View Selection for Volume Rendering

Udeepta D. Bordoloi* Han-Wei Shen†

The Ohio State Univesity

ABSTRACT

In a visualization of a three-dimensional dataset, the insights gained are dependent on what is occluded and what is not. Suggestion of interesting viewpoints can improve both the speed and efficiency of data understanding. This paper presents a view selection method designed for volume rendering. It can be used to find informative views for a given scene, or to find a minimal set of representative views which capture the entire scene. It becomes particularly useful when the visualization process is non-interactive – for example, when visualizing large datasets or time-varying sequences. We introduce a viewpoint "goodness" measure based on the formulation of entropy from information theory. The measure takes into account the transfer function, the data distribution and the visibility of the voxels. Combined with viewpoint properties like view-likelihood and view-stability, this technique can be used as a guide which suggests "interesting" viewpoints for further exploration. Domain knowledge is incorporated into the algorithm via an importance transfer function or volume. This allows users to obtain view selection behaviors tailored to their specific situations. We generate a view space partitioning, and select one representative view for each partition. Together, this set of views encapsulates the "interesting" and distinct views of the data. Viewpoints in this set can be used as starting points for interactive exploration of the data, thus reducing the human effort in visualization. In non-interactive situations, such a set can be used as a representative visualization of the dataset from all directions.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms

Keywords: viewpoint selection, view space partitioning, volume rendering, entropy, visibility

1 INTRODUCTION

With the advent of faster hardware and better algorithms, volume rendering has become a popular method for visualizing volumetric data. The traditional challenge of speeding up the rendering to achieve interactive frame-rates has been overcome for small datasets, but large datasets still pose problems for users who do not have access to supercomputing facilities. Slow frame-rates, coupled with large datasets, result in an increase in time needed to gain insights from the data. The efficiency of the visualization process can be increased by guiding the user to more informative parts of the data (or parameters) and thus saving time she would have otherwise spent in a trial-and-error search. Another option is to show more information on the screen without having a negative effect (such as, due to occlusion or cluttering). For example, various alternative rendering techniques can be used to provide a more understandable picture to the user [4][7]. Users can be guided to interesting features, isosurfaces and transfer functions by methods that suggest such candidates [9][18].

In this paper, we present a novel view selection method for volume rendering that can help improve the effectiveness of visualization by guiding the user to views that convey more information. Such interesting viewpoints are helpful both for data exploration and data presentation. For complex datasets, it is very difficult to manually find a view that maximizes the visibility of the relevant part of the data and minimizes occlusion. Currently, users can only use subjective judgment to evaluate and compare views. Our view selection technique introduces a measure to evaluate a view based on the amount of information displayed (and not displayed). It gives the users the ability to objectively compare two different views. The algorithm can be used to generate viewing positions to be used as starting viewpoints for browsing. Such suggested starting camera positions prove very beneficial in rendering situations with non-interactive frame-rates. Because of the time-lag between frames, users do not want to, and should not be made to [16][6][2], search the whole view-space for desirable views. The algorithm can also be used when presenting data in a non-interactive setting. It creates a smart partitioning of the view space, and selects representative views from each view group for rendering.

For this paper, we assume that the dataset is centered at the origin and that the camera is always looking at the origin from a fixed distance. We will refer to this set of camera positions as the view sphere. To evaluate and compare viewpoints, we use three viewpoint characteristics associated with each view:

- **View goodness:** The view-goodness measure tries to capture how closely the voxel visibilities for a given view match a user-input importance function. We define a view to be good if more important voxels in the volume are highly visible, and vice versa. It is maximized when the voxel visibilities are proportional to their importance. When selecting viewpoints, it is desirable that they have high goodness scores.

- **View likelihood :** Intuitively, the view likelihood of a given view is the number of other viewpoints on the view sphere which yield a view that is similar (defined by a threshold) to the given view. We define the view similarities in terms of voxel visibilities and importance that are used for view goodness. A highly likely view is a good candidate for representing the dataset from different views. On the other hand, low likelihood views are interesting because they display information that is not seen from most other viewpoints, and hence is likely to be missed by users during an interactive search.

- **View stability :** View stability of a view denotes the maximal change in view that can occur when the camera position is shifted within a small neighborhood (defined by a threshold). A small change implies a stable view, and a large change would make a view unstable. Unstable views make good starting viewpoints during interactive visualization, because the user can see a large change in view with a small mouse movement.

The contribution of this paper is as follows. We introduce a goodness measure of viewpoints based on the information theory

*Email: udeepta at acm.org

†Email: hwshen at cse.ohio-state.edu

concept of entropy, also called average information. We propose that good viewpoints are ones which provide higher visibilities to the more important voxels. This interpretation leads us to the formulation of viewpoint information presented in this paper. We utilize a property of our entropy definition which indicates that when the visibilities are close to their desired values, the viewpoint information is maximized. This measure allows us to compare different viewpoints and suggest the best ones to the user. The voxel importance can be specified by the users based on their domain knowledge and the desired output. Given a desired number N of views, our algorithm can be used to find the best N viewpoints over the view space. A GPU-based algorithm is used to find the visibilities at the exact voxel centers of the volume. We also use the framework to find similarity between views, which is then used to create a view space partitioning and to find the likelihood and stability of views. Representative views for each partition can be chosen either by taking the highly likely or highly unlikely views. In interactive situations, our method can suggest unstable viewpoints, so that a small change in the camera position will yield a large change in view. For time-dependent data, we present a modification of the goodness measure of a viewpoint by taking into account not only the static information but also the change in each time-step.

