# Fast content-aware resizing of multi-layer information visualization via adaptive triangulation

Chenhui Li [a], George Baciu [a,*], Yunzhe Wang [a], Xiujun Zhang [b]

[a] Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong
[b] College of Information and Engineering, Shenzhen University, Shenzhen, China

ABSTRACT

Visual graphics and image-based content have become the pervasive modes of interaction with the digital information flow. With the immense proliferation of display systems and devices, visual content representation has become increasingly challenging. Classical static image resizing algorithms are not directly suitable for the current dynamic information visualization of streaming data flows and processes because most of the visual content often consists of superimposed, multi-layered, multi-scale structure. In this paper, we propose a new adaptive method for content-aware resizing of visual information flow. Scaling is performed by deforming a hierarchical triangle mesh that matches the visual saliency map (VSM) of the streaming data. The VSM is generated automatically based on a series of predefined rules operating on a triangular mesh representation of visual features. We present a linear energy function to minimize distortions of the triangular deformations to perceptually preserve informative content. Through multiple experiments on real datasets, we show that the method has both high performance as well as high robustness in the presence of large differences in the visual aspect ratios between target displays.

## 1. Introduction

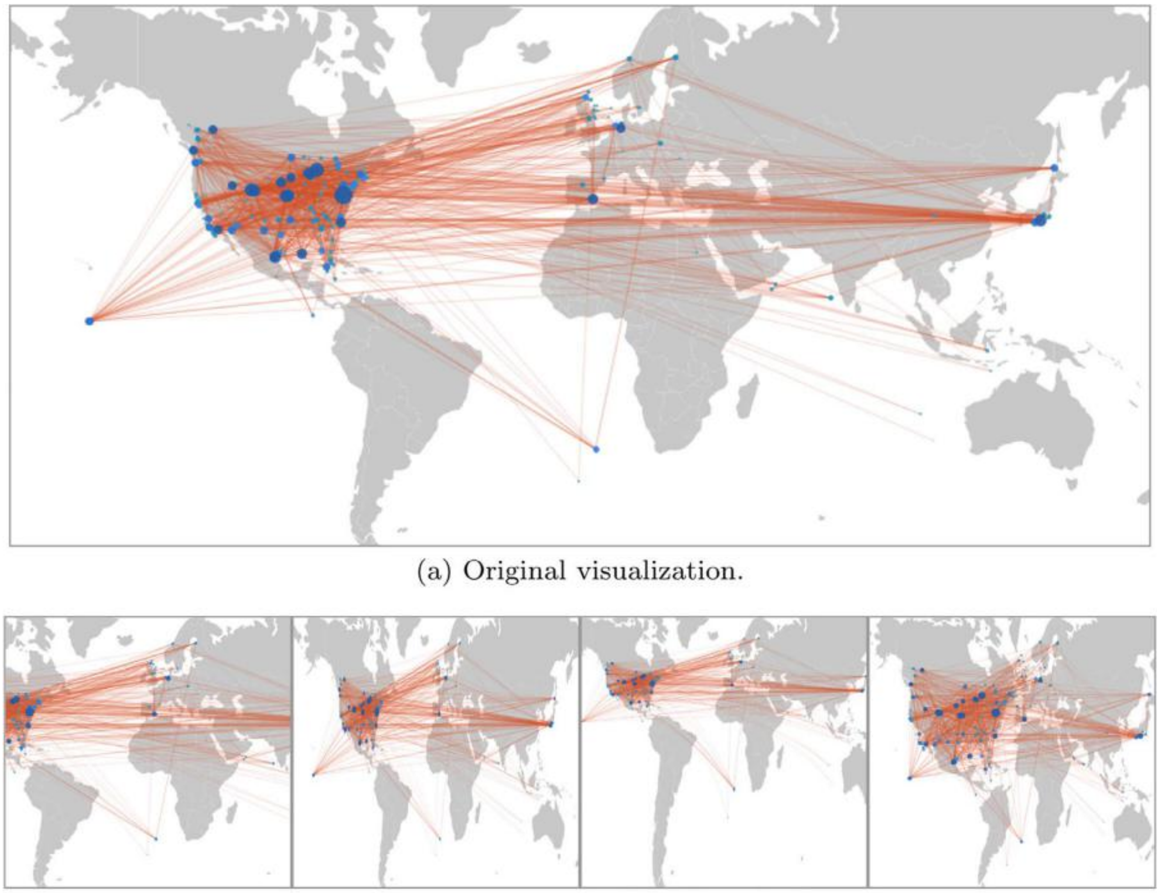
Content-aware resizing is an adaptive technique in image processing that filters out less important content and retains more important ones. This technique has also become a useful tool for information visualization because the diversity of displays for hardware is increasing. In addition, virtual displays of arbitrary size or aspect ratio require content-aware resizing techniques. Although artists, web designers, and programmers can design several available layouts for different scenarios, the task is time-consuming and costly.

The main objective of this work is to perceptively adjust the data output to any size or aspect ratio of a target display in the context of multi-layered information visualization applications. The basic resizing techniques such as cropping and linear scaling often result in information loss and visual distortions. Cropping (Fig. 1b) is the simplest operation for visualization resizing. Cropping can be used to adapt to different types of displays. However, cropping often removes important content. Linear scaling (Fig. 1c) is another approach. However, distortions often appear when more important content regions have the same scaling rate as less important regions. Missing content and added distortions in visualizations quickly lead to loss of attention, or worse, the complete misinterpretation of the information presented. Hence, it is paramount to adopt a robust approach that not only resizes the content appropriately but also retains the important content in information visualization.

Existing approaches for content-aware resizing mainly focus on natural images such as portraits, landscapes, and buildings. On the other hand, in information visualization, images normally consist of abstract mathematical representations such as vectors, points, lines, icons and geometrical shapes. The important content often represents the main subject in visual content. Fig. 1(e) shows an example of content-aware resizing of visual information produced by our method.

Most of the existing image resizing approaches are not entirely suitable for information visualization. Grid-based methods [1], as shown in Fig. 1(d), have been used for resizing such images as geometric distortions and are easily identified. Pixel-based seam carving, Avidan et al. [2], is normally used for image resizing. This technique cannot be easily extended to account for layout adjustments of geographical scatterplots and social network graphs. In addition, the criterion for significant regions in visualization is different from natural images because their color schemes are different. For example, a blue sky in a natural scene is normally classified as background. Therefore, it would often be considered less

* Corresponding author.
E-mail addresses: dawn.chli@gmail.com (C. Li), csgeorge@polyu.edu.hk (G. Baciu).



(a) Original visualization.

(b) Cropping method. (c) Scaling method. (d) Grid-based [1]. (e) Our method.

**Fig. 1.** Results of four different strategies for resizing of visualization output.

important than a person in the foreground. Unlike natural images, a blue region rendered by a visualization system may be regarded as an important region.

Information visualization often consists of multiple information layers. For example, a geographical application would normally contain several layers such as water, continents, and various location markers. If we ignore major regions such as continents in resizing, then the results will suffer from distortions. Multi-layered based resizing is rarely discussed in the previous work either on image resizing or on information visualization resizing. The resizing framework of Wu et al. [3] assumed the information visualization layer is single such as scatter-plot, network, and word cloud. However, often there are many abstract layers in information visualization designs such as a scatter-plot on a map and graph with group shapes. Therefore, it is necessary to revisit the multi-layer approaches to detect and preserve the different layers in information visualization. When the resizing content is complex and the canvas become larger, the time performance is becoming more important. Prior work such as Wu et al. [3] requires adjustment to fast resize the information visualization.

Hence, based on the resizing pipeline of Wu et al. [3], we present a different visualization resizing approach in three aspects. First, we define a visualization-related saliency map. Second, we consider the classes of information to be segregated into multi-layers for visualization. Third, the controlling mesh for resizing in our approach is adaptive so that users can emphasize the content of the visualization with fewer distortions in a shorter period of time. The contributions of our work are:

1. an abstract multi-layer model for the resizing problem of information visualization. Our model can be used to resize the output from a visualization system to automatically match the native aspect ratio of any external target display;
2. a set of criteria called the *visual saliency map* (or VSM) to describe the features of information visualizations in different saliency layers;
3. a triangle mesh-based energy optimization method to achieve better visual distribution of information features after resizing. We present the results of our experiments on different genres of multi-layered visualizations to demonstrate the performance of our approach.

## 2. Related work

In the following subsections, we review the related methods on content-aware resizing, saliency mapping, and adaptive meshing.

### 2.1. Content-aware resizing

Many researchers have been working on the content-aware resizing problem recently [4]. The problem is also known as *focus+context resizing* or *saliency-aware resizing*. Generally, content-aware resizing methods can be classified into (a) pixel-based, and (b) mesh-based. Pixel-based methods are discrete. Seam carving [2] is the first proposed pixel-based method that is related to content-aware resizing of images. Rubinstein and his colleagues improved seam carving via forwarding energy [5]. Seam carving was also improved by Xu et al. [6] by transforming the extracted image structure. Unfortunately, these methods are based on some form of pixel energy or intensity levels, and are not applicable to vector-based visualizations such as graphs and geographical maps (GGM). In addition, the content cannot be further enhanced when it is resized by seam carving. Furthermore, the iteration of seam carving is time-consuming and puts severe constraints on real-time interactive editing. Some improvements have been proposed by Yael et al. [7] and Wu et al. [8] with better visual results than the original pixel-based methods. But, these methods still lead to missing information in information visualizations.

