Mosaic Packing to Visualize Large-Scale Electric Grid Data

Adam B. Birchfield, *Member, IEEE,* and Thomas J. Overbye, *Fellow, IEEE*

***Abstract*—For large power systems, a continual challenge is to display wide-area data in a way that maximizes human users' situational awareness. This paper describes a new visualization technique that draws a mosaic of colored tiles to represent multiple data fields for electric grid objects, arranged to preserve geographic context. The key problem in creating these diagrams is packing the tiles onto the display space, minimizing the total displacement while forbidding overlaps. This paper formulates that problem and presents a horizontal-packing algorithm which is able to produce a feasible, quality solution at an interactive time scale. Illustrative examples are shown for using mosaics to monitor wide-area generator status and dispatch, bus voltages, and line and transformer limits. Mosaics can be customized in numerous ways to show different aspects of the system state, providing for human users a simultaneous sense of the wide-area summary, regional trends, and prominent outliers.**

***Index Terms*—power system visualization, mosaic displays, wide-area data visualization, packing problem.**

I. Introduction

Mosaics are pictures formed by small colored tiles arranged to impress upon the viewer a particular effect when viewed as a whole. Artists have crafted these for millennia. More recently, researchers have created mosaic-inspired data visualizations, to manifest insights for real-world datasets by conglomerating specific features of diverse, discrete objects [1]-[3]. The new mosaic-inspired visualization technique described here in the context of electric power systems has the potential to supplement existing data visualization to provide an additional level of insight and situational awareness to human users. This paper discusses how these mosaics are constructed and how they can be applied.

The application of specific interest here is large, high-voltage electric transmission interconnects. The massive datasets associated with these infrastructure systems consist of a variety of different, interrelated objects numbering from hundreds to tens of thousands, such as buses, generators, transmission lines, transformers, switched shunt capacitors and reactors, loads, and substations. The purpose of power system visualization is to help people such as engineers and operators understand the salient features of the system state that are most crucial to the study or scenario at hand. This is situational awareness, which the U.S. National Academy of Engineering has recognized as a need that new visualization techniques can help to address [4].

Since the early 1990s, technological advances in the field power system visualization have expanded the options for showing electric grid information. Instead of limiting visualization to spreadsheet-like tables of numbers and substation-specific circuit diagrams with text fields, graphical techniques have been developed for computers that include overlaying a diagram with a colored gradient contour of some numerical property such as voltage or locational marginal prices [5], or adding dynamically-size pie charts [6] or animated flow arrows [7]. Another example is adapting the display diagram to the portion of the network that is relevant to the task at hand [8]. Several recent advances have been made to use automated network diagram drawing, including methods that use only electrical data [9]-[10] and those which use geographic coordinates to guide the graph layout, adjusting to improve readability [11]-[12].

This paper presents a new visualization method, a geographically-constrained mosaic display. This idea relates to several similar techniques in data visualization. What are usually called mosaic displays are space-filling grids that indicate cross-correlation between two or more categorical properties [1], [13]. The present method differs by adding geographic constraints on the location of the tiles, with potentially thousands of tiles shown. Similarly, tree-view diagrams show nested categories using embedded rectangles [14]. Geographic rectangular cartograms are also similar to the present approach [2], [15]-[16], but these focus on distorting shapes to show relative density, rather than placing discrete

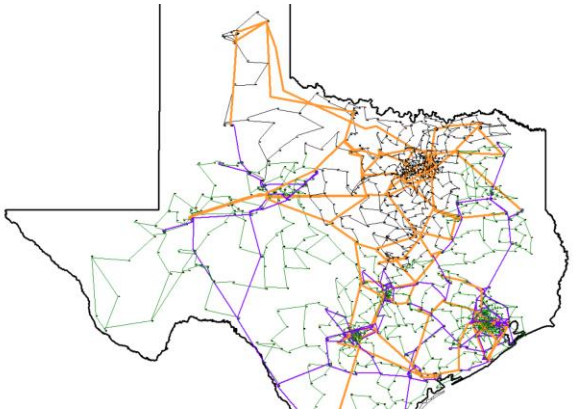

Fig. 1. One-line diagram of the synthetic 2000-bus case, a fictitious, realistic grid geo-located in Texas.

Manuscript drafted August 13, 2019, revised May 16, 2020. This work was supported in part by the Advanced Research Projects Agency-Energy (ARPA-E), United States Department of Energy, and by the National Science Foundation, Grant #1916142.

The authors are with Texas A&M University, College Station, TX, USA, 77840 (birchfieldllc@gmail.com, overbye@tamu.edu).

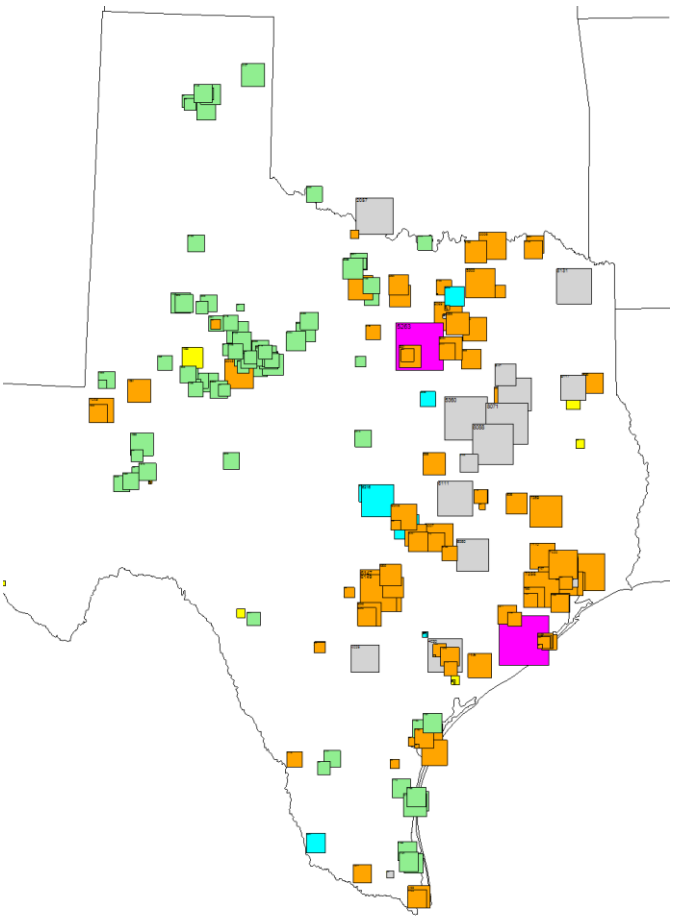

Fig. 2. The 2000-bus case generating units, shown at their actual geographic location. The generators are sized proportional to the unit's maximum generation capacity, scaled so that the total area of all the tiles sums to 30% of the figure's area. The color of the tile indicates the unit's fuel type, with magenta nuclear, gray coal, orange natural gas, blue hydro, green wind, and yellow solar.