2 Related Work

The idea of comparing different views developed much before computer graphics and visualization matured. As early as 1976, Koenderink and van Doorn [10][11] had studied singularities in 2D projections of smooth bodies. They showed that for most views (called stable views), the topology of the projection does not change for small changes in the viewpoint. The topological changes between viewpoints can be stored in an aspect graph. Each node in the graph represents a region of stable views, and each edge represents a transition from one such region to an adjacent one. These regions form a partitioning of the view space, which is typically a sphere of a fixed radius with the object of interest at its center. The aspect graph (or its dual, the view space partition) defines the minimal number of views required to represent all the topologically different projections of the object. A lot of research has been done since the early papers, mainly in the field of computer vision, which extended the ideas to more complex objects. In the case of volume rendering, a similar topology based partitioning can not be constructed. Instead, we find a visibility based partitioning by comparing visibilities of voxels in neighboring views, and clustering together viewpoints that are similar.

Viewpoint selection has been an active topic of research in many fields. For instance, viewpoint selection solutions have been proposed for the problem of modeling a three-dimensional object from range data [21] and from images [5], for object recognition [1], and also for cinematography [8]. However, the topic has not been well investigated in the fields of computer graphics and visualization, possibly because applications in these domains have relied heavily on human control. Recently, Vázquez et al. [19][20] have presented an entropy based technique to find good views of polygonal scenes. They define an entropy for any given view, which is derived from the projected area of the faces of the geometric models in the scene. Their motivation is to achieve a balance between the number of faces visible and their projection areas. The entropy value is maximized when all the faces project to an equal area on the screen. In this conference, two solutions are being presented for selecting good viewpoints for volumetric data. Takahashi et al. [17] calculate the view entropy for different isosurfaces and interval volumes, and then find a weighted average of the individual components. The weights are assigned using the transfer function opacities. The viewpoint measure presented in this paper is also based on the entropy function, but is defined on voxels as opposed to geometric objects. Each voxel in the data is assigned a visual significance, and the entropy is maximized when the visibilities of the voxels approach the respective significance values.

3 Viewpoint Evaluation

The essential goal of this paper is to have a computer suggest 'good' viewpoint(s) to the user. This naturally leads us to the question: "what is a good viewpoint?", or, "what makes a viewpoint better than another?". The answer will depend greatly on the viewing context and the desired outcome. For example, a photographer will choose the view which best contributes to the chosen mood and visual effect. For this paper, the context is the process of volume rendering, which is being used to get visual information from the data. Hence, for our purposes, *a viewpoint is better than another if it conveys more information about the dataset*. In this section, we present a method for quantifying the information contained in a view using properties of the entropy function from information theory.

The information that is transferred from a volumetric dataset to the two-dimensional screen is governed by the optical model which is used for the projection. In this paper, we assume the popular absorption plus emission model [14]. The intensity Y at a pixel D is given by

$$Y(D) = Y_0 T(0) + \int_0^D g(s) T(s) ds \qquad (1)$$

where, $T(s)$ is the transparency of the material between the pixel D and the point s. We will refer to $T(s)$ as the visibility of the location s. The first term in the equation represents the contribution of the background, Y_0 being its intensity. The second term adds the contributions of all the voxels along the viewing ray passing through D. A voxel at point s has an emission factor of $g(s)$, and its effect on the pixel intensity is scaled by its visibility $T(s)$. If two voxels have the same emission factor, then the one with a higher visibility will contribute more toward the final image.

The emission factors of voxels are usually defined by the users. They set the transfer function to highlight the group of voxels they want to see, and to make the others more transparent. We use this fact to define a *noteworthiness* factor for each voxel (section 3.2), which captures, among other things, the importance of the voxel as defined by the transfer function. Users can also specify the noteworthiness by other means – for example, using a separate importance volume. Based on the preceding discussion, we have the following two (not necessarily disjoint) guidelines for defining a good viewpoint:

1. A viewpoint is good if voxels with high noteworthiness factors have high visibilities.

2. A viewpoint is good if the projection of the volumetric dataset contains a high amount of information.

In the following section, we present the details of our view information function and its properties.

3.1 Entropy and View Information

Consider any information source X which outputs a random sequence of symbols taken from the alphabet set $\{a_0, a_1, \ldots, a_{J-1}\}$. Suppose the symbols occur with the probabilities $\mathbf{p} = \{p_0, p_1, \ldots, p_{J-1}\}$. Alternatively, we can think of it as the random variable X which gets the value a_j with probability p_j. The information associated with a single occurrence of a_j is defined in information theory as $I(a_j) = -\log p_j$. The logarithm can be taken with base 2 or e, and the unit of information is bits or nats respectively. In a sequence of length n, the symbol a_j will occur

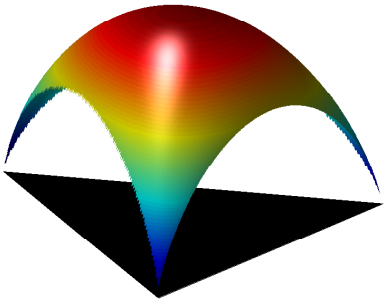


Figure 1: Entropy Function for three dimensional probability vectors $\mathbf{p} = \{p_0, p_1, p_2\}$. The function is defined only over the plane $p_0 + p_1 + p_2 = 1$, within the triangular region specified by $0 \le p_1, p_2, p_3 \le 1$. The maximum occurs at the point $p_0 = p_1 = p_2 = 1/3$, and the value falls as we move away from that point in any direction. So, increasing the entropy has the effect of making the probabilities more uniform.

np_j times, and will carry $-np_j \log p_j$ units of information. Then the *average information* of the sequence, also called its entropy, is defined as

$$H(X) \equiv H(\mathbf{p}) = -\sum_{j=0}^{J-1} p_j \cdot \log_2 p_j \qquad \text{bits/symbol} \qquad (2)$$

with $0 \cdot \log_2 0$ defined as zero [3]. Even though the entropy is frequently expressed as a function of the random variable X, it is actually a function of the probability distribution $\mathbf{p}$ of the variable X. We will use the following two properties of the entropy function in constructing our viewpoint evaluation measure:

1. For a given number of symbols J, the maximum entropy occurs for the distribution $\mathbf{p_{eq}}$, where $\{p_0 = p_1 = \ldots = p_{J-1} = 1/J\}$. (See figure 1, which gives an example of the entropy values for a three dimensional distribution.)