On the other hand, mesh-based methods for resizing provide a degree of continuity for the underlying regions. Gal et al. [9] presented a novel resizing method by using a manual feature mask and an underlying grid. Wang et al. [1] described an optimized resizing approach that overcomes the edge distortion problem which was not considered before.

Currently, the focus is increasingly put on content-aware resizing for information visualization. Examples can be found in treemaps [10] and word clouds [11]. These methods provided effective algorithms for special information visualizations. Wu et al. [3] utilized a significance map and quad-based deformation to put forward a general resizing framework for visualization. However, for complex elements in multiple layers, such as geographical information and large graphs, the existing methods are still inadequate.

### *2.2. Saliency map*

Saliency maps play a very important role in content-aware retargeting for both images and visualizations because these maps can be treated as a form of energy of pixels, which can be used to build an energy function, such as shown by Avidan [2] and Wang [1]. Itti et al. [12] took into consideration the human visual system and denoted the significance of points from a natural image. They also presented an effective feed-forward feature-extraction method to compute a saliency map from it. Frintrop et al. [13] extended the work of [12] and computed saliency at the pixel level with high performance. Wang et al. [1] used the method explained in [12] to assign a significance threshold to quads. Wu et al. [3] combined a clutter map and a DOI map into a significance map, which is another type of saliency map. Zhang et al. [14] used anisotropic diffusion equation to further improve the accuracy of saliency detection. However, in information visualizations, the methods above require further adaptation for saliency detection.

Achanta et al. [15] summarized five basic requirements for saliency detection and proposed a simple implementation, called FT. FT can be easily extended to consider the multiple visual features in images. Goferman et al. [16] presented a novel method to detect the context-aware saliency map that aimed at representing the dominant objects in an image. They argued that the context of a region should be also considered to generate a more accurate saliency map. They also demonstrated that their approach has potential applications in image resizing. Yan et al. [17] proposed a hierarchical saliency detector that can generate a multi-layer saliency map for natural images. Cellular automata with different layers was used by Qin et al. [18] to detect saliency among similar image patches. However, in information visualizations, the methods of [16–18] require further adaptation for saliency detection.

### *2.3. Adaptive mesh*

Adaptive meshes are usually used in computer graphics, particularly, in physically-based simulations. Adaptive meshing can generate various densities of meshes for different scales. For example, Busaryev et al. [19] used the adaptive meshes to simulate fracture-aware re-meshing to provide more details in fracture regions. For image retargeting, Kaufmann and his colleagues, [20], used the Finite Element Method to reduce the degrees of freedom in deformation, which stands out in real-time image resizing. For visualization resizing, Wu et al. [3] proposed the quad-tree method that can generate adaptive quads to cover the important regions as more as possible. The method proposed by Wu et al. [3] works well for single-layer information visualization, but it would require modification to deal with multi-layer resizing of information visualization and improve the performance.

## 3. Overview of our proposed method

We define information visualization resizing as a saliency detection and geometric deformation problem. The input of our model is a multi-layered rendering. Multi-layers can be viewed as more than one representation in information visualization. In the example of Fig. 1(a), the input includes a geographical map, lines and nodes with different radii. First, we detect the visual saliency through a hybrid saliency model, the VSM, that can generate different saliencies for different layers in visualization. In this step, we further improve the accuracy of the visual saliency detector by considering the lightness in color information. Second, we create an adaptive mesh that consists of controlling triangles with different levels of detail over the input visualization. The input data, such as nodes, were bound to vertices in the mesh according to their positions. Third, we formulate the controlling triangles for the resizing problem as an optimization problem according to the VSM. Finally, we can resize the input visualization by solving a large sparse linear system. After the mesh deformation, the important features of the input can be well preserved as shown in Fig. 1(e).

## 4. The visual saliency map (VSM)

The proposed saliency-based method, called the *visual saliency map* (VSM), adaptively indicates the significant regions in a information visualization. For the content-aware resizing, the deformation of each region is dependent on its corresponding saliency. Although the saliency of each region can be assigned by users manually, it is more effective to automatically detect the important regions.

The saliency concept for visualization is different than the one used in natural images in three aspects. First, the content in data visualizations is sharper than that in natural images, hence it is unnecessary to pre-process the pixels, such as smoothing or noise reduction, before saliency detection. Second, the regions with different colors but similar shapes in visualizations may have the same ground-truth saliency. However, ordinary saliency detection algorithms for natural images normally define those regions with different saliency. Third, the resizing approaches for natural images rarely take context into consideration. However, context information is also required to be preserved in multi-layer visualizations.

Normally, the background map in geographical applications and communities of social networks can be defined as context because they are closely related to important regions that users are interested in. Hence, we focus on two parts in the visual saliency map. One is the importance detector, which can detect the most important regions, and the other is the context detector, which can detect edges and the second important regions.

Our VSM method is different from the work of Wu et al. [3], who adopted a clutter map (Feature Congestion [21]) and DOI map to guide the mesh deformation. The DOI map is a fine supplement for a cluster map because it can show the degree of interests. However, when the features are more complex shapes such as a ge-

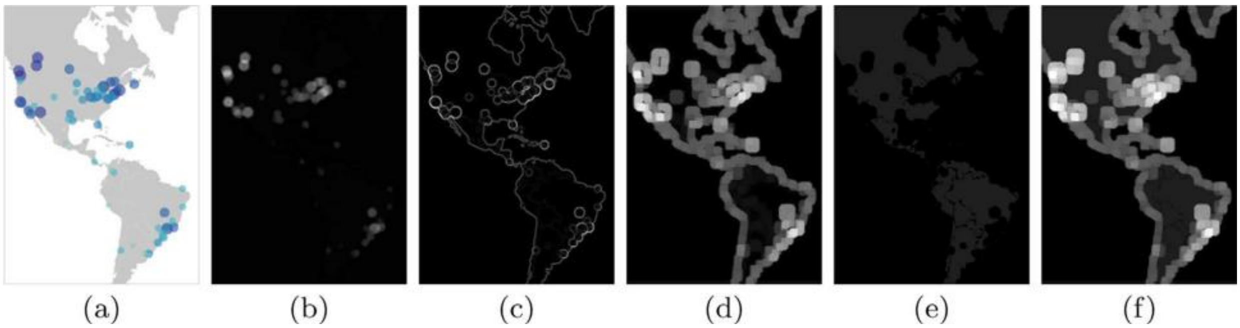



**Fig. 2.** Different parts of visual saliency map. (a) Original visualization image. (b) Important regions. (c) Sharp edges. (d) Enhanced edges. (e) second important regions. (f) Final visual saliency map.

**Fig. 3.** Color palette from Tableau. The top palette of deep colors is normally used to represent the most important regions in visualization. On the contrary, a bottom palette of lightish colors indicates the second important regions or backgrounds in visualization.

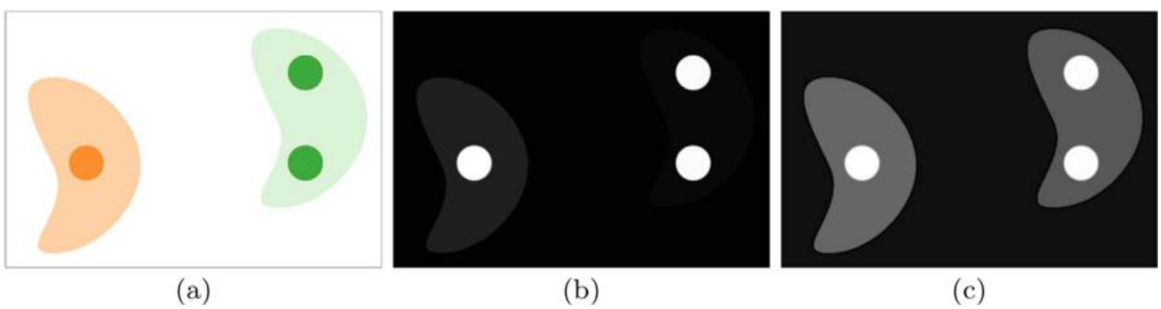



**Fig. 4.** An example of the multi-layer visualization saliency map. (a) Multi-layer visualization with lightish color. (b) The saliency map is unavailable because the light green region is not detected. (c) Visual saliency map for solving SSDC problem.

ographical map or a filled irregular shape, the generation of DOI map requires adaptation.

### *4.1. Importance detector*

We adopt a frequency-based method [15] to detect the most important region (as shown in Fig. 2b). This method can be formulated as follows:

$$G(i,j,k) = \frac{1}{2\pi k^2} e^{-(i^2+j^2)/2k^2} \tag{1}$$

$$S_{\mathrm{m}}(i,j) = \| \mathbf{G}(i,j,\infty) - \mathbf{G}(i,j,\varepsilon) \|_2 \tag{2}$$

where $\mathbf{G}(i,j,\varepsilon)$ represents the Gaussian blurred values of pixel ($i$,$j$) for image rendering following the Gaussian filter function, as shown in Equation (1. The function $\mathbf{G}(i,j,\infty)$ can be approximated through a filter template $\frac{1}{4}[1,2,1]$ with $k$=1.6. Because different features such as color and contrast should be considered in the importance detection, $\mathbf{G}$ is defined as a vector. In the method of [15], Achanta et al. adopted the LAB color space with the three features lightness (L), one color-opponent dimension (A), and another color-opponent dimension (B) as a feature vector. The feature of lightness in LAB color can represent high contrast regions, which are normally considered of high importance in information visualization.

