iView: A Feature Clustering Framework for Suggesting Informative Views in Volume Visualization

Ziyi Zheng, Nafees Ahmed, and Klaus Mueller, *Senior Member*, IEEE

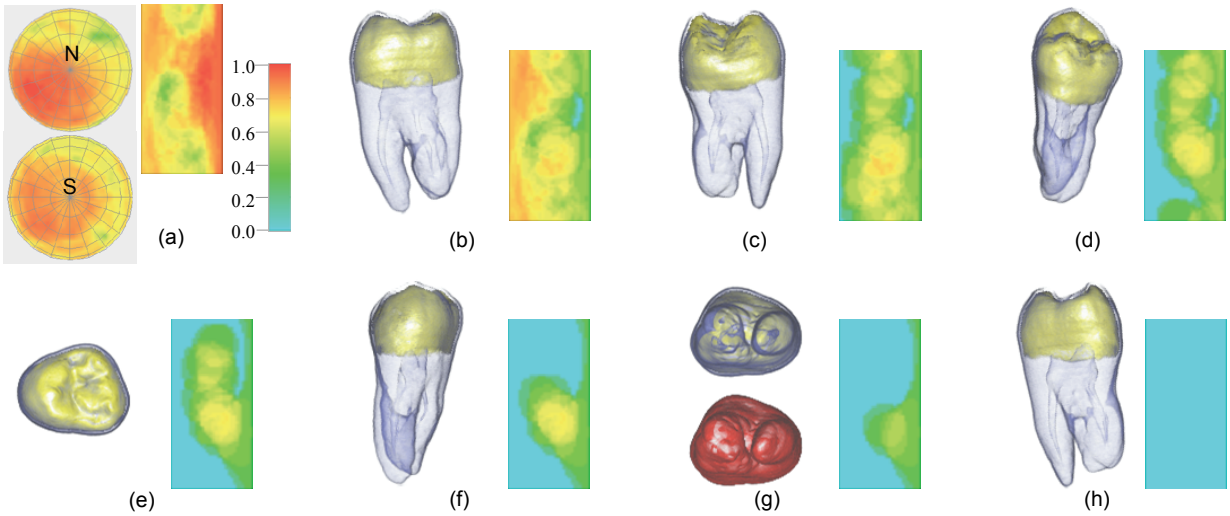


Fig. 1. The tooth dataset – the set of 7 salient representative viewpoints returned by the SCP solver. (a): the initial entropy map, (b-h): the images rendered from the suggested viewpoints (left) with remaining entropy map (right). (g): modifying the transfer function to see the detailed shape of the tooth surface.

Abstract—The unguided visual exploration of volumetric data can be both a challenging and a time-consuming undertaking. Identifying a set of favorable vantage points at which to start exploratory expeditions can greatly reduce this effort and can also ensure that no important structures are being missed. Recent research efforts have focused on entropy–based viewpoint selection criteria that depend on scalar values describing the structures of interest. In contrast, we propose a viewpoint suggestion pipeline that is based on feature-clustering in high-dimensional space. We use gradient/normal variation as a metric to identify interesting local events and then cluster these via k-means to detect important salient composite features. Next, we compute the maximum possible exposure of these composite feature for different viewpoints and calculate a 2D entropy map parameterized in longitude and latitude to point out promising view orientations. Superimposed onto an interactive track-ball interface, users can then directly use this entropy map to quickly navigate to potentially interesting viewpoints where visibility-based transfer functions can be employed to generate volume renderings that minimize occlusions. To give full exploration freedom to the user, the entropy map is updated on the fly whenever a view has been selected, pointing to new and promising but so far unseen view directions. Alternatively, our system can also use a set-cover optimization algorithm to provide a minimal set of views needed to observe all features. The views so generated could then be saved into a list for further inspection or into a gallery for a summary presentation.

Index Terms—Direct volume rendering, k-means, entropy, view suggestion, set-cover problem, ant colony optimization.

1 Introduction

The visual exploration and extraction of relevant information from 3D volume data can be a daunting task as it often requires users to try out many different combinations of views and transfer functions. In this regard, having proper views to start with can greatly improve the efficiency of the data exploration process. Hence, given an arbitrary volume dataset, the suggestion of a set of interesting views has been a research topic of great interest, but also one of challenges.

Most of the recent research in view selection uses metrics based on scalar values to locate regions of interest, that is, before selecting the views the user is either required to design a 1D transfer function [5][35] or perform a segmentation [6] to classify these regions. Then, once this has occurred, the viewpoint selection algorithm will search for the best viewpoint to display the maximum amount of information. However, there are a number of potential pitfalls with this methodology. First, before we can start searching for the best view, the user is required to know the scalar values of the hidden structures to be classified. But without having a proper look at the data first (and without the presence of strong domain knowledge), the user might not have a clear idea what these structures actually are, even in a coarse sense. Any initial guess will likely not be able to classify the hidden features successfully, and so the view selection algorithm in turn will not help to find the best viewpoint. Also, due to the fact that these methods require input in form of a transfer function or segmentation, if the user decides to change either of these a re-computation of the entire pipeline is needed to suggest the new best view. This iterative process can potentially take a long time and thus makes exploring transfer functions in an interactive manner

- *Ziyi Zheng, Nafees Ahmed and Klaus Mueller are with the Visual Analytics and Imaging (VAI) Laboratory, Center for Visual Computing, Computer Science Department, Stony Brook University, NY, USA. E-mail address: {zizhen, nuahmed, mueller}@cs.sunysb.edu.*

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

impossible. Secondly, many structures in 3D volumetric data require more than scalar values to be classified properly. They may require gradient magnitudes (or even higher-order metrics), in conjunction with multi-dimensional transfer functions. Given all these inherent shortcomings, purely scalar-based view selection algorithms can be quite limited for practical use.

Also, current view selection algorithms proposed for volume rendering were built around the concept of selecting a single best possible view to visualize the volume dataset. But in situations in which the viewer is interested in visualizing all of the interesting features in a comprehensive manner, suggesting only one view can be inadequate. While one good view might be able to extract the bulk of the salient features, more views are often necessary in the likely event that some features are occluded in this chosen best view.

Acknowledging the fact that good user-specified 1D transfer functions or proper segmentations are difficult to obtain and may be inadequate to be effective, we propose a view suggestion framework rooted in high-dimensional feature space which does not rely on transfer functions or volume segmentations as an initial input. By applying high-dimensional (low-level) feature clustering, our proposed method can automatically detect salient composite features and based on this analysis suggest promising viewpoints. The user may then inspect these promising views more closely by interactively exploring the transfer function, or run one of numerous automatic algorithms to optimize visibility. We henceforth call this method *viewpoint suggestion* since it helps users to navigate to favorable positions that potentially show interesting structures. In this way our approach is less ambitious than a full-fledged view selection pipeline, but at the same time more versatile and appropriate for data exploration. It supports the extraction of a set of good views appropriately refined with different transfer functions instead of a set of good views limited to one fixed transfer function.

Our approach also offers advantages in terms of interactive data exploration. This is especially important when users are not certain about the properties of the structures of interest and need to refine the renderings. Traditional view selection algorithms [5][6] involve full volume rendering from all possible viewpoints and calculate the viewpoint entropy from all voxels. This can result in prohibitive wait times if users want to update the transfer function during the search for the best viewpoint. In contrast, our framework splits this process into two separate and subsequent phases: (1) the determination of a set of good viewpoints using relatively inexpensive operations (GPU-accelerated clustering, cluster-based visibility test and entropy calculation), and (2) the interactive refinement of the transfer function from these promising viewpoints to generate the desired salient volume rendered images. This approach allows users to maintain a stable spatial reference (and one at which many features are visible), while exploring these features and their relationships one by one by modifying the transfer function.

To address the problem of fully covering all features in a volume, our viewpoints suggestion framework provides a two-fold solution. First, our system not only suggests a single best viewpoint, but also provides an interactive navigation interface where all the views are color-labeled with a relative measure of feature exposure. Also, the system can progressively mark visited features and only show the distribution of the unknown, yet undiscovered information. This type of interaction has already been useful in specialized applications, for example in virtual colonoscopy [18] where colon wall patches already viewed are painted in green on the fly-through, enabling doctors to focus their attention onto unpainted areas. Second, to effectively guide users in their exploration of the entire volume, our system provides an automated viewpoint suggestion module, effectively minimizing the number of views to be inspected. Our multiple-viewpoint suggestion algorithm seeks to determine the minimal set of views that can cover all features, which maps to the *Set Cover Problem* (SCP). Our set-cover problem solver together with the interactive viewpoint navigation tool then aids users in gaining a complete understanding of the features in the volume.

