

Interactive Visualization of Trivariate Georeferenced Data

Tarsus Magnus Pinheiro, Claudio Esperança

PESC – COPPE – Universidade Federal do Rio de Janeiro (UFRJ)
tarsus.pinheiro@ibge.gov.br, esperanc@cos.ufrj.br

Abstract. This paper describes an online interactive thematic map for simultaneously visualizing up to three scalar variables and which supports data filtering, panning and zooming in levels of detail. The visual encoding of the map mixes the use of colors and textures as well as simple operations like border detection and intersection identification. The user experience is enhanced by means of queries posed through manipulation tools that produce instant visual feedback. This is possible through the high rendering rates achieved by the system through the use of GPU programming to assemble and manipulate previously rasterized tiles with location information recorded in the color space of pixels. This procedure allows the implementation of interactive animated actions and spatial data decomposition.

Categories and Subject Descriptors: H.3.5 [Information Storage and Retrieval]: Online Information Services; I.3.3 [Computer Graphics]: Picture/Image Generation; H.5.2 [Information Interfaces and Presentation]: User Interfaces

Keywords: GIS, choropleth, multivariate, thematic map

1. INTRODUCTION

Data visualization studies the forms of visual communication that relate to the registration and organization of data, so as to use vision to reveal hidden characteristics, patterns and trends in massive amounts of data [Gershon et al. 1998].

Georeferenced data are data related to the geographic space. Their visualization has gained relevance with the popularization of map systems for the Internet, which have turned the use of maps into an everyday practice. Thematic maps are objects whose main function is to express geographical information by means of graphic signs designed to present relations of data similarity, ordering and quantification [Archela and Théry 2008].

Choropleth maps are a specific type of thematic map that use color as a graphic sign to communicate data relations. These types of maps are usually static and depict only one variable at a time. The system we have created allows the interactive online display of maps of that kind and supports an arbitrary number of regions with up to three variables plotted simultaneously, allowing inspection in levels of detail. The visual mapping of these variables uses color and texture that allow interactive adjustment. The system is based on the assembling of a group of previously processed raster images, and adopts Graphics Processing Unit (GPU) programming to reach a high enough speed to provide instant responses to user action, thus improving one's learning experience from the visualization of simultaneous variables. The contributions of this work can be summarized as follows:

- (1) Allows online rendering of trivariate choropleth maps with adjustable filters.
- (2) In addition to the traditional visual mapping that strictly observes the region boundaries of the original data, it is possible to compute on-the-fly mappings based on the idea of statistical grids.
- (3) All data structures are encoded as binary images, which are compact and make efficient use of browser and server caches, as well as using little network bandwidth.

- (4) Other than requesting static data images from the server, all of the system's processing is done on the client side, thus requiring very little use of server resources.
- (5) Dynamic image composition uses GPU programming, which leads to high frame rates.
- (6) A fully functional prototype is available online.

This paper is an extended version of [Pinheiro and Esperança 2017], presented at the XVII Brazilian Symposium on GeoInformatics (GEOINFO 2017). In particular, the present work (1) includes an extended discussion on the rendering of multivariate choropleth maps, (2) adds support for a third independent variable by representing two variables with a reduced color palette, (3) instruments the inspection of each plotted variable by presenting a histogram for the the plotted range, and (4) enhances the visual presentation of the map by the use of GPU shaders for edge detection, intersection computation and parametric textures. The next section presents relevant works related to multivariate data visualization with choropleth maps. The system operation and implementation is presented in Section 3. Section 4 shows some results and in Section 5 we discuss the results.

2. RELATED WORKS

The visualization of multivariate data is a relevant challenge. Examples such as the Chernoff Faces [Chernoff 1973], glyphs similar to stick figures [Pickett and Grinstein 1988], the use of parameterized naturalistic textures [Interrante 2000] and simulations of impressionist paintings in which the characteristics of brushstrokes are summarized according to the values of associated variables [Tateosian et al. 2007] show different ways of dealing with the theme. A common problem for all of them is the difficulty in represent quantification, as already discussed by Bertin [Bertin 1980]. He states that size is the only variable capable of representing proportion relations, and that other graphic signs are limited to simple visual representations, such as differentiation and ordering of categories.

One more issue in common among the cited methods is that none of them presents interactivity as a way to improve user experience. Dynamic queries, defined as the interactive control of filters over a group of data that produces an immediate visual reply in a time span of less than 100 milliseconds [Shneiderman 1994] deal with the importance of interaction in the learning process. That brings users closer to the data and turns the formality of database queries into intuitive actions to help the user to try data and improve their search for patterns that reveal new points of view about a given amount of information.

That idea counters the attempts at creating complex signs to promote multivariate visualization, being closer to the proposals of traditional cartography, where the principles set by the Semiology of Graphics [Bertin 1980] established a safe starting point for the elaboration of thematic maps. The application of technologies that improve the quality and speed of rendering, and, as a result, the experience of visualization, should enhance the result of traditional methods.

Choropleth maps are a type of cartogram which represents figures grouped into classes associated with a chromatic scale [Archela and Théry 2008, p. 8] and are the right tool for the visualization of data associated with political-administrative divisions. Creating a map like that requires correct color selection [Stone 2006], and adequate methods for the organization of classes [Andrienko 2001]. [Newman 2012] discusses the use of such maps for the visualization of data relative to American elections of 2010 showing how the changes geographic subdivisions, in color scale and in the application of cartographic anamorphosis can affect data representation, and, as a result, the assimilation of information.

Whereas univariate choropleth maps show a single variable encoded by means of a color palette, bivariate choropleth maps show two independent variables plotted with the help of a color palette organized as a two-dimensional grid. As noted by several authors [Elmer 2012; Stevens 2015], such palettes have to be chosen with even greater care so as to ensure the separability of the two variables.

A common rule-of-thumb is to use starkly different hues for each variable, and to encode category ranges as color value (luminosity) gradations. Such schemes, however, impose a severe stress on the viewer's selective attention mechanism, and rarely extend to palettes with more than 9 (3×3) colors. To alleviate this problem, Elmer [Elmer 2012] proposes using other visual mappings for the second variable, such as glyphs, size or shape. In this work we explore two alternatives for coping with this problem, namely, (1) using pattern textures to encode the second variable and (2) using a reduced (2×2) palette grid for two of the variables and pattern textures for a possible third variable.