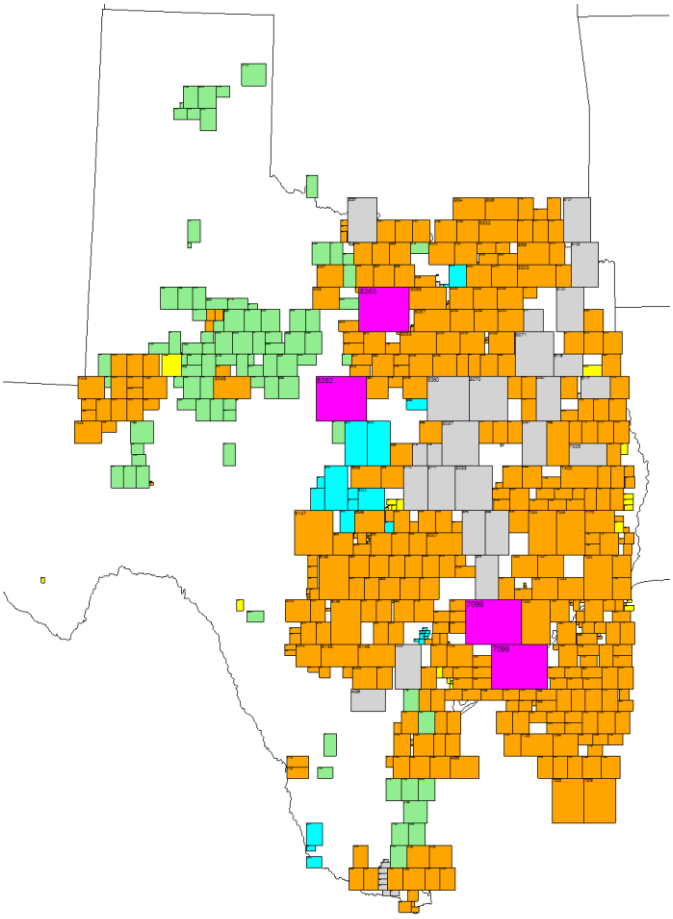

Fig. 3. The 2000-bus case generating units, shown in a mosaic display positioned with the horizontal packing algorithm. The generators are sized and colored as in Fig. 2.

objects. The mosaic diagrams described in this paper build on the initial work in [17], which uses a simple method to draw the mosaic in rows and columns to fill the whole screen.

A fundamental assumption of the mosaic displays in this paper is that geographic context is a key aspect of situational awareness. Though geo-coordinates are not necessary for many power system studies, they drive the system design; thus much of the system structure is constrained geographically and many elements are best interpreted in relation to their geographic neighbors. Increasingly, geo-coordinates are being recognized as an essential element of power system datasets, in part due to their role in simulating the impacts of geomagnetic disturbances [18]. In addition, recent developments in the creation of synthetic electric grids provide widely-available realistic test cases, including geo-coordinates, that mimic actual grid properties while not representing any actual grid [19]. The 2000-bus case shown in Fig. 1 is used for demonstration throughout this paper; all the data associated with it is freely available online [20].

II. The Mosaic Packing Problem

A. *Illustrative example*

For illustration purposes, this section uses the data visualization application of showing the generation capacity and fuel type for the 544 generating units in the 2000-bus synthetic grid. Each mosaic tile represents a generating unit, with the tile area proportional to the maximum active power generation capacity (MW) and the color indicating the fuel type of the unit. Placing the tiles purely geographically, as in Fig. 2, as is done in [21], gives some idea of the distribution of the geographic units, but this diagram has several serious limitations. Although the units are sized so that the total area used is 30% of the full figure display space, only a fraction of this space is used since nearby units overlap. The viewer cannot tell, for example, how much wind is really in the western part of the grid or natural gas in the southeast. The fact that each of the two nuclear stations is actually two units is hidden because each pair of units is placed on top of one another. Making the tiles smaller just makes them more difficult to see, whereas making them bigger only exacerbates the overlap issue.

Fig. 3 shows the corresponding mosaic display for the geographic display of Fig. 2. The same tiles are present with the same areas and colors, though the shapes and locations are somewhat distorted to remove the overlaps while keeping the

geographic locations as close as possible.

In this view, the clutter is gone and much more information is readily discernable. The proportion of the fuel types is visually accurate (with Fig. 2 one might have concluded that coal and natural gas were of nearly equal proportion). The individual unit sizes and types are more easily seen, and the relative sizes of different fuel types are clear. While the geography is distorted somewhat, it is not altogether lost. It is still quite clear that wind is located primarily in the west and south and that coal is concentrated in the center-east. A few small solar plants in the center of the state were not visible at all in Fig. 2. With an actual display application of mosaics such as Fig. 3, the identity of each tile, indicated by a small number in the upper-left corner, could be determined by zooming and panning the display, or by clicking on the tile.

B. Formulation

The overall goal in constructing mosaic displays is to pack into a rectangular screen a number of rectangular tiles of different areas. This involves setting the tiles' position and dimensions. The dimensions do not need to be square, but they should not be extremely thin; ideally the aspect ratio should be bounded, say between 0.5 and 2.0. Each mosaic tile has a preferred position: its geographic location. Optimality in this problem refers to minimizing the norm of the Euclidian displacement of all the tiles. But this placement is subject to a strict non-overlapping constraint which is the eminent feature of mosaic displays, and also the part that makes this problem so difficult.

The formulation of the mosaic packing problem is as follows:

$$\mathrm{Min}_{\bar{x},\,\bar{y}} \sum_{i \in N} ((x_i - x_{i0})^2 + (y_i - y_{i0})^2) \tag{1}$$

$$s.t. \quad x_{min} \le x_i \le x_{Max} - w_i \tag{2}$$

$$y_{\min} \le y_i \le y_{\mathrm{Max}} - h_i \tag{3}$$

$$w_i \cdot h_i = A_i \tag{4}$$

$$r_{\mathrm{m}} \le \frac{w_i}{h_i} \le r_M \tag{5}$$

$$\text{For each } i,\ j \in N,\ i \ne j \tag{6}$$

$$x_i + w_i \le x_j \ \ \text{OR} \ \ x_i \ge x_j + w_j$$

$$\text{OR} \ \ y_i + h_i \le y_j \ \ \text{OR} \ \ y_i \ge y_j + h_j$$

In this problem, it is needed to fit N tiles into a box bounded by $[x_{\min}, x_{\mathrm{Max}}]$ and $[y_{\min}, y_{\mathrm{Max}}]$. Each rectangle i has input variables area A_i and preferred position (x_{i0}, y_{io}). Its aspect ratio is required to be between r_m and r_M. The decision variables are the tile's position (x_i, y_i) and dimensions (w_i, h_i).