In this paper, we concentrate on the problem of suggesting to the user a set of potentially good views. We leave the design of the transfer function needed to highlight the exposed features to the user, supported by suitable existing interactive algorithms. There are many of these (e.g. [9][10]) and so we do not discuss this issue further. Our contributions can then be summarized as follows:

- Our high-dimensional clustering-based view suggestion framework is purely feature-based and acts *before* transfer function design. It informs users of promising views before laborious transfer function exploration even begins and so prevents "dead-end" transfer function exploration experiences.
- Our view suggestion framework is adaptive to what the user has already seen. Users can explore the entire view-space with progressive suggestions of promising viewpoints. This facilitates a fully unconstrained volume exploration, but ensures that all important features are eventually seen.
- Our system provides viewpoint set solutions that are optimal. We achieve this by using a set cover problem solver, which adapts to the set of views selected by the user so far.

Our paper is structured as follows. In Section 2 we discuss related work. Section 3 provides an overview of the theoretical basis and Section 4 presents the practical consideration of individual components in the framework. Section 5 discusses results. Section 6 presents a user study and Section 7 ends with conclusions.

2 Related Work

In the following we review previous work related to our system.

View Selection: Vazquez et al. [36][37] applied the concept of viewpoint entropy to determine the best viewpoints for polygon-based scenes. Bordoloi et al. [5] proposed to use voxel-based entropy to select viewpoints in volume rendering, assuming that transfer functions are given. Takahashi et al. [35] proposed a similar framework based on iso-surface entropy, weighted by a given transfer function. Chan et al. [6] extended Bordoloi's work by considering spatial relations between structures, after a user-specified segmentation has been given. Our viewpoint suggestion algorithm is fundamentally different from these works as it does not depend on either prior transfer functions or segmentation. Our feature-definition is different from that of Takahashi et al. where features in the volume are defined as a set of iso-surfaces. Rather, in our case the feature metric is sensitive to local structures. This allows for the detection of very delicate features giving rise to as slight normal perturbations, such as text on a surface. Other importance metrics to define interesting features found in the literature, such as suggestive-contours [11] could also be readily incorporated.

Our framework supports multiple viewpoint selection. The selection of multiple views (or view planning) has found application in many domains. A variety of methods seek to solve the *next best view* problem, such as the entropy-based method [40], the visibility-based method [14], and the silhouette-based method [1]. They have wide application in the placement of laser sensors [4] and RFID sensors [41] and for determining the best circular trajectory in cone-beam CT [2]. These view planning methods cannot be directly applied in volume rendering but they can provide useful insight in creating our pipeline, since we define local features in the volume and then solve a similar set cover problem conceptually.

Transfer Function: In volume rendering, material classification is often done via transfer functions. The traditional 1D transfer function is based on scalar values only. Recent research has investigated a plurality of new transfer function domains which have been used together with scalar values and results from these are very promising. They include gradient magnitude [21], curvature [20], features size [8], occlusion spectrum [9] and visibility [10]. Perception can be also added into the transfer function design [7]. Mai et al. [26] presented a semi-automatic 2D transfer function design method based on segmented data. These user-controlled or semi-automatic transfer functions assume a given viewpoint. Our

framework provides a potentially favorable starting point for these transfer function optimization techniques by suggesting good views at which they can be applied. As mentioned, we do not consider the transfer function design before the viewpoint suggestion, due to the additional burden caused from requiring all possible viewpoints to be volume rendered a-priori.

Works on designing transfer function based on feature clustering also exist. Sereda et al. [33] proposed to use clustering to design transfer function. Maciejewski et al. [25] proposed feature detection in 2D transfer function space automatically or semi-automatically. We focus on using clustering in the context of viewpoint suggestion.

View Enhancement: Focus+context techniques are widely used to enhance the volume rendering. Wang et al. [39] introduced the magnification lens into volume rendering. Viola et al. [38] proposed an automatic cut-away view based on assigned importance weight on segmentation. Krüger et al. devised the ClearView [23] system using spherical hot-spots based on discrete curvature based importance. There is also research on adding multiple view information in a single view. Kohlmann et al. [22] presented a deformed viewing sphere based on history. Sudarsanam et al. [34] proposed a widget to incorporate multiple views into a single image. In our work, we have focused on views without distortion and without advanced highlighting, but incorporating these advanced rendering techniques would further benefit our view suggestion framework.

3 THEORY

3.1 Viewpoint Entropy

Simply speaking, a good view onto a volume can be defined as the one that reveals the maximum amount of features relevant to the viewer. Exactly what properties of the volumetric data are relevant to the viewer and need exposure depends mainly on the kind of problem at hand and what the viewer is really looking for. But, once we have defined the feature set, what we then need to find are the views that can show the features distinctly on the screen through graphical rendering. To facilitate comparison among all these views, we assign a score to each of them. And to compute this score, we apply concepts from information theory in a similar fashion as in previous work [5][6][35].

Information theory defines entropy as a measure of uncertainty associated with an information source. Since, to resolve this uncertainty, the amount of data we need to transmit to the receiver defines the amount of information content, entropy of the source hence also measures information. Let us consider any information source A which transmits a random sequence of symbols taken from alphabet $\{a_1, a_2, \dots, a_K\}$ where occurrence probabilities are $\{p_1, p_2, \dots, p_K\}$. Entropy of this information flow is given by,

$$H(A) = -\sum_{j=1}^{K} p_j \log p_j \tag{1}$$

Now, say, in addition to the given probabilities, the receiver also has the knowledge that a certain symbol a_x is always followed by some other symbol, a_y. Presence of this knowledge to the receiver, let us define it as *E*, reduces uncertainty regarding the source. The entropy after this knowledge would be, $H(A|E)$.

In the context of volumetric data, we have an information source – the volume itself, the information receiver – the viewer and a transmission process which includes the whole pipeline of volume rendering. If the volume is not shown to the viewer, then the uncertainty associated with the volume is at maximum and it represents the total information content of the volume, say we denote this by *H*(*X*). Now, in the event that we render a particular view v_1, partial information of the volumetric features become revealed to the user. The uncertainty remaining can roughly be defined as $H(X|V = v_1)$. Since we are interested in finding the view that reveals most information, from the perspective of information theory, what we want is to find a view v_{min} that minimizes $H(X|V = v_i)$ with respect to all possible views. From the chain rule of entropy we can write,

$$H(X|V = v_i) = H(X, V = v_i) - H(V = v_i) \tag{2}$$

Here, $H(X, V = v_i)$ is the information content of the volume and its view taken together. $H(V = v_i)$ denotes entropy of a particular view and as such is a measure of information content of a rendered view. Since a view is just a projection of the volume data, we can consider $H(X, V = v_i)$ to be constant across views. Hence, minimizing $H(X|V = v_i)$ effectively means maximizing $H(V = v_i)$. So, a good view is identified as the one that has large view entropy. A straightforward way [5] [35] to measure the view entropy requires a transfer function to perform volume rendering for the view.

Our entropy estimation is different since we do not want to involve transfer function as an input and later restrict the decision on a single fixed transfer function. Instead, we propose to measure the maximum possible information (across all possible transfer functions) for a viewpoint. This is possible since the 2D image generated by computing the volume rendering equation depends on not only the transfer function but also on the shading (lighting) effect. Shading plays a significant role in conveying information about shape and is well-studied in computer graphics. We define potential information for shading as blobs of voxels and we compute the entropy based on how well one can resolve these feature-clusters at a given view based on the shading (lighting) effect. It serves as an extension of Bordoloi's work [5] in which the visibility of each voxel is computed together with the transfer function to evaluate the entropy. Here we group voxels in the volume according into clusters. Then probability distributions associated with voxel-clusters are needed to calculate entropy. Let q_j represent the contribution of cluster j in a viewpoint. q_0 is a special case indicating the background volume (containing all voxels that do not belong to any clusters). Then the view entropy for a certain view $H(V = v_i)$ is:

$$H(V = v_i) = -\sum_{j=0}^{K} q_j \cdot \log_2 q_j \tag{3}$$

$$q_j = q_j(V) = \frac{1}{\sigma} \cdot \frac{VC_j(v_i)}{W_j} \text{ where } \sigma = \sum_{j=0}^{K} \frac{VC_j(v_i)}{W_j} \tag{4}$$

Here K is the total number of feature clusters. The factor σ will make sure that all q_j sum up to 1. $VC_j(v_i)$ is the visibility of cluster j in view v_i. W_j the *noteworthiness* [5] of cluster j, is defined as:

$$W_j = I_j = -\log_2 p_j = -\log_2 \frac{|n_j|}{\sum_{i=0}^{K} |n_i|} \tag{5}$$

where p_j represent cluster j 's probability, calculated from the number of voxels in cluster j normalized by the total number of voxels. We add the consideration of background. n_0 is number of voxels that do not belong to any cluster, p_0 is the probability of background and W_0 is the noteworthiness factor for background. In Sec. 4.1 we present our method for computing the importance of features that is sensitive to shading but independent of any transfer function. Sec. 4.2 defines the computation of view entropy based on the selected metric. Finally, Sec. 4.3 shows how we present the data computed from Sec. 4.2 in both a suggestive and explorable manner.