We found that using the LAB color space to maintain feature vectors is not enough to detect available saliency regions in information visualization. For example, if the designers follow the Tableau 20 palette [22] as shown in Fig. 3, it is difficult to detect the available saliency for lightish regions as shown in Fig. 4(b). We define this problem as similar-shape-different-color (SSDC) problem. We add the mid-channel feature to the feature vector so that we can ignore the extreme value of the color channel among lightish colors. The mid-channel feature is represented via calculating middle value of three channels in RGB color. Finally, the feature vector in $\mathbf{G}$ can be defined as $\mathbf{v}_f = [l, a, b, m]$. The better visual contrast between features can be achieved by modifying the feature vector as in [15], see Fig. 4(c).

### *4.2. Context detector*

The context is normally the surrounding area of the important region. The context should also be considered because the distortion of context while resizing will also affect the visual results. We assume that the context includes two parts: edges and the second important regions. The *Sobel* and *Canny Operator* can be used to detect edges in a visual image. We select the *Sobel Operator* because it is a discrete differentiation operator that achieves a higher performance than the *Canny Operator*. The edge (as shown in Fig. 2c) of the visualization can be abstracted via *Sobel Operator* as follows:

$$S_{\mathrm{e}}(i,j) = \sqrt{g_x(i,j)^2 + g_y(i,j)^2} \tag{3}$$

where $g_x(i,j)$ is the gradient along horizontal direction and $g_y(i,j)$ is the gradient along vertical direction. Because the edge is narrow, it is difficult to be used to generate triangles. Thus, we enhance edges through a *Dilation Operator* which can be formulated as:

$$S_{\mathrm{d}}(i,j) = \max\{S_e(i+m, j+n)\} | (m,n) \in \mathbf{B} \tag{4}$$

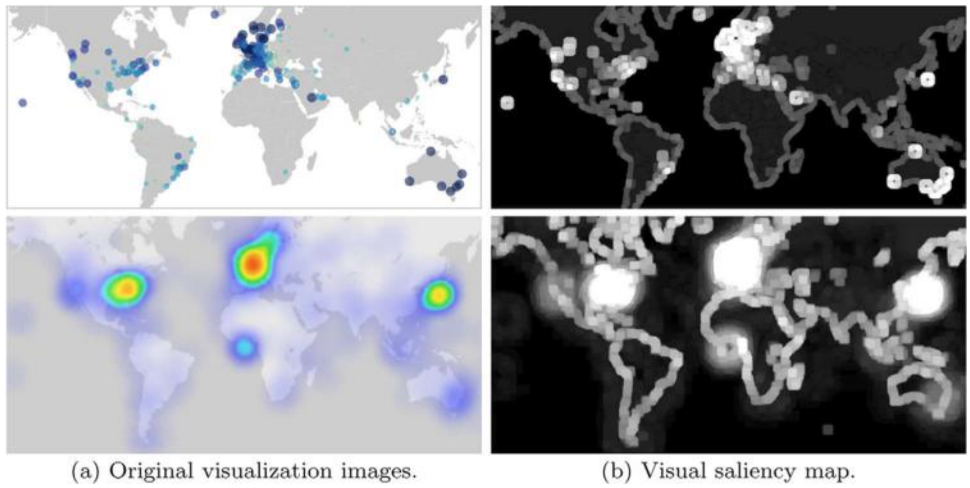



**Fig. 5.** Examples of visual saliency map. (top) Visualization of the cost of living in different countries as described in Section 6.1.1. (bottom) A heat-map of users' locations from Brightkite.com as described in Section 6.1.2.

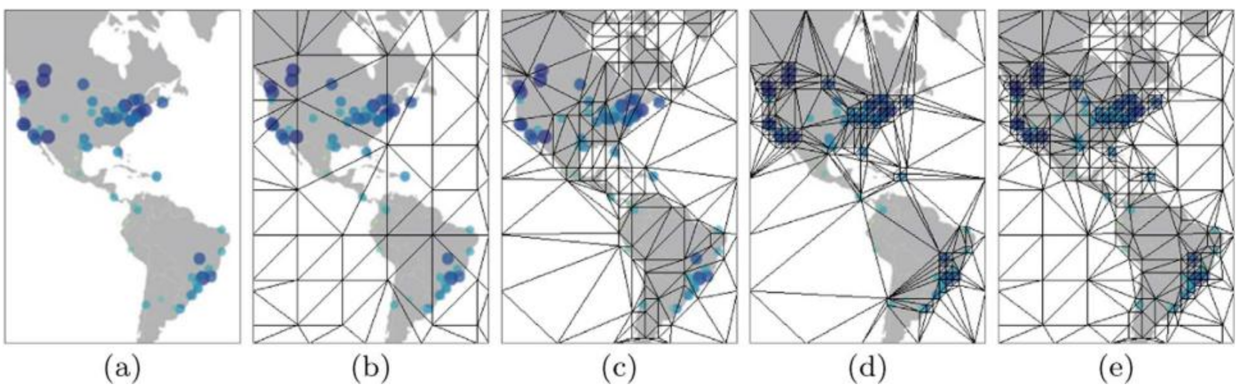



**Fig. 6.** Triangulations with different level of details where $\alpha$=20. (a) Original visualization. (b) Level=1, $S(i,j) \in [0, 0.15)$. (c) Level=2, $S(i,j) \in [0.15, 0.5)$. (d) Level=3, $S(i,j) \in [0.5, 1.0])$. (e) Level=1-3, $S(i,j) \in [0.0, 1.0]$.

where **B** is a square of radius r, (m,n) is a point in **B**. The radius of dilation operator is set to 3 in our experiments.

For the second important region, we assume it indicates the second largest regions in $S_{\rm m}$. All of those regions contains the same saliency value. Our second important region detection is achieved through a histogram of $S_{\rm m}$. First, we calculate the histogram of $S_{\rm m}$ and find the second largest peak value $p_{sec}$ in it. We ignore the first largest part because it normally indicates the background. The second important region can be defined as follow:

$$S_{sec}(i,j) = \begin{cases} 0, & S_{\rm m}(i,j) = 0 \\ \sqrt{p_{sec}}, & S_{\rm m}(i,j) = p_{sec} \end{cases} \tag{5}$$

where we use the sqrt operator to enhance the saliency value. Then, we can formulate the context detector as $S_{\rm c}(i,j) = S_{\rm d}(i,j) + S_{sec}(i,j)$ and lead to the final value of VSM as follow:

$$S_{\rm vsm}(i,j) = S_{\rm m}(i,j) + S_{\rm c}(i,j) \tag{6}$$

$S_{\rm vsm}(i,j)$ should be normalized to limit the value in $[0,1]$. With the VSM, we can adaptively indicate the saliency layers of various visual elements that are the preparation of our resizing model. Fig. 5 shows two examples of our VSM.

## 5. Adaptive resizing model

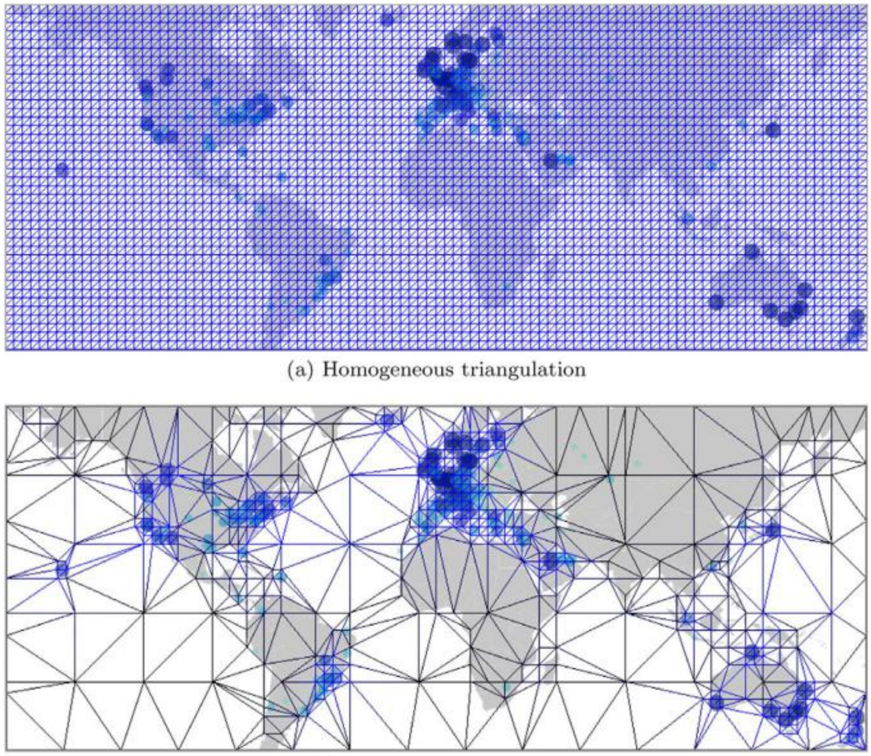
In the following sections, we describe our resizing model in detail. First, we start with the adaptive meshing. We resort to triangular meshes as triangular meshes can be more readily adapted to high-density regions.

### 5.1. Adaptive triangulation

Here we propose an adaptive triangulation method to reduce the degree of factors (DOF) that are related to the performance of resizing.

