A two-level clustering approach for multidimensional transfer function specification in volume visualization

Article *in* The Visual Computer · November 2015

DOI: 10.1007/s00371-015-1167-y

CITATION
1

READS
122

4 authors, including:

Binh P. Nguyen
National University of Singapore
29 PUBLICATIONS **114** CITATIONS

SEE PROFILE

Chee-Kong Chui
National University of Singapore
227 PUBLICATIONS **1,765** CITATIONS

SEE PROFILE

Sim Heng Ong
National University of Singapore
268 PUBLICATIONS **3,183** CITATIONS

SEE PROFILE

Vis Comput
DOI 10.1007/s00371-015-1167-y

ORIGINAL ARTICLE

A two-level clustering approach for multidimensional transfer function specification in volume visualization

Lile Cai[1] · **Binh P. Nguyen**[2] · **Chee-Kong Chui**[2] · **Sim-Heng Ong**[1]

Abstract Multidimensional transfer functions can perform more sophisticated classification of volumetric objects compared to 1-D transfer functions. However, visualizing and manipulating the transfer function space is non-intuitive when its dimension goes beyond 3-D, thus making user interaction difficult. In this paper, we propose to address the multidimensional transfer function design problem by taking a two-level clustering approach, where the first-level clustering by the self-organizing map (SOM) projects high-dimensional feature data to a 2-D topology preserving map, and the second-level clustering on the SOM neurons reduces the design freedom from a large number of SOM neurons to a manageable number of clusters. Based on the two-level clustering results, we propose a novel volume exploration scheme that provides top-down navigation to users exploring the volume. Guided by an informative volume overview, interesting structures in the volume are discovered interactively by the user selecting clusters to visualize and modifying the clustering results when necessary. Our interface keeps track of each interesting structure discovered, which not only enables users to inspect individual structures closely, but also allows them to compose the final visualization by fusing the structures deemed important.

Keywords Volume visualization · Volume exploration · Transfer function specification · Normalized cut · Self-organizing map

1 Introduction

Volume visualization aims to identify and visualize meaningful structures in a volumetric dataset. In direct volume rendering, this goal is achieved by a transfer function, which "transfers" voxel properties (e.g., intensity, gradient) to optical properties (e.g., color, opacity). The limitation of 1-D transfer function is widely recognized: volumetric objects typically comprise multiple materials, making them difficult to be differentiated by a single attribute. By taking into account more attributes, multidimensional transfer functions can perform more sophisticated classification of volumetric objects compared to 1-D transfer functions [19,42]. However, 3-D and higher-dimensional transfer function spaces are non-intuitive to visualize and manipulate.

In this work, we address the multidimensional transfer function design problem by a two-level clustering approach, where the first-level clustering serves to project the high-dimensional feature data onto a low-dimensional space, and the second-level clustering produces a manageable number of clusters that facilitates the detection and separation of meaningful volumetric objects. In the first-level clustering, the dimensional reduction ability and topology preserving nature

Electronic supplementary material The online version of this article (doi:10.1007/s00371-015-1167-y) contains supplementary material, which is available to authorized users.

✉ Lile Cai
cailile@u.nus.edu

Binh P. Nguyen
phubinh@ieee.org

Chee-Kong Chui
mpecck@nus.edu.sg

Sim-Heng Ong
eleongsh@nus.edu.sg

[1] Department of Electrical and Computer Engineering, National University of Singapore, Singapore, Singapore

[2] Department of Mechanical Engineering, National University of Singapore, Singapore, Singapore

Published online: 02 November 2015

of the self-organizing map (SOM) are exploited to transform the high-dimensional feature space to a 2-D embedding that is familiar and easy to understand to most users. Traditional transfer function design on a 2-D transfer function space is often achieved by direct manipulation of widgets in a bottom-up approach [20,30], where the user masks regions of interest in the transfer function space and the voxels falling into these regions are rendered accordingly. However, as the user can place a widget of any size at any position, the degree of design freedom is enormous, which aggravates the user's trial-and-error labor. Motivated by the need to provide reasonable constraints and guidance to the process of exploring the volume, we propose to perform a second-level clustering on the SOM neurons and to provide top-down navigation for exploration by an informative volume overview.

The main contribution of this paper is a two-level clustering approach to address the multidimensional transfer function design problem, which not only allows us to leverage on the discriminative power of high-dimensional feature vectors, but also enables effective user interaction and manipulation. Another contribution is an automatic method to generate an overview rendering of the volume. Based on the two-level clustering and volume overview, we propose a novel volume exploration scheme with top-down navigation, which reduces the trial-and-error efforts in a purely bottom-up approach.

2 Related work

2.1 Transfer function design

Transfer functions play a key role in volume visualization, as they map voxel properties to optical properties to make the volume "visible". Different volume visualization methods essentially differ in how they construct and manipulate the transfer function space.

2.1.1 1-D and 2-D transfer functions

A transfer function can be simply decided by the scalar values of the voxels. 1-D transfer function design methods usually focus on identifying important isovalues of the volume. Bajaj et al. [1] proposed the contour spectrum to assist the user in selecting relevant isovalues. Fujishiro et al. [9] identified the critical isosurfaces based on a 3-D field topology analysis of the volume. Weber et al. segmented a volume by decomposing the contour tree of the volume [45]. However, 1-D transfer functions have inherent difficulties in visualizing datasets where one data value is associated with multiple volumetric objects. To overcome the limitation of 1-D transfer functions, 2-D transfer functions were introduced. Some examples are the intensity-gradient magnitude (IGM) histogram [19], statistical transfer function space [12] and LH (Low-High) histogram [40]. More sophisticated features, such as curvature [13], feature size [7,46], and ambient occlusion [8], have also been used in the transfer function space.

Apart from investigating novel attributes to construct the transfer function space, another important direction is to improve the efficiency of manipulating this space. The traditional design method of line ramping for 1-D transfer functions and polygon drawing for 2-D transfer functions are often tedious and laborious. Pre-dividing the transfer function space into meaningful sub-regions can make the exploration process more efficient. Various clustering or segmentation techniques can be applied to divide the volume into a set of clusters that users can interact with. Šereda et al. [41] introduced a hierarchical clustering framework in LH space. Maciejewski et al. [25] divided the IGM feature space into several regions based on non-parametric kernel density estimation. Nguyen et al. [27] applied mean shift clustering to over-segment volume boundaries and then performed hierarchical clustering to group similar voxels. Wang et al. [43] trained a Gaussian mixture model on the IGM space and a transfer function was designed by modulating the resulting Gaussian ellipses. Cheuk et al. [15] applied the normalized cut algorithm to hierarchically segment the IGM histogram and introduced an information-theoretic measure to guide the exploration. The normalized cut algorithm was also employed in Cai et al.'s work for color transfer function specification [6]. Wang et al. [44] proposed to segment the IGM histogram using the theory of Morse complex. Hsieh et al. [14] employed particle swarm optimization techniques for feature extraction in transfer function design.

2.1.2 Higher-dimensional transfer functions

The discriminative power of a transfer function can be further improved by increasing its dimension beyond two. However, 3-D and higher-dimensional transfer function spaces are difficult to visualize, making the design of an effective user interface all but impossible. To address this problem, the community has developed various methods, including those based on cluster space, machine learning, parallel coordinate plot (PCP) and dimensional reduction.

Cluster space-based methods perform classification in high-dimensional feature space and transformed the volumetric data into a cluster space representation that can be manipulated by the user [5,24,35,42]. Machine learning-based methods take advantage of some well-established machine learning algorithms to "learn" a classification function and compute a voxel's likelihood of belonging to the material of interest. The uncertainty value is then mapped to the opacity value during rendering. Neural networks and support vector machines [34] are examples of these learn-

ing algorithms. PCP represents a high-dimensional point by a polyline passing through the coordinate axis of each dimension. It treats each dimension uniformly and can reveal the relationship and correlation of neighboring coordinates. The transfer function can be designed on the PCP by axis brushing, which selects a certain range in one dimension, and angular brushing, which selects line segments with similar slopes between two neighboring axes [11,48]. Dimension reduction techniques have been widely used in high-dimensional transfer function design. These techniques automatically project the high-dimensional data to a 2-D or 3-D space, so that transfer function design techniques for low-dimensional space can be applied. Kim et al. [18] employed the isomap, locally linear embedding (LLE) and principal component analysis (PCA) to perform dimensional reduction for multi-channel volumes. Zhao and Kaufman [48] applied LLE coupled with PCP in their transfer function design framework.

2.2 Self-organizing map

The self-organizing map [22] is an unsupervised learning algorithm that projects data points onto a lower-dimensional space (usually 2-D space) while preserving the original topological relations. The map is initialized randomly, but after training, the map can "self-organize" to represent the input data distribution. This property makes it a popular tool for data mining. It has been used for data visualization, clustering, classification and vector quantization [29].