When we are suggesting views onto the volume, besides finding a single view that reveals information in the best way, we also want to guide the user such that they will not miss out on any of the features. In the language of information theory, a single view v_i might be the optimal view in terms of view entropy but it may be the case that $H(X|V = v_i) \neq 0$. So, we propose to suggest to the user a set of views $\{v_1, v_2, \dots v_i\}$ so that $H(X|V = v_1, V = v_2, .., V = v_i\}=0$, that is, to find a set of views that collectively can give the viewer a total picture of all the important features of the volume data. Sec. 3.2 and Sec. 4.4 show how we compute such set by solving the SCP.

3.2 Ant Colony Algorithm for Set Cover Problem

We suggest an optimal series of viewpoints by solving the SCP. We first generate a large number of views as candidates to choose from. Each view will then cover a number of features. Thus we make each view a "set", while the features that need to be covered form

"elements". The optimization objective is to find the minimum number of views that cover all salient features.

The SCP was one of Karp's 21 NP-complete problems [17]. A mathematical model for the SCP is usually described by a 0-1 matrix, A. Let $A(a_{ij})$ be an m-row, n-column, zero–one matrix. We say that a column j covers a row i if $a_{ij} = 1$. Each column j is associated with a nonnegative real cost c_j. Let $I = \{1,\ldots,m\}$ and $J = \{1,\ldots,n\}$ be the row set and column set, respectively. The SCP can be stated as:

$$\min\{\sum_{j=1}^{n} c_j \cdot x_j\} \tag{6}$$

subject to

$$\sum_{j=1}^{n} a_{ij} \cdot x_j \geq 1 \text{ and } x_j \in \{0,1\}, \qquad \forall i \in I, \forall j \in J \tag{7}$$

where, $x_j = 1$ if set j is selected, otherwise $x_j = 0$. The matrix $A(a_{ij})$ encodes all the viewpoints strength to cover the feature clusters in the volume.

The SCP can be solved by many algorithms and the ant colony algorithm is one of the fastest solvers [30][31]. It is inspired by the observation of real ant colonies. The general mind-set behind ant colony algorithm is that a large amount of artificial ants search for an optimal solution defined by Equation (6). Each artificial ant chooses a set one by one until it achieves a complete cover defined by Equation (7). The decisions for choosing different sets are partially based on Russian-roulette. Additionally, the probability for choosing one set will increase if a large number of ants choose it, which is in the way of *pheromone information exchange*. In the following, we explain the ant colony algorithm for the SCP in detail.

The probability for an ant to choose set j is based on the state transition rule:

$$P(s_t = j|S_{t-1}) = \begin{cases} \dfrac{\tau_j h_j^{\beta}}{\sum_{q \in J \setminus S_{t-1}} \tau_q h_q^{\beta}} & if\ j \in J \setminus S_{t-1} \\ 0 & otherwise \end{cases} \tag{8}$$

where S_{t-1} denotes the partial solution constructed before step t, $J\backslash S_{t-1}$ denotes the subset of unselected columns, s_t denotes the set that will be chosen at the step t and the parameter β ($\beta >= 0$) determines the relative importance of the heuristic factor with respect to the pheromone.

The heuristic is usually defined by a greedy method. If R is the set of still uncovered elements and c_j is the cost associated to set j. The heuristic value of set j is:

$$h_j = |I_j \cap R|/c_j \tag{9}$$

The pheromone trials are stored at each set as τ_j. They are initially set to one and updated later as:

$$\Delta\tau_j = \begin{cases} \dfrac{1}{\sum_{q \in S_{gb}} c_q} & j \in S_{gb} \\ 0 & otherwise \end{cases} \tag{10}$$

$$\tau_j \leftarrow (1-\rho)\tau_j + \Delta\tau_j \qquad \forall j \in J \tag{11}$$

where S_{gb} is the current best solution across all ants, ρ is the pheromone evaporation factor (with $0 < \rho < 1$) and $\Delta\tau_j$ is the amount of new pheromone put on column j. The range of pheromone should be clamped with a range $[\tau_{min}, \tau_{max}]$ where:

$$\tau_{max} = \frac{1}{(1-\rho)\sum_{j \in S_{gb}} c_j} \tag{12}$$

$$\tau_{min} = \varepsilon \cdot \tau_{max} \qquad 0 < \varepsilon < 1 \tag{13}$$

For more detailed description of the ant colony algorithm applied for the set covering problem, we refer the reader to [31].

4 Approach

Our overall framework is shown in Figure 2. The first stage is a multi-dimensional data clustering. Given a certain noise-level for the dataset, we consider voxels with high gradient/normal variation as the important features. We perform k-means clustering to group voxels into blobs, followed by a visibility test. In the next stage we compute the information gain for all viewpoints around the object and create an entropy map that we display on a sphere that doubles as a track-ball interface used to change viewpoints. Thus, by mapping the entropy map directly on the track-ball, users can directly and intuitively identify and navigate to favorable view locations. The user can also add or delete viewpoints by clicking on the sphere and the displayed entropy map is updated accordingly.

Alternatively, the SCP solver can be used to suggest to the user a set of optimal or at least near-optimal viewpoints from which to visualize the volume. It provides a series of viewpoints that covers all features. The set of optimal views suggested by the SCP solver is annotated onto the navigation sphere and the user can continue navigating through the sphere with these added suggestions in hand. In the meantime the user can also use the transfer function designer to explore settings that expose features at the given viewpoints.

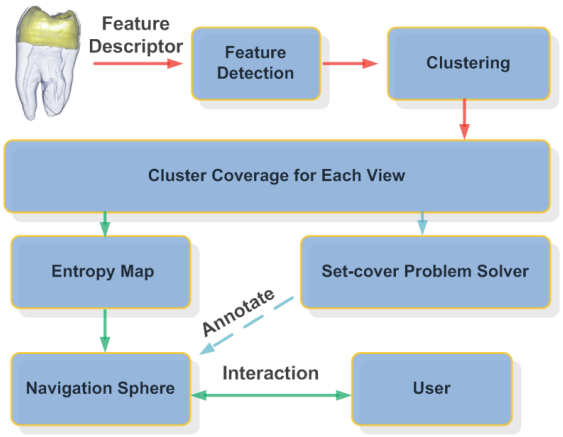


Fig. 2. Overall framework.

4.1 Feature Extraction and Clustering

Our framework is based on feature detection and high-dimensional data clustering. As for the question how to define features in high-dimensional space as potentially interesting structures, there are many choices and their suitability can be application-dependent. We chose to provide a very general and application-neutral importance metric based on *normal-variation*. Normal-variation plays a significant role in lighting. In this paper, we shall assume that an area with large normal/gradient variations contains salient information, while regions with similar gradient directions have less information. This metric is an extension of 2D curvature estimation and it generally belongs to the group of Laplacian operators. The discrete importance estimation for a voxel located at (x_0, y_0, z_0) is:

$$w(x_0, y_0, z_0) = \sum_{(x,y,z) \in N(x_0,y_0,z_0)} |\nabla f(x,y,z) - \nabla f(x_0,y_0,z_0)| \tag{14}$$

where $f(x,y,z)$ is the volumetric scalar field, ∇ is the gradient operator and $N(x,y,z)$ is the set containing a neighborhood of (x,y,z). In our estimation, only the 6 closest neighboring points are considered. This metric is different from the classic Laplacian operator which is defined as the divergence of the gradient vector field and thus can be negative. The importance weight here sums up the absolute values of gradient difference individually which can guarantee the weight to be positive. The intuition behind this metric is that we want to have a measure of the perturbations of gradient/normal in a region.

