ViSizer: A Perception-Based Framework for Visualization Resizing

Yingcai Wu, Member, IEEE, Xiaotong Liu, Shixia Liu, Member, IEEE, and Kwan-Liu Ma, Senior Member, IEEE

Abstract—Visualization resizing is useful for many applications such as collaborative business intelligence where users may use display devices with different sizes and aspect ratios. General resizing techniques (e.g., uniform scaling) and image resizing techniques suffer from several drawbacks, as they do not consider the content of visualizations. In this work, we introduce ViSizer, a perception-based framework for automatically resizing a visualization to fit any display. It can largely preserve salient features while resizing the visualization to an arbitrary aspect ratio with minimum visual clutter introduced. To achieve this, we formulate an energy function based on a perception model (feature congestion), which aims to determine the optimal deformation for every local region. We subsequently transform the resizing problem into a non-linear optimization problem by the energy function. An efficient optimization algorithm is introduced to iteratively solve the problem, allowing for automatic visualization resizing.

Index Terms—Resizing, Visualization Framework, Perception, Focus+Context, Nonlinear Least Squares Optimization

1 Introduction

Research on visualization resizing is becoming particularly important with the advance of collaborative visual analysis, in which users might use different display devices with varying sizes and aspect ratios. Although modern visualization methods can regenerate a new visualization if the display changes, the common regeneration methods do not always work. Some visualization methods such as tag clouds and force-directed graph algorithms usually regenerate a totally different layout to fit the new display, which is unacceptable in a collaborative application. Furthermore, the methods such as the force-directed algorithms need excessive time for regenerating a new layout. As a result, a generic resizing framework is needed for efficiently producing consistent visualizations, such that embedded useful patterns in the resized visualizations can still be revealed as effectively as the original one. Additionally, such a framework can relieve the burden of designing a visualization, as the developers do not need to consider the re-scaling problem any more.

There are several possible solutions to resizing a visualization. One simple approach is uniform scaling. Unfortunately, this would not work if the visualization is resized to a different aspect ratio. To tackle this problem, the visualization can simply be cropped to ensure that the uniform scaling can coincide with the new aspect ratio. However, this method may discard important or useful context information. Regions of interest can also be cluttered and overwhelmed by other less important information, especially when they are drawn in smaller regions and not visually enhanced. This problem could be solved to some extent by clutter reduction techniques [1], which can reduce visual clutter and provide more space for important visual elements. The adjusted visualization can then be resized to a smaller display. However, these methods usually discard information, require additional overhead, and may not work well for resizing the visualization to a different aspect ratio. Furthermore, they depend on a user’s ability and intuition, or the tendency of the visualization to determine the amount of information to be discarded [2].

In this work, a perception-based framework, ViSizer, is designed for effectively resizing a visualization to any display. It is built upon an image warping approach [3]. The majority of image resizing methods such as seam carving [4] keep important regions unchanged, leading to failure when the region sizes are larger than the target image sizes. In contrast, the optimized scale-and-stretch method [3] can address this problem by scaling important regions uniformly and deforming homogeneous context. ViSizer employs a similar deformation scheme, but it is much more flexible. It can be viewed as a multi-focus+context visualization technique by allowing users to explicitly specify the expected scaling factors for the regions of interest in the target visualization (see Fig. 6). In this figure, background grids are provided as visual cues for users to understand the geometric distortion and relate a resized result to its original visualization. The framework can also produce a distortion-free result...
by automatically determining the scaling factors and distorting only the white space in the visualization (see Fig. 2(g) and Fig. 4(h)).

Importantly, ViSizer employs a new perception-based significance measure designed for visualization. The measure can estimate the visual clutter magnitude in a visualization and guide the resizing process to avoid compressing visually cluttered items. A new energy function is defined based on the measure to transform the resizing problem to a nonlinear squares optimization problem. The optimization problem can then be solved by an efficient iterative algorithm.

The major contributions of this work are as follows:

- Study a new problem of how to effectively resize a visualization to any display.
- Transform the visualization resizing problem to an optimization problem with a novel perception-based energy function.
- Design and develop a generic framework for automatic visualization resizing.

## 2 Related Work

Many image resizing methods have been introduced, and they may be generally classified as discrete or continuous methods [5]. Discrete methods, i.e., seam carving [4], resize an image by judiciously inserting or removing left-to-right or top-to-bottom seams, which connect paths of pixels. Continuous methods (or image warping) [3], [6] associate an image with a grid and resize the image through deforming the grid non-homogeneously. However, these techniques are not optimal for visualization resizing for the following reasons. First, visualizations have special layouts with interactive visual elements rather than static pixels. Acceptable deformation in images may be viewed as a serious distortion or damage to the layouts. For example, non-homogeneous deformation of words in a word cloud may decrease their readability and hinder task performance. Second, in contrast to image resizing, visualization resizing would be more constrained by visual clutter - the state in which excess and/or disorganized items degrade visual task performance. This performance degradation is due to the difficulty in recognizing or searching for an item interfering with other surrounding items, especially when the item spacing is small [7].

Visual Clutter is an important factor for designing an effective visualization and user interfaces. Baldassi et al. [8] showed that visual clutter misleads users to problematic judgments and to more confidence in erroneous decisions. Researchers traditionally measured visual clutter based on information density or the number of elements [9], [10]. Some researchers argued that this traditional method was not a good measure of clutter, because the number of elements can be ill-defined [2]. Rosenholtz et al. introduced two quantitative measures, feature congestion and subband entropy, to estimate the level of clutter on a display [11]. Our framework uses the feature congestion method based on local feature variance [2] because it is effective for predicting clutter and is much more efficient than other quantitative methods.

Data Abstraction can be used to adapt visualizations to devices with small displays (e.g., mobile devices). Various data abstraction techniques [1], such as sampling [12], filtering [13], clustering [14], point/line displacement [15], and dimensional reordering [16], have been proposed to reduce information density for alleviating the visual clutter problem. They can simplify visualizations created on large-screen devices. As a result, the visualizations may be adapted to devices with small screens. However, they usually require additional overhead to redesign the visualization. They also inevitably discard information and are likely to fail when resizing to different aspect ratios. Moreover, what information to be discarded solely relies on either the user ability to navigate a view with less clutter or the heuristic rules embedded in a visualization [2].