2.2.1 Cluster detection on the SOM

Clusters on the SOM can be detected by applying well-established clustering algorithms, such as hierarchical agglomerative clustering and k-means [17,39,47], to the SOM weight vectors. Detecting clusters on a SOM can also be conducted on the visualization of the map instead of the weight vectors. The most common method to visualize a SOM is via the U-matrix [36], while more sophisticated methods, such as the P-matrix [37], U*-matrix [38] and Clusot surface [3], have also been proposed. Opolon and Moutarde [28] applied a region growing algorithm to the U-matrix visualization of the SOM for cluster detection. The approach was extended to operate on the U*-matrix for better separability in low density regions [26]. In [4], a recursive flooding algorithm was applied to detect clusters on the Clusot surface.

2.2.2 SOM in transfer function design

The SOM has also been used for high-dimensional transfer function design. The initial work was proposed by Pinto and Freitas [30], where users were asked to draw Gaussian circles on the 2-D visualization map and a transfer function was obtained by composing the user-specified circles. Khan et al. [16] employed a spherical SOM to train the volumetric data and a transfer function is obtained by direct selection of SOM neurons.

3 System overview

An overview of our method is presented in Fig. 1. In the preprocessing step, a simple region growing algorithm is applied to remove background voxels. This helps to prevent the SOM from being occupied by high-occurrence background voxels as well as to reduce the computational cost of the following processing steps. A multidimensional feature vector is extracted for each non-background voxel. We take into consideration a total of eight features in our experiment, which are intensity, gradient, second order directive along gradient direction [21] and five first-order textural features that are computed from a local histogram (i.e., mean, standard deviation, entropy, skewness and fourth order moment) [2]. A two-level clustering is then performed on the extracted feature vectors. First-level clustering is conducted by the SOM,

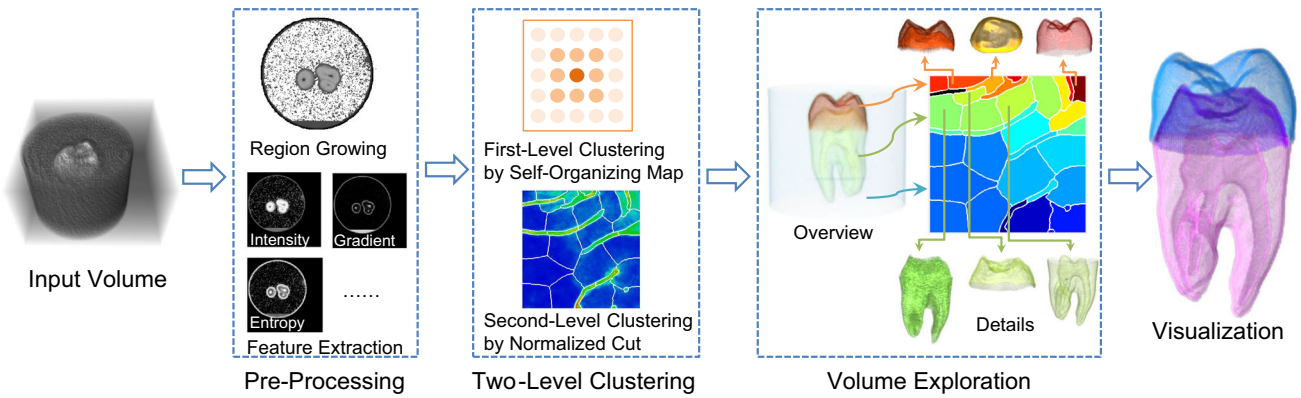


Fig. 1 System diagram of the proposed method

which projects high-dimensional feature vectors to a 2-D topology preserving embedding. This not only allows us to leverage on the discriminative power of high-dimensional feature descriptors, but also converts the volume data into a compact and easy-to-understand 2-D form. Second-level clustering is conducted by the normalized cut algorithm [31], which over-segments the 2-D SOM into semantically related sub-regions. Each of these sub-regions corresponds to a cluster that potentially represents a meaningful structure in the volume.

In the volume exploration stage, an overview of the volume is automatically generated to provide the user a quick and informative look into the volume. The overview also serves to guide the user in inspecting the clusters; if the user wishes to inspect a particular structure shown in the overview, he can find the clusters in the corresponding colored region of the map, thus avoiding an exhaustive search of the transfer function space. A final visualization of the volume is composed by combining the structures deemed important by the user.

4 Method

In this section, we describe in details the two-level clustering approach and the overview-guided exploration scheme.

4.1 First-level clustering

The dimension reduction ability and topology preserving nature of the SOM make it a suitable tool to represent the volume data in 2-D space. While other dimension reduction techniques, such as PCA [18], LLE [18,48], MDS [11], have also been applied in transfer function design, they do not lead to a compact semantic map as SOM does.

We employ a 2-D rectangular map to facilitate the design of a transfer function interface that is familiar for most users. The map is built through an iterative unsupervised learning process, comprising the following steps:

1. Initialization: Initialize the weight vectors of the SOM neurons. Random initialization or other more sophisticated initialization schemes [33] can be used for this step.
2. Sampling: Randomly choose an input vector x from the training set.
3. Best matching unit (BMU) selection: Calculate the Euclidean distances between the input vector and the weight vectors of all neurons. BMU or the winner neuron is the neuron for which the Euclidean distance is smallest, i.e.,

$$\mathrm{BMU} = \arg \min_{i \in [1,\ldots,n]} ||\mathbf{x} - \mathbf{w}_i(t)||, \tag{1}$$

where $\mathbf{w}_i(t)$ is the weight vector of neuron i at time t and n is the number of neurons.

4. Weight update: Update the weights of BMU and other neurons with the following equation:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \eta(t) h_{i,\mathrm{BMU}}(t)(\mathbf{x} - \mathbf{w}_i(t)), \tag{2}$$

where

$$\eta(t) = \eta_0 \exp\left(-\frac{t}{T}\right), \tag{3}$$

$$h_{i,\mathrm{BMU}}(t) = \exp\left(-\frac{d_{i,\mathrm{BMU}}^2}{2\sigma^2(t)}\right), \tag{4}$$

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\tau}\right), \tag{5}$$

$$\tau = \frac{T}{\log \sigma_0}. \tag{6}$$

Here, $\eta(t)$ is the time-varying learning rate, T is the total number of learning epochs, $h_{i,\mathrm{BMU}}$ is the time-varying neighborhood function and $d_{i,\mathrm{BMU}}$ is the Euclidean distance between neuron i and the winner neuron. Both $\eta(t)$ and $\sigma(t)$ are decreasing monotonically with time, which are crucial for the map to converge.

5. If the stopping criterion is met, stop; otherwise, go to 2.

The weight updating rule defined by Eq. (2) implies that the winner neuron updates not only itself, but also its neighborhood neurons. This is a feature of SOM, which was inspired by neurobiological evidence for lateral interactions, where a neuron that is firing tends to excite neurons in its intermediate neighborhood more than those that are farther away. This neighborhood cooperation allows the weight vectors of the map to follow the distribution of the input space and leads to a topological ordering in the sense that neurons that are adjacent in the lattice will tend to have similar weight vectors. The topology preserving nature of SOM results in a semantic map where neurons are spatially organized in a meaningful way, and thus facilitates the generation of semantically related clusters.

4.2 Second-level clustering

4.2.1 SOM neurons clustering by normalized cut

The number of neurons contained in the trained SOM is much larger than the number of meaningful structures or sub-structures contained in the volume. Typically, each neuron represents a small portion of a meaningful structure and a group of neighboring neurons combine to represent a complete structure. This motivates us to conduct a second-level clustering on the SOM neurons so that users can interact with

a manageable number of clusters instead of a large number of SOM neurons.

We adopted the normalized cut as our second-level clustering algorithm. Compared to other clustering or segmentation algorithms such as k-means and the watershed algorithm, the normalized cut is more sophisticated and can produce more robust results. The normalized cut represents the set of points to be clustered as a weighted undirected graph $G = (V, E)$, where vertices V are the input points, and an edge is formed between each pair of nodes. The edge connecting nodes i and j are associated with weight $w(i, j)$, which is a function of the similarity between the two nodes. The normalized cut algorithm finds a cut to bipartition the graph into two disjoints sets A and B by minimizing the following disassociation measure:

$$\text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}, \tag{7}$$

where $\text{cut}(A, B) = \sum_{u \in A, v \in B} w(u, v)$, $\text{assoc}(A, V) = \sum_{u \in A, t \in V} w(u, t)$ and $\text{assoc}(B, V) = \sum_{u \in B, t \in V} w(u, t)$.

Minimizing Eq. (7) exactly is NP-complete. By relaxing the problem in the real value domain, an approximate solution can be obtained by solving a generalized eigenvalue problem:

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda \mathbf{D}\mathbf{y}, \tag{8}$$

where $\mathbf{W}$ is the affinity matrix of the graph and $\mathbf{D}$ is a diagonal matrix with each element being the sum of elements of the corresponding row in $\mathbf{W}$. The eigenvector $\mathbf{y}$ with second smallest eigenvalue is then quantized to bipartition the graph and the resulting segments can be recursively sub-divided to create more segments if necessary.