Based on the VSM, we vary the density of triangles for different important regions. First, a point set will be extracted on the basis of intensities in saliency map. *Delaunay Triangulation* [23] can be used to generate triangles. For two-dimensional triangulation, a set of key points **V** will construct a mesh **M. M** should fulfill three constraints. First, the edges in **M** do not contain any key points in **V** except for the start and the end point. Second, an edge cannot intersect another edge. Third, all elements in **M** are triangles. These elements form the convex hull of **V**.

We utilize the VSM to create a set of key points, the **V** band that satisfies these three constraints to generate **M** that includes several triangles. The threshold is a quick method to create key points from a saliency map via extracting the points with high saliency, but a large number of key points will be extracted if there are many pixels with high saliency. Therefore, we construct a constraint to determine the interval of triangulation sampling to limit the total number of points in **V**. We define the constant number of points in **V** through $C_p = \frac{wh}{\alpha^2}$, where $w$ and $h$ are respectively the width and height of the canvas in the visualization, and $\alpha$ is a free parameter that is used to adjust $C_{\rm p}$. Different values of saliency can be mapped to different levels of triangulations. We use $t_{k0}$ and $t_{k1}$ to indicate the range of saliency value for level $k$. The interval of



(a) Homogeneous triangulation

(b) Adaptive triangulation

**Fig. 7.** Two triangulation strategies.

triangulation sampling for different levels $k$ can be formulated as follows:

$$I_k = \frac{C_p}{\sqrt{\sum_{t_{k0} \le S(i,j) < t_{k1}} S(i,j)}} \tag{7}$$

When the saliency of a region is between $t_{k0}$ and $t_{k1}$, it should follow the interval of sampling $I_k$. The interval of sampling indicates the count of interval pixels of each key point $v_k$ in $\mathbf{V}$. Fig. 6 shows an example of triangulation with different level of details. In our resizing model, we merge three levels of triangulations into one as shown in Fig. 6(e). By using VSM and *Delaunay Triangulation*, we can generate content-aware triangles as shown in Fig. 7(b), which is more flexible than the homogeneous triangulation in Fig. 7(a).

### 5.2. Mesh resizing

#### 5.2.1. Resizing energy optimization

We represent the topology of the mesh with $\mathbf{M} = (\mathbf{V}, \mathbf{E}, \mathbf{T})$, where $\mathbf{V} = [\mathbf{v}_0^T, \mathbf{v}_1^T, \ldots, \mathbf{v}_n^T]$, $(\mathbf{v}_i \in \mathbb{R}^2)$ denotes the vertices, $\mathbf{E}$ are the edges and $\mathbf{T}$ depicts the triangles that respectively have three vertices and three edges. We use $\mathbf{V}' = [\mathbf{v}_{0'}^T, \mathbf{v}_{1'}^T, \ldots, \mathbf{v}_{n'}^T]$ to indicate the deformed vertices. $\mathbf{E} = [\mathbf{e}_0^T, \mathbf{e}_1^T, \ldots, \mathbf{e}_n^T]$, $(\mathbf{e}_i = \mathbf{v}_a - \mathbf{v}_b)$ was used to indicate the edges. The edges divide the visualization into several patches. The basic idea of mesh resizing is optimizing a linear energy function as shown in [9]. The linear energy function can be defined as $\int_y^h \int_x^w S(x,y) \| \mathbf{J_F} - \mathbf{P} \|_2^2 dxdy$, where $\mathbf{J_F}$ is a Jacobian matrix, $\mathbf{P}$ represents the linear transformation, $w$ and $h$ is the width and height of the canvas, respectively.

Inspired by Wang et al. [1], we assume that triangles bound to areas with high saliency will be assigned a smaller scaling factor, while triangles that cover less important areas can be distorted due to linear scaling operations. Our method is different from Wang et al. [1] because we change the basic controlling unit from a rectangular grid to a triangular mesh. The resizing energy for a triangle can be formulated as follows:

$$R(t) = \sum_{\{m,n\} \in E(t)} \| (v'_m - v'_n) - s_t(v_m - v_n) \|^2 \tag{8}$$

where $t \in \mathbf{T}$ indicates a triangular, $s_t$ is a scaling factor for each vertex v in the triangle and we assume that the scaling factors for x-axis and y-axis are the same, $E(t)$ is a set of edges in $t$, $v$ and $v'$ respectively indicates the vertex of an original triangle and its corresponding deformed vertex, $\| \cdot \|_2$ is the $L_2$ norm of a vector. We can differentiate $R(t)$ and continue getting a direct solution of $s_t$, as shown in [1]. As we can see, $R(t)$ calculates the resizing energy of edges, however, different triangular should have different warp due to their different saliency. Hence, we need to take the saliency value of each triangle into consideration. The resizing energy of all triangles is defined as follows:

$$R = \sum_{t \in \mathbf{T}} \lambda_t R(t) = \sum_{t \in \mathbf{T}} \frac{1}{n} R(t) \sum_{\mathbf{x} \in P_t} S(\mathbf{x}) \tag{9}$$

where $\lambda_t = \frac{1}{n} \sum_{\mathbf{x} \in P_t} S(\mathbf{x})$ is the saliency of each triangle, $\mathbf{P_t}$ is the set of all pixels in t and $\mathbf{x}$ indicates the position of a pixel, $\mathbf{n}$ is the number of pixels in $\mathbf{P_t}$. Because the saliency of each pixel can be calculated from the VSM, we can use a scan–line algorithm to detect the entire pixel saliency contained in one triangle. This leads to the triangle saliency $\lambda_t$. Because R is a quadratic energy function, we can minimize the total resizing energy to obtain the final deformed vertices $\mathbf{V}'$.

#### 5.2.2. Constraints

In order to get better resizing results, constraints should be taken into account. The constraints for resizing includes overlapping preventing, boundary and smoothing. We adopt a method as illustrated in [24] to guarantee that the mesh is resized accurately without overlap. Then, we formulate two constraints such as boundary checking and mesh smoothing for resizing.

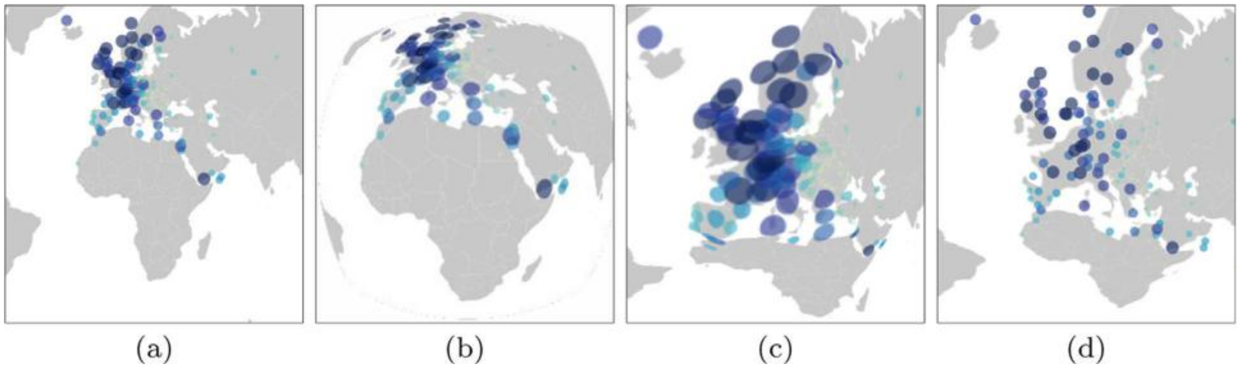



**Fig. 8.** Results of three different strategies for content enhancement. (a) No enhancement. (b) Fisheye enhancement. (c) Content-aware enhancement without vectorial adjustment. (d) Content-aware enhancement with vectorial adjustment.

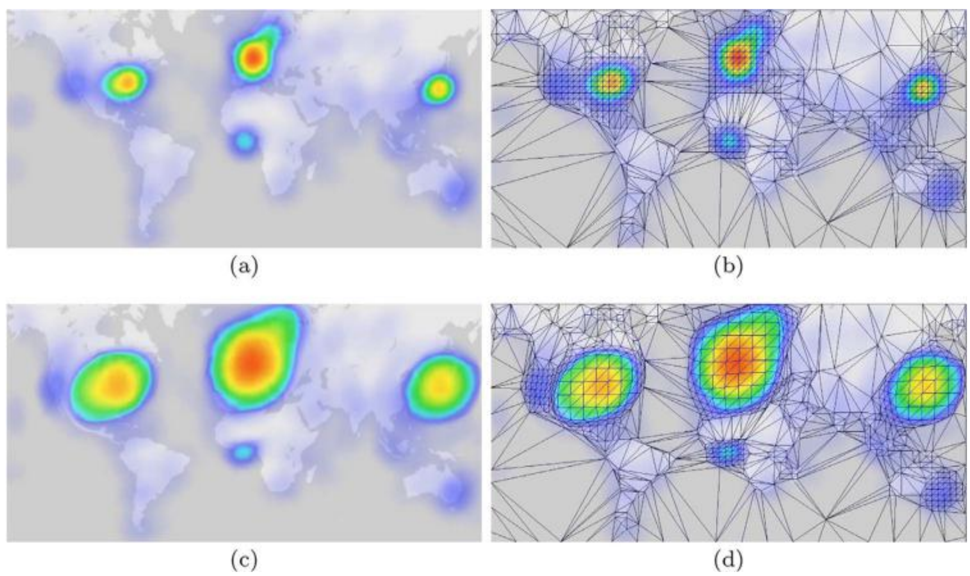



**Fig. 9.** The result of enhancement without canvas resizing. The content in this visualization is a heat-map of users' locations from Brightkite as described in Section 6.1.2. (a) Original visualization. (b) Triangulation of original visualization. (c) Enhanced visualization without canvas resizing. (d) Triangulation of enhanced visualization.