Overview+Detail Visualization [17] shows both the overview and the detail simultaneously, either by displaying them in separate views or overlaying the overview on the detail in the same view. Voto et al. [18] and Karstens et al [19] used Overview+Detail techniques in navigating web pages and maps on mobile devices, respectively. Some researchers argued that the technique is less effective on mobile devices because the limited screen space assigned to the overview prevents the user relating the overview to the detailed view easily [20]. Bürging et al. [21] indicated that showing only a larger, detailed view on small displays is more effective. The performance of experienced users could be hindered by the Overview+Detail technique on mobile devices [21].

Focus+Context Visualization, which enhances data in focus and compresses the context, is a popular solution to visualizing data on mobile devices [19]. Fisheye [22] and bifocal display [23] are among the most widely used focus+context techniques. These techniques are useful in visualization, but they may lead to target acquisition problems and impaired spatial comprehension [24]. Zanella et al. [25] suggested using grids and shading to tackle these problems. Our method can be regarded as a focus+context technique, but we novelty apply the technique in resizing visualizations. It allows users to specify the expected scaling factors of regions of interest. Compared with the existing focus+context techniques, it is naturally supported in the framework. Furthermore, the important regions are uniformly scaled, and the distortions are distributed across the whole visualization rather than only the local regions as handled in the existing techniques. Guided by a perception-based significance map, ViSizer can also minimize the chance of task performance degradation caused by visual clutter.
### 3 Design Challenges and Benefits

There are three typical important scenarios where our visualization resizing framework is useful.

- Several users are collaboratively analyzing a huge amount of data simultaneously by using visualization techniques. They use computing devices with different display sizes and aspect ratios.
- A user may have different computing devices with different displays to work at different places.
- A visualization designer and a viewer may have displays with different sizes and aspect ratios.

These scenarios present several challenges for designing an effective resizing approach. First, the resized visualization must be consistent with the original one. The inconsistency may convey incorrect information, mislead the discussion in the collaboration, or even draw a wrong conclusion. This poses a challenge to some visualization techniques such as tag clouds and graph layout methods that usually create totally different layouts for different displays. Second, the technique must be efficient. In a collaboration scenario, a new user may join in the collaboration at any time and the visualization under discussion should be resized to fit his device display in real time, such that he can start to collaborate with others immediately. Therefore, the algorithms that require excessive time to regenerate layouts is inappropriate for the application. Third, the method should naturally support multi-focus+context visualization as well as necessary visual cues for users to comprehend the geometric distortion. Usually, a user using a mobile device often does not have an appropriate display to show the original visualization and thus may want to show a deformed version. Finally, it should minimize the chance of introducing additional clutter when resizing a visualization to a smaller display.

To address these challenges, we design a flexible and efficient resizing framework with a seamless integration of multi-focus+context visualization. To improve the accuracy when deformation is needed, as suggested by other researchers [25], [26] we use the background grids to support the user’s comprehension of geometric distortion. Additionally, the framework enables different levels of distortion in the resized results to meet different user requirements.

- Distortion-free: all visual items except the empty space in the visualization are uniformly scaled. The empty space will be greatly compressed while the relative positions of the visual items are preserved.
- Controlled distortion (multi-focus+context visualization): the visual items will be deformed based on the expected scaling factors specified by users. The primary benefit of this generic framework is that it can meet different resizing requirements. Users can determine whether distortion is allowed or not. By measuring the visual clutter in the original visualization, the framework can avoid compressing the cluttered regions in the resized result. Additionally, it can also relieve the burden of visualization designers for handling the re-scaling problem.

### 4 Overview of the Framework

Fig. 1 shows an overview of ViSizer, which includes a preprocessing component and an optimization component. The preprocessing component first partitions a visualization with a grid. A significance map, a combination of a DOI map and a visual clutter map, are then created to encode the significance values of every quad in the grid. Next, a significance-aware grid is created by deforming the initial grid based on the significance map to reduce linearization artifacts and to approximate the nonlinear deformation better.

The optimization component starts by deriving an appropriate initial guess to ensure better results and quick convergence. The resizing problem is transformed into a nonlinear least squares optimization problem through an energy function based on the significance map, quad deformation, and edge bending. The optimization problem can be iteratively solved to find a good solution. The scaling factor for every grid...
quad will be adjusted (or smoothed) at each iteration to minimize potential distortion. The iteration repeats until a certain convergence condition is reached, i.e., all vertex movements are very small in the current iteration. Finally, the optimization generates a deformed grid and we utilize the grid to adjust the visualization accordingly by interpolation.

5 PRE-PROCESSING

In pre-processing, the framework first associates an input visualization with a grid used for warping the visualization. It then creates a significance map for encoding the significance of different regions in the visualization. Finally, the grid is adjusted to be significance-aware, which means that more important regions are covered by more quads.

5.1 Initial Grid Placement

ViSizer employs a grid-guided resizing optimization scheme. It partitions a visualization with a grid (see Fig. 8(a)), then iteratively adjusts the grid in an judicious manner under some constraints (size and boundary) to achieve an optimal deformation of the grid. Finally, the visualization can be resized according to the deformed grid by forward mapping. We choose to deform the grid rather than the visualization in the optimization because of the efficiency and flexibility of the grid-guided method. The efficiency is achieved through the iterative optimization scheme widely used in image warping and resizing, while the flexibility is achieved by the energy function associated with the grid-guided optimization method. Moreover, the grid can provide sufficient visual cues for a viewer to comprehend the deformation.

5.2 Significance Measure

The significance measure is an image-based measure and is a core part of the resizing framework. It is used to create a significance map for guiding the significance-aware grid adjustment and to determine the vertex movements in the optimization process. The significance of each local region can be estimated by the measure based on the DOI and the magnitude of clutter of the visual items in the region. Only quads that are both locally important and cluttered should be protected against distortion.