Political boundaries are not the only possible representation for choropleth maps. Statistical grids are a method of cartographic visualization that has become quite popular recently and establishes an arbitrary division of geographic space as a grid formed by squares that rasterize the plane and redistribute information according to specific requirements [Bueno 2014; IBGE 2015]. As some advantages of this option we can mention formal stability, once this subdivision is not subject to the history and the setting of political-administrative boundaries [IBGE 2010], and the best visualization of regions which might not be visible at more detailed levels of display. It is also worth mentioning the fact that data result, many times, from spatial decomposition and that can lead to distortion [Bueno and D'Antona 2014].

3. SYSTEM DESCRIPTION

The ideas and concepts proposed in this work were implemented in a proof-of-concept prototype which is readily available online at <http://www.tarsusmagnus.com.br/mestrado/visgl/>. Older versions are also available. Interested readers can try these prototypes which only require a modern browser and a good internet connection. The next sections explain the main ideas of our proposal and how they were assembled in our prototypes.

3.1 Data sets

In our experiments, we have used the 2013 Municipal Grid [IBGE 2013] and selected, from the IBGE's channel Cities, 140 socioeconomic variables originated from count processes, such as the Population count, the municipal vehicle fleet, and the number of votes in the second round of the 2014 Presidential Elections, with information about all 5570 Brazilian municipalities.

3.2 Interface

The work proposed herein considers the visualization possibilities provided by the statistical grid. We use an instance of a real time decomposition based on statistical information associated with regions, in order to show a representation where abstract limits are derived from the classification of data, adapted and changed by user interaction.

Figures 1 and 2 show the system interface, which is divided into a visualization panel on the left and a set of graphical user interface (GUI) elements on the right. These GUI elements control several aspects of the visualization and the selection of the scalar quantities to be visualized. Each such quantity or "theme" is a rate defined as a fraction between two socioeconomic variables selected from the data sets.

3.3 Architecture and Data Structures

The system is implemented in HTML5, the current version of the markup language and content organization that is the basis for crafting pages for the Internet. It standardizes several new technologies adopted to meet the growing demands of more elaborate multimedia content such as sound, video and elaborate imaging techniques. In particular, it establishes WebGL and its associated Javascript API

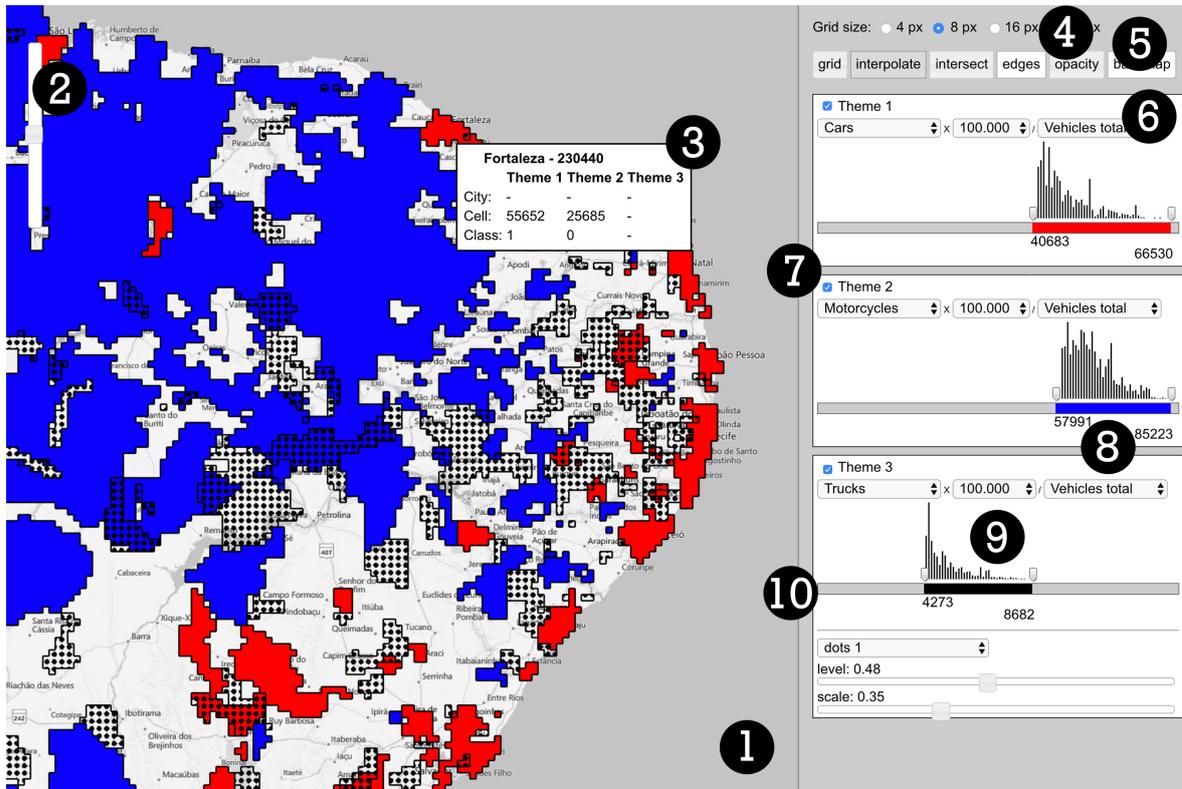


Fig. 1. System interface. 1. Area for map rendering where panning is realized by click and drag. 2. Zoom slider to change the level of detail. 3. Box with information about the cell; 4. Radio buttons set the cells size; 5. Toggle buttons controlling grid display, visualization smoothing, intersection area, edge detection, themes transparency, and background map. 6. Theme control panel. 7. Theme on/off button. 8. Theme variable selection. 9. Slider controlling category class range, associated to a real time estimated histogram and 10. Parametric texture controls associated with the third theme.

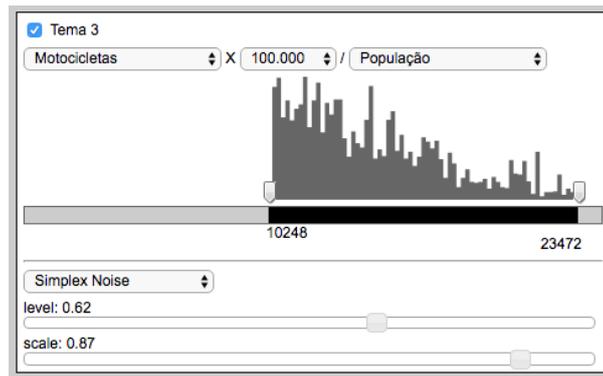


Fig. 2. Theme toolbox and parametric texture controls.

as a means to take advantage of modern graphics cards, making it possible to write code that runs directly in the card's GPU (Graphics Processing Unit) thus yielding high rendering rates directly in standard web browsers.