Most practical volume data contain a certain level of noise which will affect the feature detection. In the pre-processing stage, we take the ambient noise level as an input to threshold the scalar values. We also consider the noise removal as a thresholding procedure on gradient variation. After applying these thresholds, the resulting voxels are considered the important voxels and are clustered in a

five-dimensional space: scalar value, gradient magnitude, and (*x, y, z*) coordinate.

The k-means algorithm is one of the most well-known clustering algorithms. Given an input value k, it can partition n objects into k clusters based on some similarity (distance) measure. We apply k-means clustering to get k blobs in 5D space. We also record the voxels inside each cluster and remove a cluster if the number of voxels is too small (less than 5). The gradient/normal direction for each cluster is computed as a Gaussian-weighted average which enhances spatial coherence.

An example of computed gradient vectors is shown in Figure 3. The clustering phase can employ automatic feature detection if the ambient noise level is known. We build our system in the high-dimensional feature domain. Hence it can detect local structures with high gradient variation and adjust views for these local features. This provides a general importance metric well suited for non-expert users, to minimize user invention. But we note that our data-clustering pipeline can readily support other more specific metrics if more specific domain knowledge about dataset and task is available.

An important aspect in k-means clustering is the input value k. The value of k controls the resulting number of clusters and as such the grouping of similar features of the input dataset. For a given dataset, to identify the features distinctly, the algorithm requires a certain k that will ensure separation of features such that the clusters truthfully represent the features. Having a smaller k value will merge a set of features into a big cluster, whereas a larger value will produce clusters covering fine details. From a multi-resolution point of view, the choice of k affects the resolution of the features to be extracted. Therefore, the value k will reflect the average size of the feature clusters. We base the choice of k on the average cluster size (and therefore desired level of detail structures), chosen by the user via a slider interface. Then the value k is found by dividing the total number of noise-free voxels by the desired average cluster size. But alternative approaches such as the elbow criterion [19] and X-means [28] may also be utilized to obtain an appropriate k. Finally, users may also inspect the clusters by visualizing them in the 3D interface.

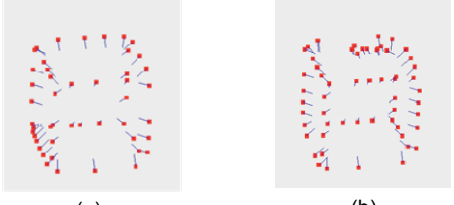
Another important issue with k-means clustering is also the choice of the initial k seeds, which can produce a certain level of randomness in the clusters formed. To overcome this problem we use the standard practice of clustering the data multiple times with different random seeds and picking the clustering that has the least overall L_2 error with respect to the clustered data points. Since we use the GPU-based standard-version k-means library [13] the performance hit is relatively minor.

4.2 Entropy Calculation

The k-means algorithm outputs a series of 5D clusters and gradient/normal values at the centers of the clusters. For each cluster, we extract its spatial information as a 3D ellipsoid and estimate the visibility based on the cluster's normal direction.

We assume the center of a cluster to fall within the clipping window. Then there are three major factors that contribute to a good view: (1) the angle between the cluster's normal and the viewing direction (the eye ray), (2) the number of clusters that can be shown, and (3) the total number of voxels within potentially visible clusters.

Our goal is to calculate the maximum potential entropy for the feature clusters. We establish a set of criteria for a feature cluster to be classified as invisible. We first set a threshold on a clusters' normal range and later use entropy to measure viewpoint quality.



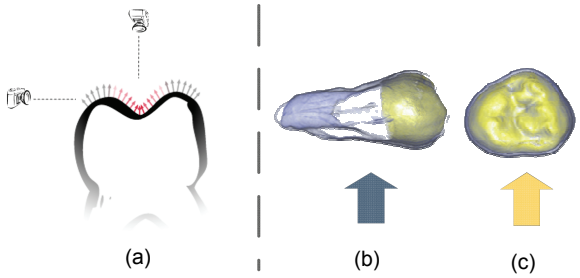
(a) (b)

Fig. 3. K-means feature clustering with gradient vectors shown for (a) a standard cube and (b) a cube with text on the back surface.

The first criterion is that the gradient direction and eye-ray should be within a certain range, enabling shading effects to enhance small details in the volume rendering. Shading conveys a strong cue for shape (see "shape from shading" in computer vision, graphics and robotics [42]). An important observation is that at 45° the Lambertian cosine shading functions starts to loose strength, since the derivative of the cosine function has an extreme point in the vicinity of 45°. So if the normal vector of the cluster and viewing vector make an angle of greater than 45°, our framework rates the viewpoint as inadequate to cover the feature cluster's information.

Silhouettes can also be salient in conveying shape information (this is a popular method in non-photorealistic rendering). Silhouettes start to become visible when the view and normal vectors are close to 90°. Consequently, we may allow clusters that are close to 90° to be visible as well. This criterion is especially effective in dynamic flow visualization, in which interesting wave fronts can be represented as silhouettes. But in static data with concave shapes, it may produce poor results as shown in a simple example (Figure 4). Figure 4(b) is a good view via the silhouette criterion but it provides only little information. In contrast, Figure 4(c) emphasizes only normal deviations and is a more informative view. Hence, we find that shading effects tend to be a safer way to test the visibility of the features, especially when the user is facing non-convex shapes. Since in this paper we only focus on volume rendering of static datasets, we prefer the shading-based method to point out salient details.

Our visibility test so far did not account for the occlusion among clusters, since our target was the maximum possible exposure of all details within a given viewpoint. This is less of an issue since advanced frameworks (such as occlusion-spectrum based transfer function) have the capability to explore data with occlusions. In addition, conceptually we place no limitation on the number of images or transfer functions per viewpoint. As such, at a given viewpoint, the user may theoretically see all the structures within a normal range by applying different transfer functions one by one. In fact, this was our initial design choice: giving the user the freedom to choose any type of transfer function or take any number of rendering results later on. But practically speaking, the occlusion effect among the ellipsoids represents an additional time overhead (and therefore cost) in the data exploration process. So we provide a weight by which the user can set a preference on less occlusion which in turn eases the transfer function design, as discussed in Section 4.5.



(a) (b) (c)

Fig. 4. Silhouettes fail to convey concave shape.

By applying the visibility test, we measure the quality of a viewpoint in terms of view entropy (Equation (3) and (4)), and find the largest entropy by extensive search. As discussed, the major difference between ours and other work [5][6] is that instead of representing information according to scalar value, we define it on important features in a high-dimensional feature descriptor domain.

For the cluster-based entropy, we mark all voxels that do not belong to any clusters as 'background', which is similar to the background feature definition in viewpoint selection methods for polygonal models [36][37]. This will remove singularities where only one cluster is shown in a viewpoint but its entropy is 0. After considering all background voxels, if no feature cluster is shown, the view will have zero entropy. In contrast, if any feature cluster is shown then the entropy will be non-zero.

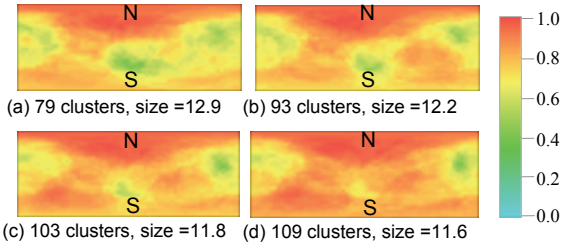


Fig. 5. The effect of the number of clusters on the entropy map. The Initial *k* for k-means: (a) k=100. (b) k=110. (c) k =120. (d) k =130.

Finally, Figure 5 shows how different settings of k affect the entropy map of the tooth dataset. The 79-cluster case (Figure 5a) has an average voxel set size of 12.9 and the 109-cluster case (Figure 5d) has an average voxel-set size of 11.6. We observe the growing entropy pattern with increasing cluster number. However, the relative importance is quite stable. All entropy maps suggest that the most important view is from the north-pole. Then the next interesting view is from the south-pole. Finally, in the middle region there are two fairly interesting viewing areas and three relatively non-interesting areas.

4.3 Viewpoint Information Exploration

If we assume that all viewpoints have the same camera distance, the possible projection locations will be on a sphere with radius R and can be parameterized by longitude φ and latitude λ. We further assume that in each view position we use the same field-of-view (FOV) and we are looking at the center of the sphere. Rendering the viewpoint entropy map onto a sphere, every single point on this sphere then represents one view position and the intensity of a point denotes the amount of information this viewpoint can possibly cover. As shown in Figure 6, the nearest point to the user (screen center) is the current viewpoint. The navigation window and volume rendering are displayed side by side. The user can rotate the sphere in the arc-ball interface and in the meantime the volume will rotate in a synchronous fashion. The user can then check on the map which view position would possibly be more interesting to look at.