5.2.1 Degree of Interest

Degree of interest (DOI) was first introduced by Furnas [27] to indicate that visual items in visualization have different levels of importance. Furnas defined a DOI function for an item \( x \) given the user’s focus item \( y \) as: 
\[
\text{DOI}(x|y) = \alpha \cdot \text{API}(x) + \beta \cdot D(x,y),
\]
where \( \text{API}(x) \) represents the general importance of \( x \), \( D(x,y) \) computes the distance from \( x \) to \( y \), and \( \alpha \) and \( \beta \) denote the weights for \( \text{API}(x) \) and \( D(x,y) \), respectively. Clearly, the DOI function is application-specific and different applications may have different definitions. With an appropriate DOI function, important regions can be differentiated from less important regions. For simplicity and clarification, we let \( \alpha = 1 \) and \( \beta = 0 \) in our experiments. In scatterplots and line charts, we view all visual items equally important and so we have \( \text{API}(x) = 1 \). In word clouds and graphs, \( \text{API}(x) \) of a visual item \( x \) is assigned based on the size of the item (word or graph node). The approach creates a DOI map, an image with the same resolution of the visualization, based on the DOI function.

5.2.2 Clutter Estimation

The DOI function is used to preserve regions of interest in visualization resizing. However, relying only on the DOI is insufficient for determining the shrinking or stretching operations of a visualization. This is because some regions may become crowded with excess, unorganized visual items when a user repeatedly resizes the visualization. As a consequence, the visualization would be cluttered and the performance of visual tasks, such as visual searching, could be degraded [7]. Fig. 2(f) shows an example in which visual clutter becomes severe when the words in the green ellipses get closer and closer. To tackle this problem, a quantitative measure of visual clutter estimation is introduced. In this scenario, the regions with high magnitudes of clutter should not be shrunk to avoid being even more cluttered.

Our framework employs an efficient method called Feature Congestion [2] to estimate the clutter magnitude in every local region. This method can produce an image called clutter map with the same resolution of the visualization for revealing the clutter magnitude at every pixel. It uses the level of feature congestion to indicate the degree of clutter in an image. The congestion level can be measured by a statistical saliency model based on the observation that unusual items are usually salient [2]. Whether or not an item is unusual depends on how different the feature vector of the item is from the local distribution of other feature vectors. A feature vector is composed of the color, the luminance-contrast, and the orientation of the item. Thus, the statistical saliency for a feature vector \( X \) can be evaluated by the Mahalanobis distance [28] as

\[
\Delta = \sqrt{(X - \mu)^T S^{-1}(X - \mu)}
\]

where \( \mu \) and \( S \) denote the mean matrix and covariance matrix of the local distribution of the feature vectors, respectively. This saliency model actually uses a set of covariance ellipsoids, determined by the covariance matrix \( S \), in the feature space to represent the local feature distribution (see Fig. 3). With the model, the difficulty of adding a new important item to a local area can be simply measured by the size of the local covariance ellipsoid represented by \( S \). Given a type
of feature space with a limited volume, such as color gamut, the larger size of the local ellipsoid indicates less space for adding a new salient item. This is because the item has to be outside of the ellipsoid to ensure that it appears to be unusual to the existing items inside the ellipsoid. In other words, there is little feature space excluding the large local ellipsoid for choosing an appropriate feature vector for a new salient item. Thus, the feature space is likely to be congested with a large covariance local ellipsoid.

We follow the procedure in [11] based on the saliency model for revealing the magnitude of visual clutter across an image. Interested readers can refer to [11] for more details about the procedure. Fig. 2(a) shows a word cloud and its visual clutter map (Fig. 2(b)) estimated by the method.

5.3 Significance-Aware Grid

The initial uniform grid can simplify the implementation and support faster performance. However, the uniform grid places an equivalent number of quads in every local region in spite of the significance of the regions, leading to linearization artifacts in the optimization process (see Fig. 4(b), where the words in the right side are shrunk too much). To reduce the artifacts and better approximate the optimal deformation, we adjust the initial grid to ensure that significant regions are covered by more quads than less important regions. The resulting grid is called a significance-aware grid. Two types of significance-aware grids are derived, such that the proposed framework is applicable to most visualizations. The first type is to directly deform the initial grid to attract the quads of less important regions to those of significant regions. It is adapted from the method in [3] by optimizing the following energy function:

$$\sum_{(i,j) \in E} \frac{1}{1 + \sqrt{w_{ij} \cdot (v_i - v_j)^2}}$$

where $E$ is a set of edges in the grid, $v_i$ and $v_j$ are two positions of nodes $i$ and $j$, and $w_{ij}$ is the average weight of the quads that share the edge $(i,j)$. In an optimal scenario in which the energy is the minimum, the nodes in the interior of the significant...
Fig. 4. Results obtained by resizing Fig. 2(a) horizontally. (a) Uniformly resized result. (b) Bad result produced by the initial grid where the words on the right are overly compressed. (c) and (d) Significance-aware grids of Fig. 2(a) created using the energy functions defined in [3] and in Equation (2), respectively. (e) Adaptive grid of Fig. 2(a). (f)-(h): Results produced by the grids shown in (c), (d), and (e), respectively. (h) is the best result because the words are nicely packed with their relative sizes and positions preserved.

regions become closer, thus attracting the surrounding nodes to the regions (see Fig. 4 (c) and (f)). Compared with the energy function in [3], our energy function uses \( \sqrt{1 + w_{ij}} \) rather than \( 1 + w_{ij} \) to prevent a quad attracting too many neighboring nodes (see Fig. 4 (d) and (g)). We also tested with other choices such as \( w_{ij} \) and \( w_{ij}^2 \), and found that \( \sqrt{1 + w_{ij}} \) produced better results in general. This optimization problem is a nonlinear least squares optimization problem and can be solved iteratively to approximate the optimal node positions of the grid. This approach is simple for implementation and does not require changing the grid topology. In addition, because this is done only once in the preprocessing step, the resizing performance remains the same.