- Boundary Constraints

Because the boundary of the visualization should fulfill the linear scaling to avoid the warp, R should obey the constraints on the boundary. We define the vertices set on the boundary as B, which includes four corner vertices and several vertices on four side edges of the boundary. The vertex set on four side edges can be defined as $\mathrm{B}(left)$, $\mathrm{B}(top)$, $\mathrm{B}(right)$ and $\mathrm{B}(bottom)$. Hence, the constraints for B can be defined respectively as:

$$\begin{cases} \mathbf{v}_{left,top} = (0,0)^T, \\ \mathbf{v}_{right,top} = (w,0)^T, \\ \mathbf{v}_{left,bottom} = (0,h)^T, \\ \mathbf{v}_{right,bottom} = (w,h)^T \end{cases} \tag{10}$$

$$\begin{cases} \mathbf{v}'_i = (0, v_{iy}), i \in \mathrm{B}(left), \\ \mathbf{v}'_i = (v_{ix}, 0), i \in \mathrm{B}(top), \\ \mathbf{v}'_i = (w, v_{iy}), i \in \mathrm{B}(right), \\ \mathbf{v}'_i = (v_{ix}, h), i \in \mathrm{B}(bottom). \end{cases} \tag{11}$$

where $w$ indicates the width of the canvas and $h$ indicates the height of the canvas.

- Smoothing Constraint

It is necessary to further avoid distortion via smoothing scaling factors as mentioned in [1,3]. We formulate a scaling energy function as follows to smooth out the scaling factor of each triangle:

$$E_s = \sum_{t \in T} \sum_{n \in \mathrm{Adj}(t)} \frac{1}{2}(a_t + a_n)(\lambda_t s_t - \lambda_n s_n)^2 \tag{12}$$

where $\mathrm{Adj}(t)$ is a set of adjacent triangles of $t$, $n$ indicates one adjacent triangle, $a_t$ and $a_n$ respectively represent the acreage of $t$ and $n$. $E_s$ is an energy function that represents the distortion scaling factor of adjacent triangles. We can achieve better scaling factors by optimizing $E_s$.

### 5.3. Vector adjustment

For image resizing of visualizations, our algorithm also preserves the intended salient features. Furthermore, our algorithm can be easily extended to vector resizing of the visualization with feature preservation. Vector-based resizing means that some vector elements such as points and edges can be bound to the vertices in triangulated mesh. The positions of elements in visualization will be adjusted through the triangulated mesh resizing. We take the scatterplot in a world-map as an example. We assume points in

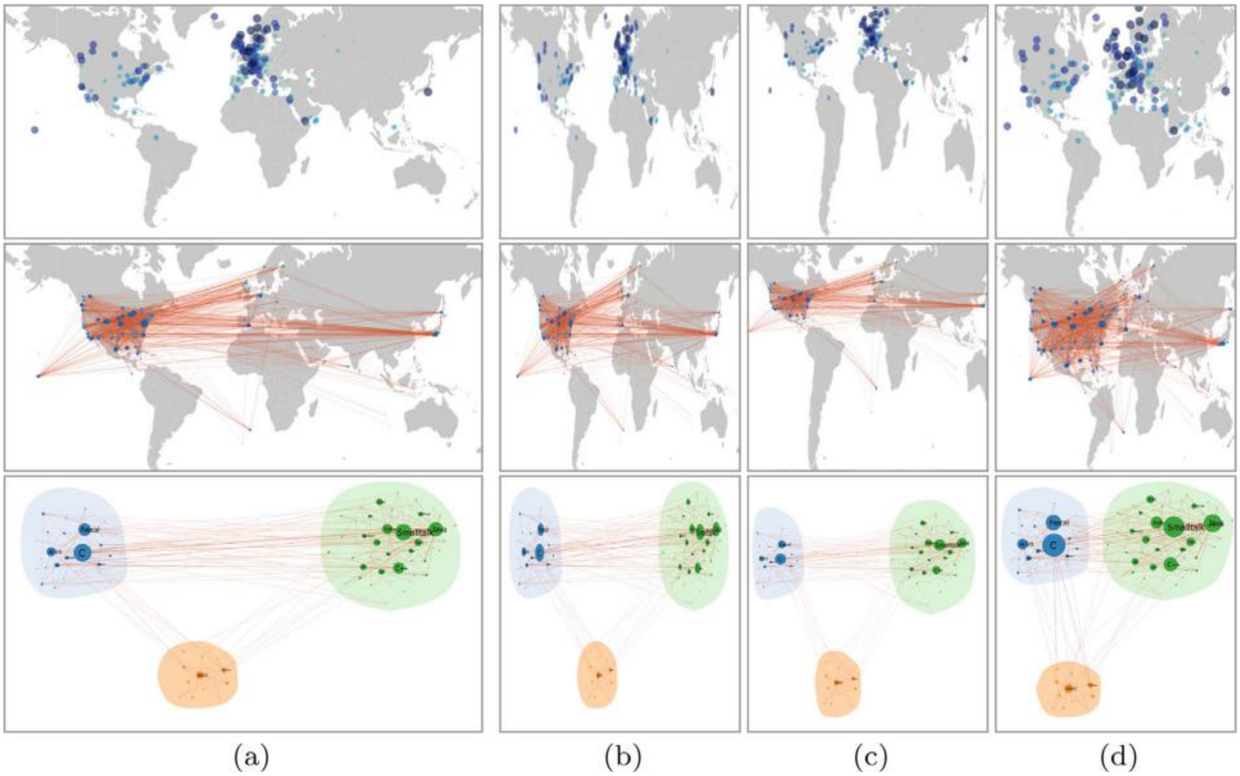


**Fig. 10.** Results of 3 different strategies for horizontal resizing of SGM 6.1.1, GGM 6.1.2 and GC 6.1.3. (a) Original visualization images. (b) Linear scaling. (c) Grid-based resizing [1]. (d) Our method.

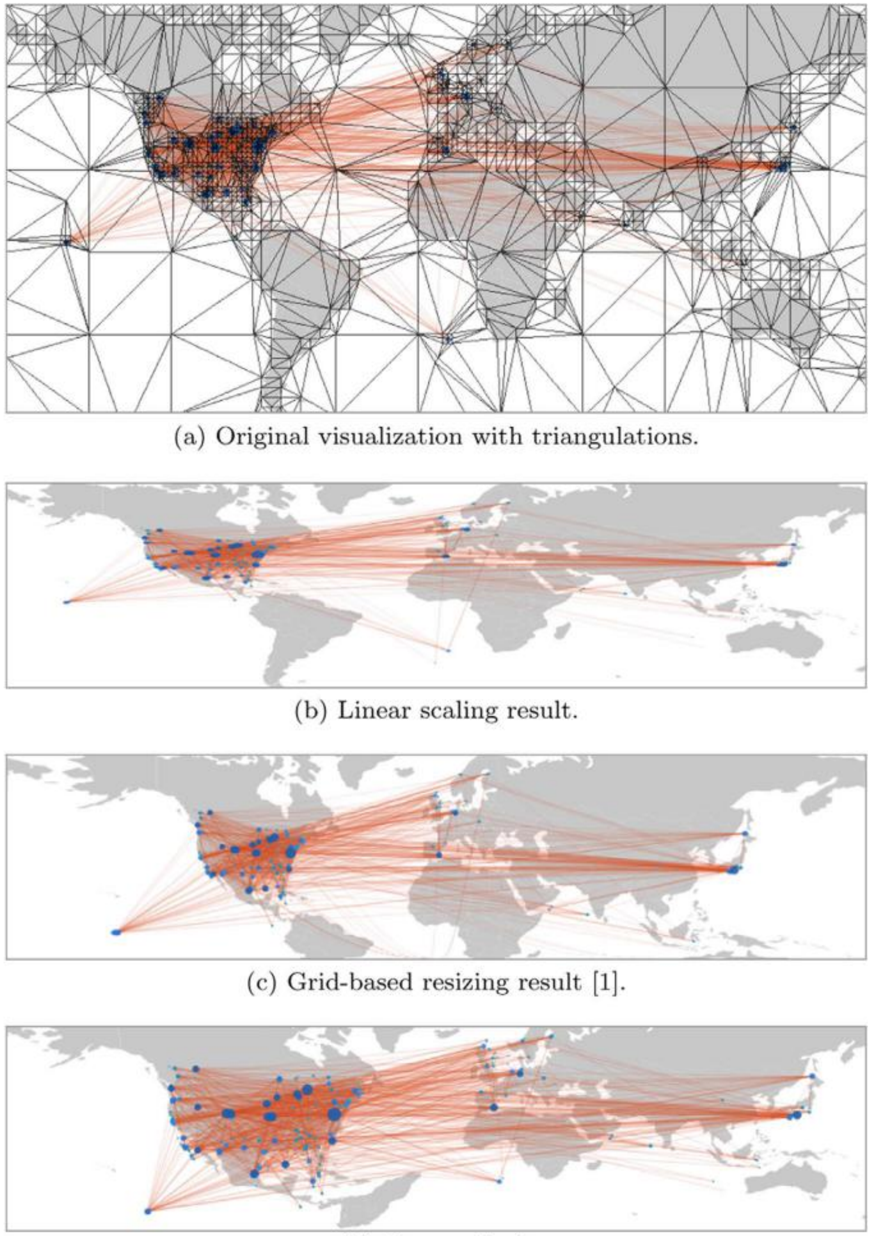
**Table 1**
The performance and evaluation of mesh deformation methods.