Javascript is a scripting language based on the technologies available on the Internet that allows

manipulation of the Document Object Model (DOM) generated by browsers from the organization arranged in an HTML page, thus enabling dynamic editing of its content. WebGL, in turn, is a Javascript API that allows manipulation of the HTML5 canvas element, thus providing support for rendering 2D and 3D graphics. That is possible due to the architecture of modern graphics hardware, which is organized around a rigid flow of operations that only allow interference from specific GPU programs called shaders. WebGL supports two types of these: vertex shaders, responsible for the processing of each vertex of the object's geometry; and fragment shaders, which process each pixel of the image output by the rasterization (conversion to a pixel matrix) of the graphics primitives. The normal output of this flow is the computer screen, but it is possible to record the result in memory buffers so that it will return to the flow and be reprocessed aiming at more complex operations. This technique is called "render to texture" and the proposed system uses it extensively to process the map.

Vector graphics files are appropriate for the elaboration of choropleth maps and can be implemented for the Web by using SVG elements. An SVG is element is an image composed using a markup language similar to HTML that specifically allows the creation of animated vector graphics. It was incorporated by HTML5 and is supported natively in modern browsers. However, maps converted into vector files usually require detailed polygons, very often with a large number of vertices. Raster images are also common, and most popular map services are based on raster images. However, raster maps are frequently more costly in terms of bandwidth and memory. For more information about these technologies, see [World Wide Web Consortium (W3C) 2017; W3Schools Online Web Tutorials 2017; The Khronos Group Inc 2017].

As the system uses GPU programming directly, it was possible to implement it based on raster images. In order to do so, the images need to be pre-processed, so that location information (municipal codes, in our case) is recorded in the color space of each one of its pixels, thus making it possible to associate the map with its data.

The map displayed on the screen is an assembling of smaller pieces cut from bigger maps, previously rasterized at several levels of detail, as explained in [Schwartz 2016]. Each level of detail is composed of a set of square image tiles forming a pyramid starting at level 0 - a single tile - with twice that size at each new level until it hits the maximum level established for the system, which is 32. The dimensions ($mapWidth \times mapHeight$) change according to the level of detail (lod) and are defined by: $mapWidth = mapHeight = 256 \times 2^{lod}$. It is possible to project a point, (lat, lng) , on a pixel, (x, y) , as in:

$$x = \left(\frac{lng + 180}{360} \right) \times mapHeight$$

$$y = \left(0.5 - \frac{\log \left(\frac{1 + \sin lat}{1 - \sin lat} \right)}{4 \times \pi} \right) \times mapHeight$$

The selected images, or tiles, are squares sized 256×256 organized into a $m \times m$ matrix, where $m = 2^{lod}$, and where the position of a tile is defined by two indices ranging from $(0, 0)$ to $(m - 1, m - 1)$. That way, the tile that contains pixel (x, y) is at $(\lfloor \frac{x}{256} \rfloor, \lfloor \frac{y}{256} \rfloor)$.

Tiles are identified by Level-Of-Detail (LOD) codes, which are strings composed according to a combination of matrix coordinates. Those codes have important characteristics, since they identify the level of detail of a tile as well as its position in the hierarchy of the tree.

The size of the screen determines the quantity of tiles and the center can be projected onto the space of pixels of the map. So, to load the images and form the screen, one can simply compute the corresponding level of detail codes. The adjustment of position (panning) is done with the displacement vector obtained from the division that defines the tile and the moving of the map consists in

changing the center in sequence. The change of level of detail (zooming) only requires loading the respective *lod* tiles (see Figure 3).

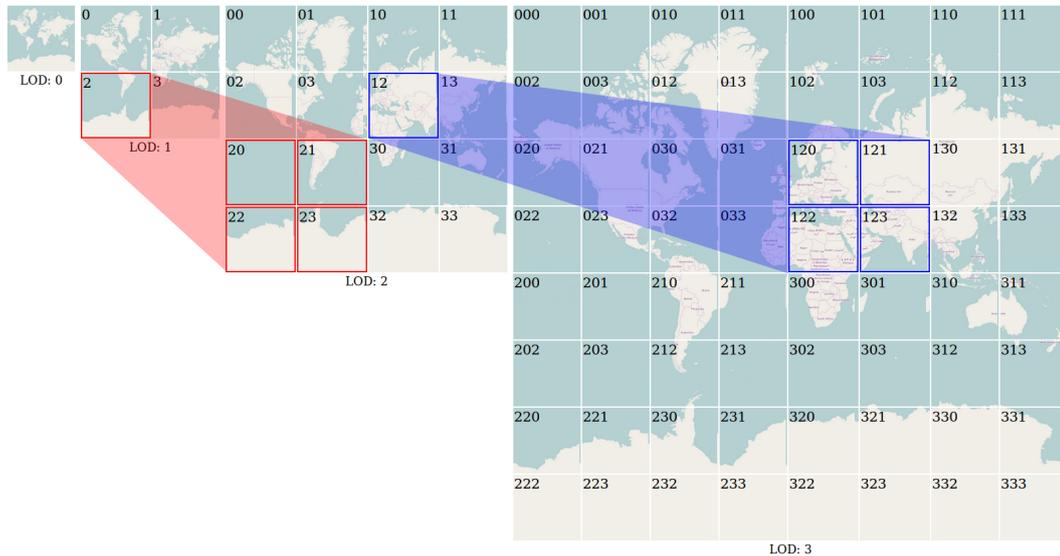


Fig. 3. World map tiles organized by LOD code strings, that have important properties: total map size is duplicated at every new LOD, whose level is represented by the string code length and its last character – 0, 1, 2 or 3 – sets the new tile's relative position.

In order to handle the visualization of the Brazilian territory, the system requires pre-computed image tiles to cover the world map starting at level 4 – a 4096×4096 pixel map, big enough to fit Brazil within a 1024×1024 pixel viewport – up to level 10, fine enough to allow the visualization of the smallest Brazilian municipality, Santa Cruz de Minas (MG) [IBGE 2016], within this same window.

The system's interface requires the assembling of two layers of tiles, a background image which serves as topographic reference and another one containing the location images of the same color of the pixels, where index values are responsible for the association of data and the regions. The RGBA channels of raster images are integers ranging from 0 to 255, allowing the formation of 256^4 (4294967296) color possibilities.