2. Entropy is a concave function, which implies that the local maximum at $\mathbf{p_{eq}}$ is also the global maximum. It also implies that as we move away from the equal distribution $\mathbf{p_{eq}}$, along a straight line in any direction, the value of entropy decreases (or remains the same, but does not increase).

We will use probability distributions associated with views to calculate their entropy (average information). For each voxel j, we define an importance factor W_j. We will call it the *noteworthiness* of the voxel, and it indicates the visual significance of the voxel. (More details about W_j are given in section 3.2). Suppose, for a given view V, the visibility of the voxel is $v_j(V)$. We are using the term 'visibility' to denote the transparency of the material between the camera and the voxel. It is equivalent to $T(s)$ in equation (1). Then, for the view V, we define the *visual probability*, q_j, of the voxel as

$$q_j \equiv q_j(V) = \frac{1}{\sigma} \cdot \frac{v_j(V)}{W_j} \qquad \text{where, } \sigma = \sum_{j=0}^{J-1} \frac{v_j(V)}{W_j} \qquad (3)$$

where the summation is taken over all voxels in the data. The division by σ is required to make all probabilities add up to unity. Thus, for any view V, we have a visual probability distribution $\mathbf{q} \equiv \{q_0, q_1, \ldots, q_{J-1}\}$, where J is the number of voxels in the dataset. Then, we define the entropy (average information) of the view to be

$$H(V) \equiv H(\mathbf{q}) = -\sum_{j=0}^{J-1} q_j \cdot \log_2 q_j \qquad (4)$$

The view with the highest entropy is then chosen as the best view. This satisfies the two guidelines presented earlier in section 3:

1. The best view has the highest information content (averaged over all voxels).

2. The visual probability distribution of the voxels is the closest (of all the given views) to the equal distribution $\{q_0 = q_1 = \ldots = q_{J-1} = 1/J\}$, which implies that the voxel visibilities are closest to being proportional to their noteworthiness.

To calculate the view entropy, we need to know the voxel visibilities and the noteworthiness factors. Visibilities can be queried through any standard volume rendering technique such as ray casting. The noteworthiness, described in the next section, is view independent, and needs to be calculated only once for a given transfer function.

3.2 Noteworthiness

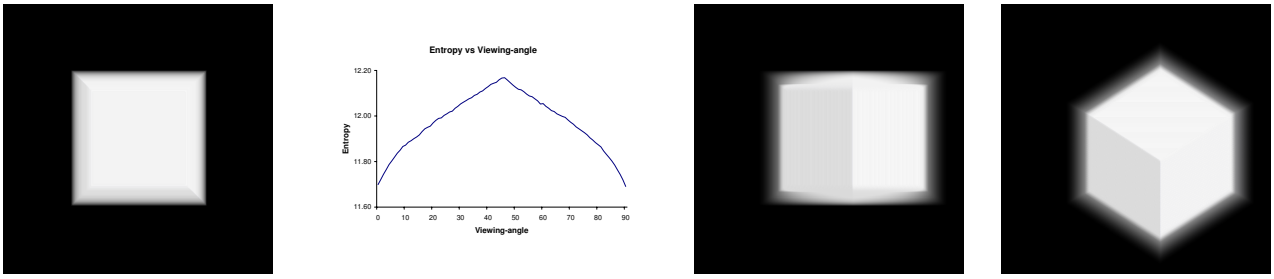
The noteworthiness factor of each voxel denotes the significance of the voxel to the visualization. It should be high for voxels which are desired to be seen, and vice versa. Considering the diverse array of situations volume rendering is used in, it is practically impossible to give a single definition of noteworthiness that satisfies expectations of all users. Instead, we can rely on the user-specified transfer functions to deliver us a definition which is tailor-made for the particular situation. The opacity of a voxel, as assigned by the transfer function, is part of the emission factor $g(s)$ in equation (1), and controls the contribution of the voxel to the final image. We use opacity as one element of the noteworthiness of the voxel. Another consideration is that voxels of a particular color can be more visually meaningful to the viewer than voxels of another color. Consider a scene with voxels of two different colors. If there are fewer voxels of the first color compared to the second, then it is possible that the second group of voxels completely occludes the first one. Also, Gestalt principles [15] suggest that the human mind extrapolates the larger object (called ground) behind the smaller one (called figure). Hence, other factors (such as opacities) being equal, we would want the voxels of the first color to have a higher importance than the others.

Based on these observations, we construct the noteworthiness W_j of the jth voxel as follows. We assign probabilities to voxels in our dataset by constructing a histogram of the data. All the voxels are assigned to bins of the histogram according to their value, and each voxel gets a probability from the frequency of its bin. The information I_j carried by the jth voxel is then $-\log f_j$, where f_j is its probability (bin frequency). Then, W_j for the voxel is $\alpha_j I_j$, where α_j is its opacity. We ignore voxels whose opacities are zero or close to zero, and these voxels are not included in the evaluation of equation (4). This reduces the computational and memory requirements for the entropy and similarity calculations (section 4.1).

