appealing imagery is just one side of the coin; for a text-based visualization such as ours to be useful, it has to be accurate as well as readable. While our technique at its core is designed for accuracy (unlike techniques such as the digital micrograms of Maharik et al. [27], which first and foremost prioritize a visually appealing appearance), there are still several aspects in which it could be improved. For example, we think there is still work to be done for our algorithm in improving readability by delimiting labels, as well as reducing visual clutter in geographically dense areas. Furthermore, when rendering small regions and paths, it is clearly important that at least one complete label is visible in the textualized representation. Alternatively, if the region or path is too small to fit even one complete label, perhaps the geographical feature should be omitted entirely, or shown using an iconified representation—a little like semantic zooming for typographic maps. Adding such advanced importance-based rendering and filtering functionality is left for future work.

At the same time, cartographic labeling often deliberately uses imprecise labeling to indicate the fluid nature of geographic features; for example, labeling a mountain range does not necessarily commit to a specific extents or precise location for the range. This approach is very much in line with what has been called Tobler's *first law of geography* [38]—"everything is related to everything else, but near things are more related than distant things"—and we look forward to seeing how these ideas can be extended in the future given this new take on making the labels the actual map.

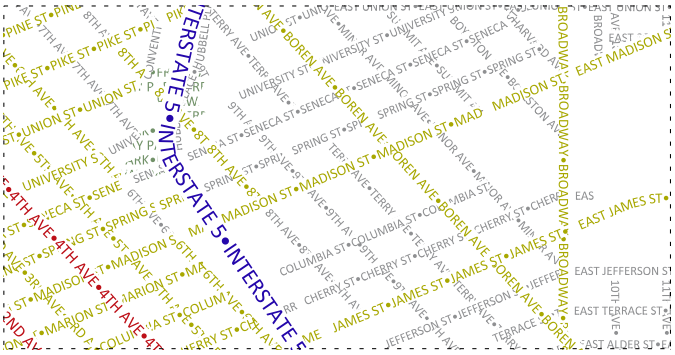


Fig. 10. Detailed area around the VisWeek 2012 hotel in Seattle, WA.

8 Conclusions and Future Work

We have presented an automatic algorithm for generating typographic maps—geographic maps consisting entirely of text as the only graphical features—using the OpenStreetMap web service, allowing us to generate SVG renditions of any region on the globe within a matter of seconds. Using this framework, we have shown how to use our algorithm for creating thema-typographic maps, where the size of the text on the map can also be scaled based on spatial data features such as crime, demographics, or traffic information. We have also presented several examples that showcase the utility of this technique.

We see many avenues for future research. We are very interested in continuing to explore the use of spatial data features as a means of visualizing data using the typographic map representation. The step is also not far to spatialized Wordles, or Wordle Maps, where the labels no longer convey the name of the geographic feature, but instead carries semantic meaning specific to the dataset. Finally, our maps are currently static SVG files and do not incorporate any interaction, and an obvious extension is to add operations for navigating, drilling down, and changing the map layout.

Acknowledgements

We thank Axis Maps for generously giving us the permission to use their Typographic Maps in this article as well as for providing details on how these maps are currently constructed. This work was supported

in part by the U.S. Department of Homeland Security's VACCINE Center under award no. 2009-ST-061-CI0002 and the Defense Threat Reduction Agency under award no. HDTRA1-10-1-0083.

REFERENCES

[1] Axis Maps. Typographic maps. http://www.axismaps.com/typographic.php.

[2] S. Bateman, C. Gutwin, and M. A. Nacenta. Seeing things in the clouds: the effect of visual features on tag cloud selections. In *Proceedings of the ACM Conference on Hypertext and Hypermedia*, pages 193–202, 2008.

[3] J. Bertin. *Sémiologie graphique: Les diagrammes - Les réseaux - Les cartes*. Editions de l'Ecole des Hautes Etudes en Sciences, Paris, France, les réimpressions edition, 1967.

[4] J. Bertin. *Semiology of graphics*. University of Wisconsin Press, 1983.

[5] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. DeRose. Toolglass and Magic Lenses: The see-through interface. In *Computer Graphics (ACM SIGGRAPH '93 Proceedings)*, volume 27, pages 73–80, Aug. 1993.

[6] M. D. Byrne. Reading vertical text: Rotated vs marquee. In *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting*, volume 10, pages 1633–1635, 2002.

[7] F. Chevalier and S. Diamond. The use of real data in fine arts for insight and discovery: Case studies in text analysis. In *IEEE VisWeek Discovery Exhibition*, 2010.

[8] J. Clark. Clustered word clouds. http://neoformix.com/2008/ClusteredWordClouds.html, Oct. 2008.

[9] C. Collins, F. B. Viégas, and M. Wattenberg. Parallel tag clouds to explore faceted text corpora. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 91–98, 2009.

[10] J. Dykes, J. Wood, and A. Slingsby. Rethinking map legends with visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):890–899, 2010.

[11] D. S. Ebert and P. Rheingans. Volume illustration: Non-photorealistic rendering of volume models. In *Proceedings of the IEEE Conference on Visualization*, pages 195–202, 2000.

[12] H. Edelsbrunner and R. Waupotitsch. A combinatorial approach to cartograms. In *Proceedings of the Annual Symposium on Computational Geometry*, pages 98–108, 1995.