C. Solution approaches

Without the non-overlapping constraint (6), the solution would be a trivial geographic display as given in Fig. 2. Because of this constraint, the feasible space is non-convex and disjoint. Since the goal here is to keep this constraint strict, a natural direction is to take a combinatorial, discrete programming approach. This could be done by discretizing the space and making x, y, w, and h into integer variables. Doing so would make a grid with resolution as appropriate to balance the screen size, accuracy, and speed requirements. This framework would convert the problem to a constrained assignment problem of placing N' sub-tiles onto a $x_M \times y_M$ sized grid.

$$\mathrm{Min} \sum_{(i,j \in N', M)} d_{i,j}\, a_{i,j} \tag{7}$$

$$s.t. \sum_{i \in N'} a_{i,j} = 1, j \in M \tag{8}$$

$$a_{i,j} \in \{0,\ 1\} \tag{9}$$

Here, $a_{i,j}$ are elements of an assignment matrix assigning the sub-tiles to locations. Assignment problems have known solution approaches [21]; however, each of the sub-tiles has additional constraints in that they must be arranged particularly near the other sub-tiles in its tile, a constraint that precludes the use of many typical algorithms.

One solution approach that would exactly solve this integer program is a depth-first branch-and-bound [22]. This approach works with the following steps, starting with no tiles on the placement stack and an infinite upper bound:

1) If the total displacement of the stack is not less than the upper bound, pop the top tile and return to step 1. (If the stack becomes empty End.)
2) If the top of the stack is overlapping another tile, or if the last move was to pop from the stack: move the top tile slightly, radiating systematically, and go to step 1.
3) If all tiles are placed: save the current stack as the best solution so far, update the upper bound, pop the top tile, and go to step 1.
4) Push the next tile (any order) to the stack and put it in its preferred location, then return to step 1.

This approach bounds each placement by the best found position so far. This approach was implemented and gave reasonable results for a simple 7-tile problem.

Unfortunately, solving the problem this way, along with any standard or exact combinatorial method tried, is far too slow. Beyond 7 tiles the branch-and-bound approach computation time expands exponentially. Even if it could be reduced somewhat, computational performance is paramount in this application, since ideally users should be able to create a new display quickly, even interactively, when the data metric corresponding to size changes. Hence, an absolute ceiling for computational order is $O(N^2)$ with the number of possible configurations greater than $N!$. Ideally, a solution should be faster than $O(N^2)$, since N could be 10,000 or more.

A simplistic, quick solution is a greedy approach, which adds each tile in sequence in a locally-optimal place. Such an approach is straightforward to implement at $O(N^2)$, and with appropriate data structures can be reduced in practice to nearly $O(N)$, since only a few locations need to be checked with each addition. This approach is decidedly sub-optimal, but meets all constraints and the computational order requirements. If the problem is not very constrained, this approach works well; however, as the problem becomes more constrained the results become quite dependent on the order in which the tiles are added. Tiles added last are placed very far from their geographic location, causing the distortion to be concentrated on the last set of tiles added rather than evenly spread. The

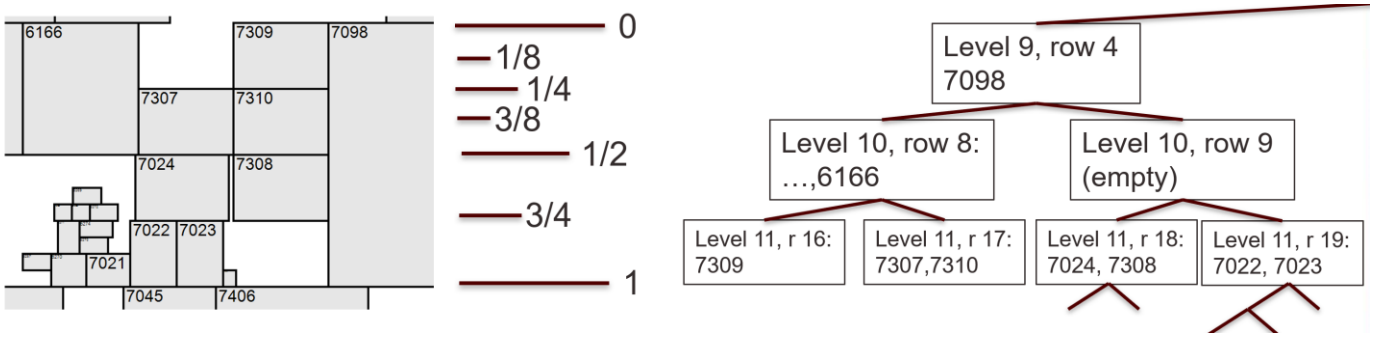

Fig. 4. Data structure used for the addition step of the horizontal packing algorithm. Left, the dividing of rectangles into levels and rows; right, the part of the binary tree representing this set of tiles.

following horizontal-packing approach improves on the greedy approach, still sub-optimal but better while still maintaining a quick computational performance.

III. The Horizontal Packing Algorithm

The method used to create the mosaic display shown in Fig. 3, along with the mosaic applications highlighted in Section V, employs a horizontal packing approach, which this section describes. The basic intuition is to limit the tile heights and vertical positions to discrete power-of-two values so that they line up horizontally and can easily slide left and right, allowing the algorithm to "pack" the tiles along the horizontal direction.

The first step is to divide the vertical direction into levels and rows. Level 0 corresponds to the height of the entire map space, with only one row. Hence row 0, level 0 means an enormous tile that covers the entire map vertically. Level 1 is half that height, with two rows: row 0 for the top half and row 1 for the bottom half. Each succeeding level involves twice as many rows that are half as tall as the previous level. So each tile will be assigned a level L_i, which will determine its height as $h_i = h_w 2^{-L}$, where h_w is the height of the world. Then its vertical position will be $y_i = h_w 2^{-L} R_i$ based on its row R_i, which will be an integer between 0 and $2^L - 1$. The width of the tile must be $w_i = A_i / h_i$. There is exactly one level L_i for any given area A_i which will guarantee an aspect ratio in the range [0.5, 2.0]. Thus each tile has a pre-determined level L_i and width and height, fulfilling constraints (4) and (5). All that is left is to determine the discrete row R_i and the horizontal position x_i.

