Displaying large amounts of spatial data in GIS

H. Keserica[1], S. Sučić[1] and Ž. Mihajlović[2]
[1]Control and Protection Systems
Končar – Power Plant and Electric Traction Engineering Inc.
Fallerovo šetalište 22, 10 000 Zagreb, Croatia
Tel: (+385) 1 365 5030, E-mail: hrvoje.keserica@koncar-ket.hr
[2]Department of Electronics, Microelectronics, Computer and Intelligent Systems,
Faculty of Electrical Engineering and Computing, University of Zagreb,
Unska 3, 10 000 Zagreb, Croatia
Tel: (+385) 1 612 99 44, E-mail: zeljka.mihajlovic@fer.hr

Abstract – Data visualization based on geographic maps introduces issues when lots of geo-spatial data have to be shown. These issues are caused by usual non-uniform density of the data. Map symbols that have to be seen can overlap in some parts of the map and become unclear; on the other hand, there can be poorly filled, unused space on the map. This paper proposes a method for continuous area deformation dependent on the amount of information (attributes) that need to be displayed. Two instances of algorithm have been realized, one that shows qualitative results and the other that generates results faster. This shows practical realization of different map visualization using cartograms where different layouts can be realized with adjustable application parameters that can manipulate with results area, shape and topology.

I. INTRODUCTION

Many existing applications collect and reference data using their geographic locations. For example, most of the devices that are monitored by Supervisory Control And Data Acquisition (SCADA) systems spread over a large geographical area. Number of such devices can be very large, and information about them usually contains, along with the other attributes, geographic location on which they are placed. Large clusters of data that contain several thousand of records are almost impossible to understand quickly just by looking at the raw data. Visualization is essential to identify their geographic location and introduce better overview. Therefore, SCADA systems are in need to show large amounts of data using GIS system. Model of widespread computer usage is typical in terms that they contain many devices which usually have GPS included and their visualization can become an issue.

Although geographic visualization has been studied for over a decade, it still brings new challenges. Displaying large amount of information points using standard maps is problematic. Areas with densely filled data points are not clear enough because there is high probability that they will cause over-plotting while less dense area unnecessarily exploit a lot of space that may come of a better use.

Data point in GIS system can represent any element (e.g. substation as an element of power network) and it can be found in a state of alarm, but it may not be visible when it is in the area of dense distribution due to data point overlapping. In this case, it is of extreme importance to ensure that the map view is shown in the way which will enable visibility of the points that are in the state of alarm. To ensure better visibility of geo-spatial location of these data points it is often necessary to change the view criteria using panning or zooming functions. There are contradictory requirements. On the one hand it is necessary to have an insight into the overall data; on the other hand, due to high density in certain areas, it is necessary to have enlarged map view. If the SCADA system considers only parts of the map, complete set of data points will never be displayed at the same time, and it is possible that alarming situation occurs on the field which, at that point, is not visible. For this reason it is important that the entire area supervised by SCADA system is visible at any given time so that all data points are shown, but it is also important that, for better visibility, they are not too dense so that operators that oversee the system can respond to its changes on time.

Recent research in the field of information visualization has addressed types of transformation functions that generate spatially-transformed maps with recognizable shapes. These types of spatial-transformation are called global shape functions. In particular, cartogram-based map distortion has been studied [2].

This article describes a method for distorting regions of the map in order to provide space for attributes display depending on their density in certain areas. Those areas can contain important geo-spatial information such as alarm tags, so the goal is to achieve their equal density to minimize the occurrence of overlapping points or their labels and, by that, make the display of larger amount of data possible.

II. METHODS FOR GENERATING CARTOGRAMS

Cartograms show the usual map with special display technique and they are typically used in geographically related statistical information such as population, demographics and epidemiological data. There are different types of cartograms and their goal is, on one side, to maintain recognition of the geographical shape and, on the other side, to enter the information to the display in a transparent way. There are cartograms in which it is important to preserve the neighborhoods, or the topological relationships, with other cartograms it is important to preserve shape, while with Dorling cartograms full emphasis is based on the display of the attributes using circles, for example. Size and color of circles then dominate the view.

Cartographers and geographers have used cartograms long before computers were available to make displays.