[13] Y. Hassan-Montero and V. Herrero-Solana. Improving tag-clouds as visual information retrieval interfaces. In *Proceedings of the International Conference on Multidisciplinary Information Sciences and Technologies*, 2006.

[14] D. Heyman. Axis Maps, personal communication, Mar. 2011.

[15] E. Imhof. Positioning names on maps. *The American Cartographer*, 2(2):128–144, 1975.

[16] V. Interrante and C. Grosch. Strategies for effectively visualizing 3D flow with volume LIC. In *Proceedings of the IEEE Conference on Visualization*, pages 421–424, 1997.

[17] K. G. Kakoulis and Ioannis G. Tollis. A unified approach to labeling graphical features. In *Proceedings of the ACM Symposium on Computational Geometry*, pages 347–356, 1998.

[18] T. Kameda and K. Imai. Map label placement for points and curves. *IEICE Transactions on Fundamentals of Electronics, Communications & Computer Sciences*, E86–A(4):835–840, Apr. 2003.

[19] D. A. Keim, S. C. North, C. Panse, and J. Schneidewind. Efficient cartogram generation: A comparison. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 33–36, 2002.

[20] K. T. Kim, S. Ko, N. Elmqvist, and D. S. Ebert. WordBridge: using composite tag clouds in node-link diagrams for visualizing content and relations in text corpora. In *Proceedings of the Hawaiian International Conference on System Sciences*, 2011.

[21] K. Koh, B. Lee, B. H. Kim, and J. Seo. ManiWordle: Providing flexible control over Wordle. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1190–1197, 2010.

[22] A. Lau and A. V. Moere. Towards a model of information aesthetics in information visualization. In *Proceedings of the International Conference on Information Visualization*, pages 87–92, 2007.

[23] R. Laurini and D. Thompson. *Fundamentals of Spatial Information Systems*. Academic Press, New York, 1992.

[24] B. Lee, N. H. Riche, A. K. Karlson, and M. S. T. Carpendale. SparkClouds: Visualizing trends in tag clouds. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1182–1189, 2010.

[25] S. Lohmann, J. Ziegler, and L. Tetzlaff. Comparison of tag cloud layouts: Task-related performance and visual exploration. In *Proceedings of INTERACT*, volume 5726 of *Lecture Notes in Computer Science*, pages 392–404. Springer, 2009.

[26] A. M. MacEachren. *How Maps Work: Representation, Visualization and Design*. Guilford Press, New York, 1995.

[27] R. Maharik, M. Bessmeltsev, A. Sheffer, A. Shamir, and N. Carr. Digital micrography. *ACM Transactions on Graphics (Proc. SIGGRAPH 2011)*, 30(4):100:1–100:12, 2011.

[28] A. V. Moere and H. C. Purchase. On the role of design in information visualization. *Information Visualization*, 10(4):356–371, 2011.

[29] OSMF. OpenStreetMap. http://www.openstreetmap.org/. accessed March 2012.

[30] A. H. Robinson, J. L. Morrison, P. C. Muehrcke, A. J. Kimerling, and S. C. Guptill. *Elements of Cartography*. John Wiley & Sons, 1995.

[31] D. Roscover. Steven Paul Jobs. http://www.gorosco.com/#438803/Steven-Paul-Jobs.

[32] D. Roscover. Burdened. In *Time Magazine*, Feb. 2010.

[33] L. Shi, F. Wei, S. Liu, L. Tan, X. Lian, and M. Zhou. Understanding text corpora with multiple facets. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology*, pages 99–106, 2010.

[34] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343, 1996.

[35] A. Skupin. From metaphor to method: Cartographic perspectives on information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 91–98, 2000.

[36] T. A. Slocum, R. B. McMaster, F. C. Kessler, and H. H. Howard. *Thematic Cartography and Geovisualization*. Prentice Hall, third edition, 2009.

[37] N. A. Svakhine and D. S. Ebert. Interactive volume illustration and feature halos. In *Proceedings of the Pacific Conference on Computer Graphics and Applications*, pages 347–354, 2003.

[38] W. Tobler. A computer model simulating urban growth in the Detroit region. *Economic Geography*, 46(2):234–240, 1970.

[39] W. Tobler. Cartograms and cartosplines. In *Proceedings of the Workshop on Automated Cartography and Epidemiology*, pages 53–58, 1976.

[40] E. R. Tufte. *Visual Explanations: images and quantities, evidence and narrative*. Graphics Press, Cheshire, Connecticut, 1997.

[41] F. B. Viégas and M. Wattenberg. Tag clouds and the case for vernacular visualization. *interactions*, 15(4):49–52, 2008.

[42] F. B. Viégas, M. Wattenberg, and J. Feinberg. Participatory visualization with Wordle. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1137–1144, 2009.

[43] W3C. Scalable Vector Graphics (SVG). http://www.w3.org/SVG/. accessed March 2012.

[44] F. Wei, S. Liu, Y. Song, S. Pan, M. X. Zhou, W. Qian, L. Shi, L. Tan, and Q. Zhang. TIARA: a visual exploratory text analytic system. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 153–162, 2010.

[45] J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 51–58, 1995.

[46] P. C. Wong, P. Mackey, K. Perrine, J. Eagan, H. Foote, and J. Thomas. Dynamic visualization of graphs with extended labels. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 73–80, 2005.

[47] J. Wood and J. Dykes. Spatially ordered treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1348–1355, 2008.

[48] Yahoo. TagMaps. http://tagmaps.research.yahoo.com/. accessed March 2012.