Besides the georeferenced tiles encoding municipalities, the values of the 140 socioeconomic variables are also encoded as images, in order to load them directly in GPU memory and operate the data also in shader programming. Each such variable is stored as an image where each pixel encodes the value of that variable for a given municipality. This image has a width of 256 pixels – a convenient value since it is the size of a color channel. In order to store the 5570 municipal identifiers, an image of height $\lceil \frac{5570}{256} \rceil = 22$ is necessary. The position of a pixel in that image is defined by the municipality index, an integer number between 0 and 5569. Thus, if the identifier of a municipality is stored in a two-byte integer, the first byte corresponds to the row and the second to the column of the pixel in the image. In fact these two bytes are exactly those encoded in the blue (B) and green (G) channels of a tile pixel (see Figure 4).

Another relevant piece of information that must be encoded in images is the size, in pixels, of each municipality for every level of detail. This is necessary to compute the actual contribution of that municipality per unit area, or rather, per pixel. These contributions are averaged to obtain the value of a given variable within each cell of a grid used for the visualization. A visualization cell is a square group of pixels ranging from 1 to 32 pixels wide used to compose the visual representation of the map.

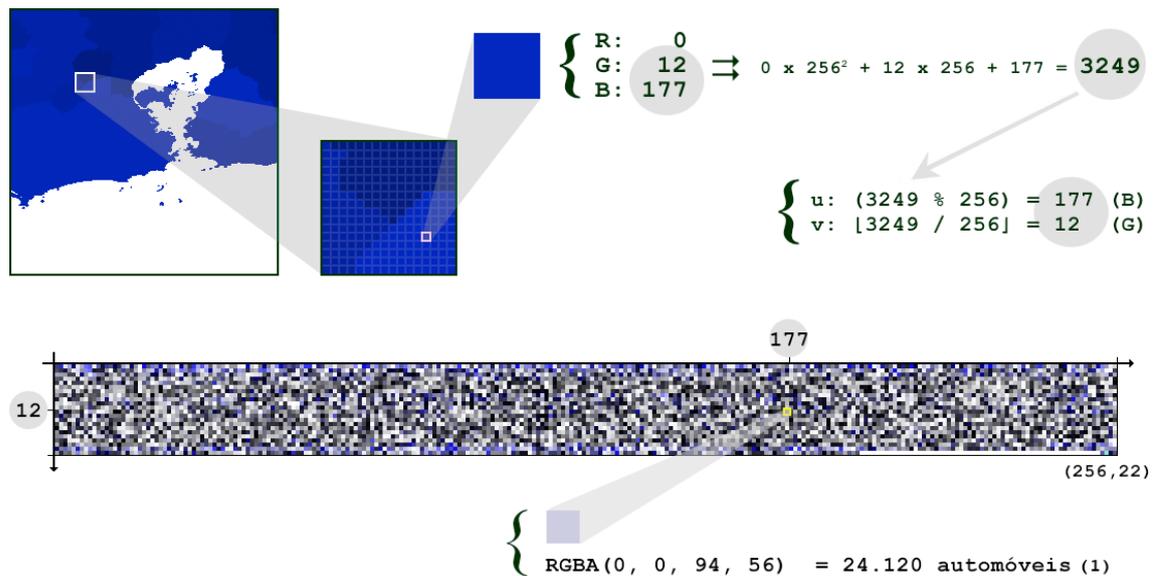


Fig. 4. Example of image encoding of a socioeconomic variable. Municipality number is retrieved from the tile (3249 in this case), corresponding to the pixel at row 12, column 177 of the variable image, which contains value 24120 encoded as a color.

The size of the visualization cells is chosen by the user. A cell of unit size will result in the traditional choropleth visual mapping, where the color of a pixel depends solely on the value of the variable for the region containing that pixel. Larger cells require averaging over all pixels within the cell, allowing a visual decorrelation from the political boundary of each municipality, as suggested by the research on statistical grids (e.g. [Bueno and D'Antona 2014]).

An example is shown in Figure 5, where a 16×16 visualization cell covers 3 municipalities represented with 31, 79 and 145 pixels, respectively. Since these municipalities are represented at that level of detail with 707, 441 and 969 total pixels, the average for the cell is given by $\frac{1}{256} (v_1 \frac{31}{707} + v_2 \frac{79}{441} + v_3 \frac{145}{969})$, where v_1, v_2, v_3 are the values of the variable in question for each municipality.

3.4 Rendering

All of the heavy lifting of the system is performed in a series of steps implemented by fragment shaders. The process starts by estimating the average value per visualization cell of each socioeconomic variable required by the visualization selected by the user. This follows the process outlined in the previous section. Up to six variable values might be estimated this way, since the visualization itself is a composition of up to three “themes”, each expressed as a rate between two variables, e.g., cars by population, schools by spending budget, etc.

Once the value for a given theme is known for each cell, a simple analysis is performed for that theme in order to gather the maximum and the minimum values, as well as a count histogram. This is done on a GPU using a parallel programming technique called *parallel reduction* [Fernando 2004]. These statistics are then used to build the range selection interfaces (see panel 6 in Fig. 1) which are used to establish the visual encoding for each theme. Two sliders are provided for each theme so the user may select a range of values. Regions of the map that satisfy this range query are painted accordingly. The histogram plotted above each range scale can be used to help the user select interesting ranges.

An earlier version of our prototype (see an example rendering in Fig. 6) allowed for the selection of three consecutive ranges for each theme, rather than a single range. These were mapped to either three

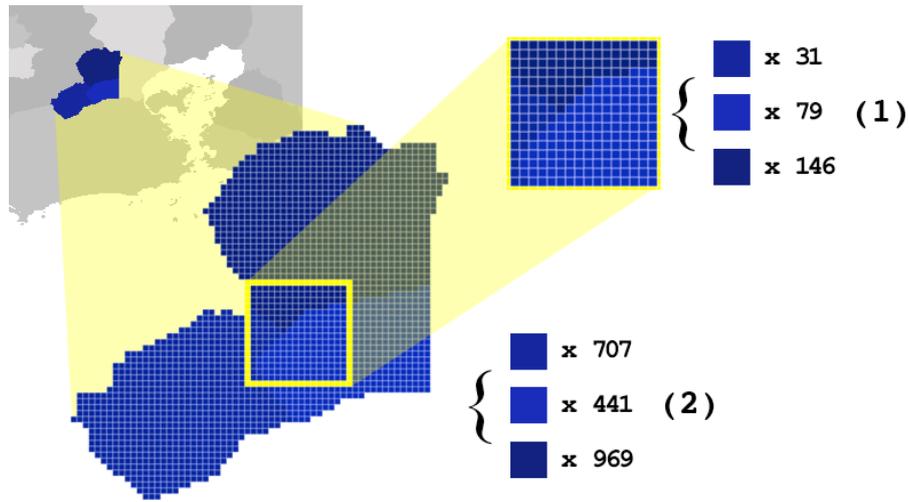


Fig. 5. Data decomposition by pixel and recomposition by visualization grid cell. The values shown in (1) represent how many pixels of each municipality are present in this 16 x 16 visualization cell. Values shown in (2) correspond to the total number of pixels of each municipality present in the current level of detail.