Because cartograms are very demanding to be made by hand, they are subject of errors and they are not suitable for achieving changes. Therefore, program implementation for making cartograms comes to importance.

Cartograms that are interesting in this work are those for which the map deforms in such way that respectable region areas become proportional to given parameters (the density of information within the regions) but with condition that the map remains recognizable.

In general case, even the simple variants of cartogram generation cannot be achieved because of contradictory requirements. On one hand it is necessary to increase some regions, reduce the others and, at the same time, preserve their shapes. In the end, in order to achieve the desired result, it is necessary to use heuristic algorithms because feasible variants of the problem are NP-complete.

III. CARTOGRAMS BASED ON AREA

The basic idea of a cartogram is to deform a map by resizing its regions by some geographically associated parameter.

An area based cartogram can be seen as a generalization of an ordinary map. In this interpretation, an arbitrary parameter vector gives the intended sizes of the cartogram's regions. In such way, the ordinary map is simply a cartogram with sizes proportional to land area. In the scenario related to this paper, the arbitrary parameter vector is given by the number of data points contained within respectable region. In that way cartograms can show region areas in relation with some additional parameters, such as for example, the population of particular region or, in this case, the number of data points. The sizes of the regions are then displayed in proportion to the given parameter instead of the actual surface. In Fig. **1**., a cartogram based on population is presented which shows that a cartogram can give much different impression of overall trends as compared with the original map.

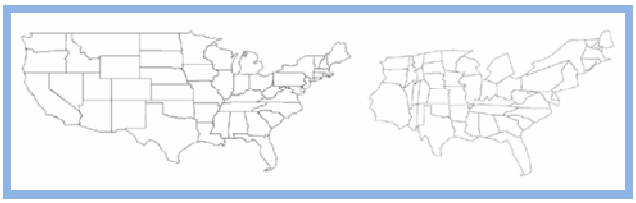

Fig. 1. Population-based cartogram. The original map is shown on the left and cartogram on the right side.

For a cartogram to be effective, an observer must be able to quickly understand the presented information and relate it to the original geographical model. Recognition, in turn, depends on preserving basic properties, such as shape, orientation, and contiguity. This, however, is difficult to achieve in the general case because, sometimes, it is impossible even just to maintain the original map's topology. Most of the currently available algorithms are very time-consuming because the generation of contiguous cartograms with simultaneous optimization of these objectives is extremely complex.

The ultimate goal is to achieve region areas proportional to the number of data points they are containing. The reason for this is to expand the areas with dense data points so that each individual data point becomes visible without changing the scale of the map. SCADA systems require ability that the map is entirely shown at any time so that changes on the map are visible on the screen regardless of the region in which they occurred.

IV. CARTOGRAM MAKING ALGORITHM

The main objective of the algorithm for making cartograms is their fast generation with acceptable quality. Due to the fact that the input maps often have much more vertices than it is necessary to calculate good cartograms, the first step is to discard redundant vertices using reduction algorithm. Next step is to determine the scanlines which are the base of the iterative algorithm execution. They can be automatically determined (horizontal and vertical lines) or may be interactively entered as arbitrarily positioned line segments of any length. After that, the CartoDraw algorithm follows. In every step of the algorithm the area error function is used to express measured deviation from wanted result and that way it controls changes that are made to the original polygonal network.

In order for cartogram to be recognizable, it is important to maintain local and global shape. For this reason, the target function for rendering cartograms includes the preservation for both, local and global shapes. To measure the degree of shape preservation, the function was used for expressing shape similarity. It is based on a difference between functions that describe geometric curvature of polygons.

Using the method for cartogram calculation described in this paper, higher performance speed can be achieved than some other techniques provide. The method is done using the efficient iterative scanline algorithm for repositioning the vertices while preserving the local and global shape. Scanlines can be generated automatically or they can be entered interactively in order to achieve better looking results at the end of the process.

V. REDUCTION ALGORITHM

Edge reduction algorithms are used as starting method for solving the cartogram problem in various algorithms for cartogram calculation [1]. Since preservation of the global shape is very important in making recognizable cartograms, used reduction algorithm takes it into account by simplifying the global and inner polygons in the same way, but with a different selection of input parameters.