We regard the clustering of SOM neurons as an image segmentation problem, where the SOM is first represented by a U-matrix [37] and then fed into the normalized cut algorithm. The U-matrix has the same size as the SOM, with each element storing the average distance between a neuron and its connected neighbors. On the U-matrix, regions with similar neurons appear as "valleys" and the boundaries between two "valleys" appear as "ridges". A cluster by definition is a group of similar neurons, thus corresponding to "valleys", while cluster boundaries correspond to "ridges". The normalized cut can facilitate the detection of clusters on the SOM by segmenting "valleys" from "ridges".

We apply the normalized cut to over-segment the SOM into a small number of segments, which ensures the fine granularity of volume exploration given that the actual number of structures contained in the volume is unknown. Figure 2a shows the over-segmentation results on the Tooth dataset. It can be seen that the normalized cut is effective in separating the "valleys" and "ridges". Due to the over-segmentation, the structure represented by one cluster may be incomplete. The topology preserving nature of SOM implies that neighboring clusters usually represent similar structures. This spatial relationship between clusters provides a useful guideline for users exploring the map, which is that neighboring clusters can be combined to form a complete structure. An example is shown in Fig. 2b.

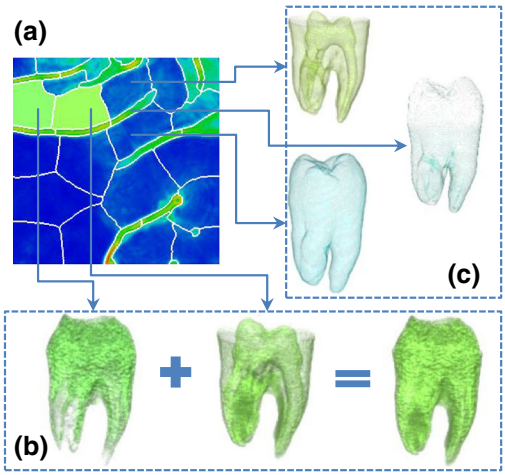


Fig. 2 Second-level clustering on the tooth dataset by the normalized cut. The U-matrix is mapped to a hot-to-cold color scale, where *hot colors* represent large values. The segment boundaries are denoted by *white lines*. **a** The over-segmentation results on the U-matrix of the SOM; **b** Neighboring clusters represent similar structures due to the topology preserving nature of the SOM and can be combined to form a complete structure; **c** "Valleys" represent clusters while "ridges" represent cluster boundaries

4.2.2 Voxel membership computation

After grouping the SOM neurons into a small number of clusters, voxels are assigned to the clusters using a weighted Euclidean distance approach. First, the center of each cluster is computed by averaging the neuron vectors within each cluster:

$$\mathbf{c}_i = \sum_{j \in \text{cluster}_i} \mathbf{v}_j \frac{h_j}{s_i}, \tag{9}$$

where $\mathbf{c}_i$ is the center of cluster i, $\mathbf{v}_j$ is the weight vector of neuron j, h_j is the number of training samples whose BMU is neuron j, and $s_i = \sum_{j \in \text{cluster}_i} h_j$ is the total number of training samples in cluster i. Voxel p is assigned to the cluster for which the weighted Euclidean distance is minimized:

$$idx_p = \arg \min_{i \in [1, \ldots, n]} \frac{||\mathbf{f}_p - \mathbf{c}_i||}{\text{weight}_i}, \tag{10}$$

where idx_p is the cluster index of voxel p, $\mathbf{f}_p$ is the feature vector of voxel p, n is the total number of clusters, and

weight_i is a positive weighted value for cluster i, which can be interactively set by the user.

It has been shown that the normalized cut can effectively separate "valleys" and "ridges". "Valleys" represent clusters and "ridges" represent cluster boundaries. Strictly, we should exclude "ridges" from the assignment of voxels, but this will incur the extra step of quantitatively distinguishing between "valleys" and "ridges". We thus include all the segments in voxel membership computation and assign a voxel to its nearest neighbor among these segments. Figure 2c shows three clustering results on the Tooth dataset. It can be seen that the two "valley" clusters represent meaningful structures (the light green tooth root and the aqua tooth shell), while the "ridge" cluster separating the two "valley" clusters only contains a small fraction of voxels that do not form a meaningful structure. This provides another useful guideline for the user exploring the map, which is that priority should be accorded to "valley" clusters when selecting clusters to visualize.

4.2.3 Cluster modification by weight adjustment

While the clusters generated by second-level clustering usually represent meaningful and well-separated structures, it is still desirable to offer a way for the user to modify the clusters when he or she is not satisfied with the automatically generated results. As voxels are assigned to the clusters according to the weighted Euclidean distance, weight adjustment serves as an effective manner to change voxel membership. We propose a cluster modification algorithm, summarized in Algorithm 1, to realize this function.

In Algorithm 1, the modification procedure takes place between the cluster whose weight is changed and the clusters which are selected by the user to take part in the modification procedure. These clusters are usually neighboring clusters as their centers are in close proximity and a change in weights will effectively change the voxel membership. To improve the efficiency of the algorithm, weight increasing and decreasing cases are processed separately.

The weight adjustment procedure is illustrated by an example in Fig. 3. Under the default weight values (1.0 for all clusters), the purple structure contains some voxels from the pink structure. As boundaries are important for visualization, the user may wish to remove the noisy voxels to obtain a clean boundary. To achieve this, the user can decrease the weight value of the purple structure so that the distances between the noisy voxels and the purple cluster are increased to the extent that the purple cluster is no longer their nearest neighbor among the three affected clusters. As a result, the noisy voxels in the purple structure are gradually reallocated to the pink structure. Once the purple structure becomes clean or the user is satisfied with the modified results, he or she can stop decreasing the weight value.

Algorithm 1 Cluster modification by weight adjustment

input
$cluster_{chg}$: the index of the cluster whose weight is changed
$cluster_{sel}$: the indexes of the clusters selected by the user to take part in the modification
n_{sel}: the number of clusters in $cluster_{sel}$
$weight$: the array that stores cluster weight values
$index$: the array that stores the cluster index for each voxel
output
updated $index$
functions
distance(v, s): compute the weighted Euclidean distance between voxel v and cluster s
procedure
  case $weight(cluster_{chg}) \uparrow$
  for each voxel p belonging to $cluster_{sel}$ **do**
    $dist_{new}$ = distance(p, $cluster_{chg}$)
    $dist_{old}$ = distance(p, $index(p)$)
    if $dist_{new} < dist_{old}$ **then**
      $index(p) = cluster_{chg}$
    end if
  end for
  case $weight(cluster_{chg}) \downarrow$
  for each voxel p belonging to $cluster_{chg}$ **do**
    $dist_{min}$ = distance(p, $cluster_{chg}$)
    $idx_{min} = cluster_{chg}$
    for $i = 1$ to n_{sel} **do**
      $dist$ = distance(p, $cluster_{sel}(i)$)
      if $dist < dist_{min}$ **then**
        $dist_{min} = dist$
        $idx_{min} = cluster_{sel}(i)$
      end if
    end for
    $index(p) = idx_{min}$
  end for
end procedure

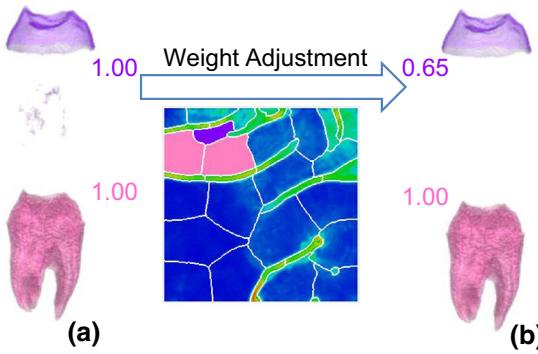


Fig. 3 Cluster modification by weight adjustment. **a** When the weight values are the same for the *purple cluster* and the *pink clusters*, the *purple* structure contains some noisy voxels from the *pink* structure. **b** By decreasing the weight value of the *purple cluster* from 1 to 0.65, a *clean purple* structure can be obtained

4.3 Overview generation

To give the user a better idea of what is to be explored and what remains to be explored in the volume, we generate an

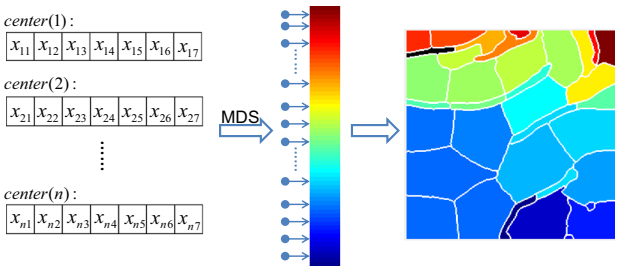


Fig. 4 The proposed color assignment scheme for overview generation

overview of the volume by rendering all the clusters with an automatic color and opacity assignment scheme.