However, this method can still introduce linearization artifacts (the important words inside the green ellipses in Fig. 4(g) are too small). To tackle this problem, an adaptive grid is used as a second type of significance-aware grid. The basic idea is simply to use a quad tree to partition the visualization and thus ensure that significant regions are covered by more quads. Fig. 4(h) shows the result by the adaptive grid (Fig. 4(e)). The implementation becomes more complicated compared with the first grid type, but the results look better.

6 Optimized Resizing

This section describes the resizing algorithm of ViSizer. It is adapted from a continuous image warping method [3] because of its two major advantages. First, it can optimally distribute the deformation caused by image resizing over the context regions of the image, regardless of the resizing direction (horizontal, vertical, or both). This feature is particularly useful for visualization resizing, because context regions can be optimally deformed over the entire visualization rather than in local regions. Second, it uses a grid for resizing an image, thus it has the potential to preserve shapes as well as relative spatial positions between objects within visualization. The grid also facilitates users’ comprehension of the geometric deformation.

Our method is different from the image warping method in three aspects. First, a completely different significance map is derived to guide the optimization process. Second, the energy function is tailored for visualizations. Third, besides automatically finding an optimal scaling factor for a focus region, our method also allows users to specify an expected one for the region. It can uniformly change the focus region(s) to the desired size(s) while resizing whole visualization. Therefore, it can be viewed as a combination of multi-focus-context visualization and visualization resizing.

Given a visualization, we place a grid on it and denote the grid by \( G = \{V, E, F\} \), where \( V \), \( E \), and \( F \) represent the nodes, edges, and quads of \( G \), respectively. Let \( n_v, n_e, \) and \( n_f \) be the numbers of nodes, edges, and quads. We have \( V = [v_1^T, \ldots, v_{n_v}^T]^T \), \( E = [e_1^T, \ldots, e_{n_e}^T]^T \), and \( F = [f_1^T, \ldots, f_{n_f}^T]^T \) represent the node position of node \( i \). The optimization is to transform \( G \) judiciously for creating a new grid \( G' = \{V', E', F'\} \), so as to ensure the uniform scaling of salient quads and the distortion distribution to other quads without increasing visual clutter. A new visualization can then be created by space interpolation according to the new grid \( G' \).

6.1 Energy Function

An energy function is defined to formulate the visualization resizing problem as an optimization problem. It has two terms: a quad deformation term for penalizing the non-uniform quad distortion and an edge bending term for minimizing the edge bending.

6.1.1 Quad Deformation Energy

During the optimization, we strive to uniformly scale the salient regions and prevent the features from being arbitrarily distorted. Thus, the quad deformation energy is formulated for a quad to penalize the situation in which a quad is distorted rather than uniformly scaled. The energy is measured based on the deviation of each local deformation from a uniform scale.

Given a quad \( f_k \in F \), it can be represented by its four edges \( e_{k0}, e_{k1}, e_{k2}, \) and \( e_{k3} \), i.e., \( f_k = E_{f_k} = [e_{k0}^T, e_{k1}^T, e_{k2}^T, e_{k3}^T] \). The \( u^{th} \) edge of \( f_k \) can be computed...
as \( e_{k} = v_{i} - v_{j} \) with its two nodes \( v_{i} \) and \( v_{j} \). When we uniformly deform \( f_{k} \), we have \( f'_{k} = s_{k} f_{k} \) where \( s_{k} \) is a scalar value indicating the scaling factor for \( f_{k} \). The distortion energy of \( f_{k} \) is then defined as
\[
\omega_{f_{k}} || f_{k} - s_{k} f_{k} ||^{2} = \sum_{e_{k} \in E} || \sqrt{\omega_{f_{k}}} (v'_{i} - v'_{j}) - \sqrt{\omega_{f_{k}}} s_{k} (v_{i} - v_{j}) ||^{2}
\]
where \( \omega_{f_{k}} \) represents the average significance value inside the quad. The energy for all quads \( F \) is
\[
\sum_{f_{k} \in F} \sum_{\{i,j\} \in E_{f_{k}}} || \sqrt{\omega_{f_{k}}} (v'_{i} - v'_{j}) - \sqrt{\omega_{f_{k}}} s_{k} (v_{i} - v_{j}) ||^{2}
\]
To simplify the following discussion, we represent the energy function by matrices as
\[
|| W_{F} F' - W_{F} S F ||^{2}
\]
where \( F = [f_{0}, f_{1}, \ldots, f_{n_{v} - 1}]^{T} \), and \( W_{F} \) and \( S \) are matrices whose element at the \( u^{th} \) row and the \( v^{th} \) column is defined as
\[
S_{uv} = \begin{cases} 
    s_{k} & \text{if } u = v \\
    0 & \text{otherwise}
\end{cases}
W_{F,uv} = \begin{cases} 
    \sqrt{\omega_{f_{k}}} & \text{if } u = v = k \\
    0 & \text{otherwise}
\end{cases}
\]
The quad can be represented as \( f'_{k} = q_{k} V \) where \( q_{k} \) is a \( 4 \times n_{v} \) matrix and its element at the \( u^{th} \) row and the \( v^{th} \) column is defined as
\[
q_{k,uv} = \begin{cases} 
    1 & \text{if } v = i \\
    -1 & \text{if } v = j \\
    0 & \text{otherwise}
\end{cases}
\]
where \( i \) and \( j \) are the node indices of the \( u^{th} \) edge of \( f_{k} \). We let \( Q = [q'_{0}^{T}, q'_{1}^{T}, \ldots, q'_{n_{v} - 1}^{T}]^{T} \), the quad set \( F \) can be derived by \( F = Q V \). Therefore, the total quad deformation energy is
\[
|| W_{F} Q V' - W_{F} S Q V ||^{2}
\]

### 6.1.2 Edge Bending Energy

Minimizing only the quad deformation to is insufficient. Fig. 5 (b) shows an example where the grid edges bend too much and some regions of interest (in the green ellipses) are distorted too much. As a result, grid edges should be prevented from bending, such that the prominent objects under multiple connected quads retain their shapes. To achieve this, we allow a grid edge to scale its length but penalize the change of the edge orientation.