The answer to what is interesting and what is not is very subjective. Our algorithm can be made to suit the goals of any particular visualization situation just by changing the noteworthiness factors. Domain specific knowledge can be readily incorporated into the framework – for example, using a separate importance volume which gives importance values for each voxel. It can be used in conjunction with, or in place of, the noteworthiness criteria described in this section. Irrespective of the method used to specify the interestingness of the voxels, maximizing the entropy serves to give better visibility to the more interesting voxels.

3.3 A Simple Example

To demonstrate our concept of view information, we constructed the test dataset shown in figure 2. The voxel opacities of the cube



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

Figure 2: An illustration of the change in view entropy (equation 4) with camera position for a test dataset. Figure (a) shows the initial position of the camera. Figure (b) shows the behavior of entropy as the camera revolves around the dataset (around the vertical axis in the figure) at 1° increments. The entropy increases steadily and reaches a maximum for a movement of 45° (figure (c)), and then begins to decrease again. The maximum entropy for the whole view space is obtained for the view in figure (d).

dataset increase linearly with distance from the boundary of the cube. Figure 2(a) shows the volume rendering the dataset when the camera is looking directly at one of its faces. Next, we revolve the camera about the vertical axis of the dataset at 1° increments (or equivalently, rotate the dataset in the opposite direction about the vertical axis). The view entropy steadily increases (figure 2(b)) as more and more voxels on the side face start becoming visible. It reaches a maximum when camera has moved by 45°, which is the view that shows the two faces equally (figure 2(c)). Further movement of the camera results in greater occlusion of voxels near the first face, and the entropy begins to drop again. Upon evaluating the entropies for all camera positions around the dataset, the view in figure 2(d) results in the highest entropy. Clearly, this is one of the more informative views about the cube dataset for a human observer.

3.4 Finding the Good View

The view selection proceeds as follows. The dataset is centered at the origin, and the camera is restricted to be at a suitable fixed distance from the origin. This spherical set of all possible camera positions defines the view sphere, and represents all the view directions. The view space is then sampled by placing the camera at sample points on this sphere. We create a uniform triangular tessellation of the sphere and place the viewpoints at the triangle centroids. The camera position and the origin specify the eye and the center points respectively for the modelview transformation. Since the roll of the camera does not affect the visibilities, the up vector can be arbitrarily chosen.

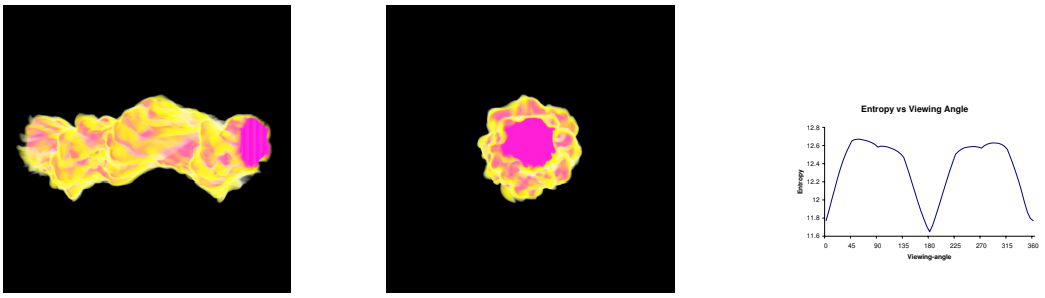
Next, the voxel visibilities are calculated for each sample view position. Our technique is not dependent on any particular volume rendering method, and both software and hardware renderers can be used by modifying them to output voxel visibilities. (Please note that the transparency or voxel visibility as given in equation (1) is numerically the same as the accumulated opacity subtracted from unity.) Since we will be comparing the visual probability distributions ($\mathbf{q}$) and the entropies ($H(\mathbf{q})$) of different views, it is desirable that we compute the visibilities at the same locations for all views (say, at the voxel centers). Most renderers, however, do not perform the opacity calculations exactly at the voxel centers. Ray-casters accumulate opacities along the rays, and texture based renderers accumulate opacities at frame-buffer pixel locations, neither of which are necessarily aligned with voxel centers. We use the GPU to calculate the visibilities at the exact voxel centers by rendering the volume slices in a front-to-back manner using a modified shear-warp algorithm. We give a brief description of our implementation below.

The object-aligned slicing direction is taken along the axis which is most perpendicular to the viewing plane. We use a floating point p-buffer with the same resolution as the volume slices. The first slice has no occlusion, so all the voxels in this slice have their visibilities set to unity. We iterate through the rest of the slices in a front-to-back order. In each loop, we calculate the visibilities of a slice based on the data opacities and the visibilities of the immediate previous slice. In each iteration, the following actions are performed. The frame-buffer (p-buffer) is cleared, and the camera set such that the current slice aligns perfectly with the frame-buffer. Then, the immediate previous slice is rendered with a relative shear[12], and a fragment program combines its opacities with its visibility values. The frame-buffer now contains the visibilities of the current slice, and these will be used in the next loop. Render to texture techniques are used to prevent a copy from the frame-buffer to a texture. After all the slices are processed, the entropy for the given view direction is calculated using equations (3) and (4).