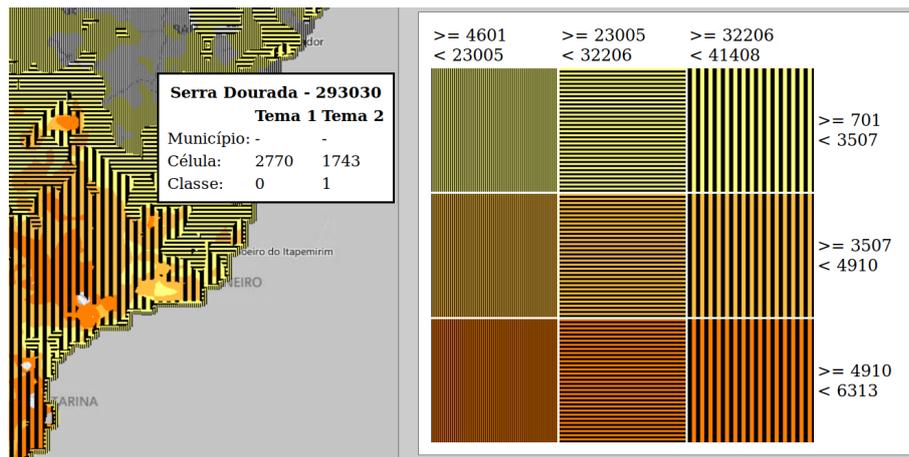


Fig. 6. Bivariate choropleth map using three categories for each theme, where the first theme is plotted with colors and the second with patterns. The 3 x 3 grid on the right is used as a legend to help the visual decoding of the map.

categorical colors (for the first theme) or three textured patterns (for the second theme). Although patterns and colors are more amenable to visual separation than using two colors with different hues, we noticed that the resulting rendering was hard to understand. For this reason, we adopted the choropleth design shown in Figure 1, i.e., three themes, each mapped to a single categorical visual encoding, where the first two themes are rendered with two primary colors (red and blue) and the third rendered as a user-selectable pattern. Areas within the range of both color-coded themes are shown in magenta, which is the secondary color obtained by mixing red and blue. Notice that, if desired, the two color-coded themes can be mapped to the same variable, but with different ranges selected, thus reproducing the design where the variable is drawn with three colors. Although this design may seem too restrictive, we must remember that the map thus produced is not static, but rather rendered at interactive rates as the user adjusts the range selection scales. In a static map, the relationship between value ranges and category colors is established a priori in order to cover the combinations of interest. In a dynamic interactive visualization, range scales can be adjusted to

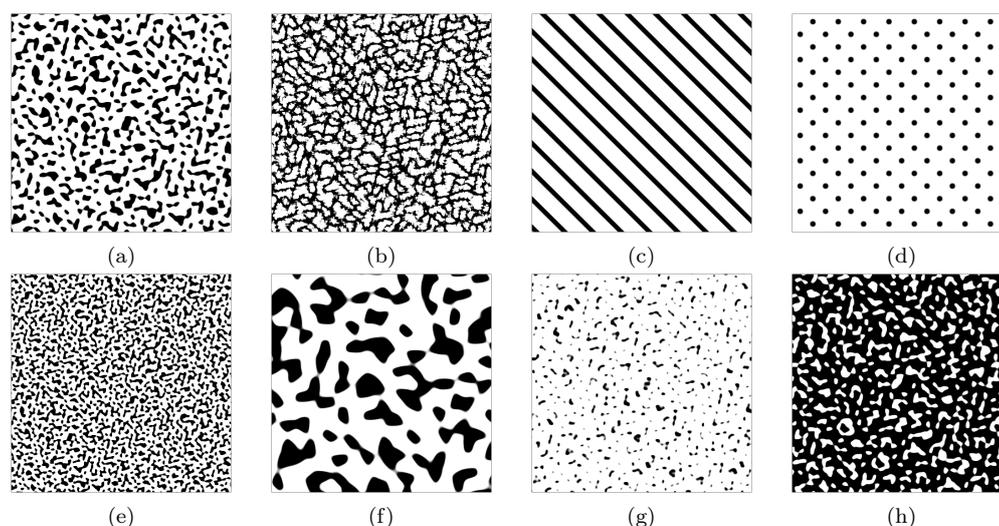


Fig. 7. First row shows examples of patterns made available in the prototype: (a) simplex noise, (b) turbulence, (c) stripes and (d) dots. The first two are irregular patterns, while the last two are regular. Second row shows examples of variations of the simplex noise pattern that can be obtained using the scale and level sliders (see Fig. 2), namely: (e) finer-grained, (f) coarser-grained, (g) lighter and (h) heavier.

explore any desired combination.

The use of a patterned texture rather than color to render the third theme aims at helping the user's selective attention mechanism. Our earlier prototype allowed for three kinds of patterns to be used, but the results were rather unsatisfactory since it was hard to distinguish one pattern from another in small areas. Nevertheless, distinguishing patterned from non-patterned regions is relatively straightforward, which led us to adopt a single type of pattern for the whole map.

Other important issues related to the use of patterns are those of granularity, regularity and weight. Coarse-grained textures are more visible for large areas, but might not be perceived when covering small areas. Similarly, regular patterns such as dots or stripes are pleasing and mostly fine for large areas but can interfere with the perception of the region boundaries. Finally, the weight of a pattern, i.e., the ratio between the areas covered by opaque and transparent pixels, influences the visibility of underlying areas. Light patterns let us see more of the colors beneath it but are harder to distinguish, whereas heavy patterns have the opposite effect. In order to address all of these issues, our prototype generates custom patterns on-the-fly using procedural textures, i.e., patterns generated by an algorithm and controlled by parameters. Six such algorithms can be selected using the interface (see Fig. 2), which generate both regular and irregular patterns. Variations with coarser or finer grain, as well as lighter or heavier weight can be commanded through the `scale` and `level` sliders. These algorithms were implemented based on examples discussed in [Vivo and Lowe 2017]. See Fig. 7 for a sample of these.