A key observation is that the importance of polygon's vertices can vary greatly. Vertices on angles near 180° and those with short edges make almost no noticeable difference in the shape of a polygon, while others with sharp angles or long edges have a significant effect. The basic idea of the global polygon reduction algorithm is to rate the importance of each vertex according to these criteria. Then, iteratively, the least important vertices are removed. To maintain the topology, only vertices that do not belong to multiple polygons are removed. To formalize the global reduction algorithm, we first define the notion of a vertex's importance as:

$$I(v) = Sig(\alpha^v) \cdot |e_1^v| \cdot |e_2^v|, \quad (1)$$

where e_1^v and e_2^v are the two edges of vertex v, absolute value indicates that this is the length of edge, and $Sig(\alpha^v)$ is a function denoting the significance of the angle α^v at vertex v. The significance function is important because different angles have a specific impact on the shape of the polygons. Sharp angles and angles close to 90° are more important than obtuse angles and the significance function therefore assigns higher values to sharp angles and lower values for obtuse angles. In this algorithm for the significance function is used following expression:

$$Sig(\alpha) = \sum_{\mu \in \{0°,90°,270°,360°\}} \frac{\exp(\alpha - \mu)^2}{2\sigma^2}. \quad (2)$$

This function has peaks for α=0°, 90°, 270°, 360° and is close to zero for α=180°. The function is defined for α=[0°,360°], and σ is a constant chosen to be 0.2·π. Fig. **2**. shows a plot for this function.

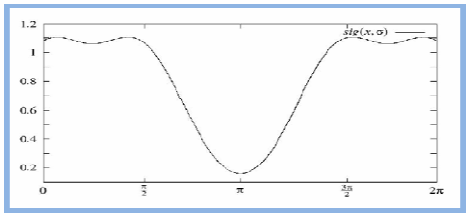

Fig. 2. Angle significance function.

To formalize the global reduction algorithm it is necessary to define the global polygon as a subset of the vertices of $\mathcal{P}$. For each polygon p_j ,j = *1*,...,k, the portion of the global polygon can be defined as:

$$gp_j = \{v \in p_j : |edges(v)| > |polygons(v)|\}. \quad (3)$$

The global polygon is then a union of (3) where j = *1*,...,k. The algorithm for the reduction of the global polygon considers only those vertices that do not belong to multiple polygons and they are only removed if the induced area difference is smaller than a given constant *MaxAreaDiff* in order from the less important vertex to those with higher value of significance function (1).

For reduction of inner polygon, the algorithm for reduction of global polygon can be re-used but with different groups of the vertices that are considered for removal. Vertices belonging to that group are all inner vertices that are not common to more than two polygons.

Fig. **3**. shows the map before and after reduction of global and inner vertices.

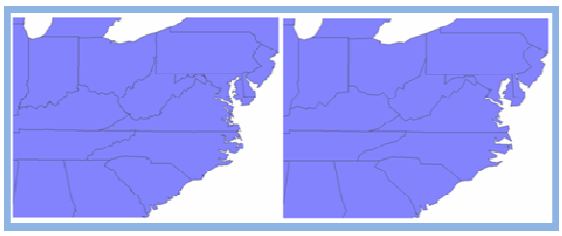

Fig. 3. Reduction algorithm example.

VI. THE CARTODRAW ALGORITHM

The main idea of the CartoDraw algorithm is to incrementally reposition the vertices along a series of scanlines. Fig. **6**. shows an example of a scanline. A scanline is a line segment of arbitrary length and position. Each scanline defines a scan section, orthogonal to the scanline. All points within a scan section are repositioned in a single step. For each section on a scanline, a target scaling factor for each of its polygons is determined according to their area error factors. Vertices are then repositioned according to the polygon scaling factors and distances to the scanline. The repositioning may be parallel or orthogonal to the scanlines. If the shape error introduced by applying a scanline exceeds some threshold, its candidate vertex repositioning is discarded.

Scanlines should be applied to parts of the map where the area error is large and there is still potential for improvement. A simple approach to scanline generation is to use horizontal and vertical line segments. Significantly better results can be obtained by a manual scanline placement, guided by the shape of the input polygons and the local potential for improvement. Incremental repositioning of vertices per scanline application is intentionally small compared to the expected change in area. This means the same scanline may need to be applied many times to make large adjustments in the area.

Before describing the main CartoDraw algorithm there must be defined its three main components: the area error function, the shape similarity function, and the scanline algorithm.