Deviating from the previous works that use single viewpoint selection interface, we provide progressive navigation tools. Users can select a complete set of views to render the volume data in a greedy manner. The system helps the user to navigate and allows them to select a series of viewpoints. When the user marks a point on the sphere to represent the selection of a viewpoint, the system shows the rendered image and updates the entropy information interactively, reducing the entropy map to that of the undiscovered features. Users may then continue to choose several more views until not much color is left. Also, the user may undo the latest selection and the system will then add the affected clusters back into the map. The user may also undo selections multiple times which will be reflected on the entropy map in reverse order.

We next explain how to update the entropy calculation after a user's interaction. Initially, all clusters are set as unknown, denoted as $u_j = 1$ for the j^{th} cluster. The clusters that have been explored by the user are marketed as inactive. Thus:

$$u_j = \begin{cases} 0 & \text{if user visited cluster } j \text{ and } j \neq 0 \\ 1 & \text{otherwise} \end{cases} \quad (15)$$

$$q_j = u_j \cdot \frac{1}{\sigma} \cdot \frac{VC_j(V)}{W_j} \quad (16)$$

Undoing a user selection will mark the corresponding u_j as active again. We do not perform further normalization of the noteworthiness factors during user selection, considering the fact that the amount of total information is constant. In this way, the user will observe the color fading from red to blue to convey the amount of information that has now been explored.

As most users tend to use this tool in a greedy manner, it may not be the optimal way for choosing the camera positions that cover all detected features. For this purpose, we incorporate our SCP solver to help users to find the optimal viewpoints. This is explained next.

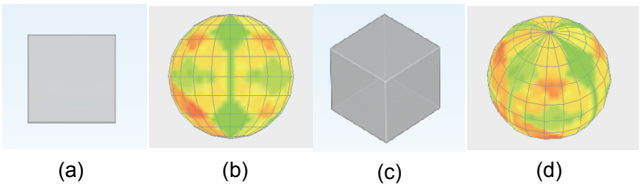


Fig. 6. Viewpoint navigation for a cube dataset. (a): display window for the rendered object. (b): view navigation sphere (track-ball). Upon rotating the track-ball. Both (c) the rendered object and (d) the view navigation sphere will rotate in a synchronized manner.

4.4 Suggesting Best Combination of Views

As mentioned, a greedy search is not always the most optimal approach when searching for a set of views covering all clusters. The entropy-rated viewpoints are only a result of a local heuristic which is not adequate to find a global optimum. Hence we provide the user with the minimum number of viewpoints needed for full exploration based on an ant colony optimization of the set covering problem.

The ant colony optimization method creates L artificial ants to search for the viable solution to Equation (6) and (7). After all L ants find viable solutions the system keeps track of the best ant (with minimum cost), updates the pheromone and runs L more artificial ants until the desirable cost is found by an ant. In our problem, the user can specify the number of views he/she wants to have to expose all feature clusters. Then the SCP solver will run multiple ants to search for the solution. Each ant will make a decision on what is the next viewpoint to choose based on heuristic and pheromone (Equation (8)). The heuristic is proportional to the number of unknown clusters that can be covered (Equation (9)). And the pheromone reflects how many other ants previously chose this viewpoint, as shown in Equation (10-13). After all ants have finished, the ant with the minimum number of viewpoints will deposit the pheromone to the viewpoints it chose. The system will report a success if an ant has found the desired solution. If in a limited amount of time, the ants cannot find a set of views under the desired cost, the system will report that no solution has been found.

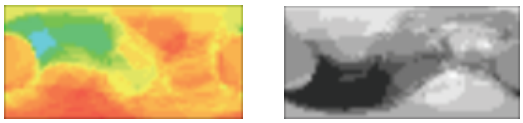
We have implemented the ant colony algorithm in C++ and validated it against several test problem cases [3] with known solutions.

4.5 Viewpoint Preference

The user can further set preferences to refine the viewpoint suggestion results. We have added several metrics to that effect.

Cluster Occlusion: In our approach, multiple transfer functions are typically required to extract information from a suggested view when there are occlusions between features. We can add support for preferring views with non-occluded features by adding a cost penalty for views with occluded features. This cost is determined by the maximum number of occlusions for a given view. We compute the penalty during the splatting-based cluster visibility test (Sec. 4.2). As discussed, the clusters are represented as 3D ellipsoids, and we record the maximum number of ellipsoids projected on each pixel. The cost is initially assigned to 1 if there are no overlaps, and it grows with the number of overlapping clusters discovered in the visibility splatting.

To visualize the amount of cluster overlap and so the potential difficulty in the subsequent transfer function exploration, we provide two interfaces: (1) we render the cost in grayscale on a separate map/sphere, and (2) we fuse the occlusion cost into the entropy map by using it as a weighting inside the SCP solver. The former can be justified since the cost of occlusions does not add to the concept of entropy which only encodes the maximum potential exposure of information. In Figure 7, we show the entropy map alongside the occlusion cost map for the bluntfin dataset. There are some areas with similar entropy but with different overlap counts. During the



(a) the entropy map (b) the cost map

Fig. 7. (a) Entropy map and (b) cost map for the bluntfin dataset. In the cost map, the maximum number of overlaps is 8 which is mapped to white color.

user interaction, the user mainly relies on the entropy map for viewpoint exploration, but treats the cost map as a guide to estimate the potential difficulty for the subsequent transfer function design.

Viewpoint Separation: The SCP solver will generate results with reasonable separations among the viewpoints. If there are several solutions, the user can set a preference for viewpoint separation. This will reflect how evenly the viewpoints are distributed over the sphere. We gauge this evenness of the distribution by calculating the variance of the distances between the selected views on the sphere.

Viewpoint Up-Direction: Given a fixed view, the information exposed by the volume on the screen does not depend on the orientation of the image. But users may have a preferred up-direction of the camera based upon common viewing experiences with the given dataset. In our framework there is a default vector specified as *up-direction*. If the data does not appear to be properly oriented with the default up direction, the user has the option to specify this vector.

5 Results

In our experiments, we set the viewing angle limit to 45° and assuming a voxel distance of 1 mm we set the viewing distance to 5,000 mm. The image size is always 512^2 pixels. The view positions on a sphere are sampled on a 60×30 grid, with a total of 1,800 viewpoints. All volume-rendered results shown in this paper were obtained using the rendering software Imagevis3D [12][15].

We first tested our framework on a simple cube dataset. The cube's size is 80^3, residing in a 256^3 volume grid. We added a shift vector $(10, 20, 30)$ to the cube which moves it off the volume grid center. Figure 8(a) shows this dataset, and Figure 9 displays the rendering results obtained with our system. In this case, the SCP solver automatically suggests 4 different views, looking down the cube diagonals. All of these 4 views coincide with the best views provided by Bordoloi et al. [5]. It appears that two images are not sufficient since each viewpoint will resolve three edges in the center through shading, while the rest of the edge features are partially but not completely visible. As seen in Figure 8(c), it is not safe to only have two views to visualize the cube, since we do not have good information about the 6 edges appearing in the silhouettes. Conservatively speaking, only 4 viewpoints will be able to see all 12 edges with full exposure of all features. The corresponding entropy map is shown in the bottom row of Figure 9. The initial high entropy map with no views selected shows 8 favorable regions, identifying the cube's 8 vertices as best views. The entropy map is then updated gradually as views are selected. Here selecting a given view typically removes more than one local maximum from the entropy map.

For our next experiment, we add the text "Vis2011" on one of the surfaces of the cube (normal perturbation), as shown in Figure 8(b). The results for this modified cube are shown in Figure 10. In this

(a) (b) (c)

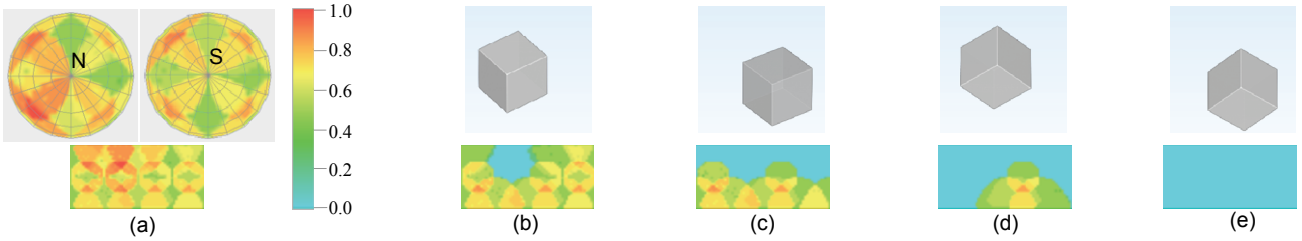
Fig. 8. The cube datasets. (a): a standard cube. (b): a cube with text on a face. (c): a cube with features on the edges.

case, we need 5 views to fully visualize the dataset. In navigation mode, the entropy map clearly highlights the surface with text, indicating that in this region there is something important. In the automatic view suggestion mode, one of the resulting views specifically targets the text while the other viewpoints aim to look along the diagonals. In contrast, scalar-value based methods [5][35] will not be able consider the text as an important feature and so will display the entropy map of a uniform cube. Our method, on the other hand, can faithfully detect this type of intricate surface features and suggest something of possible interest is hiding in a certain view.