As a result of this discretization of the vertical direction, sliding in the horizontal direction is easier. Notice in Fig. 3 that pushing on any tile left or right only affects a narrow band of tiles (the same is not true of the vertical direction). This arrangement lends itself to a binary tree data structure for the levels and rows, as illustrated in Fig. 4, where each non-empty row has a list of tiles placed on that row, ordered left to right.

The algorithm proceeds as follows:

1) Ordering tiles by x_{i0}, split them along the median tile into two queues: a left queue and a right queue.
2) Take one tile at a time, alternating between the left and right queues, ordered by x_{i0} from the median outward. For each tile do two steps:
   a. Addition step: add the tile to the best available spot only to the left (or right) of already placed tiles.
   b. Pushing step: push the tile and its neighbors inward horizontally towards the median an optimal distance.

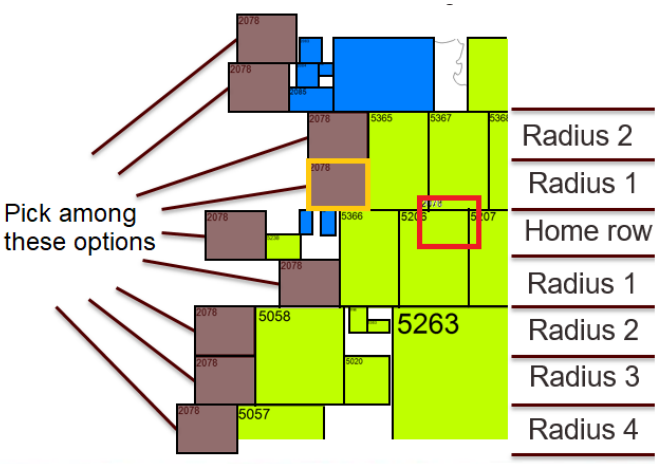

Fig. 5. During the addition step, a row for a tile is chosen. In this example, yellow-green and blue tiles have already been placed, and the rectangle outlined in red on the right shows the new tile's preferred location. The brown rectangles are options for addition; the one outlined in orange is best.

Thus this algorithm adds the tiles starting at the median outward, alternating left and right. For each tile, there are two steps: addition and pushing. The further description below will assume a tile from the left queue is being placed, the right queue process being symmetric.

The addition step ensures that the tiles are at least initially placed commensurate with the horizontal ordering of their preferred location. Since the tile to be added has preferred location left of all placed tiles' preferred location, restricting its placement to the left of placed tiles is reasonable. A tile only needs to search the rows of its level, checking upward and downward on the binary tree to find the x-position just to the left of already placed tiles. The tile then picks the row where the displacement function is minimized as in (1). The most efficient search starts at the home row, radiating up and down until the best displacement found is better than $(y_i - y_{i0})^2$ for any further radiating. Fig. 5 illustrates the addition step. If all placed tiles are to the right of x_{i0}, the tile can be assigned $x_i =$

Fig. 6. The effect of the choice of pushing distance. The brown tile is being pushed from left to right. Left, before pushing. Center, smaller pushing distance. Right, larger pushing distance.

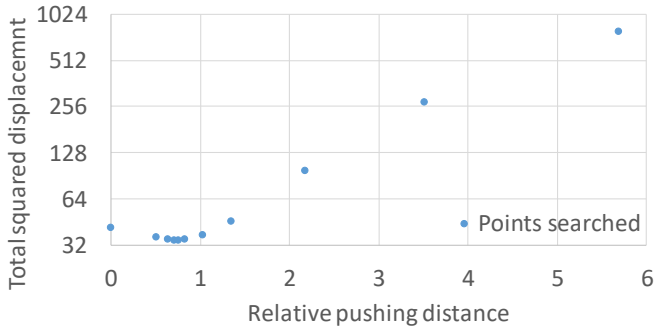

Fig. 7. Golden section line search to determine pushing distance.

x_{i0}, leaving a horizontal gap between it and existing tiles.

The pushing step allows the tiles to be "packed," by sliding the new tile and its neighbors rightward and potentially closing the gaps created by placing earlier tiles. For efficient implementation, the pushing step requires a new data structure, a neighbor-graph where each tile has a set of left- and right-neighbors. During the addition step, this data structure is updated by traversing the binary tree and adding connections between the new tile and any tiles directly to its right.

The pushing involves first gathering the set of tiles that will be pushed, the "push group," by traversing the neighbor graph links. For a given push distance, each element in the push group, ordered from left to right, has its x_{i0} increased if it was pushed.

Then the final question of the pushing step is how far to push the push group. Fig. 6 illustrates different options for the pushing distance, which to different degrees disturbs the position of existing tiles but may move the new tile closer to its preferred location. In selecting the pushing distance, many distance are tried, with the final decision based on what minimizes the total displacement (1) of the pushing group. This is a line-search problem, which can be solved very efficiently by a golden section line search, shown in Fig. 7. The area searched is convex due to the left-right ordering of the tile placement; hence this line search is guaranteed to converge, and converges very rapidly. The exact number of iterations needed depends on the tolerance chosen; for this application 10% of the tile's width is used as the tolerance and usually about 10 iterations is plenty for convergence.

The addition and pushing steps to place the tiles will thus enforce (6) while seeking (not guaranteed) to minimize (1). Constraints (4) and (5) are already met as described above, and if the addition step refuses to search rows that violate (3) then that constraint will be met as well. The main constraint that still needs to be met is (2), which is the horizontal bounds. (Enforcing the boundary box is optional, but it is often desired to show all the tiles on a single screen.) In this algorithm, constraint (2) is relaxed by adding a barrier function to make the objective function as follows:

$$\sum_{i\in N} \begin{matrix} ((x_i - x_{i0})^2 + \alpha(y_i - y_{i0})^2 \\ +10^{12-p(x_i - x_{\min})} \\ +10^{12-p(x_{\max}-(x_i+w_i))}) \end{matrix} \quad (10)$$

In this augmented objective function, $p = 200/(x_{\max} - x_{\min})$, and α is a heuristic factor, set to 1.5, to encourage the addition step to prefer vertical searching to horizontal. This is the only constraint that is relaxed.

IV. Performance Evaluation Results

A. Computational order

The computational order of the horizontal packing algorithm is proportional to both the number of tiles N and the sum of the time taken for the addition and pushing steps.

The addition step speed depends on the number of rows searched R_{ni} and the number of tiles that must be checked when searching a row S_{ni}. The latter involves searching a binary tree—only the left-most object in each row must be searched. Searching upwards is $O(\log_2 N)$, and searching downwards could be $O(N)$ for the largest nodes, but it cannot be so for all nodes; in fact, for leaves it will be zero. This search time amortizes over all nodes to $O(\log_2 N)$. For R_{ni} the worst-case is 2^L since there are that many rows to search, but this is guaranteed to be limited by the $N-1$ other tiles that could be in the way and approximately by $\sqrt{N}$, provided the bounds are not excessively restrictive, since the most constrained cases will form a square and only one column needs to be searched. The total computational order of the addition step is $O(\sqrt{N}\log_2 N)$, but in practice this step is much faster than the pushing step.