| Cases | Original size | New size | Method | Vertices | DOF | Time cost (ms) |
|---|---|---|---|---|---|---|
| SGM Section 6.1.1 | 876×374 | 385×374 | Grid-based [1] | 741 | 4104 | 175 |
| | | | Our method | 324 | 586 | 19 |
| GGM Section 6.1.2 | 876×414 | 438×414 | Grid-based | 1012 | 5670 | 263 |
| | | | Our method | 589 | 3242 | 64 |
| GC Section 6.1.3 | 1500×800 | 750×800 | Grid-based | 2886 | 16644 | 4739 |
| | | | Our method | 853 | 4824 | 161 |
| IG Section 6.1.4 | 678×680 | 1066×680 | Grid-based | 1849 | 10584 | 1337 |
| | | | Our method | 611 | 3498 | 58 |
| Hex Points | 750×495 | 375×495 | Grid-based | 950 | 3800 | 206 |
| | | | Our method | 408 | 2310 | 19 |
| Graphs | 600×363 | 300×363 | Grid-based | 465 | 2520 | 42 |
| | | | Our method | 405 | 2322 | 19 |
| Social Network | 600×299 | 300×299 | Grid-based | 589 | 2356 | 54 |
| | | | Our method | 280 | 1560 | 12 |
| Heat-map | 1126×563 | 663×563 | Grid-based | 1653 | 6612 | 1024 |
| | | | Our method | 479 | 2694 | 30 |
| China Map | 1320×791 | 660×791 | Grid-based | 2688 | 10752 | 3932 |
| | | | Our method | 393 | 2142 | 32 |

the scatterplot are the main regions and the background map is the context that also needs to be preserved. Each point will be bound to the vertex in the triangulated mesh. The adjustment of positions can be calculated from the controlling mesh via resizing the visualization image. We use the whole visualization image to generate the saliency map and respectively resize the main part and the context part. In the final step, adjusted points will replace the main part and combine with context part as final result according to their adjusted positions. As shown in Fig. 10, in our experiments we adopt a hybrid method which takes advantage of both image resizing and vectorial resizing. This improves the resulting visual features.

### 5.4. Content enhancement

Content enhancement is aimed at further enhancing the important regions while resizing the canvas. Let's take Fig. 8 as an example, there are so many overlapping plots in the region of Europe that users are impeded to get integrated information. Fish-eye distortion can be a solution for this problem. However, there are two drawbacks of the fish-eye distortion method. First, manually selecting the significant region is tedious for users. Second, simple spherical representation is not enough to enhance irregular regions.



(a) Original visualization with triangulations.

(b) Linear scaling result.

(c) Grid-based resizing result [1].

(d) Our method.

**Fig. 11.** Results of 3 different strategies for vertical resizing of GGM 6.1.2.

Next, we present a method that can achieve better content enhancement for information visualization. In addition, our method also has immediate applications in resizing content from a large display device to a very small one. In our method, we add an enhancing factor called $\delta_k$ to control the enhancement of different layers with different saliency layer in visualization because a scaling factor $s_t$ has been formulated in resizing energy function Equation (8). $k$ denotes the layer of saliency. We can formulate an enhancing energy function for content enhancement as shown in Equation (13). Moreover, the values of $\delta_k$ can be adjusted by users to interactively enhance the desired regions. The interactive operation can be brushing over the triangles with a different value. Figs. 8 and 9 show two examples of content enhancement.

$$E(t) = \sum_{\{m,n\}\in E(t)} \| (v'_m - v'_n) - s_t \delta_k \lambda_t (v_m - v_n) \|^2. \tag{13}$$

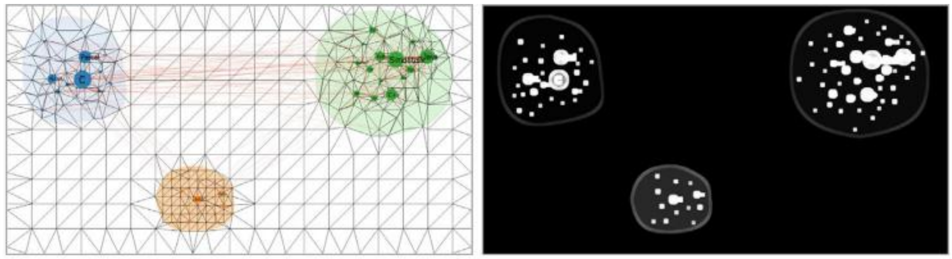
## 6. Experiments and results

All the experiments in this paper were performed on a computer with Windows 7 OS, an Intel i7 CPU 2.8 GHz and 8 GB RAM. We implemented the algorithm in C++ and used CGAL [25] library to generate the triangulation. Lapack++ and C++ library were employed to solve the large sparse linear system of equations.

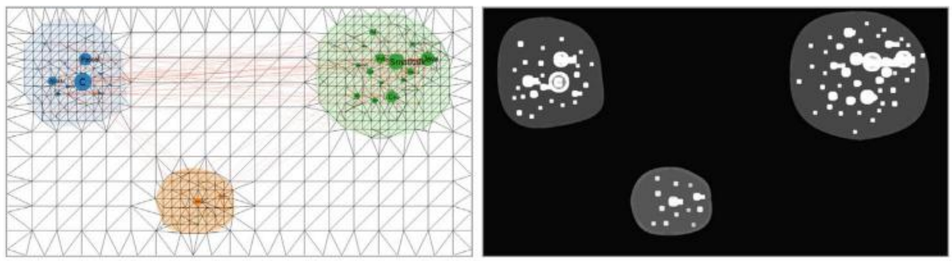
We tested our method on several datasets and obtained better results than previous methods. Because we use the adaptive method to generate triangles, better performance can be achieved than previous methods for visualization resizing. We show the performance of our method on Table 1 and evaluate our work through the remaining saliency (Fig. 15) in the resized visualization.

### 6.1. Results

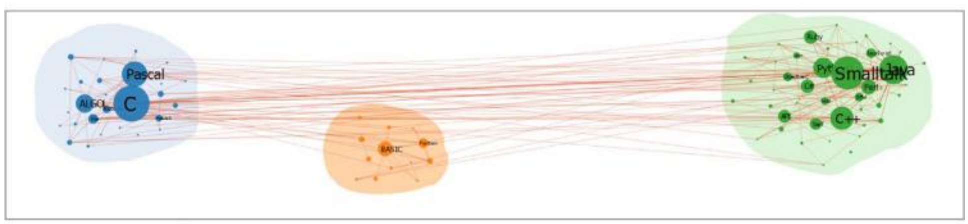
We base our experiments on multi-layered information features for visualization. We analyze three cases of resizing using the



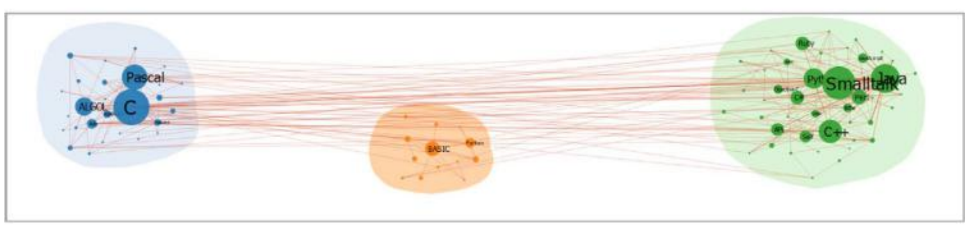
(a) Triangulation and saliency map without SSDC fixing [15]



(b) Triangulation and saliency map with SSDC fixing (our method).




(c) The result of vertical resizing of GC without SSDC fixing.




(d) The result of vertical resizing of GC with SSDC fixing.

**Fig. 12.** Results of 2 different strategies for vertical resizing of GC 6.1.3.

method presented in this paper. We compared our approach with the linear scaling and the grid-based method. Our results demonstrate that adaptive content-aware method has better visual feature representation than the grid-based method for multi-layered visualization resizing. Furthermore, we show additional resizing results in Fig. 14.