We also tested our pipeline on medical data, which typically do not have strong regular edges like the cube. Our experiment uses the well-known tooth dataset. Figure 1 shows the resulting 7 views suggested by the SCP solver (the progressive entropy maps are shown on the right of each image). We note that both the rendering results and the entropy maps have been re-oriented using the user-defined up-directions since the default up-direction does not conform to the user-preferred tooth direction. In accordance with Bordoloi's result, the entropy map of this dataset indicates the north-pole and the south-pole as the two most interesting regions. The SCP solver subsequently chooses the north-pole and the south-pole as the first two viewpoints, and therefore the resulting 2 images are deemed to reveal the most interesting information on the data. We also see that after the first 2-3 views have been selected the remaining entropy is rather low and sparse, but our system includes them in the gallery to provide complete coverage. Another important aspect to note in this particular experiment is the utilization of multiple transfer function in the same view. For some of the viewpoints, the potential information is hidden if only one transfer function is considered, as shown in the 6^{th} viewpoint in Figure 1(g). But once the user is given a view that can guarantee the presence of useful information, he can always refine the transfer function to freely explore the interesting structures best revealed by this view.

Next, we tested the framework on a practical dataset obtained from Computed Tomography (CT). Figure 11 shows the results for the carp dataset. According to the entropy maps, the first 3 views cover the most important features of the carp. The map also indicates that one side of the carp is slightly more interesting than the other side, due to the carp's bent body as shown in Figure 11(d). The remaining two views are less important, but again we provide them for completeness of the gallery.

Figure 12 shows a gallery obtained for the engine dataset where panels (d) and (g) each show images of the same view but rendered with different transfer function settings to visualize different aspects of the engine. This study illustrates the benefit of the two-phase design of our system. First obtain a view direction at which many different types of features can be observed well, and then maintain



(a) (b) (c) (d) (e)

Fig. 9. The standard cube dataset (4 viewpoints computed by the SCP solver). The cube is shifted from the center. (a): the initial entropy map. (b-e): the suggested viewpoints rendered (top) and the maps with remaining entropy (bottom).

that view and visualize these features in a number of ways to accentuate their various relationships in turn. This approach allows the user to maintain a coherent spatial reference, while learning about the dataset through dynamic feature exploration.

Finally, we also tested our framework on a fluid simulation dataset, e.g., the well-known blunt fin. Figure 13 shows the 5 views suggested by the SCP solver. The view in Figure 13(a) is suggested as the most important view, which in fact is close to the most commonly selected view onto this dataset.

All of our experiments were conducted on an NVIDIA GTX 480 GPU, programmed with CUDA 3.2 runtime API, hosted by an Intel Core 2 Duo CPU @ 2.66GHz. Table 1 shows the performance of the different stages of our framework. The feature extraction part includes the thresholding and randomization of the resulting voxels. The visibility test portion is for testing the cluster normal directions at all views and also includes the splatting-based occlusion number computation. The most time-consuming part is the visibility test which would be much slower without GPU acceleration. The total processing time is about 2-8 times faster than a volume rendering of 1,800 views with a fixed transfer function. Table 2 shows the optimal number of viewpoints computed by the ant colony-based SCP solver in 10 seconds and the number it finally converges to (marked by ∞). For practical datasets, the voxels were filtered at above 3% of the maximum scalar value to remove noise, and the average cluster width was in the range of 10-20 voxels.

Table 1. The Performance of Different Stages of Our Viewpoint Suggestion Pipeline

Dataset	Datasize	Feature Extraction / K-Means Clustering / Visibility Test	Rendering Time / Speedup
Cube	256x256x256	0.9s / 0.9s / 7.3s	18s / 2.0×
Cube + Text	256×256×256	1.2s / 0.5s / 7.9s	18s / 1.9×
Tooth	256×256×161	1.2s / 1.3s / 8.9s	93s / 8.2×
Engine	256×256×110	1.1s / 0.6s / 10.4s	53s / 4.4×
Blunt Fin	256×128×64	0.7s / 0.6s / 8.2s	26s / 2.7×
Carp	256×256×512	2.5s / 2.5s / 9.2s	87s / 6.1×

Table 2. The Problem Size of the K-Means Clustering and the Minimum Number of Views Found by the SCP Solver

Dataset	Voxels	Initial k / Resulting k	Averaged Cluster Width	Views (10s / ∞)
Cube	968	50 / 47	2.7	4 / 4
Cube+Text	1278	60 / 58	2.8	5 / 5
Tooth	170228	100 / 79	12.9	7 / 6
Engine	529050	120 / 111	16.4	7 / 6
Blunt Fin	65053	50 / 50	10.9	6 / 5
Carp	331894	80 / 69	16.9	5 / 5

6 Evaluations

We performed a simple user study to evaluate the effectiveness of our iView interface. For this, we invited 9 graduate students, all familiar with volume rendering and transfer function design. At all time, the subjects were permitted to use the 1D or 2D transfer function editor and choose any preferred view to look at the volume, with a fixed front-light and using the track-ball interface. The only testing condition was that they either had access to our entropy map or not. In the latter case the track-ball surface was simply left blank.

Each subject/user would render two different dataset in turn – the tooth and the carp. For each dataset, a user was asked to construct two galleries (that is, select a set of viewpoints) that would best expose the salient information of the given dataset. The first gallery was always constructed with no entropy-map guidance, while for the second gallery this guidance was available. Prior to using the system, each user was trained on how to navigate with the entropy sphere (if provided), how to interpret its data, how to observe the clustering results and also how to use keys to add/delete views from the gallery.

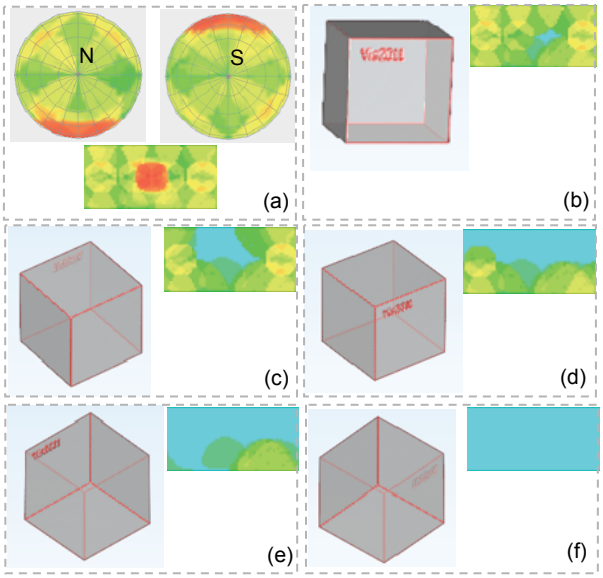


Fig. 10. The cube with text on one face (5 viewpoints computed by the SCP solver), with transfer function highlighting the text. (a): the initial entropy map. (b-f): the suggested viewpoints rendered (left) and the maps with remaining entropy (right).

We compared the views selected with and without the iView guidance. We found that in general users would pick fewer views without guidance. The mean of the difference between two sets of views was 0. 90 (tooth) and 0.56 (carp). We used the dependent t-test for paired samples to analyze the view numbers with the hypothesis that those two means (with/without entropy map) are identical. The p-values for tooth and carp were 0.003 and 0.05, respectively. Thus, the fact that users would consistently pick fewer views without guidance indicates that iView helps users in locating commonly overlooked regions.