Our coloring scheme is illustrated in Fig. 4. The centers of the clusters as defined in Eq. (9) are utilized to determine the color assignment. This is motivated by the fact that these cluster centers essentially reflect the similarity between the clusters, and a color mapping based on them will be able to differentiate dissimilar structures. Multidimensional scaling (MDS) is performed on the cluster centers to convert them to 1-D points. Each of these 1-D points determines a position on a color scale. We employ the rainbow color scale, which takes into consideration hue and lightness to maximize its discriminating ability. The cluster is then assigned the corresponding color at the projected position. Under such a coloring scheme, clusters that are close in the feature space are assigned similar colors and those that are far away are assigned distinct colors.

Our opacity assignment scheme utilizes the two fundamental properties of the volume data, intensity and gradient, to emphasize high-intensity and high-gradient regions, which usually correspond to important structures. Letting $\mathrm{Int}(i)$ and $\mathrm{Gm}(i)$ be the average intensity and gradient magnitude of voxels in cluster i, the opacity for cluster i, $o(i)$, is obtained by:

$$o(i) = \alpha \cdot \mathrm{Int}(i) \cdot \mathrm{Gm}(i) + \beta, \tag{11}$$

where $\mathrm{Int}(i) \cdot \mathrm{Gm}(i)$ is normalized to the range $[0, 1]$, and α and β are parameters controlling the contrast and opaqueness of the overview. The influence of α and β on the overview is shown in Fig. 5. It can be seen that different α and β values can create different visual effects that emphasize or suppress different structures in the volume. To allow users to obtain as much information as possible from the overview, the adjustment of α and β is implemented as sliders in our interface, with the range of α set to $[0, 0.5]$ and that of β set to $[0, 0.1]$.

4.4 Volume exploration

Based on the two-level clustering results and the volume overview, we propose a volume exploration scheme with top-down navigation. In our interface, the clusters are shown in their default color on the segmented SOM. The volumetric objects in the overview are also rendered in their default color, and users can thus locate the corresponding clusters for a particular structure from the color, thus avoiding an exhaustive search of the entire map. At the beginning of the exploration, the user adjusts the α and β sliders to obtain a quick and informative view of the volume. When the user wants to inspect a particular structure shown in the overview, he or she can just select the clusters in the corresponding colored region to look for it. The two exploration guidelines discussed in Sect. 4.2, i.e., (a) neighboring clusters can be combined to form a complete structure, and (b) priority should be accorded to "valley" clusters when selecting clusters to visualize, will be helpful for the user when exploring interesting structures. Once a meaningful structure is found, the user can save it and move on to explore other structures. Finally, the user creates a visualization of the volume by combining the structures deemed important.

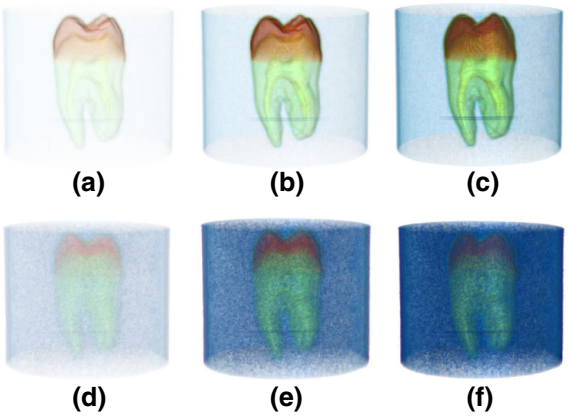


Fig. 5 Overviews rendered with different α and β values. *Top row*: fix $\beta = 0$, vary α. **a** $\alpha = 0.05$; **b** $\alpha = 0.2$; **c** $\alpha = 0.5$. *Bottom row*: fix $\alpha = 0$, vary β. **d** $\beta = 0.01$; **e** $\beta = 0.05$; **f** $\beta = 0.1$

The proposed exploration scheme is illustrated by an example in Fig. 6. In Fig. 6a, the overview reveals that the volume contains three major structures: the orange red crown, the lawn green root and the light blue container. The container is usually of little interest to the user, so it is removed from the overview by deselecting the blue clusters on the segmented SOM (Fig. 6b). To study the details of the crown, the user selects the clusters at the top of the segmented SOM for closer inspection, which will reveal the sub-structures composing the crown. Similarly, to study the details of the root, the user selects the light green clusters, which will allow the user to gain a deeper understanding of the internal structures of the root (Fig. 6c). After exploring the volume, the user can create a final visualization by fusing interesting structures and modifying the optical properties accordingly (Fig. 6d).

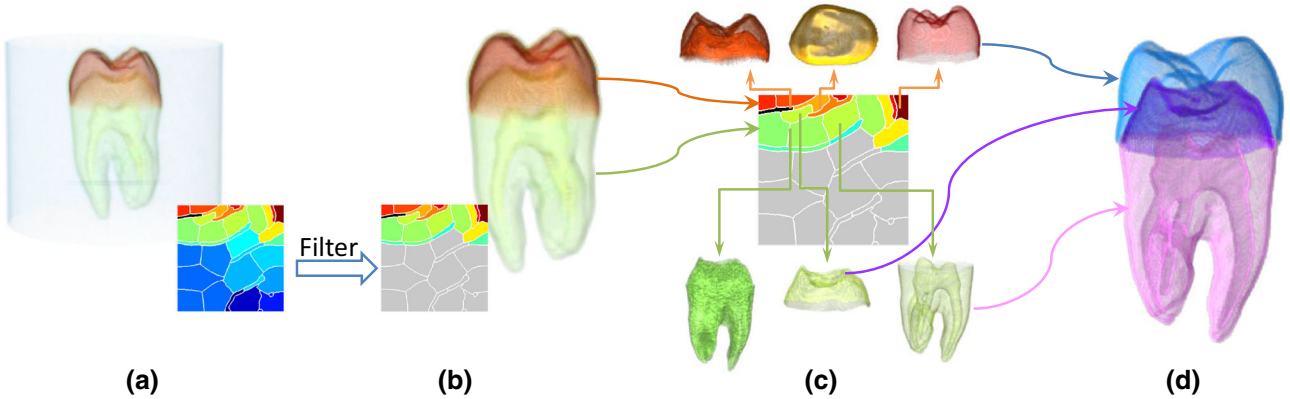


Fig. 6 The proposed overview-guided exploration scheme. **a** Automatically generated volume overview. **b** Deselect unimportant clusters (colored in *gray*) from the overview to avoid visual clutter. **c** Cluster inspection guided by the overview. **d** Volume visualization by fusing interesting structures

5 Results

We applied our method to a variety of volumetric datasets. For each dataset, we show the automatically generated volume overview along with the default colored clusters arranged on the segmented SOM to demonstrate how the overview can guide the exploration process. We also present the volume structures that can be discovered when the user explores the volume, with the corresponding clusters displayed in the same color of the structure that they represent. A final visualization of the volume, which is generated by combining interesting structures, is also provided. The colors of the chosen structures (indicated by red boxes) may have been modified to achieve more visually pleasing results. Readers are encouraged to watch our video demo in the supplementary material to better understand the interactivity of our method.

Tooth (16-bit CT): The tooth dataset is visualized in Fig. 7. From the overview, we observe that the volume contains three major structures: the orange red crown, the lawn green root and the light blue container, corresponding to the top, top-middle and middle-bottom of the SOM. By inspecting the corresponding clusters, we see the detailed structures such as enamel, the inner and outer boundaries of the enamel, the dentine, the upper and lower halves of the dentine boundary, the pulp, etc. We compose a visualization by combining the enamel, the inner and outer boundaries of the enamel, the pulp surface and a thin shell encompassing the tooth, with the color and opacity properly modulated to achieve better visual effects. Note that a different visualization of the tooth is composed in Fig. 7 compared to Fig. 6, which serves to demonstrate that our method allows users to easily create various visualizations of the same dataset based on their particular interests. The SOM was trained with seven features, namely intensity, gradient magnitude, second order derivative along the gradient direction and four first-order

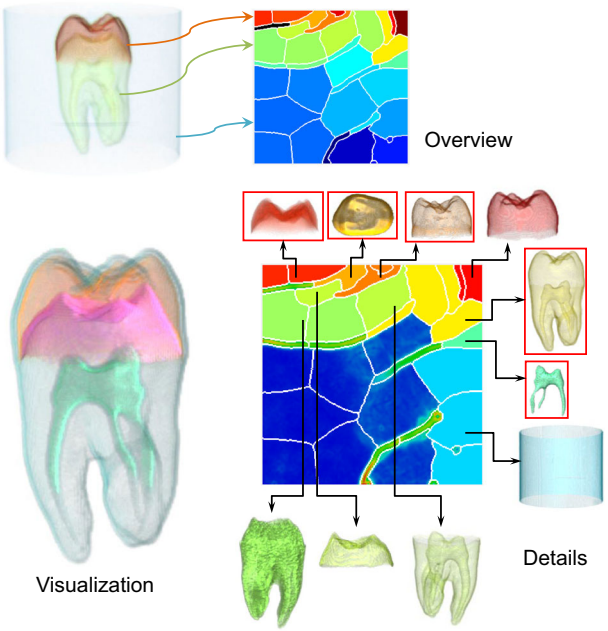


Fig. 7 Visualization of the tooth dataset

textural features (mean, standard deviation, skewness and entropy).