A. Area Error Function

The objective of cartogram generation is to obtain a set of polygons where the area of the polygons corresponds to values given in a data vector X. In each step of the algorithm, the area error function is needed to determine the reduction of the area error achieved by applying a given scanline. The relative area error E_{rel}^j of a polygon p_j can be computed as:

$$E_{rel}^j = \frac{\left| A_{wanted}^j - A_{actual}^j \right|}{A_{wanted}^j + A_{actual}^j}. \quad (4)$$

Therefore, the area error for the set of polygons $\mathcal{P}$ is defined as:

$$E_{rel}^{\mathrm{P}} = \sum_{j=1}^{k} \left(E_{rel}^j \cdot \frac{A_{wanted}^j}{\sum_{j=1}^{k} A_{wanted}^j} \right). \quad (5)$$

B. Shape Error Function

Along with the reduction of area errors, the process of generating cartogram also tries to preserve original shapes. To assess shape preservation, we need a shape similarity function that compares the new shape of a polygon with its original shape. Defining a useful shape similarity function is a difficult problem since the similarity measure should be translation-invariant, scale-invariant, and rotation-invariant. Function of the curvature of polygons and

normalize arc length of the polygons to 2π is used to gain invariance against translation, rotation, and scaling.

In the following, it is assumption that the polygons are transformed into a normalized parameterized polygon contour function $p:[0,2\pi]\rightarrow \mathbb{R}^2$. Then, we can define the curvature C of the polygons as:

$$C:\left(\mathbb{R}\rightarrow\mathbb{R}^2\right)\rightarrow\left(\mathbb{R}\rightarrow\mathbb{R}^2\right). \quad (6)$$

The shape similarity of two polygons p and p dash can then be defined as:

$$d_s\left(S(p),S\left(\bar{p}\right)\right)=d_{Euclid}\left(C(p),C\left(\bar{p}\right)\right). \quad (7)$$

In the following, a more detailed description of the curvature transformation C is given.

C. Determining the Curvature of a Polygon

In general, the curvature of a polygon defined as a parameterized function is mathematically undefined because the second derivative is not continuous. This problem can be avoided by approximating the polygon by replacing each vertex with very small circular arcs, as shown in Fig. **4**. So get a new geometric object of which the first derivative is continuous. The curvature of this structure is defined in sections, concatenating these sections gets the curvature as a square wave function.

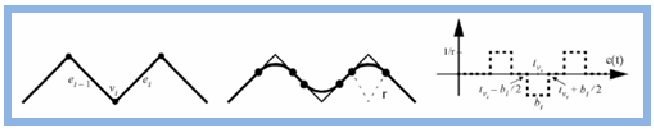

Fig. 4. Curvature approximation of the polygon. Left is original polygon, in the middle approximated polygon, and right is curvature of approximated polygon [1].

To describe the curvature transformation in more detail, focus is on two consecutive edges e_{i-1} and e_i. These edges coincide in vertex v_i with an angle α_i. For the polygon containing v_i, the curvature function c_i can easily be computed, describing the differential geometric curvature of the approximated polygon because the curvature of a circle segment with radius r is a constant function $1/r$ and the curvature of a straight line is a constant zero function. The arc length of the circle segment can be calculated by substituting vertex v_i with $b_i=|\alpha_i|\cdot r$. For $c_i(t)$ therefore we obtain:

$$c_i(t)=\begin{cases}\frac{1}{r}, in\ the\ case\ of\left(t_{v_i}-\frac{b_i}{2}>t>t_{v_i}+\frac{b_i}{2}\right)\\ 0, in\ other\ cases\end{cases}. \quad (8)$$

The curvature of an arbitrary polygon p is $c(t)=\sum c_k(t)$. Fig. **4**. (right) shows the graph of the curvature function $c(t)$ for the approximation of the polygon section in Fig. 4. (left). Fig. **5**. shows the curvature functions for two polygons which are identical under translation, rotation and scale invariance.

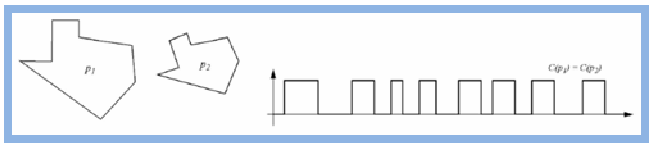

Fig. 5. Curvature transformation of two identical polygons. Right is original and the left one is transformed.