The final rendering of the map takes as input the visual encodings of all three themes. For each cell, the colors of the first two themes are added and the result is multiplied by the texture of the third theme. Optionally, several special effects implemented as shaders can be applied to this primary image. These are commanded through the buttons shown in panel 5 of Fig. 1, namely: `grid` shows the cells of the visualization grid, `interpolate` smoothes out the cell boundaries in order to reduce the pixelation effect, `intersect` removes from the display regions not covered by all of the selected ranges, `edges` uses an edge detection algorithm to paint in black the pixels bounding same-colored regions, `opacity` toggles the opacity of the rendering so that features of the political map can be seen underneath the colored/patterned regions, and `basemap` toggles the rendering of the political map.

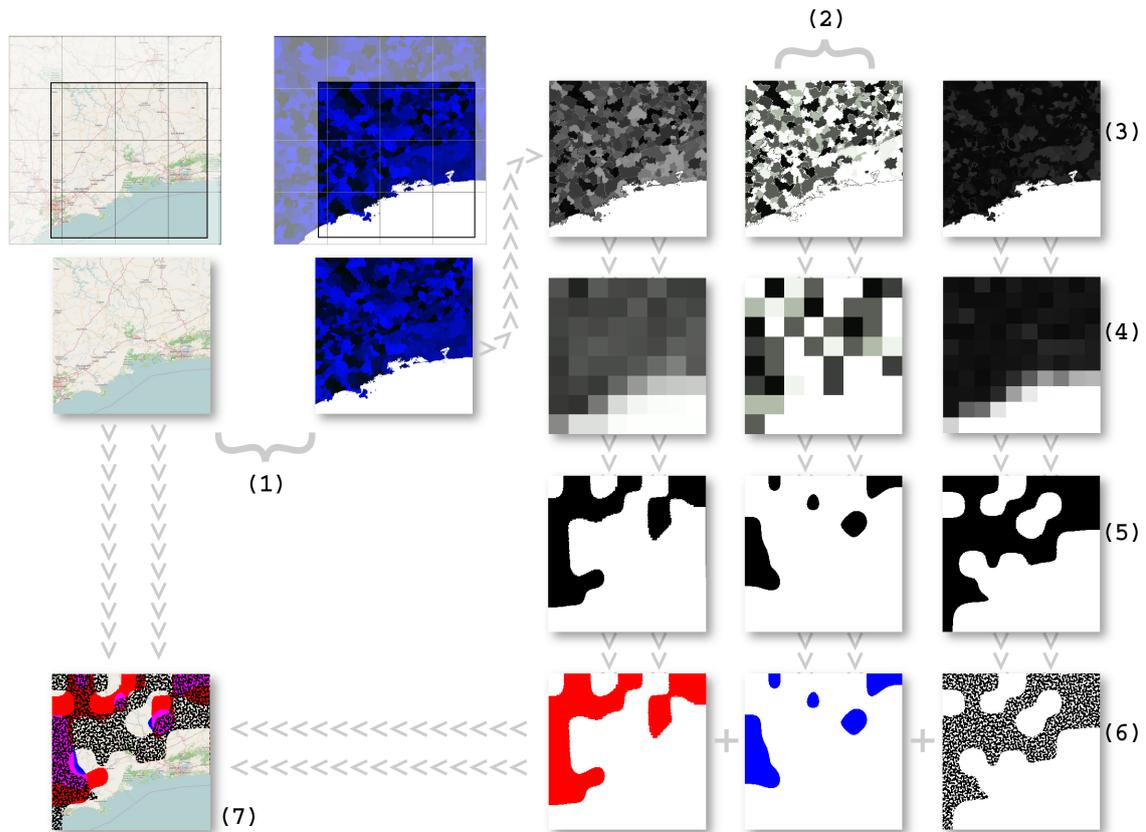


Fig. 8. Functional flow: assemble the tiles to display the map (1); for each theme (2) associate the location to its respective value, according to the amount of pixels contained in that area (3); equalize the values according to the grid size (4); perform interpolation, if this option is checked (5); assign the respective visual sign to the theme (6); blend the layers over the map (7).

The entire process restarts at every new triggered frame. Since rendering is instantaneous, the result is animated transitions according to user actions. Figure 8 illustrates the complete process.

4. RESULTS

To evaluate the system, we conducted experiments for the same two case studies presented in [Pinheiro and Esperança 2017]. Figure 9 shows the situation of vehicle fleet data, presenting the quantity of cars, motorcycles and trucks in relation to the total number of motor vehicles. In a single map, it is easy to see that the Southern and Southeast Regions concentrate the largest proportion of cars, probably because they concentrate Brazil’s wealthiest population. Differently from cars, motorcycles are more common in the North and Northeast Regions of Brazil, which is understandable since they are easier and cheaper to assemble and distribute in these parts of the country. On the other hand, trucks are spread throughout the Southern, Southeast and Central-West regions, coinciding with important agricultural areas. A clearer picture of each theme can be shown instantaneously by toggling the associated check boxes as shown in Figure 10.

In the second case, we selected three variables related to the second round of 2014’s Brazilian presidential elections: Total votes for the two top contenders: Dilma Rousseff and Aécio Neves and the total of valid votes are presented in Figure 11, that shows the votes received by each candidate for each 100 thousand votes. Theme 1, in red, shows the votes received by Dilma Rousseff, the winner.