*6.1.1. Scatterplots and geographical maps (SGM)*

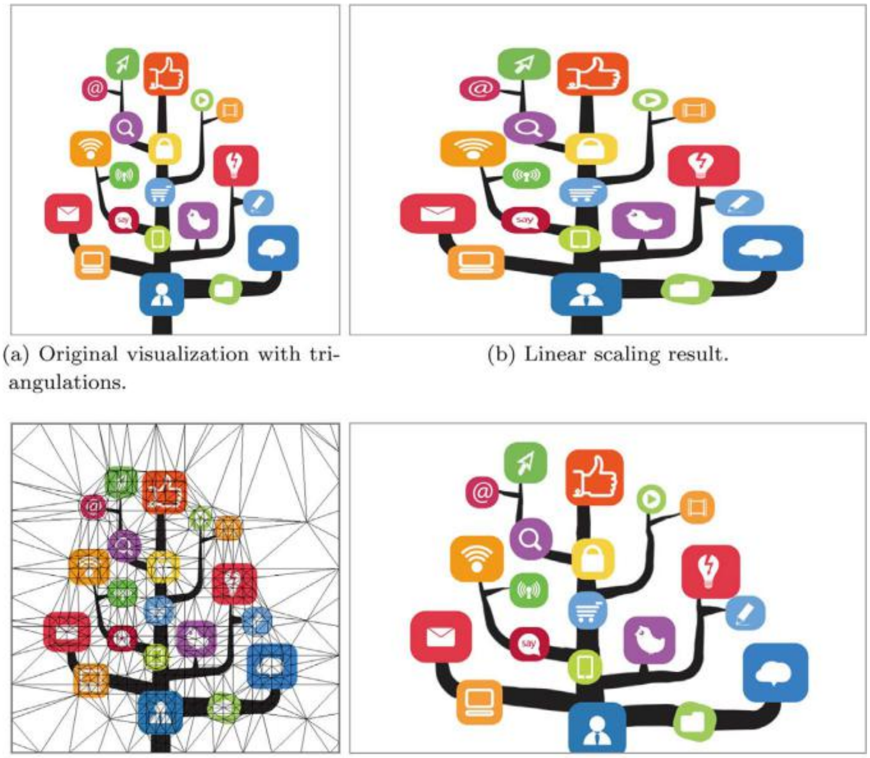
In the first multi-layer visualization case, we select a dataset from numbeo.com that records the cost of living in the world. The scatterplots and geographical map (SGM) can be regarded as two layers of the visualization of this dataset. Scatterplots show the cost of living in different countries. Geographical map demonstrates the distribution of countries. The algorithm in [3] acts on just one layer in the visualization such as the scatterplots, and we take another layer such as the geographical map into consideration. Because the geographical map occupies a large area, computing the cluster map in [3] requires more calculations. We can see that all the circles in Fig. 10(top) suffer less distortion after the resizing using our method. We can also adjust the enhancing parameter to enhance important regions in the visualization as shown in Fig. 8(d).

*6.1.2. Graph and geographical maps (GGM)*

In the second case, we select a large dataset called Brightkite [26]. Each record in this dataset includes a geographical location. When the canvas is resized without the content-aware constraints, two main problems emerge. First, re-layout algorithm such as the direct-force method is time-consuming for large graphs because due to a large number of nodes. Second, the important regions are not preserved. Although regular grid-based methods can preserve important regions, the high density of grid will result in excessive calculation because the computation is related to the grid density. In our method, we bind each node with a nearby vertex in the controlling mesh and redraw the nodes after resizing the mesh, so that we can get high performance of content-aware resizing. The resizing results of the GGM are shown in Fig. 10(middle) and Fig. 11.

*6.1.3. Graph and community (GC)*

The third example demonstrates the advantages of our method on dealing with the SSDC problem as discussed in Section 4.1. We select a graph-based dataset, which indicates the relationship between computer languages. This dataset can be considered as a social network of computer language. In visualization, this dataset



(a) Original visualization with triangulations.

(b) Linear scaling result.

(c) Original visualization with triangulations.

(d) Our method.

**Fig. 13.** Results of 2 different methods for horizontal resizing of the visualization in IG 6.1.4.

can be represented as graph and community. All nodes in the graph are assigned different colors according to their *Modularity Classes*, which were computed by a statistical method presented by Blondel et. al [27]. The nodes with similar color will build a community which is presented as an ellipse-like shape in the visualization image. We draw the visualization image with the color palette from Tableau. Our result in Fig. 10(bottom) and Fig. 12 show that our method can get a better result and avoid the problem of SSDC.

### 6.1.4. *Infographic (IG)*

The fourth example demonstrates the advantages of our method for handmade multi-layer visualization. Infographic is a visual representation of information that utilize text, line, and graphics to improve the visual pattern discovery. Unlike splatterplot and graph, the layout of the infographic is usually created by vector graphics editor such as Adobe Illustrator. We selected a vector graphic from shutterstock.com and applied our method to resize the vector graphic. Our result in Fig. 13(d) shows that our method can get a better result than linear scaling for infographics.

## 6.2. *Discussion*

### 6.2.1. *Performance*

The performance of our method is better than previous resizing algorithms for information visualizations as shown in Tables 1, 2. The performance of mesh deformation has been significantly improved compared with the work of [1] because we generate an adaptive geometric mesh for content deformation. Our approach can substantially improve the performance when an image contains large empty regions, i.e. more than 50%. Table 1 shows that our method can reduce the vertex number, which is highly related to time cost of solving a linear equation. Wu et al. [3] also presented an adaptive grid implementation with high accuracy. Since it is slower than using a regular grid method, as mentioned by Wu et al. [3], we do not compare adaptive grid method with ours.

In addition, we adopt a fast method to create the VSM, so that more time-consuming calculations, such as estimating the kernel

**Table 2**
The time cost of clutter map [21] and visual saliency map (ours).

| Cases | Canvas size | Method | Time cost (ms) |
|---|---|---|---|
| SGM Section 6.1.1 | 876×374 | Clutter map [21]<br>Our method | 3441<br>472 |
| GGM Section 6.1.2 | 876×414 | Clutter map<br>Our method | 3813<br>525 |
| GC Section 6.1.3 | 1500×800 | Clutter map<br>Our method | 12177<br>1749 |
| IG Section 6.1.4 | 678×680 | Clutter map<br>Our method | 4631<br>653 |
| Hex Points | 750×495 | Clutter map<br>Our method | 3762<br>534 |
| Graphs | 600×363 | Clutter map<br>Our method | 2370<br>303 |
| Social Network | 600×299 | Clutter map<br>Our method | 1973<br>254 |
| Heat-map | 1126×563 | Clutter map<br>Our method | 6477<br>943 |
| China Map | 1320×791 | Clutter map<br>Our method | 10456<br>1553 |

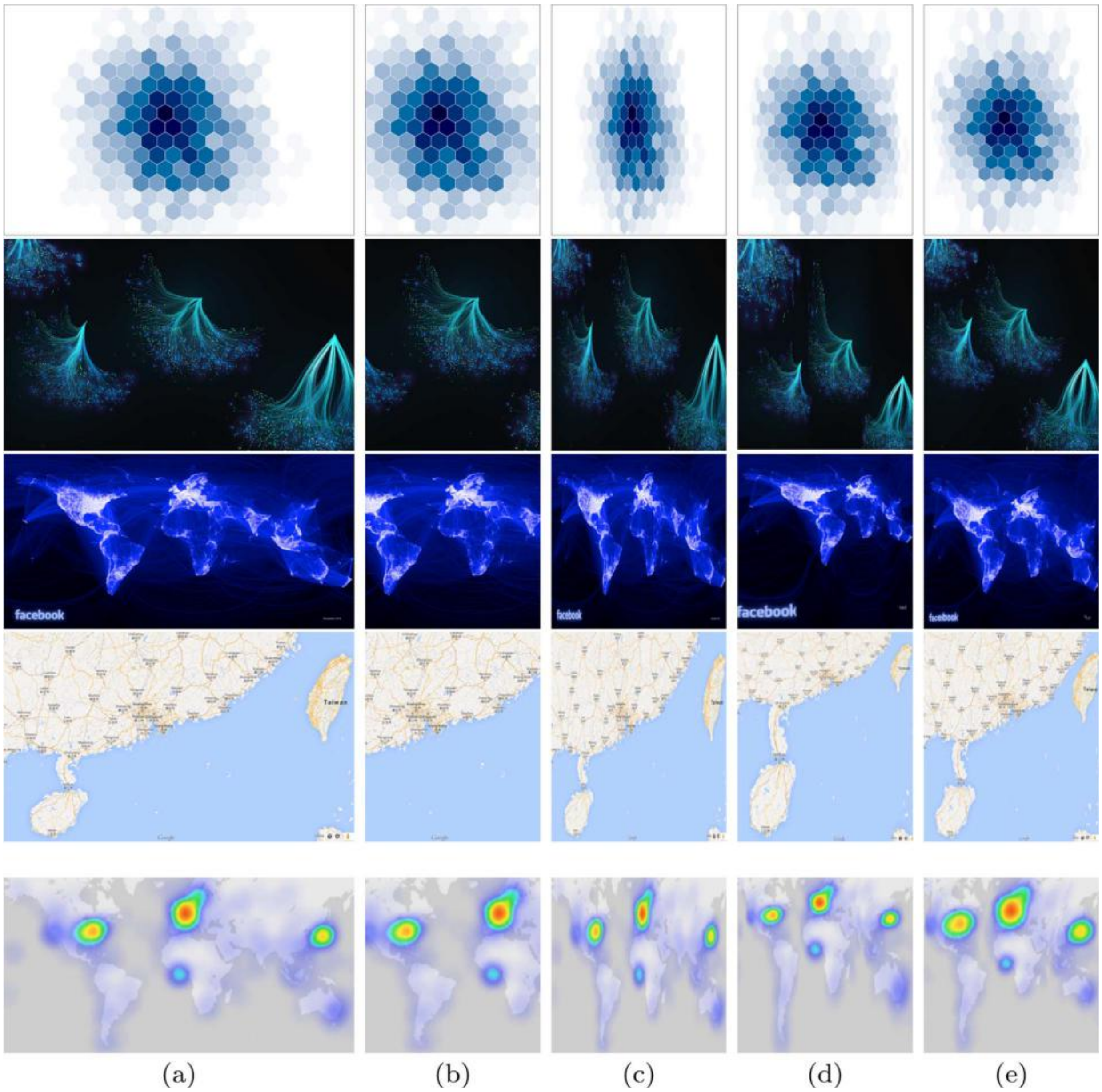



**Fig. 14.** Additional results of four different methods for reducing the width of the visualization. Fig. 15 shows the saliency preservation of each method using the bottom case in this figure. (a) Original visualization. (b) Cropping. (c) Linear resizing. (d) Grid-based resizing. (e) Ours.

density, can be avoided. Table 2 shows that our method is faster than using cluster map [21].