Given an edge \( e_{k} \in E \), it can be computed as \( e_{k} = v_{i} - v_{j} \) with its two nodes \( v_{i} \) and \( v_{j} \). When we uniformly deform \( e_{k} \), we have \( e'_{k} = l_{k} e_{k} \) where \( l_{k} \) is a scalar value indicating the scaling factor for \( e_{k} \). The edge bending energy of \( e_{k} \) is then formulated as
\[
\omega_{e_{k}} || e'_{k} - l_{k} e_{k} ||^{2} = || \sqrt{\omega_{e_{k}}} e'_{k} - \sqrt{\omega_{e_{k}}} l_{k} e_{k} ||^{2}
\]
where \( \omega_{e_{k}} \) is the average significance value of the two quads sharing the edge. The energy for all edges \( E \) is
\[
\sum_{e_{k} = (i,j) \in E} || \sqrt{\omega_{e_{k}}} (v'_{i} - v'_{j}) - \sqrt{\omega_{e_{k}}} l_{k} (v_{i} - v_{j}) ||^{2} = 0
\]
To simplify the following discussion, we represent the energy function by matrices as
\[
|| W_{E} E' - W_{E} LE ||^{2}
\]
where \( E = [e'_{0}^{T}, e'_{1}^{T}, \ldots, e'_{n_{e} - 1}^{T}]^{T} \), and \( L \) and \( W_{E} \) are matrices whose element at the \( u^{th} \) row and the \( v^{th} \) column is defined as
\[
L_{uv} = \begin{cases} 
    l_{k} & \text{if } u = v \\
    0 & \text{otherwise}
\end{cases}
W_{E,uv} = \begin{cases} 
    \sqrt{\omega_{e_{k}}} & \text{if } u = v = k \\
    0 & \text{otherwise}
\end{cases}
\]
Let $e_k = v_i - v_j = h_k V$, where $h_k$ is a $1 \times n_v$ vector and its $v^{th}$ element $h_{k,v}$ can be defined as

$$h_{k,v} = \begin{cases} 1 & \text{if } v = i \\ -1 & \text{if } v = j \\ 0 & \text{otherwise} \end{cases}$$

Let $H = [h_0^T, h_1^T, \ldots, h_{n_v-1}^T]^T$, we can obtain $E = HV$. Therefore, the total edge bending energy is

$$||W_EHV' - W_ELHV||^2 = \sum_{k,v} (L_{k,v})^2 H_{k,v}^2$$

6.2 Non-Linear Least Squares Optimization

Total energy is derived by summing the quad deformation energy and the edge bending energy. The optimal nodes positions $V'$ of the grid can be approximated by minimizing the total energy as follows:

$$\argmin_{V',S,L} ||W_F(V') - SQV')||^2 + ||W_E(HV' - LHV)||^2$$

which is a non-linear least squares optimization problem. It can be viewed as an over-determined system $AV' = b(V)$, where $A = [Q^T W_F^T, H^T W_F^T]^T$ and $b(V) = [V^T Q^T S^T W_F^T, V^T H^T L^T W_F^T]^T$. Hence, we minimize

$$\argmin_{V'} ||AV' - b(V)||^2$$

which is composed of $4n_f + n_v$ linear equations. The $x$ (or $y$) coordinates of the nodes on the vertical (or horizontal) grid boundaries are constrained to remain constant to ensure a rectangular shape. We directly substitute these constraints into Equ. (5) before the optimization starts. The optimal solution of the non-linear optimization problem can be approximated by iteratively updating the node positions.

6.3 Linear Solution

The resizing technique has a solid math foundation and it is easy to implement by repeatedly solving a linear system derived from Eq. (5).

$$AV' - b(V) = 0$$

Through this linear system, we can approximate the solution of Eq. (5). Let $V^i$, $S^i$, and $L^i$ denote the nodes, quad transformations, and edge transformations at the $i^{th}$ iteration, respectively. In our experiments, the initial guess $V^0$ is obtained by homogeneously resizing the original visualization to the target display. The scaling transformations $S^i$ are computed based on the ratio between the deformed and original quad sizes. For instance, we estimate the scaling factor $s_{k}^i$ for quad $k$ at the $i^{th}$ iteration according to its current deformed size $\lambda_k^i$ and original size $\lambda_0^i$, i.e., $s_{k}^i = \sqrt{\lambda_k^i/\lambda_0^i}$. We compute $L^i$ similarly based on the ratio between the deformed and original edge lengths.

Next, we obtain $b(V)$ by $S^i$, $L^i$, and $V^i$. We then multiply Eq. (6) by $A^T$ and solve the system $(A^T A)V' = A^T b(V)$ by LU decomposition to derive $V^{i+1}$. This process is repeated until all the node movements are smaller than 0.5 or the newly obtained total energy is bigger than that of the previous iteration. A resulting grid for a particular target display is obtained after the optimization process. The visualization can be reconstructed by interpolating the visual elements according to the new grid.

6.4 Quad Transformation Smoothing

During the optimization process, the scaling transformations of the neighboring prominent quads need to be similar because the salient objects usually span several quads. Fig. 7(a) shows a resized visualization without quad transformation smoothing. The salient nodes marked by the green ellipses are clearly distorted. Thus, the scaling factors in each local region must be smoothed out to prevent the distortion of an important object caused by the different scaling factors of the surrounding regions. We formulate the smoothing problem as an optimization problem.

$$\sum_{k \in F} \sum_{q \in N(k)} w_q (s_{k}^i - s_q^i)^2 + \sum_{k \in F} w_k (s_{k}^i - s_k) = 0$$

where $N(k)$ denotes the quads surrounding the quad $k$, $w_q$ is the average significance of all nodes in the grid, and $w_k$ indicates the significance of the quad $k$. Compared to the smoothing function defined in [3], our function uses $w_k$ rather than $0.5(w_q + w_f)$ to weight $(s_{k}^i - s_q^i)^2$. Fig. 7 shows the different resizing results using the original function (Fig. 7(b)) and ours (Fig. 7(c)). Generally, our tailored function can produce better results with less distortion to the salient objects in the green ellipses. The smooth scaling $s_{k}^i$ can be estimated by minimizing the function. This process is repeatedly performed after we have obtained $s_k$ at every iteration of the optimization of Eq. (6).