Figure 3 shows a time-step of a 256-cube shockwave dataset. The camera was rotated about the vertical axis in a complete circle, with the dataset centered at the origin. Entropy was evaluated at 5° increments. The first figure shows the view at 55° rotation which was the view with the highest entropy. Figure(b) shows the worst view, which occurred at 180°. Figure 4 shows a $128 \times 128 \times 80$ tooth dataset. The view sphere was sampled at 128 points. Figures (a) and (b) have the highest view entropy values. Figures (c) and (d) have the lowest entropy, and not surprisingly, are highly occluded views. It is notable that the viewpoints for (c) and (d) are not very far apart, and that (a) and (b) show much of the same voxels. This shows that if the user wants a few (say, N) good views from the algorithm, returning the N highest entropy views might not be the best option. Instead we can try to find a set of good views whose view samples are well distributed over the view sphere. The next section presents such a solution.

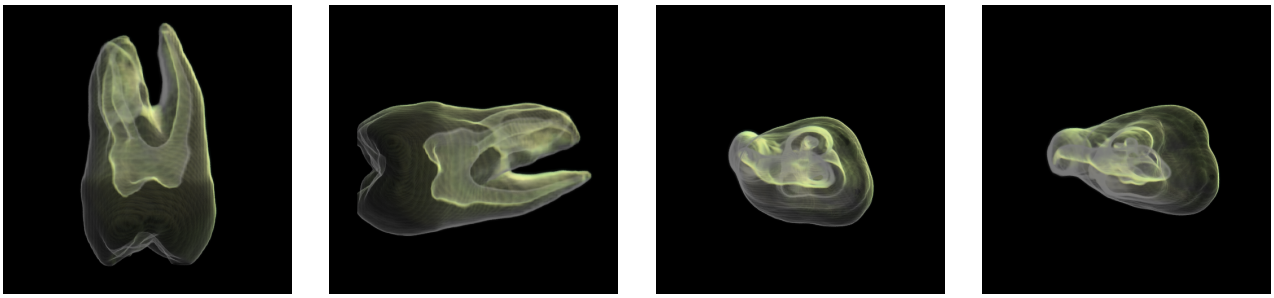
4 View Space Partitioning

The goodness measure presented in the previous section can be used as a yardstick to measure the information captured by different volume rendering views and select the best view. But the calculation of goodness considers information from only the given view, and ignores the information that might be contained in other views. In particular, neighboring viewpoints tend to have similar visibilities, and comparing a viewpoint with its neighbors can provide additional properties of the viewpoint. Also, for most datasets, a single view does not give enough information to the user. The user will almost certainly want to look at the dataset from another angle. Instead of a single view, it is desirable to present to the user a set of



(a) (b) (c)

Figure 3: The figure shows a time-step of a 256-cube shockwave dataset. The camera was rotated about the vertical axis in a complete circle, with the dataset centered at the origin. Entropy was evaluated at 5° increments. Figure (a) shows the view at 55° rotation which was the view with the highest entropy. Figure(b) shows the worst view, which occurred at 180°. Figure (c) plots the change of entropy with change in angle.



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

Figure 4: The two highest entropy views for the tooth dataset are shown in (a) and (b), and the two worst ones in (c) and (d).

views such that, together, all the views in the set provide a complete visual description of the dataset. This can also be thought of as a solution to the best N views problem: given a positive number N, we want to find the best N views which together give the best visual representation of the dataset.

We propose to find the N views by partitioning the view sphere into N disjoint partitions, and selecting a representative view for each partition. A similar partitioning is defined by aspect graphs [10][11], where each node (aspect) of the graph represents a set of stable views. Each set shows the same group of features on the surface of the object. However, the aspect graph creation methods deal mostly with algebraic and polygonal models and their topology, and cannot be applied in a straightforward manner to volume rendering. Instead, we compute the partitioning by grouping similar viewpoints together.

4.1 View Similarity

To find the (dis)similarity of viewpoints, we use the visual probability distributions associated with each viewpoint (section 3.1). Popular measures for computing the dissimilarity between two distributions $\mathbf{p}$ and $\mathbf{p}'$ are the relative entropy (also known as the Kullback-Leibler (KL) distance), and its symmetric form (known as divergence) which is a true metric [3]. (Please note that some texts refer to the KL distance as divergence instead.)

$$D(\mathbf{p}\|\mathbf{p}') = \sum_{j=0}^{J-1} p_j \log \frac{p_j}{p'_j} \quad (5)$$

$$D_s(\mathbf{p},\mathbf{p}') = D(\mathbf{p}\|\mathbf{p}') + D(\mathbf{p}'\|\mathbf{p}) \quad (6)$$

Although these measures have some nice properties, there are some issues with these measures that make them less than ideal. If $p'_j = 0$ and $p_j \neq 0$ for any j, then $D(\mathbf{p}\|\mathbf{p}')$ is undefined. In our case, any voxel which is fully occluded (zero visibility) will get a visual probability q_j of zero (equation (3)). If it is visible in one view but occluded in the other, we cannot evaluate equation 5 for these views. Also, $D(\mathbf{p}\|\mathbf{p}')$ and $D_s(\mathbf{p},\mathbf{p}')$ do not offer any nice upper-bounds. To overcome these problems, we instead use the Jensen-Shannon divergence measure [13]:

$$JS(\mathbf{p},\mathbf{p}') = JS(\mathbf{p}',\mathbf{p}) = K(\mathbf{p},\mathbf{p}') + K(\mathbf{p}',\mathbf{p}) \quad (7)$$

$$\text{where, } K(\mathbf{p},\mathbf{p}') = D(\mathbf{p}\|(\frac{1}{2}\mathbf{p} + \frac{1}{2}\mathbf{p}')) \quad (8)$$