### 6.2.2. Evaluation

There are many methods to evaluate the effect of visualization resizing as shown in the cognitive experiments [4]. In this paper, we mainly focus on evaluating three attributes such as the number of vertices in the controlling mesh, the degree of freedom (DOF) and time cost. DOF indicates the number of variables that are free to change in the final computation. DOF is a statistics concept and is often used in evaluating the performance of retargeting, such as [20]. Because our model has advantages on controlling mesh reduction, DOF can be used in our evaluation. The results of the evaluation are shown in Table 1. From the evaluation, we find that our method has better performance, especially for large canvas size, and has less DOF than grid-based method.

Although image energy preservation methods [2] are normally used in pixel-based retargeting problems such as seam carving, they are also applicable to our method. The visual coherency can

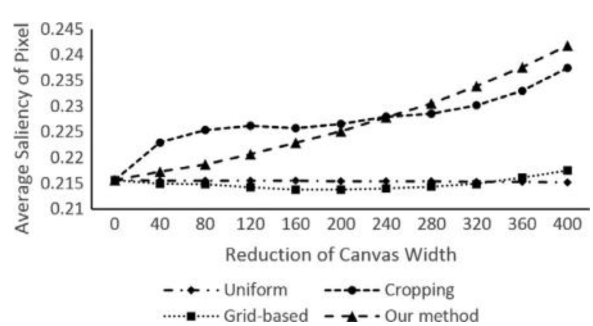



**Fig. 15.** A comparison of the preservation of content measured by the average pixel saliency using four different method of resizing. The testing visualization is the bottom case as shown in Fig. 14.

be evaluated quantitatively via calculating the percentage of energy preservation. We use the saliency of each pixel instead of the gradient to calculate the preserved energy, because our method is based on a VSM. The average saliency of pixels is shown in Fig. 15.

The figure shows the saliency preservation of four strategies and demonstrates that our method can effectively preserve the saliency content in the visualization while resizing the canvas.

## 7. Conclusion

In this paper we present an adaptive triangle-mesh based method for content-aware resizing of information visualizations. We propose a visual saliency detector that follows seven criteria. The detected visual saliency map (VSM) is not only used to generate adaptive meshes but also used to calculate the deformation factor of each triangle. A robust resizing energy function is defined to implement mesh resizing. The experiments show that our method can be used effectively in redesigning visualization features for different aspect ratio displays.

In the future, we plan to further improve our method to automatically classify features of visual representations into multiple layers, if the visualization does not have a clear multi-layered designation. The work of cartoon and texture decomposition [28] is a potential direction to abstract the structure of multiple layers in information visualization. Another potential direction is using a deep learning saliency map [29] to guide the mesh deformation. Furthermore, motion-aware image resizing [30] and visual tracking technique [31] can be adopted to implement a dynamic resizing of information visualization.

## Acknowledgment

The authors would like to acknowledge the partial support of the Hong Kong Research Grants Council Grants, GRF PolyU 5100/12E, IGRF PolyU 152142/15E and Project 4-ZZFF from the Department of Computing, The Hong Kong Polytechnic University.

## References

[1] Y.-S. Wang, C.-L. Tai, O. Sorkine, T.-Y. Lee, Optimized scale-and-stretch for image resizing, ACM Trans. Graph. 27 (5) (2008) 1–8.
[2] S. Avidan, A. Shamir, Seam carving for content-aware image resizing, ACM Trans. Graph. 26 (3) (2007) 10.
[3] Y. Wu, X. Liu, S. Liu, K.-L. Ma, ViSizer: a visualization resizing framework, IEEE Trans. Vis. Comput. Graph. 19 (2) (2012) 278–290.
[4] M. Rubinstein, D. Gutierrez, O. Sorkine, A. Shamir, A comparative study of image retargeting, ACM Trans. Graph. 29 (6) (2010) 1–10.
[5] M. Rubinstein, A. Shamir, S. Avidan, Improved seam carving for video retargeting, ACM Trans. Graph. 27 (3) (2008) 1–9.
[6] L. Xu, Q. Yan, Y. Xia, J. Jia, Structure extraction from texture via relative total variation, ACM Trans. Graph. 31 (6) (2012) 1–10.
[7] Y. Pritch, E. Kav-Venaki, S. Peleg, Shift-map image editing, in: Proceedings of the IEEE International Conference on Computer Vision, 2009, pp. 151–158.
[8] H. Wu, Y.-S. Wang, K.-C. Feng, T.-T. Wong, T.-Y. Lee, P.-A. Heng, Resizing by symmetry-summarization, ACM Trans. Graph. 29 (6) (2010) 1–10.
[9] R. Gal, O. Sorkine, D. Cohen-Or, Feature-aware texturing, in: Proceedings of the conference of the European Association for Computer Graphics, Eurographics Association, Nicosia, Cyprus, 2006, pp. 297–303.
[10] Y. Tu, H.-W. Shen, Balloon focus: a seamless multi-focus+context method for treemaps, IEEE Trans. Vis. Comput. Graph. 14 (6) (2008) 1157–1164.
[11] X. Liu, H.-W. Shen, Y. Hu, Supporting multifaceted viewing of word clouds with focus+context display, Inf. Vis. (2014) 1473871614534095.
[12] L. Itti, C. Koch, E. Niebur, A model of saliency-based visual attention for rapid scene analysis, IEEE Trans. Pattern Anal. Mach. Intell. 20 (11) (1998) 1254–1259.
[13] S. Frintrop, T. Werner, M. G. Germán, Traditional saliency reloaded: a good old model in new shape, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[14] X. Zhang, C. Xu, M. Li, R.K. Teng, Study of visual saliency detection via nonlocal anisotropic diffusion equation, Pattern Recognit. 48 (4) (2015) 1315–1327.
[15] R. Achanta, S. Hemami, F. Estrada, S. Sabine, D. L. Epfl, Frequency-tuned salient region detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1597–1604.
[16] S. Goferman, L. Zelnik-Manor, A. Tal, Context-aware saliency detection, IEEE Trans. Pattern Anal. Mach. Intell. 34 (10) (2012) 1915–1926.
[17] Q. Yan, L. Xu, J. Shi, J. Jia, Hierarchical saliency detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1155–1162.
[18] Y. Qin, H. Lu, Y. Xu, H. Wang, Saliency detection via cellular automata, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.
[19] O. Busaryev, T.K. Dey, H. Wang, Adaptive fracture simulation of multi-layered thin plates, ACM Trans. Graph. 32 (4) (2013) 1–6.
[20] P. Kaufmann, O. Wang, A. Sorkine-Hornung, O. Sorkine-Hornung, A. Smolic, M. Gross, Finite element image warping, Comput. Graph. Forum 32 (2–1) (2013) 31–39.
[21] R. Rosenholtz, Y. Li, J. Mansfield, Z. Jin, Feature congestion: a measure of display clutter, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '05, 2005, pp. 761–770.
[22] S. Lin, J. Fortuna, C. Kulkarni, M. Stone, J. Heer, Selecting semantically-resonant colors for data visualization, Comput. Graph. Forum 32 (2013) 401–410.
[23] J.R. Shewchuk, Triangle: engineering a 2D guality mesh generator and delaunay triangulator, Appl. Comput. Geom. Eng. (1996) 203–222.
[24] Y. Jin, L. Liu, Q. Wu, Nonhomogeneous scaling optimization for realtime image resizing, Vis. Comput. 26 (6–8) (2010) 769–778.
[25] J.-D. Boissonnat, O. Devillers, S. Pion, M. Teillaud, M. Yvinec, Triangulations in CGAL, Comput. Geom. 22 (1–3) (2002) 5–19.
[26] E. Cho, S.A. Myers, J. Leskovec, Friendship and mobility: user movement in location-based social networks, in: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, ACM, San Diego, California, USA, 2011, pp. 1082–1090.
[27] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech.-Theory E. 2008 (10) (2008) P10008.
[28] H. Yu, X. Chen, B. George, L. Min, I.M. Robiul, Cartoon and texture decomposition-based color transfer for fabric images, IEEE Trans. Multimed. 19 (1) (2017) 80–92.
[29] G. Li, Y. Yu, Visual saliency based on multiscale deep features, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5455–5463.
[30] Y.-S. Wang, H. Fu, O. Sorkine, T.-Y. Lee, H.-P. Seidel, Motion-aware temporal coherence for video resizing, ACM Trans. Graph. 28 (5) (2009) 1–10.
[31] C. Xu, W. Tao, Z. Meng, Z. Feng, Robust visual tracking via online multiple instance learning with fisher information, Pattern Recognit. 48 (12) (2015) 3917–3926.