For the pushing step, which dominates the computation time, the time is proportional both to the number of golden section iterations G and the size of the push group P which much be checked at each iteration. As discussed above, G depends on the tolerance, but is constant with respect to N. The push group could be in the worst case $O(N)$ if all the tiles are lined up horizontally, but unless the bounding box is very unusual this will not happen, since the addition step will move tiles to other rows. Thus the typical worst case will involve a push group representing one row of a square, hence the speed will be $O(\sqrt{N})$.

Therefore, the predicted overall worst-case computational order of the horizontal packing algorithm, barring unusual conditions, is $O(N\sqrt{N}) = O(N^{1.5})$. An upper bound considering unusual conditions would be $O(N^2)$.

B. Experimental computation speed

To test the computational order of the horizontal-packing algorithm, four types of mosaics were built. Two involve the 2000-bus case already described, and two with a larger 20,000 bus case to show the scalability. For each system, a mosaic

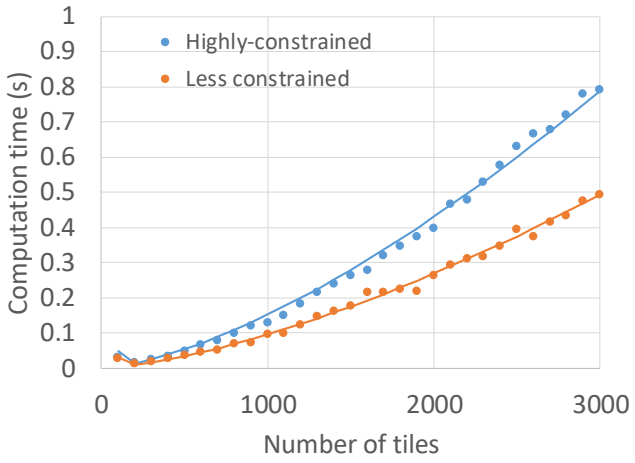

Fig. 8. Computation time for 2000-bus case experiments, and trend-lines with $O(N^{1.5})$.

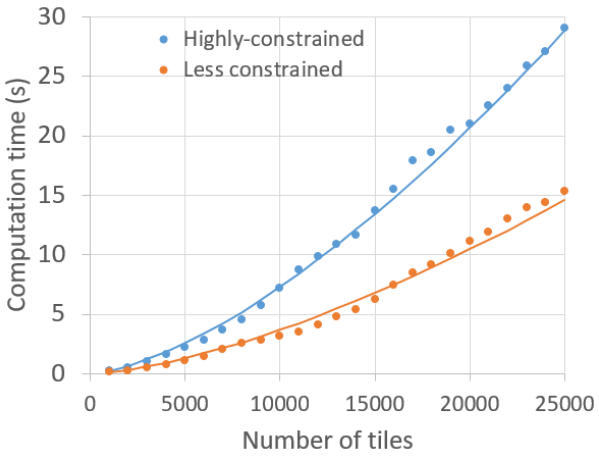

Fig. 9. Computation time for 20,000-bus case experiments, and trend-lines with $O(N^{1.5})$.

problem was defined where the tiles represent system branches (lines and transformers), sized proportional to the branch's apparent power limit. Both systems' mosaics were created both with a highly-constrained configuration with the tiles sized to take up 80% of the display space, and a less-constrained problem where the tiles were sized to take up 5% of the display space.

For each of these four mosaics, many tests were run, selecting a random sampling of the actual branches from the cases. For the 2000 bus case, the times were recorded in increments of 100 tiles up to 3000 total tiles. For the 20,000 bus case, the times were recorded in increments of 1000 tiles up to 25,000 total tiles. The results are shown in Figs. 8-9.

Regression analysis shows that an $O(N^{1.5})$ computational order, as shown in Figs. 8-9, fits the data with > 0.99 correlation coefficient. This confirms the scaling principal predicted in IV.A. In numerical terms, all sizes of the 2000-bus case mosaics take under 1 second to compute, meaning that they can be created and updated dynamically and interactively. This is still true up to about 5000 tiles (3 seconds). For a very large interconnect case, such as the 20,000-bus case, creating a mosaic with this algorithm is still very feasible, with a computation time up to 30 seconds.

V. Applications of Mosaics

Variations in the construction of mosaic displays allow them to be used for a wide variety of planning and operations applications. This section presents and analyzes an initial selection of these variations.

An impactful design decision in creating mosaic displays is how large to scale the tiles. Since the tiles will never overlap,