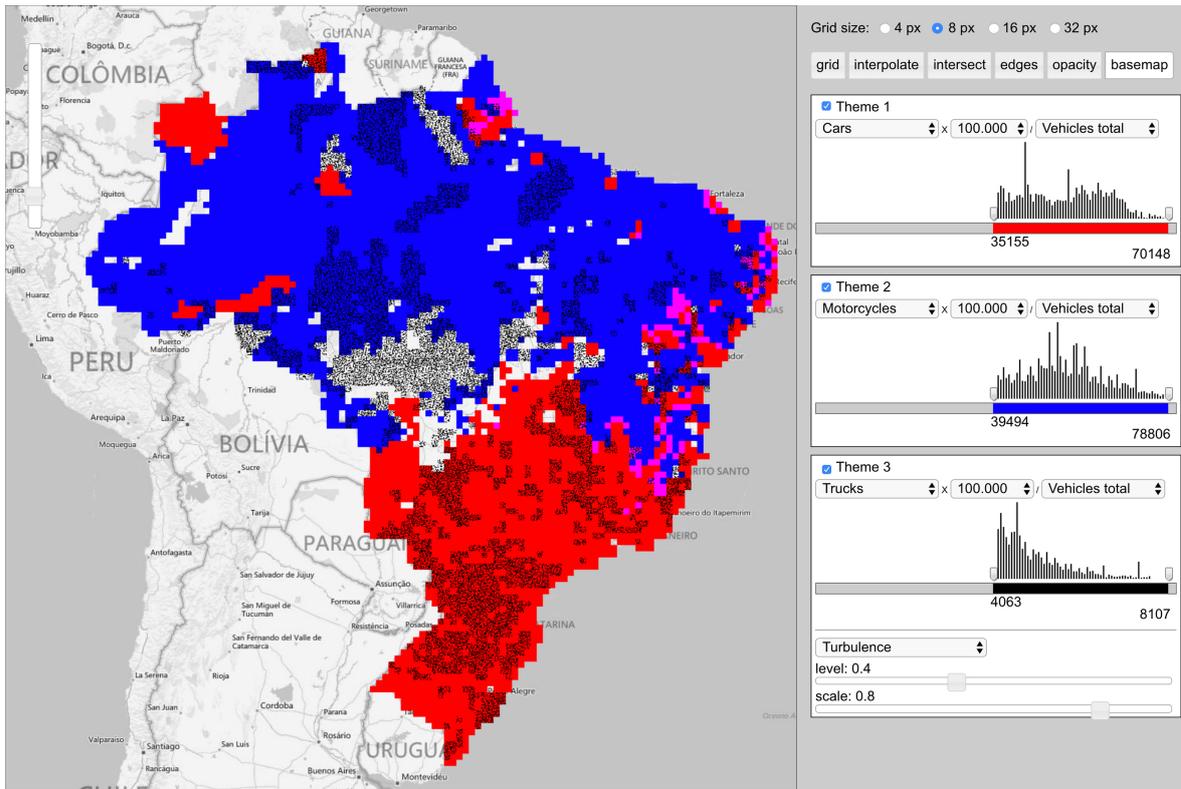


Fig. 9. Cars, motorcycles and trucks for every 100 thousand vehicles.

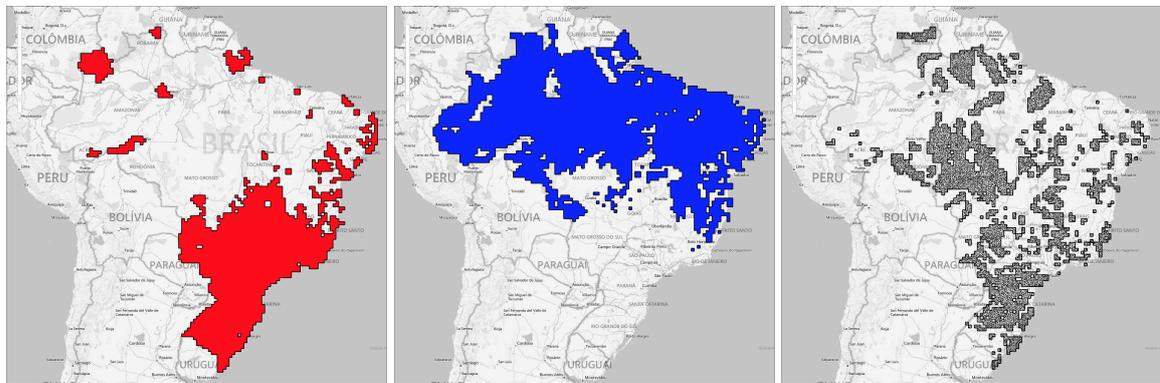


Fig. 10. Cars, motorcycles and trucks for every 100 thousand vehicles are shown one at a time by toggling their selection checkboxes and selecting the edge detection button.

Theme 2, in blue, shows the votes received by Aécio Neves. By adjusting each theme range slider it is revealed that Dilma Rousseff stood out in the country’s North and Northeast regions while Aécio Neves had a significant majority in the Southeast, Southern and Central-West regions of the country.

To better understand the behavior of the data organized in the map, Figure 12 shows a sequence of images obtained by moving the range slider for each theme/candidate. The real-time response of the system results in an animation capable of revealing how the data is distributed, as the colors appear and disappear in concert with the range sliders.

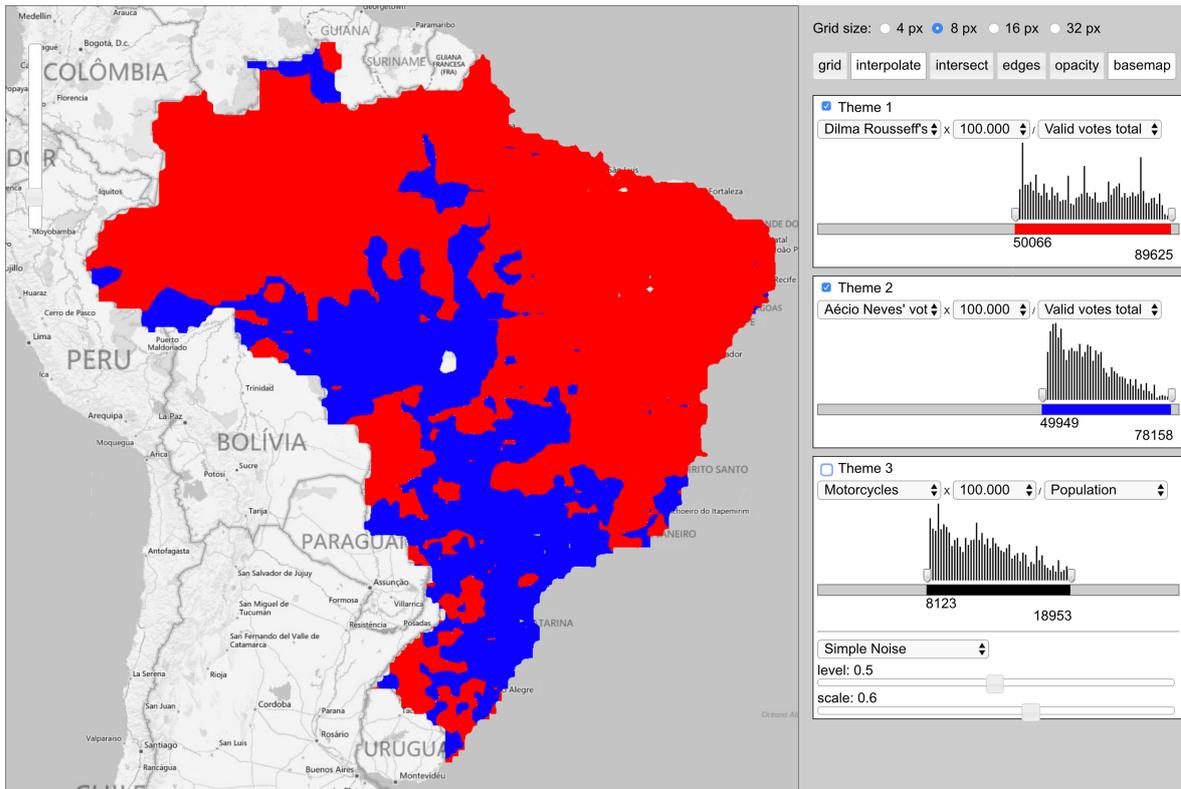


Fig. 11. Status of the second round of the Brazilian presidential elections.