6.5 Multi-Focus+Context Visualization

The scaling factors $S$ in $b(V)$ in Eq.(6) are automatically determined during the optimization. This allows for creating distortion-free results, as the all visual items are uniformly scaled. The distortion is largely absorbed by the empty space. Fig. 2(g) and Fig. 4(h) present two examples of distortion-free results in which the words are uniformly scaled based on the change of the display size. The spatial relations between the words are mostly preserved and thus the results are consistent with the original ones.

Many applications usually prefer the distortion-free results. However, some applications might not have enough empty space. Other techniques such as data abstraction might have more or less limitations as we discussed in Section 2. Our framework naturally supports multi-focus+context visualization during the resizing process to address this issue. It allows a user to specify a desired uniform scaling factor $\delta$ for any object in the resized visualization. Thus, the
quad scaling factors of the quads covering the objects are fixed to the constant value $\delta$ specified by the user during the optimization. This can produce a result similar to multi-focus+context visualization (see Fig. 6). Therefore, it can be regarded as a combination of significance-aware focus+context visualization and visualization resizing techniques. ViSizer transforms a visualization through a grid and thus can seamlessly provide the background grid to support the user’s comprehension of geometric distortion. It has been reported that the background grid can help improve the accuracy of visualization performance [25], [26].

7 Results and Discussion

In this section, we demonstrate the effectiveness of our framework and show how we can apply it to different visualizations. The techniques described in this work were implemented by Java and Prefuse. All results were generated on an Apple Macbook Pro with Intel Core i7 2.66GHz CPU and 4 GB Ram. Interactive time performance was achieved. Our system produced most of the results within one second (when using a significance-aware grid) or three to five seconds (when using an adaptive grid).

7.1 Experiments

7.1.1 Word Clouds

In the first experiment, we tested our technique for showing its usefulness for resizing word clouds to arbitrary display sizes. The resizing technique is im-

Fig. 6. Multi-focus+context visualization. (a) Result created by uniform resizing. (b)-(d) Results created with the specified scaling factors: $s_k = 1$, $s_k = 2$, and $s_k = 4$ for important graph nodes, respectively.

Fig. 7. Results created by resizing Fig. 5(a). (a) Result without any smoothing. (b) and (c) Results in which the scaling factors are equalized using the original and our adapted energy functions, respectively. (c) is the best result because the relative sizes of the nodes (indicated by the green ellipses in (a)) are preserved better than those of (a) and (b).
important for word cloud visualizations. It is usually difficult to resize a word cloud to fit a new display. Uniformly scaling a word cloud to a smaller display may create a word cloud of which words are too tiny to be easily recognized. Re-generating a new word cloud might be another option. However, the word cloud, e.g., re-created by [29] or [30], can be totally different from the original one. As they are based on either a random algorithm [29] or a force-directed algorithm [30], they fail to create appropriate word clouds with a specified aspect ratio. Additionally, context-preserving word clouds [30] use the relative positions of the words in the clouds to encode important semantic information in the original text. This requires that the relative spatial positions between words in the original word cloud should be preserved in the new word cloud.

The framework views a word as a visual item and uses its size as $API(x)$ to produce a DOI map. Each word is attached to a grid quad via an anchor point. After the grid is deformed, the anchor point positions are adjusted by interpolating the four nodes of the quad (see Section 6.3 for details). The size of each word is changed based on how the size of the associated grid quad changes.

We generated a context-preserving word cloud by a force-directed algorithm [30] using a real dataset with 13,828 news articles spanning one year (from 2008 to 2009) that were related to American International Group (AIG). In the word cloud, semantically-similar words get close to each other. As we wanted to shrink the word cloud for a small display, we filtered out tiny words in grey that is almost unrecognizable in the target display (see Fig. 2(a)). Figs. 2(b)-(d) show the visual clutter, DOI, and significance maps of the word cloud used for guiding the resizing optimization process. Fig. 2(a) (right) presents the color encoding scheme for these maps. We resized the word cloud vertically to reduce it to its half size.

Fig. 2(e) shows a uniformly resized result in which previously large words become unnoticeable, not to mention the smaller ones. Fig. 2(f) is a result created by our method guided only by the DOI map. Compared to Fig. 2(e), it distributed most of the distortion to the empty space, thus reserving more room for important keywords. However, the words inside the green ellipses were overly packed, making it challenging for users to recognize the words quickly. We can remedy this problem with the help of the visual clutter map (see Fig. 2(b)) that can inform the optimization process of the crowding degree in every local region, preventing the words from being excessively packed. Fig. 2(g) presents the result created by using the significance map (Fig. 2(d)). We can clearly observe that the overly crowding problem was fixed using the perception-based clutter measure.

We further tested different types of grids. Fig. 4(a) is a uniformly deformed result in which most of the words become tiny. We can use a uniform grid for resizing the word cloud in most cases. However, it cannot always produce a good result due to the linearization artifacts in the optimization process. Fig. 4(b) shows such an example where only the left part of the word cloud is deformed correctly. We then used three types of grids (Figs. 4(c)-(e)) from the original word cloud (Fig. 2(a)) to reduce the linearization artifacts. Fig. 4(f) shows a result based on the grid in Fig. 4(c) generated by the original energy function [3]. We can see that the grid is distorted too much. The words in the green ellipse in Fig. 4(f) are separated far away and their relative sizes change a lot.

Fig. 4(g) shows the result based on the grid in Fig. 4(d) created by our adapted energy function (Eq. (2)). The grid was modestly adjusted without too much distortion. However, some words originally neighboring to each other, e.g., in the green ellipse, are still far away. To solve the problem, we used an adaptive grid (Fig. 4(e)) to resize the word cloud. Fig. 4(h) presents the result in which the words are nicely packed and their relative sizes and positions are mostly preserved. We found that the adaptive grid generally worked better than other grids. However, it took more time (2.553 seconds in this experiment) than the significance-aware grids (within 1 second).