The approximation of the original polygon and, in particular, the choice of r, influences the curvature function. If we reduce the radius r of the circle segment, $1/r$ will be increased while b_i will be decreased. This causes $c(t)$ to become narrower and the amplitude of square waves to become higher, while the approximation of the polygon becomes difficult to handle numerically. An adequate value for r which has proven useful for this application is $\pi/50$ for polygons with a normalized length of 2π. Chosen value for r is good as long as r is smaller than half of the length of the shortest edge since, otherwise, individual square wave functions may overlap.

D. Scanline Algorithm

The key to the CartoDraw algorithm is the scanline heuristic, which incrementally repositions vertices along scanlines. A scanline sl is a line segment of arbitrary position and length and is partitioned into n portions of length $|sl|/n$. The scanline section $sp_i,(i=0..n)$ define $n+1$ sections of the polygon mesh, which are orthogonal to the scanline (Fig. **6** left).

Fig. 6. Scanline algorithm example. Left is scanline section, and right scanline section with limited range [1].

In one step of the scanline algorithm, all vertices $v\in V_i$ within a certain distance ($\xi=\frac{|sl|}{2n}$) of l_i are considered for incremental repositioning (Fig. **6** left). Let SP_i be the set of polygons which have at least one vertex in scanline section i, (i=0,...,n). Then, the scaling factor SF_i is determined according to the area error of all polygons p in section i:

$$SF_i=const\cdot\sum_{r\in S_i}\left(\frac{x_r-A(p_r)}{x_r+A(p_r)}\cdot\frac{x_r}{\sum_{l\in S_i}^{n}x_r}\right). \quad (9)$$

Still remains to determine the direction $o(v)$ of a vertex v and apply the scaling factor SF_i to reposition the vertex. The repositioning can be done either in the direction of the scanline or in the direction of the section line l_i. The scanline sections always span the full range orthogonal to the scanline of the polygon net. If we want to restrict the changes to be local in both directions, we can optionally limit the polygons considered to those close to the scanline (Fig. **6**. right).

E. Main CartoDraw Algorithm

Having defined the components of the *CartoDraw* algorithm, its main procedure now can be described. The algorithm assumes as input a set of polygons $\mathcal{P}$, a scaling vector of the desired statistical parameter X, and a set of scanlines SL, which can be determined automatically or manually. Output is the modified set of polygons $\mathcal{P}$ which describes the cartogram. The algorithm works as follows.

For each scanline, the algorithm applies the scanline transformation and checks the results. If the area difference E_{rel} introduced by the scanline transformation is below a certain threshold ε_A and the shape distortion (7) is below a certain threshold ε_S, then the changes are retained and, otherwise, discarded. Then, the algorithm proceeds with the next scanline until all scanlines are applied in the same way. At this point, the algorithm checks whether, in applying all scanlines, an improvement of the area error has been obtained. If this is the case, the algorithm applies all scanlines again and repeats the entire procedure until no further improvement is reached (area improvement below ε). Since the area error improvement must be positive and above the threshold ε in each iteration, the area error is monotonously decreasing and termination of the algorithm is guaranteed. Applying an individual scanline allows the algorithm to potentially increase the area error to allow escaping local optima.

VII. RESULTS

After the application launches, it is possible to include the layer in which scanlines are displayed. On the example below, scanlines are set automatically (horizontal and vertical lines) and scanline section points that define the scope which local algorithm effect are shown (Fig. **7**.).

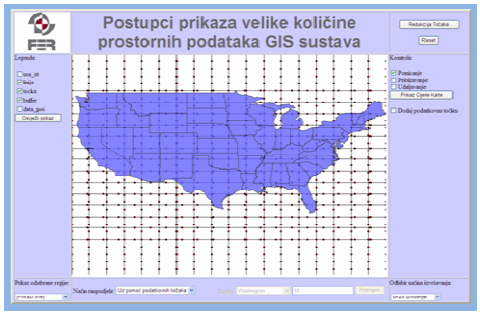

Fig. 7. Scanlines *sl* and scanline section points *sp* used in the application.