The distance between two views V_1 and V_2, with distributions $\mathbf{q}_1$ and $\mathbf{q}_2$, is then defined as $JS(\mathbf{q}_1,\mathbf{q}_2)$. This measure does not have the zero visual probability problem, since the denominator of the log term is zero iff the numerator is zero. It is also nicely bounded by $0 < JS(\mathbf{q}_1,\mathbf{q}_2) < 2$. Moreover, it can be expressed in terms of entropy [13], which allows us to reuse the view information calculations given in equation (4):

$$JS(\mathbf{q}_1,\mathbf{q}_2) = 2H(\frac{1}{2}\mathbf{q}_1 + \frac{1}{2}\mathbf{q}_2) - H(\mathbf{q}_1) - H(\mathbf{q}_2) \quad (9)$$

4.2 View Likelihood and Stability

We can now use the definition of view-distance given by equation (9) to define two additional characteristics of viewpoints – view

likelihood and view-stability. View likelihood of a view V is defined as the probability of finding another view anywhere on the view-sphere whose view-distance to V is less than a threshold. In our scenario, it is given the number of view samples on the view sphere that are within the threshold of V. If a view has a (relatively) high likelihood, it implies that the object or dataset projects a similar image for a (relatively) large number of views. On the other hand, a view with low likelihood provides information that is unique to a few views. This property is indirectly taken into consideration when we partition the set of all the view samples (that is, the view sphere). Large partitions have views with high likelihoods.

Sometimes it is not the view itself but the change in view that provides important information. If the view is changed from one viewpoint to another very similar view, the user is not shown much new information. But, if the rendering changes by a large amount, the user sees not just the new information in the visualization but also derives knowledge from the change that has occurred. Occlusion is one of the most important depth cues that is available to the user when visualizing three-dimensional renderings on a two-dimensional surface. A large change in occlusion implies a large change in visibilities, which results in a large JS distance between two viewpoints. View stability is a view property that captures this information and can be used to select viewpoints during interaction. It is defined as the maximal change that occurs when the viewpoint is moved anywhere within a given radius from its original position. The greater the change, the more unstable a viewpoint is. We calculate the (un)stability as the maximum view-distance between a view sample and its neighboring view samples in the triangular tessellation of the view-sphere. The 180° viewpoint in figure 3(b) is an unstable viewpoint for this particular viewpoint sequence.

4.3 Partitioning

Once the visual probability distributions ($\mathbf{q}$) and their entropies ($H(\mathbf{q})$) are calculated as described in section 3, we use the *JS*-divergence to find the (dis)similarities between all pairs of view samples. We then cluster the samples to create a disjoint partitioning of view sphere. The number of desired clusters can be specified by the user. Each partition represents a set of similar views, i.e., these views show the voxels at similar visibilities. If desired, the *JS*-measure can be weighted using the physical distance between the view samples to yield tight regional clusters.

The best (highest entropy) views within each partition are selected as representatives of the cluster and displayed to the user. Together, this set of images gives a good visualization of the dataset from many different viewpoints. Sometimes, it might happen that the selected representatives of two neighboring partitions lie on the common boundary and next to each other. If the view distance between two selected view samples is less than a threshold, we use a greedy approach and select the next best sample.

Figure 5 shows the results of a 5-way partitioning of the view space for the tooth dataset. 128 view samples were used with a *JS*-divergence measure. The largest partition contains 39 samples, while the smallest one has 18. The representative views from four of the partitions are shown. The view for the fifth partition is figure 4(a). We would like to point out that figures 4(a) and 4(b) both are in the same partition. In fact, the top ten high entropy viewpoints all belong to the same partition, illustrating the need for selecting representative views from different partitions.

5 Time Varying Data

Suggestion of good views becomes all the more useful in the case of time dependent data. The time required to compute a volume rendering animation of the dataset grows with the number of time steps. In an interactive setting, this creates a large lag between a viewpoint update and the completion of rendering all the frames. Moreover, it takes more tries by the user to find the desired viewpoint because the data changes with time, and the user has to consider not only the current time step but also the previous and future ones. The user's job is made harder by cases where an interesting view in a few time steps turns out to be a dull view in the rest.

In section 3, we discussed the notion of a good view and presented a measure of view information for a volume dataset. For time-dependent data, using equation (4) separately for each time-step is not the best solution – it can yield viewpoints in adjacent time-steps that are far from each other, thus resulting abrupt jumps of the camera during the animation. A natural solution is to constrain the camera, but it still does not guarantee the most informative viewpoint. For instance, it can result in a viewpoint which has a high information value for each individual time-step, but does not show any time-varying changes. It is contrary to what is expected from an animation – it should show both the data at each time-step, and also the changes occurring from one frame to the next. In the next section, we present an alternate version of viewpoint information tailored to capture the view information present in one time-step, taking into account the information present in the previous step.

5.1 View Information

Consider two random variables X and Y with probability distributions $\mathbf{r}$ and $\mathbf{s}$ respectively. If X and Y are related (not independent), then an observation of X gives us some information about Y. As a result, the information carried by Y, conditional on observing X, becomes $H(Y|X) \equiv H(\mathbf{s}|\mathbf{r})$. Then the information carried together by X and Y is $H(X,Y) = H(Y,X) = H(X) + H(Y|X)$, as opposed to $H(X) + H(Y)$. We will use this concept to create a modified viewpoint goodness measure for time dependent data.