7.1.2 Graph Visualization

The second experiment was conducted to show the usefulness of our technique in graph visualization. Regenerating a new graph layout by, e.g., a force-directed algorithm, for a different display is usually time-consuming and the new layout could be totally different from the original one. Furthermore, most existing algorithms do not take into account the different display aspect ratios and could not make efficient use of the screen space. In resizing a graph, every graph node is regarded as a visual item and its size is used as $API(x)$ to produce a DOI map. Each node is attached to a grid quad via an anchor point. After the grid is deformed, the anchor points are adjusted by interpolating the four nodes of the quad. The size of the graph node is changed based on how the size of the associated grid quad changes.

We tested ViSizer by two real graph datasets. One is a social network dataset from Prefuse with 129 nodes and 161 edges, while the other contains major airline routes of Northwest Airlines in the United States with 235 nodes and 2,101 edges. We used both the size and color of a graph node to encode its degree. Fig. 5(a) shows a graph of the network data generated by a force-directed algorithm. Its significance map is shown in Fig. 5(a). Fig. 6(a) presents a uniformly resized result where nodes become too small to analyze. Figs. 6(b)-(d) demonstrate the resized results with an increasing $s_k$ ($s_k = 1$, $s_k = 2$, and $s_k = 4$) that was manually specified for the important graph nodes. The quads covering the nodes were uniformly
Fig. 8. Results created by resizing a graph visualization originally shown on a 27 inch display with 1920 × 1200 pixels (top-left) to a 3.5 inch display with 960 × 640 pixels in different orientations using uniform scaling, ViSizer with a significance-aware grid, and ViSizer with an adaptive grid. ViSizer makes more efficient use of the small displays than uniform scaling. The adaptive and the significance-aware grids both work well in most cases for maintaining original information. However, the significance-aware grid might have a chance to produce artifacts. The node indicated by the green arrow is not well preserved in the results of the significance-aware grid.

expanded by the specified $s_k$ to produce results similar to focus+context visualization. ViSizer distributed the distortion to the less important nodes and empty space across the whole visualization. These results can be regarded as focus+context visualizations.

Fig. 8 shows a typical use of ViSizer for resizing a graph originally shown on a 27 inch display with 1920 × 1200 pixels to a 3.5 inch display with 960 × 640 pixels in horizontal and vertical orientations. The used airline data contains spatial information for each graph node. Uniformly scaling the graph to the small display with a very different aspect ratio produced a squeezed visualization (see Fig. 8 (b)). In addition, it is difficult for a user to explore and interact with the graph in the much smaller display (see Fig. 8 (b) and (e)) because the graph nodes are barely discernible in such a display. Simply increasing the sizes of the nodes would cause the graph nodes to overlap one another. On the other hand, we can see that there is much white space in the left part of the graph. Therefore, we could compress the white space to make room for enlarging significant regions.

Fig. 8 (c) and (f) show the results created by ViSizer with the significance-aware grid, while Fig. 8 (d) and (g) present the results created by ViSizer with the adaptive grid. These results assigned more display space to important nodes (the larger the nodes, the more important they are) by compressing the white space in the graph while still preserving the overall graph structure. Comparing these results, we can observe that the two types of grids produced similar results. Nevertheless, the adaptive grid (Figs. 8(d) and (g)) works slightly better than the other (Figs. 8(c) and (f)) in preserving the original information. For instance, the node indicated by the green arrow is distorted in the results of the significance-aware grid. Since the spatial information and the sizes of the nodes are useful and important for analysis, it is therefore desirable to preserve the information.

7.1.3 Scatterplots

The third experiment was conducted to demonstrate the use of our technique for scatterplots based on the data from IN-SPiRE [31] showing the topic distribution of IEEE Vis, IEEE InfoVis and, IEEE VAST proceedings papers published from 2006 to 2008. Fig. 9 (left) shows an original scatterplot of the data on a 21.5 inch display with 1920 × 1080 pixels. Each paper
is represented as a point with a fixed size. This scatterplot is very sparse and has much white space. Fig. 9 (middle) presents two uniformly scaled scatterplots that are shown on a 3.5 inch display with 960 × 640 pixels and a 9.7 inch display with 1024 × 768 pixels, respectively. These results (particularly the smallest scatterplot) look squeezed. Additionally, the points in the scatterplots are too small and too close to be visually distinguished from one another. For example, the points inside the cluster indicated by the green arrow look quite cluttered in these results. It would be challenging for a user to interact with these cluttered points to facilitate visual analysis. One solution might be to decrease the point sizes to reveal the relations among the points. However, this would result in very tiny points in the small scatterplots that are hardly discernible. Applying simple magnifying lens such as Fisheye does not work either for distortion-sensitive applications. In addition, these techniques do not take different aspect ratios into account and could not apply simple magnifying lens such as Fisheye does not work either for distortion-sensitive applications. In addition, these techniques do not take different aspect ratios into account and could not.

Fig. 8(a) or Fig. 8(h) shows the results of ViSizer for showing the same smaller displays. Every point in the scatterplot is regarded as a visual item and has the same aspect ratios into account and could not. Every point in the scatterplot is regarded as a visual item and has the same aspect ratios into account and could not.

This section presents two collaborative visualization applications to demonstrate the usability and usefulness of our resizing framework.

7.2.1 Word Clouds
We assume that in an analysis task, two analysts, Joe and Jack, are collaborating using two displays remotely: one is a larger display for showing Fig. 2(a), and the other is a smaller one for showing Fig. 4(a) or Fig. 4(h). The analysts want to find out what has been reported for the AIG group quickly. Fig. 2(a) shows a context-preserving word cloud created by [30] where the semantically similar words are clustered in the word cloud. For instance, the keywords related to the financial news are placed in the top-left of the word cloud and those related to political leaders are placed in the top-right of the word cloud. The semantically clustered words greatly narrow the visual search space. The analysts can use the word cloud to examine the semantic relationships between keywords. Joe viewing Fig. 2(a) can easily find an interesting pattern by examining the words surrounding “aig” and “companies”, which clearly reveals that people were most interested in the aspects such as “business”, “assets”, “loan”, and “credit” when talking about the company. He notifies Jack who is using the smaller display about his finding.