Engine (8-bit CT): Figure 8 shows the visualization of the Engine dataset. The overview reveals that the dataset contains an engine block, with the internal structures highlighted in blue. By checking the corresponding colored clusters, we can view the internal structure at the blue region of the SOM, the engine block body at the aqua region, the engine block surface at the orange region, as well as a thin surface of the internal structure at the yellow region. The final visualization is composed by combining the internal structure, the internal structure surface and the engine block surface. The features used to trained the SOM are the same as those for the Tooth dataset.

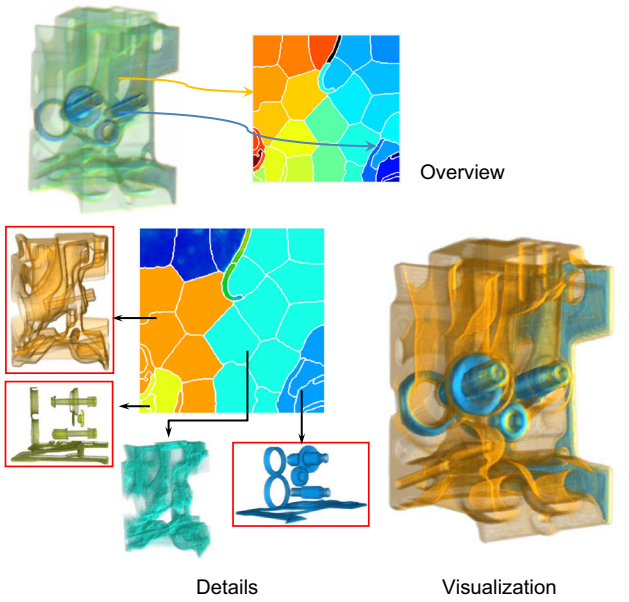


Fig. 8 Visualization of the engine dataset

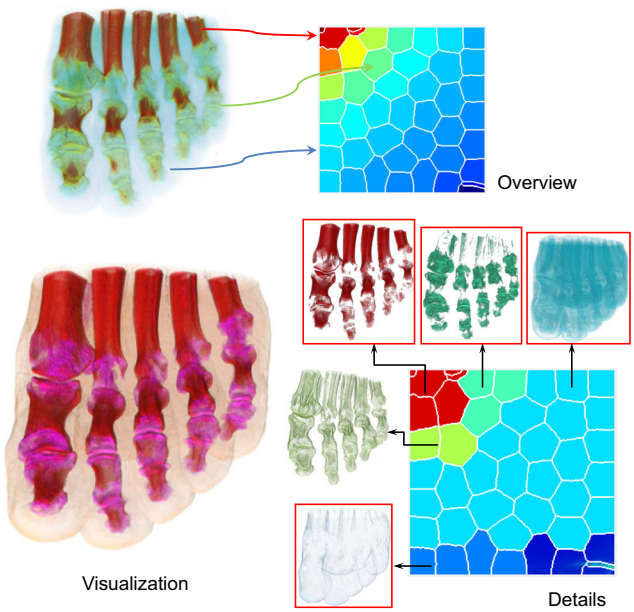


Fig. 9 Visualization of the foot dataset

Foot (8-bit CT): The foot dataset is visualized in Fig. 9. The overview shows three major structures in the dataset: the bone in red, the joint in green and the flesh in blue. Guided by the overview, we obtain the bone, the joint and the boundaries of the hard tissues at the top-left corner of the SOM. The flesh occupies a large portion of the map due to its high occurrence in the volume. The skin portion is located at the bottom of the map. The bone, joint, flesh and skin are combined to create the final visualization. Three features, including intensity, standard deviation and entropy, were used to train the map.

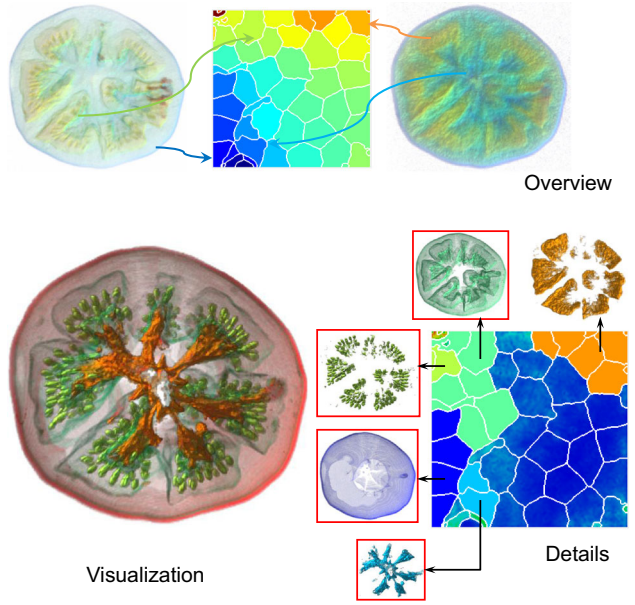


Fig. 10 Visualization of the tomato dataset

Tomato (8-bit MRI): We demonstrate our method on an MRI dataset in Fig. 10. Two overviews created with different α and β values ($\alpha = 0.075, \beta = 0$ for the left view and $\alpha = 0, \beta = 0.02$ for the right) are shown in Fig. 10 to illustrate the different information that can be obtained by varying α and β. The left overview informs us of the presence of the seed, valve and peel, while the right one emphasizes the orange pulp and the blue columella. With the guidance from the two overviews, we can go on to obtain the detailed structures at the corresponding colored regions of the map. The final visualization is created by combining the peel, the seeds, the valve and the columella. The features used for this dataset are gradient, mean, standard deviation, skew and fourth order moment.

Carp (16-bit CT): The visualization result of the carp dataset is displayed in Fig. 11. The overview presents an informative look at the dataset, which consists of the skeleton (in orange), skin and air bladder (in blue) and a thin surface of the skeleton and air bladder (in yellow). We construct the final visualization by combining the skeleton, skin and air bladder. The map was trained with three features: intensity, gradient magnitude and mean.

Visible Female Feet (16-bit CT): Figure 12 presents the test result on the visible female feet dataset. As shown in the overview, the dataset consists of the bone (in orange), joints (in yellow green), skin (in blue) and a plate (in deep blue). Four meaningful structures are provided for individual inspection and the final visualization comprises the bone, joints and skin. We used three features, intensity, mean and standard deviation, to train the SOM for this dataset.

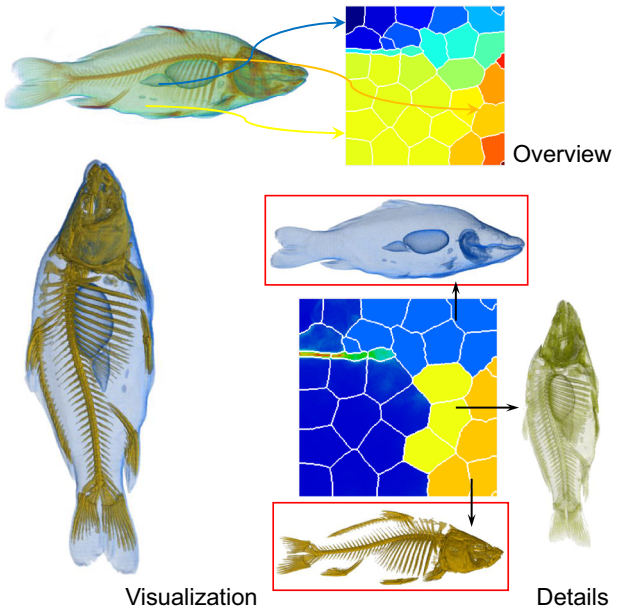


Fig. 11 Visualization of the carp dataset

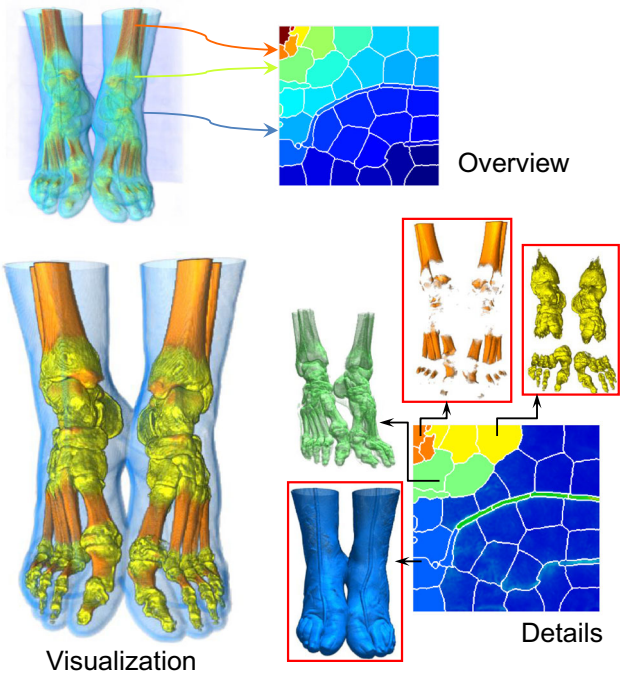


Fig. 12 Visualization of the visible female feet dataset

Pig (16-bit CT): This dataset was obtained from an experiment on a live pig to test a device for secondary tracheoesophageal puncture [23]. The visualization result is presented in Fig. 13. The overview reveals that the dataset consists of the yellow green bone, the blue lung, the aqua flesh and the red device in the mouth. Due to the relatively small number of voxels comprising the mouth device, this structure was represented by a very small region on the map