Suppose there are n time-steps $\{t_1, t_2, \ldots, t_n\}$ in the dataset. For a given view V, we denote the entropy for time-step t_i as $H(V,t_i) \equiv H_V(t_i)$. The view entropy for all the time-steps together is $H_V(t_1, t_2, \ldots, t_n)$. We will assume a Markov sequence model for the data, i.e., the data in any time-step t_i is dependent on the data of the time-step t_{i-1}, but independent of older time-steps. Then the information measure for the view, for all the time-steps taken together, is given by equation (11). (10) is a standard relation [3], and (11) follows from the independence assumption.

$$\begin{aligned} H(V) &= H_V(t_1, t_2, \ldots, t_n) \\ &= H(t_1) + H(t_2|t_1) + \ldots + H(t_n|t_1, \ldots, t_{n-1}) \quad (10) \\ &= H(t_1) + H(t_2|t_1) + \ldots + H(t_n|t_{n-1}) \quad (11) \end{aligned}$$

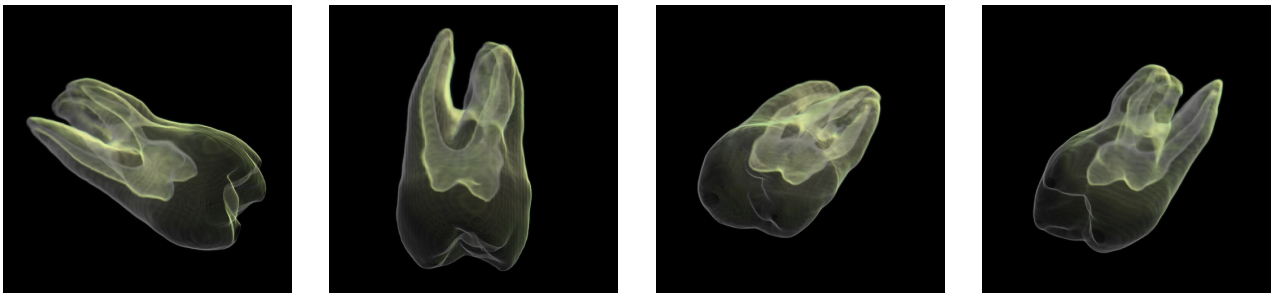
The conditional entropies will be defined following the same principles outlined in section 3. We consider a view to be good when the visibilities of the voxels are in proportion to their noteworthiness. But in the time-varying case, the significance of a voxel is derived not only from its opacity, but also from the change in its opacity from the previous time-step. For the time-step t_i, we then define the noteworthiness factor of the jth voxel as $W_j(t_i|t_{i-1}) =$

$$\{k \cdot |\alpha_j(t_i) - \alpha_j(t_{i-1})| + (1-k) \cdot \alpha_j(t_i)\} \cdot I_j(t_i) \quad (12)$$

where, $0 < k < 1$ is used to weight the effects of voxel opacities and the change in their opacities. A high value of k will highlight the changes in the dataset. Suppose the visibility of the voxel for the view V is $v_j(V,t_i)$. Then, the conditional visual probability, $q_j(t_i|t_{i-1})$, of the voxel is

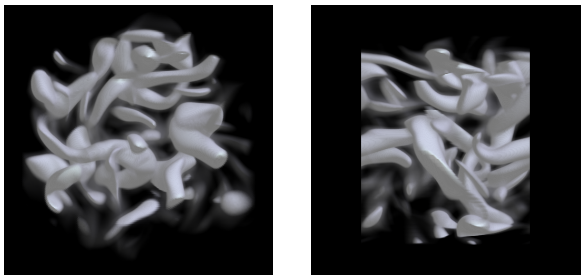
$$q_j(t_i|t_{i-1}) \equiv q_j(V,t_i|t_{i-1}) = \frac{1}{\sigma} \cdot \frac{v_j(V)}{W_j(t_i|t_{i-1})} \quad (13)$$

where, σ is the normalizing factor as in equation (3). The entropy of the view V is then calculated using equations (11) and (13). Voxels



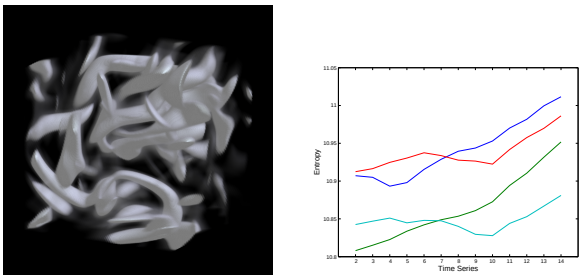
(a) (b) (c) (d)

Figure 5: Representative views for a 5-way partitioning of the view-sphere for the tooth dataset. The view for the fifth partition is figure 4(a).



(a) (b)

Figure 6: View Evaluation results for a 128-cube vortex dataset. Figure (a) shows the recommended view with a high entropy value, (b) shows a bad view for comparison.



(a) (b)

Figure 7: View Evaluation for time-varying dataset. (a) The best overall view for 14 time-steps. (b) The conditional entropies of four selected views for each of the 14 time-steps. The view in (a) is represented by the blue plot (highest curve, top-right corner).

with both low opacities and small changes (as defined by thresholds) are ignored for these calculations.

6 Results and Discussion

We have implemented our technique using a hardware-based visibility calculation (section 3.4). 128 sample views were used for each dataset. The camera positions were obtained by a regular triangular tessellation of a sphere with the dataset place at its center. The tests were run on a 2GHz P4, 8x AGP machine with a GeForce 5600 card. For a 128-cube dataset, the visibility calculations for all 128 views were completed in 42s, resulting in an average time per view of 0.33s. In case of a 256-cube dataset, the calculations for 128 views took 310s, with an average time per view of 2.42s.