Although notified by Joe, Jack might still find it difficult to identify and understanding Joe’s finding with Fig. 4(a) produced by uniformly resizing Fig. 2(a), since the related words become too tiny to be easily identified. Moreover, they are placed far away from “aig” and “companies”, which cannot convey the original semantic meaning embedded in 4(a) by the closeness of the words. In contrast, this pattern can still be detected and comprehended by Jack easily with Fig. 4(h) because the words such as “assets” and “business” are still closely surrounding “aig” and “companies”. Moreover, because of compressing the white space, the word cloud in Fig. 4(h) can show the words in larger sizes and preserve the relative positions between the words. Therefore, the analyst viewing Fig. 4(h) can quickly find and understand the similar pattern indicated by the first analyst in the collaboration using Fig. 2(a).

7.2.2 Graph Visualization
We assume a scenario in which a user (Joe) is visualizing the airline network using a 27 inch display (Fig. 8(a)). Joe finds six major airline hubs of Northwest Airlines in the east of the United States. He then wants to convey his finding to Jack and discuss how these hubs play a role in the whole flight network. Jack is using a smart phone with a 3.5 inch display at another city. If a uniformly resized result such as Fig. 8 (b) or (e) is shown in Jack’s small display, he would feel difficult to locate these hubs because the graph nodes are too small to be easily distinguished. Even if Jack finally finds the airline hubs, it would be challenging for him to interact with the graph nodes on such a small touch screen. Zooming into the graph loses the graph overview, while the overview+detail techniques hinder the performance of experienced users [21].

These problems can be readily resolved by ViSizer (see Fig. 8(d) and (g)) by compressing the empty space of the graph and assigning more space for the nodes. The relative positions between the graph nodes are well preserved, thus allowing for retaining the overall graph structure and providing necessary and correct context for further analysis. Our results suit
the small screen very well. ViSizer does not discard any information and maintains the clutter level with the perception-based significance map. Now, Jack can immediately identify the airline hubs and interact with them using his small touch screen. For example, he can easily use his index finger to touch and select the node indicated by the green arrow on the small screen. The airline routes connecting the selected hub will be highlighted, such that he can analyze how the associated hub influences other major hubs.

7.3 Discussion

Our system provides two types of grids for resizing a visualization: a significance-aware grid obtained by optimizing an energy function defined in Eq. (2) and an adaptive grid. The adaptive grid generally creates slightly better results than the significance-aware grid. A possible reason is that the adaptive grid has a regular shape which can better preserve the overall structure of a visualization. However, this advantage is gained at the cost of performance. In addition, the implementation of the resizing framework using an adaptive grid is more complicated and challenging. Based on our experiences and experimental results, we suggest that the adaptive grid should be used when users need more accurate results and do not care much on the performance. We recommend the significance-aware grid for most applications although it might slightly deform the items.

We consider the clutter magnitude in the optimization to avoid introducing additional visual clutter in the resized result based on a quantitative perception model of visual clutter. ViSizer mainly focuses on minimizing the chances for increasing visual clutter and it does not deal with the originally presented clutter. In other words, the framework cannot reduce visual clutter presented in a visualization. Therefore, it does not discard information in the resized results, which can be regarded as an advantage of this approach. Clutter reduction is another fundamental topic in information visualization and numerous techniques have been developed. We believe that our resizing framework can leverage the existing techniques in case of information overload. One possible solution to reduce clutter is to leverage a clutter reduction technique at each iterative step of the optimization. However, this may introduce additional overhead to the system. Thus we plan to thoroughly study this issue in our future work.

ViSizer can automatically adapt a visualization to any display. Its main advantage is that it can uniformly scale important items through diverting deformation to other regions without increasing clutter magnitudes with the help of the perception-based significance map. Moreover, by distributing deformation to only the white space, ViSizer can produce distortion-free results in which every local non-empty region is well preserved. ViSizer deforms the whole visualization through a grid and thus it can also largely retain the overall pattern of the original visualization, which has been demonstrated in our results (e.g., Fig. 2(g), Fig. 4(h), and Fig. 9).

The framework can also produce multi-focus+context results by highlighting important regions while compressing others. The important regions are uniformly scaled to the sizes specified by a user. This is an additional feature of our framework, which is particularly useful in a scenario in which there is no white space in the visualization to be compressed. Nevertheless, as other focus+context visualizations, these distorted results might lead to...
the target acquisition problem and impaired spatial comprehension. As suggested by other researchers [25], [26], ViSizer remedies this problem by showing the background grid for providing a necessary visual cue, which helps improve comprehension of the distortion and task accuracy.

The experimental results have demonstrated that ViSizer can tackle various visualization techniques including scatterplots, word clouds, and graphs. However, it may fail to resize some visualizations in which the positions of visual items are constrained to a regular shape. For instance, resizing radial graphs requires that the resized visual items should be always on a circular ring. Nevertheless, it is still feasible that our technique can be tailored for them by incorporating the layout constraints. This is worth further study.

8 CONCLUSIONS AND FUTURE WORK

In this work, we introduce a perception-based visualization resizing framework, ViSizer, for automatically resizing a visualization to any display size without introducing additional visual clutter. Prominent objects can either be uniformly scaled or fixed to a size specified by the user during the resizing process. The deformation introduced by the resizing operation is distributed to less important regions globally over the visualization. ViSizer is especially useful for applications that require preserving the stability and consistency of the visualization, such as context-preserving word clouds, in which resizing a visualization should maintain the spatial relations among visual items. In the future, we plan to test our framework on other visualizations such as radial graph and parallel coordinates. We also want to further evaluate the effectiveness of the resized results from the perspective of human perception and cognitive.

REFERENCES