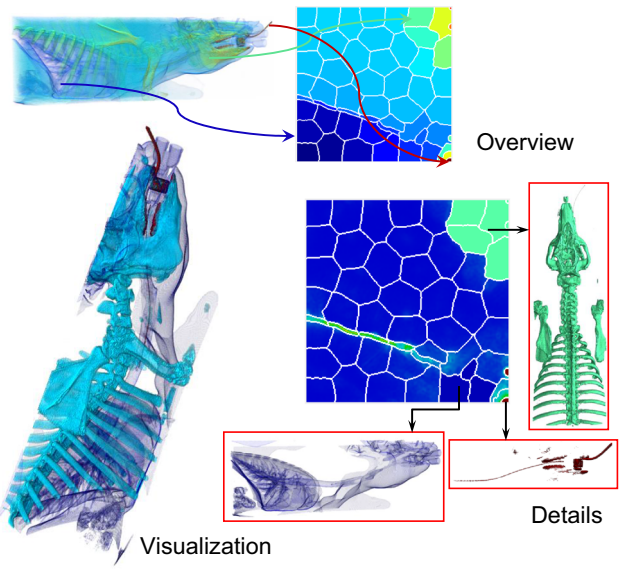


Fig. 13 Visualization of the pig dataset

and would be difficult to discover by trial-and-error methods. With our approach, users can easily locate the clusters corresponding to the device. The final visualization comprises the bone, lung and mouth device. Four features (intensity, gradient, mean and fourth order moment) were employed to train the SOM for this dataset.

The parameters of the evaluation datasets and the time needed for each major operation in our system are listed in Tables 1 and 2, respectively. Our method was implemented on a PC with an Intel i7-2600 CPU and an Nvidia GTX680 graphics card. The code provided by Shi and Malik [32] was used for the normalized cut segmentation. All the operations listed in Table 2 need to be done only once for a given dataset and the transfer function design is conducted at interactive frame rates.

As can be seen from Table 2, SOM training is the most time-consuming step in our method. Its time complexity is $O(m^2 d_f n_s T)$, where m is the map width (and height), d_f the feature dimension, n_s the number of training samples and T the number of training epochs. The size of SOM in our experiments was set to be 256×256. To reduce the time cost for SOM training, two measures were taken:

(a) The SOM algorithm was implemented in CUDA C to utilize the parallel computing power of Nvidia GPU. The sequential learning algorithm proposed by Gajdoš and Platoš [10] was implemented for SOM training, where each dimension of the weight vector was processed by one thread.
(b) Since the number of voxels is huge, it is unnecessary to use all of them for training. A subset of the voxels is sufficient to represent the volume. In our work, 1 % of the total voxels were randomly sampled to train the SOM.

Table 1 Parameters of the evaluation datasets

Dataset	Resolution	No. of non-background voxels	No. of features	No. of clusters
Tooth	256 × 256 × 161	1971836	7	35
Engine	256 × 256 × 256	1731738	7	35
Foot	256 × 256 × 256	5066965	3	45
Tomato	256 × 256 × 64	1733102	5	55
Carp	256 × 256 × 512	6322815	3	35
Visible female feet	512 × 512 × 256	5275796	3	45
Pig	256 × 256 × 584	28217983	4	55

Table 2 Time costs of the evaluation datasets (in seconds)

Dataset	Region growing	Feature extraction	SOM training	Normalized cut	Voxel assignment
Tooth	8.9	3.6	49.6	12.5	2.3
Engine	13.9	3.2	43.8	12.9	2.0
Foot	10.4	8.2	74.8	12.2	5.7
Tomato	2.4	2.9	34.1	16.5	2.0
Carp	24.3	10.8	93.1	12.0	7.3
Visible female feet	58.8	8.6	77.8	12.3	6.0
Pig	10.7	43.8	482	14.2	32.8

The training of SOM consists of a self-organizing or ordering phase to construct the map and a convergence phase to fine-tune the map. In our application, the convergence of the neuron weights is not critical, hence the second phase is not needed. For the first phase, we found that 10 training epochs were sufficient for the input samples to self-organize themselves into a reasonably good map.

6 Discussion

6.1 Parameter choices

To ensure the effectiveness of volume exploration and visualization, the clusters proposed by the two-level clustering should represent well-separated and meaningful volumetric objects. The parameters that most greatly affect the quality of the resulting clusters are the features used for SOM training and the number of clusters in the second-level clustering.

6.1.1 Feature selection

In our experiment, eight candidate features, which are intensity, gradient, second order directive along gradient direction and five first-order textural features (i.e., mean, standard deviation, entropy, skewness and fourth order moment), are considered in the feature selection process. Different feature combinations can produce different results. We find that for some datasets (e.g., the Tooth dataset), using more features in the training of the SOM can result in clusters with better separability, while for other datasets (e.g., the Carp dataset), selecting more features for SOM training does not bring any improvement, probably due to the fact that the additional selected features are redundant or noisy. As the number of candidate features is small, we select the features empirically by visualizing the feature values on 2-D slices, and from there deciding whether to select a feature by observing how well it can discriminate different structures. However, manual feature selection is undoubtedly subjective and inefficient, especially when more candidate features are taken into consideration. A quantitative analysis of the feature's separability will assist the user in the feature selection process and facilitate the design of an automatic feature selection algorithm.

Though an efficient algorithm to determine the optimal feature combination is lacking in the current work, the robustness of the two-level clustering approach serves to remedy this problem to some extent, as we find that the results are not very sensitive to the selected features. This is illustrated by the examples in Fig. 14a, b. The results shown in Fig. 14a are trained with 7 features, which are intensity, gradient magnitude, second order derivative along the gradient direction and four first-order textural features (mean, standard deviation, skewness and entropy), while those shown in Fig. 14b are trained with only intensity and its derivatives (gradient magnitude and second order derivative along the gradient direction), excluding the textural features. Though the quality of some structures shown in Fig. 14b is degraded (e.g.,