Before launching the CartoDraw algorithm it is necessary to reduce the number of vertices in order to achieve the algorithm execution faster. The result of the reduction algorithm is shown in Fig. **8**. The screenshot shows the border with the jagged line (Mississippi) in order to display effectiveness of the algorithm at its best.

Also, before main CartoDraw algorithm execution the input data needs to be prepared. Input data determines how the distribution of areas in the cartogram will look at the end. Data can be prepared through entering the information points, or manually - by entering values for each region that will represent the number of information points for each region. After application starts, some data points are set initially. Each region contains ten information points on which the algorithm (with no additional changes) seeks the uniform area distribution for each region.

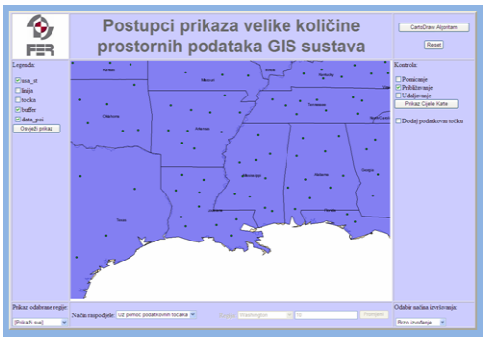

Fig. 8. Look at the application after the reduction algorithm.

A. Measurement Results

The tests have been done using the fast and the qualitative algorithm. The difference between these two modes is the method that considers whether the individual point of *sp* belongs to polygon on which the algorithm is operating. Since this test is often performed, it becomes a big time-consumer and the performance of the algorithm decreases. Therefore, for faster performance, the problem resolves using approximation so that affiliation of the sp point to the polygon is determined using polygon's bounding box and not the polygon itself and thus gain a great time saving.

In the example, with 10 data points in each region, there are additional 75 data points inserted in California. List of parameters are shown in Table I.

TABLE I
PAREMETERS USED IN APPLICATION

Mode	*MaxAreaDiff*	r	ε	ε_S	ε_A
Fast	0.013	0.0001	0.009	2.57	0.0007
Qualitative	0.013	0.0001	0.009	2.7095	0.0007

MaxAreaDiff from Table I is the parameter used in reduction algorithm and it is described in chapter V. The value r is the radius of circle segments listed in the chapter VI.C.. Parameters ε, ε_S and ε_A are the values of CartoDraw algorithm. Table II shows the time consumption for test runs and the number of iterations for fast and qualitative algorithm. Testing is done on the computer: Intel Core Duo CPU T2350@1.86GHz 1.86GHz, 2 GB of RAM.

TABLE II
NUMBER OF ALGORITHM ITERATIONS
AND EXECUTION TIMES.

Mode	Iterations	Execution run	Average iteration time
Fast	207	04:49,8 min	00:01,4 min
Qualitative	213	21:18,0 min	00:06,0 min

B. Algorithm Execution Results

Following figures show the cartogram rendering results.

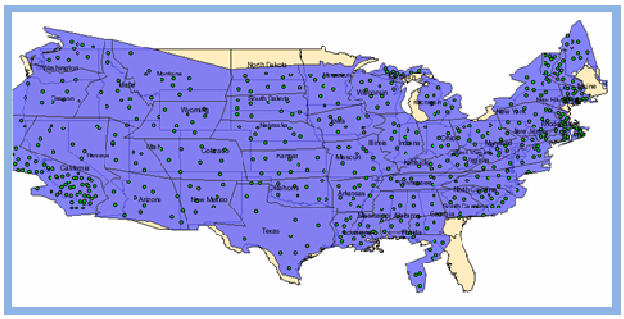

Fig. 9. Fast algorithm – California contains 85 data points and all other states contain 10 data points.

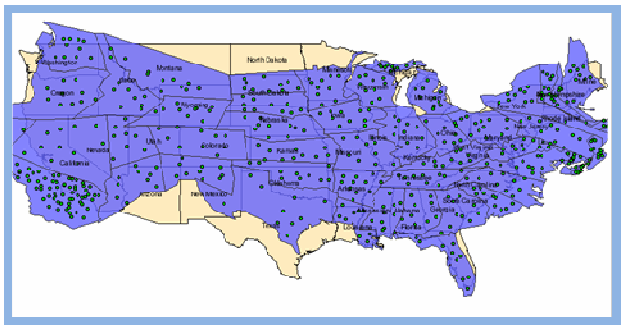

Fig. 10. Qualitative algorithm – California contains 85 data points and all other states contain 10 data points.