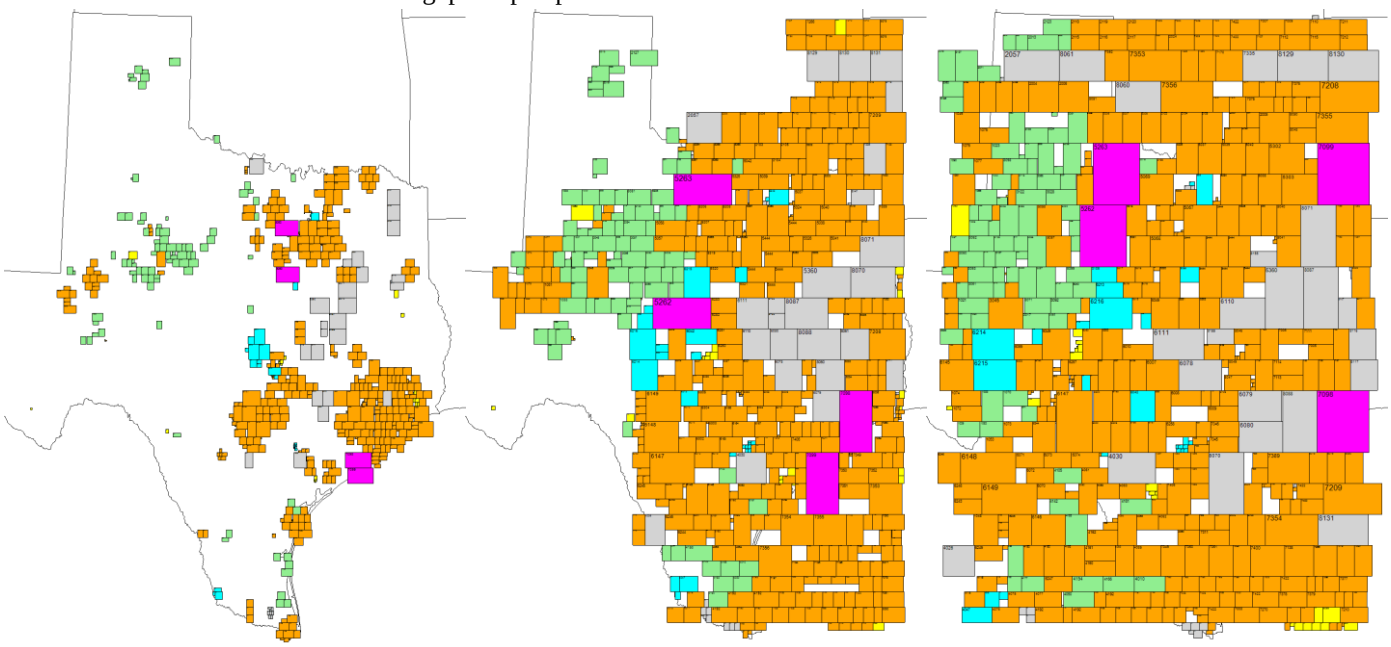

Fig. 10. Generator mosaic for the 2000-bus case, scaled to 10%, 50%, and 80% of the display space, respectively. Relative size and coloring represent generation capacity and fuel type, as in Fig. 2.

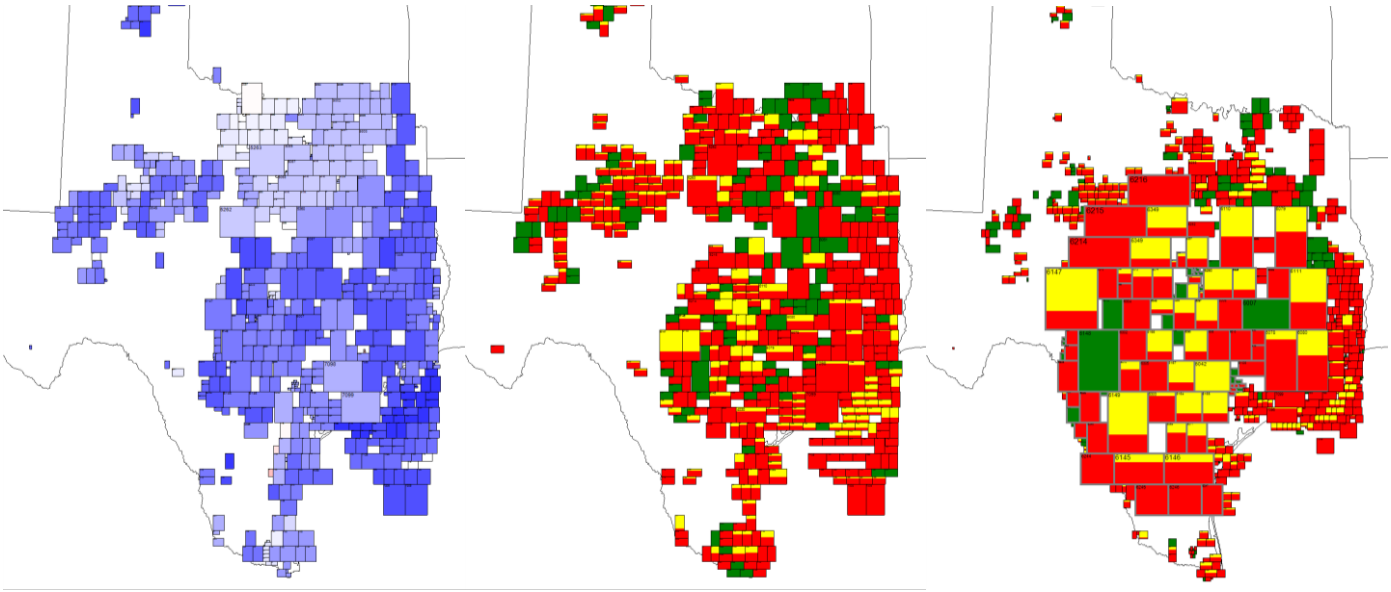

Fig. 11. Variations on the generator mosaic for the 2000-bus case. Left shows coloring by per-unit voltage, where blue is high, white is near nominal, and red is low. Center and right, coloring by dispatch (green is offline, red is dispatched, yellow is available on-line) with two variations in tile sizing: right, one area highlighted by making it larger with thicker borders, and center, enforcing a minimum tile size so smaller generators can be seen.