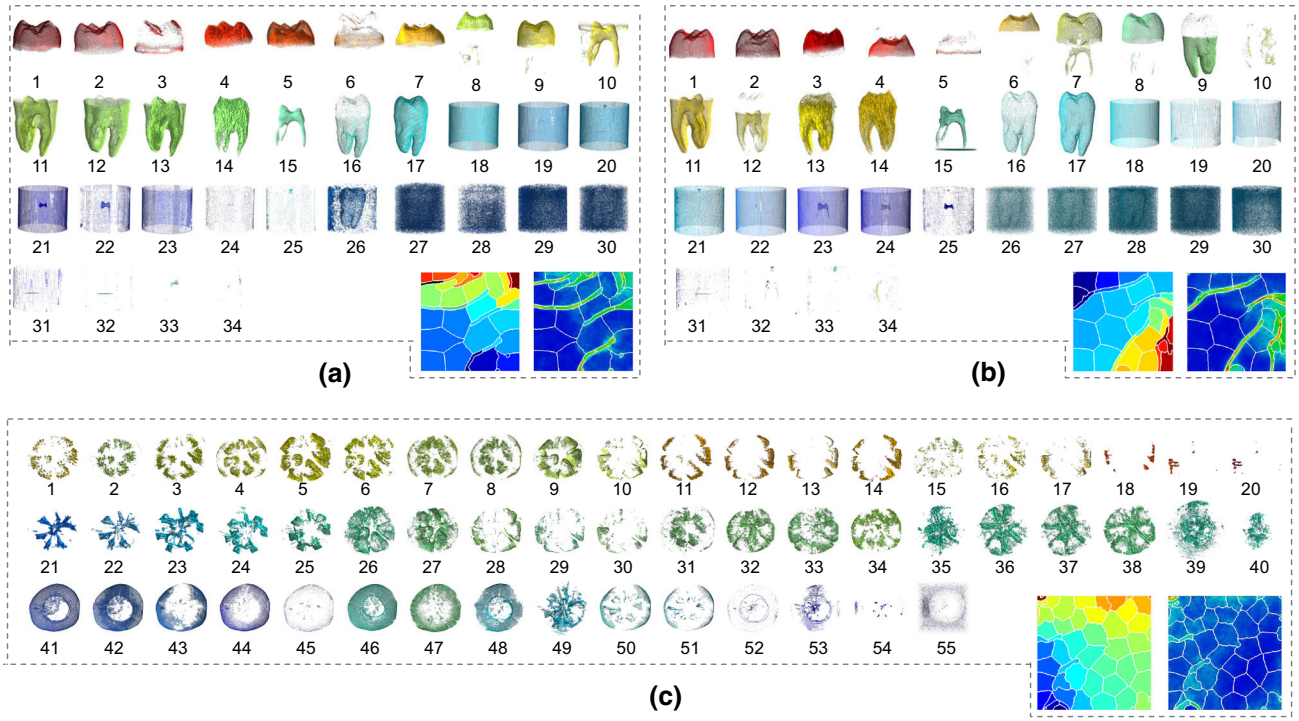


Fig. 14 The corresponding volumetric objects of the clusters produced by the two-level clustering approach. Each subfigure consists of the collection of volumetric objects (each labeled with a number and rendered in the default *color*), the default colored clusters arranged on the segmented SOM and the U-matrix visualization of the SOM. **a** The tooth dataset trained with 7 features (the same set of features as mentioned in the Sect. 5, the 35th cluster is empty). **b** The tooth dataset trained with 3 features (intensity, gradient magnitude and second order derivative along the gradient direction, the 35th cluster is empty). **c** The tomato dataset

objects 2, 7, 15 in Fig. 14b), other structures including important boundaries (e.g., objects 1, 6, 11 in Fig. 14b) remain well defined. Therefore, depending on the user's requirements and interests, a non-optimal feature combination may be sufficient for volume exploration and visualization, thereby easing the reliance on fine-tuned feature selection.

6.1.2 Number of clusters

The number of clusters in the second-level clustering can also influence the resulting clusters. This number is set to 35–55 in our experiment. Typically, using a larger number can result in a smaller and finer structure represented by each cluster at the cost of more exploration burden for the user. A difficult dataset will generally benefit from using a larger number of clusters. Figure 14a, c shows the structures represented by each cluster on two typical datasets. For the Tooth dataset, first-level clustering generates a SOM where the "valleys" are clearly separated from the "ridges". This indicates that clusters are well separated in the SOM, and a cluster number of 35 will be sufficient for effective visualization. On the other hand, the Tomato dataset presents a difficult case by not having clear "valleys" and "ridges" in its SOM. This implies that clusters are not well defined by the first-level clustering and thus a larger number of clusters (i.e., 55) is used in the second-level clustering to generate clusters with finer granularity. In other words, users can decide the number of clusters empirically based on the result of the first-level clustering.

6.2 Comparison with other transfer function design methods

We are aware of two other high-dimensional transfer function design methods that also employ the SOM to perform dimensional reduction [16,30]. These two methods only conduct one-level clustering and the transfer function is directly designed on the SOM visualization map in a bottom-up manner. In Pinto and Freitas's work [30], users are required to draw Gaussian circles on the map with the transfer function obtained by composing the user-specified Gaussian circles; in Khan et al.'s work [16], the transfer function is obtained by direct selection of SOM neurons. Our method uses the trained SOM as the input to a second-level clustering and the transfer function design is conducted on a manageable number of clusters, thus avoiding direct interaction with a large number of SOM neurons and facilitating the detection and separation of interesting regions on the SOM. Moreover, our method enables the generation of an informative volume overview to provide useful guidance for volume exploration,

which is not available in the other methods. For visual comparison, readers may refer to Fig. 3 in [30], Fig. 2 in [16], and Figs. 8, 9 and 11 in this paper to compare the visualization results on the Engine, Foot and Carp datasets. It can be seen that our method can separate and highlight important structures more clearly (e.g., the internal structure of the engine block, the bone and joints of the foot, and the bone and air bladder of the carp).

The normalized cut has been used in Cheuk et al.'s work [15] to segment the intensity-gradient histogram for 2-D transfer function design, while in our work, the normalized cut is used to segment the SOM for higher-dimensional transfer function design. The accuracy of volume structure classification in their work is limited by the number of features used to construct the transfer function space. For instance, our method is able to separate and reveal more structures on the foot and tomato dataset than their method (comparing Figs. 9 and 10 in this paper, and Figs. 10 and 11 in [15]).

7 Conclusions

In this work, a two-level clustering approach was proposed for semiautomatic multidimensional transfer function design. We employed the SOM to conduct first-level clustering, which was able to "self-organize" the high-dimensional feature data on a 2-D rectangle map. To facilitate the detection of meaningful volumetric objects, we applied the normalized cut on the SOM neurons to perform a second-level clustering, which resulted in clusters that represented well-separated and meaningful volumetric objects. We proposed an automatic color and opacity assignment scheme to generate an informative volume overview, which can effectively guide the user in the volume exploration process. By selecting and modifying clusters to visualize, the user gradually understands the volume's content, i.e., what structures the volume contains and the relationship between them. The user can then compose the final visualization based on their understanding of the volume. The effectiveness of our method has been demonstrated on a large variety of volumetric datasets.

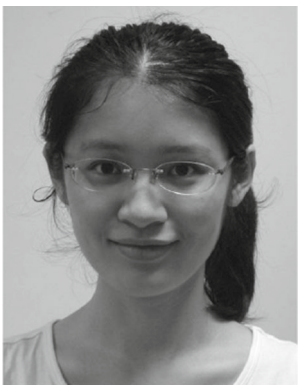
In future work, we plan to take more features into consideration for the training of the SOM, which can further increase the separation ability of the clusters. As the number of features increases, selecting features empirically as adopted in current work would be of low efficiency, thus requiring investigation into an effective feature selection algorithm for voxel data clustering. Another direction of advancement is to study an automatic method to determine the suitable number of clusters for each volumetric dataset, which can increase the flexibility of the system.

Acknowledgments The authors would like to thank all the anonymous reviewers for their insightful comments. This work is supported in part by National University of Singapore FRC Tier 1 Grant (WBS: R265-000-446-112).

References

1. Bajaj, C., Pascucci, V., Schikore, D.: The contour spectrum. In: Proceedings of IEEE Visualization (VIS '97), pp. 167–173 (1997)
2. Bevk, M., Kononenko, I.: A statistical approach to texture description of medical images: a preliminary study. In: Proceedings of the 15th IEEE Symposium on Computer-Based Medical Systems, pp. 239–244 (2002)
3. Bogdan, M., Rosenstiel, W.: Detection of cluster in self-organizing maps for controlling a prostheses using nerve signals. In: Proceedings of European symposium on artificial neural networks, pp. 131–136 (2001)
4. Brugger, D., Bogdan, M., Rosenstiel, W.: Automatic cluster detection in Kohonen's SOM. IEEE Trans. Neural Netw. **19**(3), 442–459 (2008)
5. Caban, J.J., Rheingans, P.: Texture-based transfer functions for direct volume rendering. IEEE Trans. Vis. Comput. Graph. **14**(6), 1364–1371 (2008)
6. Cai, L., Tay, W.-L., Nguyen, B.P., Chui, C.-K., Ong, S.-H.: Automatic transfer function design for medical visualization using visibility distributions and projective color mapping. Comput. Med. Imaging Graph. **37**(7), 450–458 (2013)
7. Correa, C.D., Ma, K.-L.: Size-based transfer functions: a new volume exploration technique. IEEE Trans. Vis. Comput. Graph. **14**(6), 1380–1387 (2008)
8. Correa, C.D., Ma, K.-L.: The occlusion spectrum for volume visualization and classification. IEEE Trans. Vis. Comput. Graph. **15**(6), 1465–1472 (2009)
9. Fujishiro, I., Azuma, T., Takeshima, Y.: Automating transfer function design for comprehensible volume rendering based on 3D field topology analysis. In: Proceedings of IEEE Visualization (Vis '99), pp. 467–563 (1999)
10. Gajdoš, P., Platoš, J.: GPU based parallelism forself-organizing map. In: Proceedings of the third internationalconference on intelligent human computer interaction (IHCI 2011), vol. 179 of advances in intelligent systems and computing, pp. 231–242. Springer, Berlin, Heidelberg (2011)
11. Guo, H., Xiao, H., Yuan, X.: Multi-dimensional transfer function design based on flexible dimension projection embedded in parallel coordinates. In: Proceedings of IEEE pacific visualization symposium (PacificVis), pp. 19–26 (2011)
12. Haidacher, M., Patel, D., Bruckner, S., Kanitsar, A., Gröller, M.: Volume visualization based on statistical transfer-function spaces. In: Proceedings of IEEE pacific visualization symposium (PacificVis), pp. 17–24 (2010)
13. Hladuvka, J., König, A., Gröller, E.: Curvature-based transfer functions for direct volume rendering. Proc. Spring Conf. Comput. Graph. **16**, 58–65 (2000)
14. Hsieh, T.-J., Yang, Y.-S., Wang, J.-H., Shen, W.-J.: Feature extraction using bionic particle swarm tracing for transfer function design in direct volume rendering. Vis. Comput. **30**(1), 33–44 (2014)
15. Ip, C.Y., Varshney, A., JaJa, J.: Hierarchical exploration of volumes using multilevel segmentation of the intensity-gradient histograms. IEEE Trans. Vis. Comput. Graph. **18**, 2355–2363 (2012)
16. Khan, N.M., Kyan, M., Guan, L.: Intuitive volume exploration through spherical self-organizing map. In: Advances in self-organizing maps, pp. 75–84. Springer (2013)
17. Kiang, M.Y.: Extending the kohonen self-organizing map networks for clustering analysis. Comput. Stat. Data Anal. **38**(2), 161–180 (2001)