It was also interesting to observe that no gallery generated without guidance would include all of the top three views found with guidance (or with the SCP solver). There were often redundant views in the uninformed gallery or views with low entropy. To capture this behavior more quantitatively, we measured a given gallery's information coverage by the sum of entropy left in the map after gallery composition. When a user was allowed to use the map, the sum of the entropy left dropped from 51% to 24% for the tooth and from 37% to 19% for the carp, on average. Likewise, the percentage of entropy covered per view increased from 11% to 15% for the tooth and from 14% to 16% for the carp. This demonstrates that our navigation interface can clearly help users to optimize a set of viewing positions and with it the information seen.

7 Conclusions

We have presented a feature clustering approach that suggests users promising viewpoints for volume visualization prior to transfer function design activities. We believe that such a transfer function neutral approach cuts down on the volume exploration effort since it selects potentially interesting views before laborious transfer function exploration begins. As such, our approach promotes a data exploration procedure in which users first navigate to views that promise to show many features well and then explore and enhance these features and their relations via transfer functions in a stable spatial context. We believe that this is cognitively less challenging than changing viewpoint and transfer function at the same time.

Our approach strongly favors interactive volume exploration, mapping the navigation aids directly on the track-ball interface used for spatial transformation. In addition, the system updates the navigation aids in an adaptive manner based on the views selected. Finally, a set cover problem solver is also available to choose a set of optimal views automatically to compose an overview gallery.

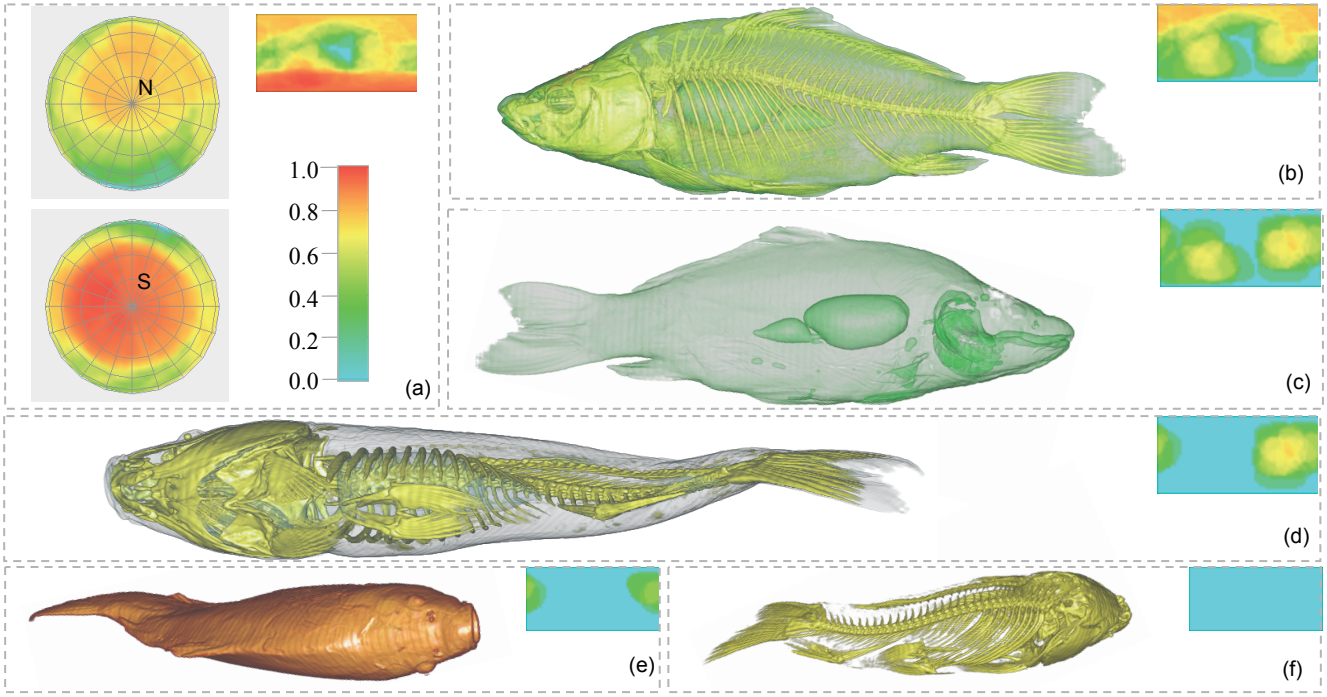


Fig. 11. The carp dataset (5 viewpoints computed by the SCP solver). (a): the initial entropy map. (b-f): the suggested viewpoints rendered (left) and the maps with remaining entropy (right). The transfer function could be changed in different views.

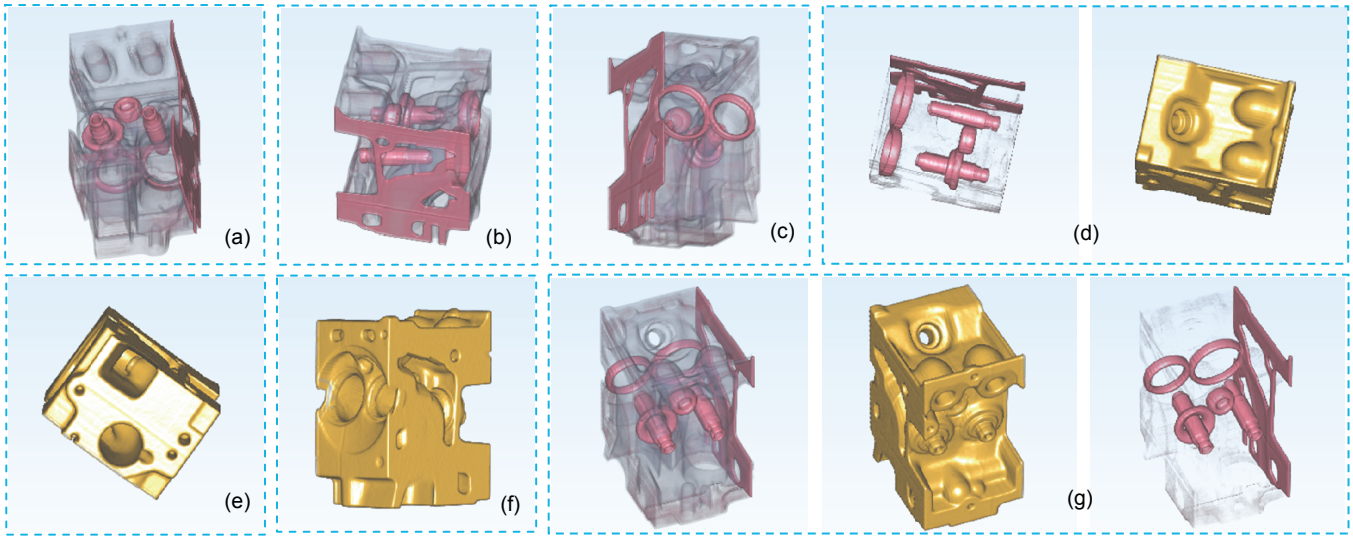


Fig. 12. The engine dataset (7 viewpoints computed by the SCP solver). (a-g): the suggested viewpoints rendered with different transfer functions. (d) and (g) shows the need to use multiple transfer functions to explore features.

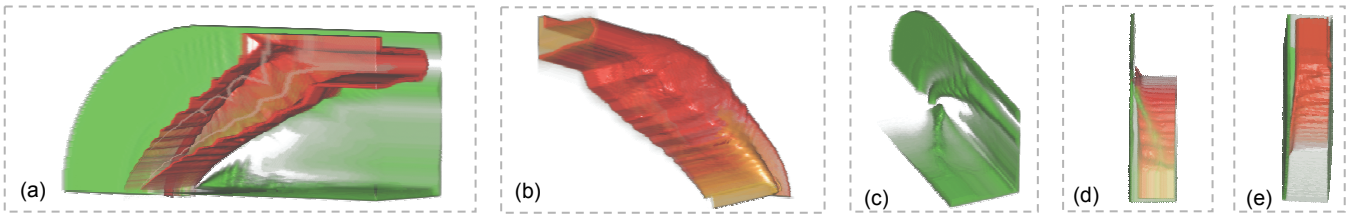


Fig. 13. The bluntfin dataset (5 viewpoints resulting from the SCP solver).

For future work, we plan to extend our feature descriptor to other known metrics, such as suggestive contours (where the derivative of the normal vector is 0) and other well-known descriptors from computer vision: the multi-scale Harris detector [16] or SIFT [24], which we used already in other work for feature detection [27]. Further, we also believe that the silhouette metric would be a promising candidate for dynamic flow visualization and we plan to research this more thoroughly. Finally, we also plan to implement the ant colony optimization on the GPU to reduce response time.