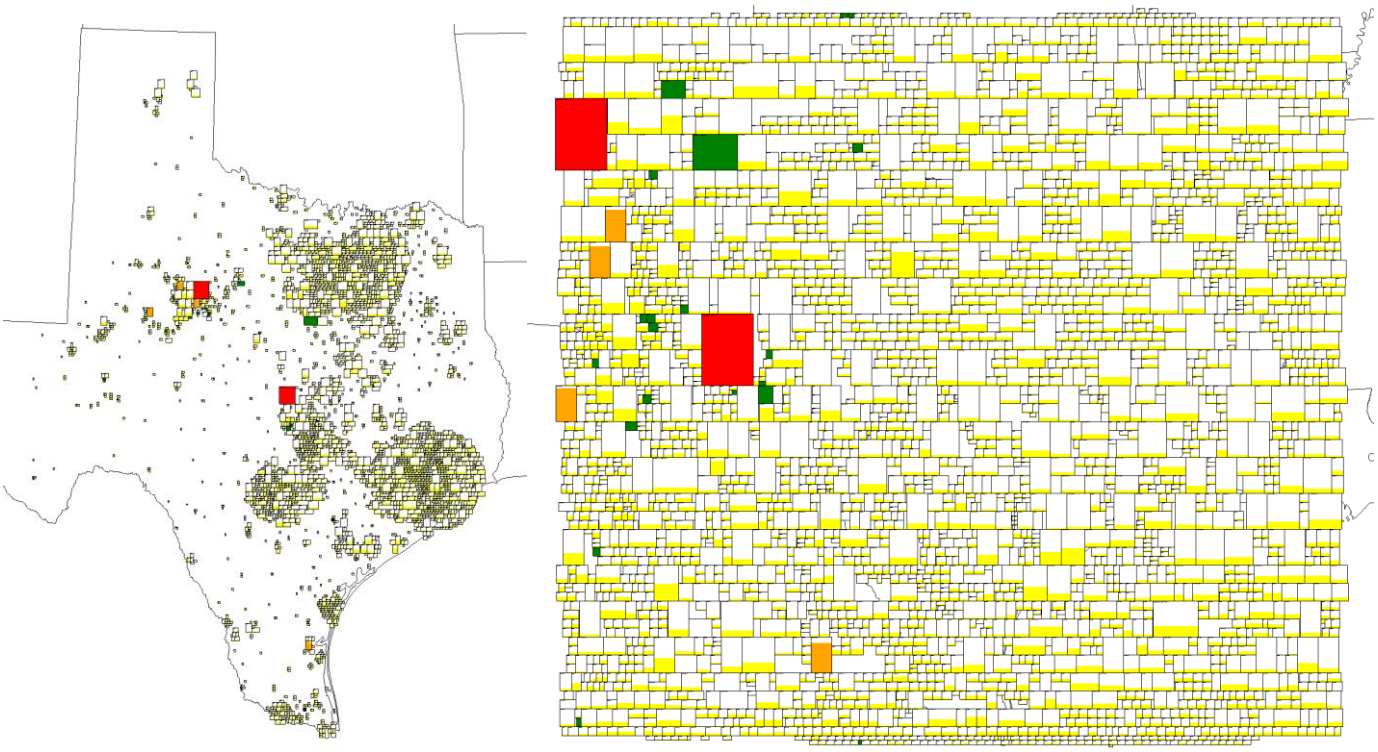

Fig. 12. Mosaic of branch limits for the 2000-bus case. Tile size indicates the line or transformer limit, with overloaded lines shown larger. The fraction of yellow and white indicates how close the branch is to a limit. Green branches are out of service; orange are near their limit; and red are over their limit. Left, a smaller, more geographic mosaic; right, size expanded to fill the screen.

the display can be meaningful at different sizes. As Fig. 10 (along with Fig. 3) shows, decreasing the size can help make geographic context more clear, though there will nearly always be some distortion, such as for units at the same plant. The larger, more filled mosaics, such as the third one in Fig. 10, allow nearly all the display space to be used, keeping some degree of relative geographic connection despite significant distortion. For users with significant familiarity with a system, this sort of display might be best, since the larger tiles allow more detail such as text to be added to each tile.

Many quantitative and qualitative variables could be indicated by the color of the mosaic tiles, such as the control area number or per-unit voltage. As in the center and right mosaics of Fig. 11, the coloring could be compound, such as

showing out-of-service units in green and in-service units in a proportion of red and yellow that indicates whether the unit is closer to its minimum or maximum limit. Other variations in the presentation methods are also shown in Fig. 11. Though the relative size of tiles is a meaningful way of showing data, the very smallest objects can be too small to see. An alternative is to enforce a minimum tile size as a fraction of the largest tile size. In the extreme, one could set up a mosaic with all tiles the same size, which the horizontal packing algorithm would still handle well to create a grid-like uniform mosaic. In contrast, one advantage of allowing different size tiles is that an area of particular interest can be emphasized, by making tiles from that area say 10 times as large as those from other areas. Tiles can also be highlighted by fill color or border color. Another presentation choice is the background color, which could be dark for a display monitor or light for a printed page.

The objects representing tiles need not be generating units. Fig. 12 shows an example of a branch mosaic, with size indicating the limit of a transformer or line, and the fraction of the white rectangle that is filled up with yellow indicates how close the circuit is to its limit. Green lines and transformers are out of service; red ones have exceeded their limit; and orange ones are close. This diagram also shows that size could be a dynamic quality, with overloaded lines (red) shown much larger for emphasis.

VI. Comparison to other Visualization Techniques and Assessment

These mosaic-type displays are a supplement to other forms of power system data visualization, adding another possible tool that can be used to provide additional insight into these complex datasets. Table I compares a selection of existing data visualization techniques to mosaic packing. Most power system data is discrete, so this method fits in with many other techniques in using discrete objects to show the data. Key tradeoffs in visualization techniques are computational speed, geographic accuracy, and whether the data can easily be seen. Mosaic packing, as shown in Table I, provides a high level of geographic accuracy—though not exact—while avoiding the challenge of overlapping and crowding that happens in many other techniques. Its moderate speed allows it to scale to reasonably large datasets in online applications.

To investigate the potential impacts of the mosaic packing visualization technique, a user study was performed. This study follows a similar format to [25]. The primary initial target user group of this method is engineers, so the study group was a set of 10 respondents, each with a B.S.E.E or equivalent and at least 5 years of experience in power engineering. The average respondent had 15.9 years of experience. The respondents were asked to (1) rate a list of visualization characteristics by importance for wide-area situational awareness, (2) answer specific questions from three examples of mosaic tile visualizations and discuss the helpfulness of the diagram in answering the question, and (3) provide comments on the advantages and disadvantages of these mosaic diagrams, how easy they are to understand, and their potential usefulness in power engineering.

The results of this user study were very positive, with many respondents commenting that the diagrams have much potential to be useful for understanding wide-area geographic system data. For part (1), the most important characteristics were that the diagrams be generated automatically (rated 4.3 out of 5) and that the computation speed be fast (rated 4.1 out of 5). Respondents indicated that some correspondence with geographic coordinates was important (rated 3.6 out of 5) but that some distortion was acceptable (exact geographic locations were rated 2.6 out of 5).

For part (2), respondents answered the questions with 91% accuracy, and they indicated that these diagrams were most helpful for answering questions that related to system-wide information, such as “In what area of the system are most generators at their capacity?” Numerical questions such as “How many voltage violations are there?” are better suited to tabular displays than graphical ones, and can lead to some errors through not counting correctly. In purely geographic displays, however, these questions would definitely be answered incorrectly because some of the violations would be obscured by other data.

The respondents overwhelmingly agreed in part (3) that the diagrams were intuitive and easy to understand, noting the importance of having an informative legend and color key. They pointed out main advantages of the diagrams in identifying patterns in the data, getting a quick general view of the system, with usefulness both for engineers and potentially for operators as well. Several respondents mentioned these displays may be well suited to presenting patterns to people not familiar with a particular grid.

VII. Conclusions and Future Work

The mosaic-type displays presented in this paper are well suited to wide-area, multi-dimensional situational awareness, helping the user to comprehend the big picture, highlight key

TABLE I
Comparison to Other Visualization Techniques

Visualization Method	Speed	Overlaps	Crowded Areas	Continuous /Discrete	Size Emphasis	Geographic Accuracy
Mosaic Packing	Moderate	No	No	Discrete	Yes	High
Line color/flow arrows [5]	Fast	Yes	Yes	Discrete	Yes	Exact
Tabular data	Fast	No	No	Discrete	No	None
Space-filling grid [17]	Fast	No	No	Discrete	Yes	Low
Color contour/heatmap [7]	Moderate	No	Yes	Continuous	No	High
Geographic data view/glyps [21]	Fast	Yes	Yes	Discrete	Yes	Exact
Hand-placed data views	Slow	No	No	Discrete	Yes	High
Graph drawing layouts [12], [24]	Slow	No	Yes	Discrete	Yes	None

outliers, and focus on what is most important. This paper formulates the mosaic packing problem and the horizontal-packing algorithm used to solve it. This benefits of the horizontal-packing approach used here is that it is flexible to allow different types and sizes of tile inputs, positioning them near the actual geographic location without overlapping. The approach is fast enough for interactive applications for large systems, and has a variety of possible customizations.