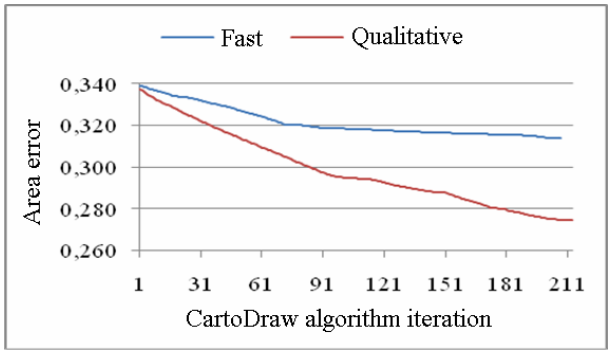

Fig. 11. Area error graph of CartoDraw algorithm measured after every iteration.

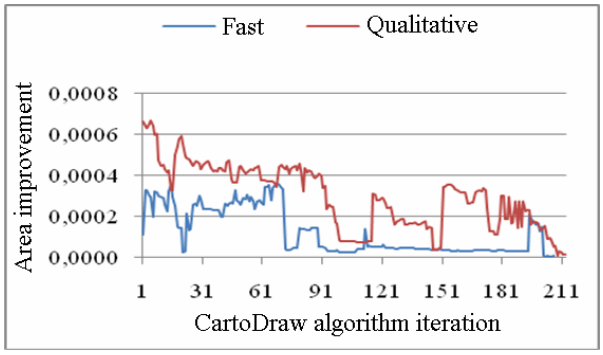

Fig. 12. Improvement graph of CartoDraw algorithm measured after every iteration.

In the examples presented on the left, methods show notecably different results. This differences are conformed with area error graphs (Fig. 11.) that show fairly better result in the case of qualitative algorithm.

Using the fast algorithm, some unnatural changes were made to the original map in particular regions (Fig. 9.). The main reason for these changes lies in the fact that fast algorithm uses bounding boxes for affiliation of *sp* points so skewed regions can give more distorted results. These distortions are mostly noticable in the area of Florida and Maine making those regions look wierd and recjective to the human eye. On the other hand, the bounding box approach came up with the result almost five times faster than it is the case when using precise determination of *sp* point affiliations to the specific regions. Quality algorithm shows significantly better results (Fig. 10.) on the side of area error function. The improved result is mostly seen in the west part of the map where dense data points of California are better spread and the mid part of the map with scattered data points is significantly narrowed compared to the fast algorithm result.

IV. CONCLUSION

This paper presents the method for viewing large amounts of spatial data in GIS system with the help of cartogram drawings. The fast CartoDraw algorithm was used and implemented in two ways: fast and qualitative algorithm. Qualitative algorithm shows better results in all tested examples considering the original region shapes preservation which is highly important for better map overview.

Speed of cartogram calculation for usage in SCADA systems is not so important because it is enough that the algorithm for the calculation is carried out once after entering a new data points. Deformed map after the calculation will always be available for usage. But because the main purpose of the CartoDraw algorithm is usage in real time data acquiring and in that purpose this work is testing the ability of generating cartograms using faster approach. The results clearly indicate that the data obtained in real time should not make large differences in the visualization within ten minutes so that images can be consistent with the system from which the data arrives.

It can be noted that the deformation of the maps does not remove the problem of mutual point overlapping, but only reduces the need for it. One of the possible upgrades of the application would be usage of local data point placement by PixelMaps method [3].

REFERENCES

[1] D. A. Keim, S. C. North and C. Panse, *CartoDraw: A Fast Algorithm for Generating Contiguous Cartograms*, IEEE Transactions on visualization and computer graphics, 2004.

[2] C. Panse, M. Sips, D. A. Keim and S. C. North, *Visualization of Geo-spatial Point Sets via Global Shape Transformation and Local Pixel Placement*, IEEE Transactions on visualization and computer graphics, 2006.

[3] D. A. Keim, C. Panse, M. Sips and S. C. North, *Pixel Based Visual Mining of Geo-Spatial Data*, 2003.