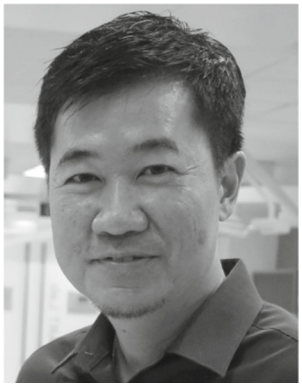
18. Kim, H.S., Schulze, J.P., Cone, A.C., Sosinsky, G.E., Martone, M.E.: Dimensionality reduction on multi-dimensional transfer functions for multi-channel volume data sets. Inf. Vis. **9**(3), 167–180 (2010)
19. Kindlmann, G., Durkin, J.W.: Semi-automatic generation of transfer functions for direct volume rendering. In: Proceedings of IEEE symposium on volume visualization, pp. 79–86 (1998)
20. Kniss, J., Kindlmann, G., Hansen, C.: Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In: Proceedings of IEEE visualization (VIS '01), pp. 255–262. IEEE Computer Society (2001)
21. Kniss, J., Kindlmann, G., Hansen, C.: Multidimensional transfer functions for interactive volume rendering. IEEE Trans. Vis. Comput. Graph. **8**(3), 270–285 (2002)
22. Kohonen, T.: The self-organizing map. Proc. IEEE **78**(9), 1464–1480 (1990)
23. Lau, D.P., Chng, C.B., Chui, C.K.: New device for single-stage in-office secondary tracheoesophageal puncture: animal studies. Head Neck (2014)
24. Linsen, L., Van Long, T., Rosenthal, P., Rosswog, S.: Surface extraction from multi-field particle volume data using multi-dimensional cluster visualization. IEEE Trans. Vis. Comput. Graph. **14**(6), 1483–1490 (2008)
25. Maciejewski, R., Chen, W., Woo, I., Ebert, D.S.: Structuring feature space—a non-parametric method for volumetric transfer function generation. IEEE Trans. Vis. Comput. Graph. **15**(6), 1473–1480 (2009)
26. Moutarde, F., Ultsch, A. et al.: U∗ F clustering: a new performant "cluster-mining" method based on segmentation of self-organizing maps. In: Proceedings of workshop on self-organizing maps (2005)
27. Nguyen, B.P., Tay, W.-L., Chui, C.-K., Ong, S.-H.: A clustering-based system to automate transfer function design for medical image visualization. Vis. Comput. **28**(2), 181–191 (2012)
28. Opolon, D., Moutarde, F.: Fast semi-automatic segmentation algorithm for self-organizing maps. In: Proceedings of European symposium on artifical neural networks (2004)
29. Petrilis, D., Halatsis, C.: Two-level clustering of web sites using self-organizing maps. Neural Process. Lett. **27**(1), 85–95 (2008)
30. Pinto, F.d.M., Freitas, C.M.: Design of multi-dimensional transfer functions using dimensional reduction. In: Proceedings of eurographics/IEEE-VGTC symposium on visualization, pp. 131–138. Eurographics Association (2007)
31. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**, 888–905 (2000)
32. Shi, J., Malik, J.: Normalized cut segmentation code. University of Pennsylvania, Computer and Information Science Department (2004)
33. Su, M.-C., Liu, T.-K., Chang, H.-T.: An efficient initialization scheme for the self-organizing feature map algorithm. In: Proceedings of international joint conference on neural networks, vol. 3, pp. 1906–1910. IEEE (1999)
34. Tzeng, F.-Y., Lum, E.B., Ma, K.-L.: An intelligent system approach to higher-dimensional classification of volume data. IEEE Trans. Vis. Comput. Graph. **11**(3), 273–284 (2005)
35. Tzeng, F.-Y., Ma, K.-L.: A cluster-space visual interface for arbitrary dimensional classification of volume data. In: Proceedings of eurographics/IEEE-VGTC symposium on visualization, pp. 17–24 (2004)
36. Ultsch, A.: Self-organizing neural networks for visualisation and classification. In: Information and classification, pp. 307–313. Springer (1993)
37. Ultsch, A.: Maps for the visualization of high-dimensional data spaces. In: Proceedings of workshop on self-organizing maps, pp. 225–230 (2003)
38. Ultsch, A.: U∗-matrix: a tool to visualize clusters in high dimensional data. Fachbereich Mathematik und Informatik (2003)
39. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Trans. Neural Netw. **11**(3), 586–600 (2000)
40. Šereda, P., Bartroli, A.V., Serlie, I.W.O., Gerritsen, F.A.: Visualization of boundaries in volumetric data sets using LH histograms. IEEE Trans. Vis. Comput. Graph. **12**(2), 208–218 (2006)
41. Šereda, P., Vilanova, A., Gerritsen, F.A.: Automating transfer function design for volume rendering using hierarchical clustering of material boundaries. In: Proceedings of Eurographics/IEEE-VGTC symposium on visualization, pp. 243–250 (2006)
42. Wang, L., Zhao, X., Kaufman, A.E.: Modified dendrogram of attribute space for multidimensional transfer function design. IEEE Trans. Vis. Comput. Graph. **18**(1), 121–131 (2012)
43. Wang, Y., Chen, W., Zhang, J., Dong, T., Shan, G., Chi, X.: Efficient volume exploration using the gaussian mixture model. IEEE Trans. Vis. Comput. Graph. **17**(11), 1560–1573 (2011)
44. Wang, Y., Zhang, J., Lehmann, DirkJ., Theisel, H., Chi, X.: Automating transfer function design with valley cell-based clustering of 2D density plots. In: Proceedings of eurographics conference on visualization, pp. 1295–1304 (2012)
45. Weber, G.H., Dillard, S.E., Carr, H., Pascucci, V., Hamann, B.: Topology-controlled volume rendering. IEEE Trans. Vis. Comput. Graph. **13**(2), 330–341 (2007)
46. Wesarg, S., Kirschner, M.: 3D visualization of medical image data employing 2D histograms. In: Proceedings of second international conference in visualisation (VIZ '09), pp. 153 –158 (2009)
47. Wu, S., Chow, T.W.: Clustering of the self-organizing map using a clustering validity index based on inter-cluster and intra-cluster density. Pattern Recogn. **37**(2), 175–188 (2004)
48. Zhao, X., Kaufman, A.: Multi-dimensional reduction and transfer function design using parallel coordinates. In: Proceedings of the 8th IEEE/EG international conference on volume graphics, pp. 69–76. Eurographics Association (2010)



Lile Cai received the B.Eng. in Electronic Engineering and Information Science from the University of Science and Technology of China in 2011. She is currently a Ph.D. student in the Department of Electrical and Computer Engineering, National University of Singapore. Her research interests include scientific visualization, GPU-based algorithms and high performance computing.

Binh P. Nguyen received his B.Eng. (Hons.) and M.Sc. degrees from Hanoi University of Science and Technology, Vietnam. He was a lecturer in the School of Information and Communication Technology, Hanoi University of Science and Technology prior to pursing a Ph.D. in the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. Currently, he is a Research Fellow in the Department of Mechanical Engineer-

ing, National University of Singapore. His research interests include medical image analysis and visualization, computational intelligence and human-machine interface.



Chee-Kong Chui is an Associate Professor in the Department of Mechanical Engineering, National University of Singapore. His current focus on immersive media involves the provision of visual and haptic cues to assist humans in the training of hand-eye coordination, and furthermore, to augment the hand-eye coordination in a mixed reality environment. Chui received a Ph.D. from the University of Tokyo, Japan.

Sim-Heng Ong is an Associate Professor in the Department of Electrical and Computer Engineering, National University of Singapore. He received his B.E. (Hons.) from the University of Western Australia and his Ph.D. from the University of Sydney. His major research areas are computer vision and medical image analysis and visualization.