With the speed and quality of mosaics that the horizontal packing approach can produce, it is anticipated that many new possibilities for applications will be developed in the future. One future research topic is in enhancing the continuity between successive packing approaches when tile size changes significantly. The nature of the problem requires that if tile sizes change, the placement must be affected. Overlaps are not allowed. If a small tile becomes large, nearby tiles must move out of the way to make room for it. Fortunately, the geographic constraints mean that for lower-constrained problems tiles will always show up near their previous spot. To combat potential discontinuity between time points, one approach in applications is to use the size variable for data that changes infrequently, such as generator size, line limits, or peak load. Then the color, shading, text, and blinking can indicate more quickly-changing variables. Distortion will be more significant in highly-constrained cases. An idea for future work is to add a feature to the algorithm so that, rather than optimizing to the geographic location, it optimizes to maintain consistency with the last layout generated.

REFERENCES

[1] M. Friendly, "A brief history of the mosaic display," *Journal of Computational and Graphical Statistics*, vol. 11, no. 1, pp. 89-107, Mar. 2002.

[2] E. Raisz, "The rectangular statistical cartogram," *Geographical Review*, vol. 24, no. 2, pp. 292-296, Apr. 1934.

[3] P. C. Wong, et al., "A space-filling visualization technique for multivariate small-world graphs." *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 5, pp. 797-809, Jun. 2011.

[4] *National Academies of Sciences, Engineering, and Medicine, Enhancing the Resilience of the Nation's Electricity System.* Washington, DC, USA: Nat. Acad. Press, 2017, pp. 4–34.

[5] J. D. Weber and T. J. Overbye, "Voltage contours for power system visualization," *IEEE Transactions on Power Systems*, vol. 15, no. 1, pp. 404-409, Feb. 2000.

[6] T. J. Overbye and J. D. Weber, "Visualization of large-scale power systems," *Proc. EPSOM 1998*, pp. 152-161, Zurich, Switzerland, Sept. 1998.

[7] T. J. Overbye et al., "A virtual environment for interactive visualization of power system economic and security information," *Proc. IEEE PES 1999 Summer Meeting*, Edmonton, Canada, pp. 682-687, Jul. 1999.

[8] M. Mahadev and R. D. Christie, "Minimizing user interaction in energy management systems: Task adaptive visualization," *IEEE Trans. Power Syst.*, vol. 11, no. 3, pp. 1607–1612, Aug. 1996.

[9] P. Cuffe and A. Keane, "Visualizing the electrical structure of power systems," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1810–1821, Sep. 2017.

[10] Y. S. Ong, H. B. Gooi, and C. K. Chan, "Algorithms for automatic generation of one-line diagrams," *EE Proc., Gener., Transm. Distrib.*, vol. 147, no. 5, pp. 292–298, Sep. 2000.

[11] A. de Assis Mota and L. T. M. Mota, "Drawing meshed one-line diagrams of electric power systems using a modified controlled spring embedder algorithm enhanced with geospatial data." *J. Comput. Sci.*, vol. 7, no. 2, pp. 234–241, 2011.

[12] A. B. Birchfield and T. J. Overbye, "Techniques for drawing geographic one-line diagrams: Substation spacing and line routing," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 7269-7276, Nov. 2018.

[13] J. A. Hartigan and B. Kleiner, "Mosaics for contingency tables," *Proc. of the 13th Symp. on the Interface*, pp. 269-273, New York, NY, 1981.

[14] B. Schneiderman, "Tree visualization with tree-maps: A 2-D space filing approach," *ACM Transactions on Graphics*, vol. 11, pp. 92-99, Feb. 1992.

[15] M. Van Kreveld and B. Speckmann, "On rectangular cartograms," *Computational Geometry*, vol. 37, no. 3, pp. 175-187, Aug. 2007.

[16] R. G. Cano, et al., "Mosaic drawings and cartograms," *Computer Graphics Forum*, vol. 34, no. 3, pp. 361-370, Jun. 2015.

[17] T. J. Overbye, J. Wert, A. Birchfield, and J. D. Weber, "Wide-area electric grid visualization using pseudo-geographic mosaic displays," *Proc. 2019 North American Power Symposium (NAPS)*, Wichita, KS, Sept. 2019.

[18] *Transmission System Planned Performance for Geomagnetic Disturbance Events*, NERC Std. TPL-007-1, Jun. 2014.

[19] A. B. Birchfield, T. Xu, K. M. Gegner, K. S. Shetye, and T. J. Overbye, "Grid structural characteristics as validation criteria for synthetic networks," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3258-3265, July 2017.

[20] Power Flow Cases, 2016. [Online]. Available: http://electricgrids.engr.tamu.edu.

[21] T. J. Overbye, E. M. Rantanen, S. Judd, "Electric power control center visualizations using geographic data views," Bulk Power System Dynamics and Control – VII. Revitalizing Operational Reliability – 2007 IREP Symposium, Charleston, SC, Aug. 2007, pp. 1-8.

[22] D. P. Bertsekas, "The auction algorithm: A distributed relaxation method for the assignment problem." *Annals of operations research*, vol. 14, no. 1, pp. 105-123, Dec. 1988.

[23] E. L. Lawler and D. E. Wood, "Branch-and-bound methods: A survey." *Operations Research*, vol. 14, no. 4, pp. 699-719, Aug. 1966.

[24] P. Cuffe, A. Keane, "Visualizing the Electrical Structure of Power Systems," IEEE Systems Journal, vol. 11, pp. 1810-1821, September 2017.

[25] H. Mitsui and R. D. Christi, "Visualizing voltage profiles for large scale power systems," *IEEE Computer Applications in Power*, vol. 10, no. 3, pp. 32-37, July 1997.

Adam B. Birchfield (S'13, M'19) received the B.E.E. degree from Auburn University, Auburn, AL, USA, in 2014, the M.S. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 2016, and the Ph.D. degree in electrical engineering at Texas A&M University (TAMU), College Station, TX, USA., in 2018. He is now a research consultant at Texas A&M University through Birchfield Consulting LLC.

Thomas J. Overbye (S'87–M'92–SM'96–F'05) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Wisconsin-Madison, Madison, WI, USA, in 1983, 1988, and 1991, respectively. He is currently a TEES Eminent Professor in electrical and computer engineering with Texas A&M University, College Station, TX, USA.