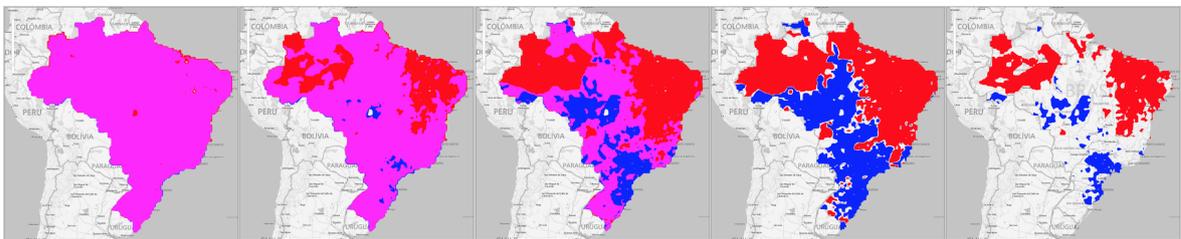


Fig. 12. Sequence of frames from the animation obtained by interactively adjusting the range sliders for the themes representing the votes received by each candidate in 2014's presidential elections.

In Figure 13 the sliders for the two candidates were set to show regions where each had 40% or more of the votes. The areas shown in magenta correspond to the intersection of these, i.e., they depict regions where the competition was fiercer. A third theme was added as a pattern showing GDP per capita in order to reveal if behavior of voters was influenced by their income in this suffrage. Indeed, a cursory examination shows that the richest regions of the country are more aligned with the areas where Aécio Neves did better.

5. CONCLUSION

The visualization of multiple variables on a map is a research challenge and rendering efficiency plays an important role in this process. The system described herein proved to be able to perform this task, allowing fast responses to user interactions in spite of the multiple processing steps. The good performance leads us to consider the use of the techniques applied here to visualize challenging vector

- BUENO, M. D. C. D. AND D'ANTONA, A. D. O. Avaliação de métodos de desagregação para geração de grades de população. *Revista Espinhaço* 3 (1): 127–137, 2014. Access: July 3, 2016.
- CHERNOFF, H. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association* 68 (342): 361–368, 1973.
- ELMER, M. E. *Symbol Considerations for Bivariate Thematic Mapping*. Ph.D. thesis, University of Wisconsin-Madison, 2012. Access: May 20, 2018.
- FERNANDO, R. *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics*. Pearson Higher Education, 2004.
- GERSHON, N., EICK, S. G., AND CARD, S. Information Visualization. *Interactions* 5 (2): 9–15, 1998.
- IBGE. Evolucao da Divisao Territorial do Brasil 1872 - 2010. ftp://geoftp.ibge.gov.br/organizacao_territorial/divisao_territorial/evolucao_da_divisao_territorial_do_brasil_1872_2010/evolucao_da_divisao_territorial_mapas.pdf, 2010. Access: December 27, 2015.
- IBGE. Malhas Digitais - Municípios 2013. ftp://geoftp.ibge.gov.br/malhas_digitais/municipio_2013/, 2013. Access: November 16, 2015.
- IBGE. Grade Estatística - Guia de Utilização. ftp://geoftp.ibge.gov.br/malhas_digitais/censo_2010/grade_estatistica/ge_guiu_utilizacao.pdf, 2015. Access: November 29, 2015.
- IBGE. Cidades@ - Minas Gerais - Santa Cruz de Minas. <http://cidades.ibge.gov.br/xtras/perfil.php?lang=&codmun=315733>, 2016. Access: August 26, 2015.
- INTERRANTE, V. Harnessing Natural Textures for Multivariate Visualization. *IEEE Computer Graphics and Applications* vol. 20-6, pp. 6–11, 2000. Access: July 3, 2016.
- NEWMAN, M. Maps of the 2012 US presidential election results. [http://www-personal.umich.edu/~sim\\$ejn/election/2012/](http://www-personal.umich.edu/~sim$ejn/election/2012/), 2012. Access: July 21, 2015.
- PICKETT, R. AND GRINSTEIN, G. Iconographic Displays For Visualizing Multidimensional Data. In *Proceedings of the 1988 IEEE International Conference on Systems, Man, and Cybernetics*. Vol. 1, 1988.
- PINHEIRO, T. M. AND ESPERANÇA, C. VisGL: an Online Tool for Visualization of Bivariate Georeferenced Data. In *Proceedings XVIII GEOINFO*. pp. 62–73, 2017. Access: May 17, 2018.
- SCHWARTZ, J. Bing Maps Tile System. <https://msdn.microsoft.com/en-us/library/bb259689.aspx>, 2016. Access: August 12, 2015.
- SHNEIDERMAN, B. Dynamic queries for visual information seeking. *IEEE Software* 11 (6): 70–77, 1994.
- STEVENS, J. Bivariate choropleth maps: A how-to guide. <http://www.joshuastevens.net/cartography/make-a-bivariate-choropleth-map/>, 2015. Access: May 20, 2018.
- STONE, M. Choosing Colors for Data Visualization. *Perceptual Edge*, 2006. Access: July 3, 2016.
- TATEOSIAN, L. G., HEALEY, C. G., AND ENNS, J. T. Engaging Viewers Through Nonphotorealistic Visualizations. *Proceedings of the Fifth International Symposium on Non-Photorealistic Animation and Rendering*, 2007.
- THE KHRONOS GROUP INC. WebGL Overview. <https://www.khronos.org/webgl/>, 2017. Access: November 8, 2017.
- VIVO, P. G. AND LOWE, J. The book of shaders. <https://thebookofshaders.com>, 2017. Access: May 17, 2018.
- W3SCHOOLS ONLINE WEB TUTORIALS. HTML - The language for building web pages. <https://www.w3schools.com/>, 2017. Access: November 8, 2017.
- WORLD WIDE WEB CONSORTIUM (W3C). Web Design and Applications. <https://www.w3.org/standards/webdesign/>, 2017. Access: November 8, 2017.