ACKNOWLEDGEMENTS

This work was funded in part by NSF EAGER grant 1050477 and NSF grant 0959979. It was also made possible in part by the imageVis3D software from the NIH/NCRR Center for Integrative Biomedical Computing, 2P41 RR0112553-12.

REFERENCES

[1] S. Abbasi and F. Mokhtarian, "Automatic view selection in multi-view object recognition," *In Proc. of International Conference on Pattern Recognition*, volume 1, pages 13-16, 2000.

[2] A. Amirkhanov, C. Heinzl, M. Reiter, and M. E. Gröller, "Visual optimality and stability analysis of 3DCT scan positions," *IEEE Transactions on Visualization and Computer Graphics,* 16(6): 1477-1487, 2010.

[3] J. E. Beasley, "OR-Library: distributing test problems by electronic mail," *Journal Operational Research Society*, 41(11):1069-1072, 1990.

[4] P. S. Blaer and P. K. Allen, "View planning and automated data acquisition for 3-D modeling of complex sites", *Journal of Field Robotics*, 26(11): 865-891, 2009.

[5] U. Bordoloi and H.-W. Shen, "View selection for volume rendering," *In Proceedings of the IEEE Visualization,* pages 487-494, 2005.

[6] M.-Y. Chan, H. Qu, K.-K. Chung, W.-H. Mak, and Y. Wu, "Relation-aware volume exploration pipeline". *IEEE Transactions on Visualization and Computer Graphics*, 14(6): 1683-1690, 2008.

[7] M.-Y. Chan, Y. Wu, W.-H. Mak, W. Chen, and H. Qu, "Perception-based transparency optimization for direct volume rendering," *IEEE Transactions on Visualization and Computer Graphics*, 15(6): 1283-1290, 2009.

[8] C. Correa and K.-L. Ma, "Size-based transfer functions: a new volume exploration technique," *IEEE Transactions on Visualization and Computer Graphics*, 14(6): 1380-1387, 2008.

[9] C. Correa and K.-L. Ma, "The occlusion spectrum for volume classification and visualization," *IEEE Transactions on Visualization and Computer Graphics*, 15(6): 1465-1472, 2009.

[10] C. Correa and K.-L. Ma, "Visibility histograms and visibility-driven transfer functions," *IEEE Transactions on Visualization and Computer Graphics*, 17(2): 192-204, 2011.

[11] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, "Suggestive contours for conveying shape," *ACM Transactions on Graphics*, 22(3): 848-855, 2003.

[12] T. Fogal and J. Krüger, "Tuvok - an architecture for large scale volume rendering," *In Proceedings of the 15th International Workshop on Vision, Modeling, and Visualization*, 2010.

[13] W. Fang, K.-K. Lau, M. Lu, X. Xiao, C.-K. Lam and P. Y. Yang, B. He, Q. Luo, P. V. Sander, and K. Yang, "Parallel data mining on graphics processors," Technical Report, HKUST-CS08-07, 2008.

[14] S. Fleishman, D. Cohen-Or, and D. Lischinski, "Automatic camera placement for image-based modeling," *Computer Graphics Forum*, 19(2):101-110, 2000.

[15] http://www.imagevis3d.org, ImageVis3D: A real-time volume rendering tool for large data. Scientific Computing and Imaging Institute (SCI).

[16] C. Harris and M. Stephens, "A combined corner and edge detector," *Proc. 4th Alvey Vision Conf*. pp. 147–151, 1988.

[17] R. Karp, "Reducibility among combinatorial problems," *Complexity of Computer Computations*. pp. 85-103, 1972.

[18] A. E. Kaufman, S. Lakare, K. Kreeger, and I. Bitter, "Virtual colonoscopy," *Communication of the ACM,* 48(2): 37-41, 2005.

[19] D. J. Ketchen and C. L. Shook, "The application of cluster analysis in strategic management research: an analysis and critique," *Strategic Management Journal*, 17(6): 441-458, 1996.

[20] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Möller, "Curvature-based transfer functions for direct volume rendering: methods and applications," *In Proceedings of the IEEE Visualization,* pages 513-520, 2003.

[21] J. Kniss, G. Kindlmann, and C. Hansen, "Multi-dimensional transfer functions for interactive volume rendering," *IEEE Transactions on Visualization and Computer Graphics*, 8(3): 270-285, 2002.

[22] P. Kohlmann, S. Bruckner, A. Kanitsar, and M. E. Gröller, "LiveSync: deformed viewing spheres for knowledge-based navigation," *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1544-1551, 2007.

[23] J. Krüger, J. Schneider, and R. Westermann, "ClearView: An interactive context preserving hotspot visualization technique," *IEEE Transactions on Visualization and Computer Graphics*, 12(5): 941-948, 2006.

[24] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Intern. J. Computer Vision*, 60(2):91-110, 2004.

[25] R. Maciejewski, I. Woo, W. Chen, and D. Ebert, "Structuring feature space: a non-parametric method for volumetric transfer function generation," *IEEE Transactions on Visualization and Computer Graphics*, 15(6): 1473-1480, 2009.

[26] W.-H. Mai, Y. Wu, M.-Y. Chan, and H. Qu, "Visibility-aware direct volume rendering," *Journal of Computer Science and Technology*, 26(2): 217-228, 2011.

[27] J. Nam, M. Mauer, and K. Mueller, "High dimensional feature descriptors to characterize volumetric data," Knowledge-Assisted Visualization Workshop, 2008.

[28] D. Pelleg and A. Moore, "X-means: extending k-means with efficient estimation of the number of clusters," *In Proc. of the 17th International Conference on Machine Learning*, pages 727-734, 2000.

[29] R. Pito, "A solution to the next best view problem for automated surface acquisition," *IEEE Trans. of Pattern Analysis and Maching Intelligence*, 21(10):1016-1030, 1999.

[30] M. Rahoual, R. Hadji, and V. Bachelet, "Parallel ant system for the set covering problem," *Lecture Notes in Computer Science*, volume 2463, pages 249-297, 2002.

[31] Z. Ren, Z. Feng, L. Ke, and Z. Zhang, "New ideas for applying ant colony optimization to the set covering problem," *Computers and Industrial Engineering*, 58(4):774-784, 2010.

[32] W. Scott, G. Roth, and J. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," *ACM Computing Surveys*, 35(1):64-96, 2003.

[33] P. Sereda, A. Vilanova, and F. A. Gerritsen, "Automating transfer function design for volume rendering using hierarchical clustering of material boundaries," *In Proc. of EuroVis 2006*. pages 243-250, 2006.

[34] N. Sudarsanam, K. Singh, and C. Grimm, "Non-linear perspective widgets for creating multiple-view images," *Symposium on Non-photorealistic Animation and Rendering*, pages 69-79, 2008.

[35] S. Takahashi, I. Fujishiro, Y. Takeshima, and T. Nishita, "A feature-driven approach to locating optimal viewpoints for volume visualization," *In Proceedings of the IEEE Visualization,* pp. 495-502, 2005.

[36] P.-P. Vazquez, M. Feixas, M. Sbert, and W. Heidrich, "Viewpoint selection using view entropy," *In Proc. of Vision Modeling and Visualization Conference*, pages 273-280, 2001.

[37] P.-P. Vazquez, M. Feixas, M. Sbert, and W. Heidrich, "Automatic view selection using viewpoint entropy and its application to image-based modeling," *Computer Graphics Forum*, 22(4):689-700, 2003.

[38] I. Viola, A. Kanitsar, and M. E. Gröller, "Importance-driven feature enhancement in volume visualization", *IEEE Transactions on Visualization and Computer Graphics*, 11(4): 408-418, 2005.

[39] L. Wang, Y. Zhao, K. Mueller, and A. E. Kaufman, "The magic volume lens: an interactive focus+context technique for volume rendering," *In Proceedings of the IEEE Visualization*, pages 367-374, 2005.

[40] S. Wenhardt, B. Deutsch, J. Hornegger, H. Niemann, and J. Denzler, "An information theoretic approach for next best view planning in 3-D reconstruction," *In Proc. of International Conference on Pattern Recognition*, pp.103-106, 2006.

[41] Y. Wu, K.-K. Chung, H. Qu, X. Yuan, and S.-C. Cheung, "Interactive visual optimization and analysis for RFID benchmarking," *IEEE Transactions on Visualization and Computer Graphics*, 15(6): 1335-1342, 2009.

[42] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah, "Shape from shading: a survey," *IEEE Trans. of Pattern Analysis and Maching Intelligence*, 21(8): 690-706, 1999.