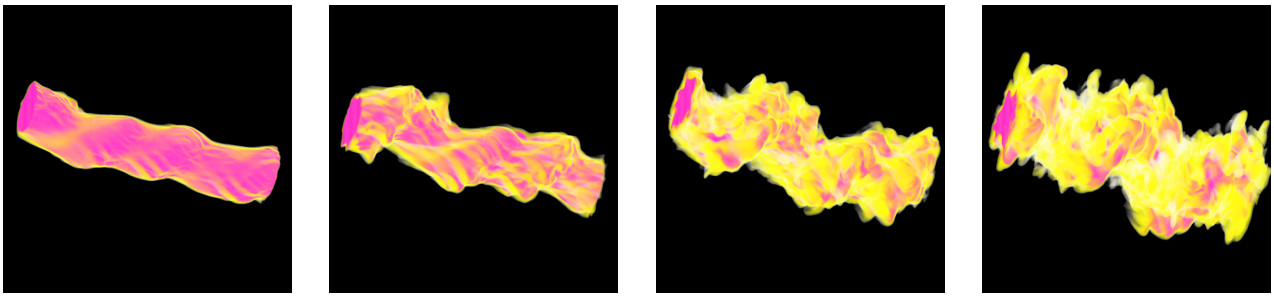
View selection results for the $128 \times 128 \times 80$ tooth dataset have been shown in figure 4. Figure 5 shows the results of a 5-way view space partitioning for the dataset using the *JS* divergence measure. The partitioning helps to avoid selection of a set of good views which happen to be similar to each other. Even though we have not considered the physical distance between the viewpoints during partitioning, it forces the selected viewpoints to be well distributed over the view sphere. Figure 6 shows view evaluation results for a 128-cube vortex dataset. Both high and low quality views are shown for comparison.

For time-varying data, we used the view information measure presented in section 5. A sequence of 14 time-steps of the 128-cube vortex data was used. The entropy for each view was summed over all the time-steps, as given by equation (11). The conditional entropy for each time-step was calculated with $k = 0.9$ in equation (12). A high value of k gives more weight to the voxels which are changing their values with time compared to high opacity voxels which remain relatively unaltered. Figure 7(a) shows the view with the best cumulative entropy for the time-series. Although the summed entropy gives a good overall view for the whole time-series, there might be other views which are better for particular segments of the time-series. Figure 7(b) plots the conditional entropies ($H(t_n|t_{n-1})$) for four selected views of the vortex dataset. The best overall view (figure 7(a)), which is represented by the blue curve (highest curve on the right edge), is not the best choice for the first half of the series. For long time sequences, it might be beneficial to consider different good views for different segments of time.

View entropy was calculated over fifty time-steps of the 256-cube shockwave dataset with 128 view samples. Figure 8 shows four time-steps from a viewpoint which had a good entropy using the time-varying criteria, and figure 9 shows the corresponding time-steps for a viewpoint which resulted in a bad score.

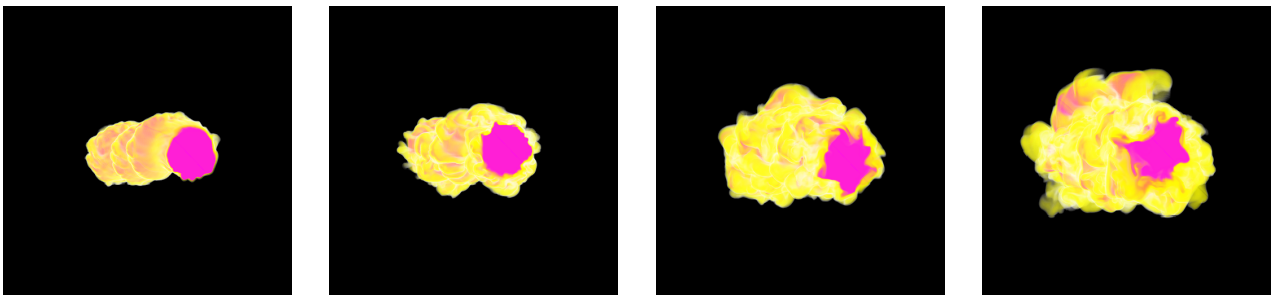
7 Conclusion and Future Work

We have presented a measure for finding the goodness of a view for volume rendering. We have used the properties of the entropy function to satisfy the intuition that good views show the noteworthy voxels more prominently. The user can set the noteworthiness of the voxels by specifying the transfer function, or by using an importance volume, or a combination of both. The algorithm can be used both as an aid for human interaction, and also as an oracle to present multiple good views in less interactive contexts. Furthermore, view sampling methods such as IBR can use the sample similarity information to create a better distribution of samples.



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

Figure 8: View entropy results over 50 time-steps of the 256-cube shockwave dataset. Time steps 1, 16, 31 and 46 for a good view.



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

Figure 9: Low entropy viewpoint for 50 time-steps of the 256-cube shockwave dataset. Time steps 1, 16, 31 and 46 are shown.

Acknowledgment

We would like to thank Antonio Garcia for his help with our implementation. This work was supported by NSF grant ACI-0329323, NSF ITR grant ACI-0325934, DOE Early Career Principal Investigator award DE-FG02-03ER25572, and NSF Career Award CCF-0346883.

References

[1] T. Arbel and F. Ferrie. Viewpoint selection by navigation through entropy maps. In *Proc. of the 7th IEEE International Conf. on Computer Vision (ICCV-99)*, volume I, pages 248–254. IEEE, 1999.

[2] R. E. Barber and H. C. Lucas Jr. System response time operator productivity, and job satisfaction. *Comm. of the ACM*, 26(11):972–986, 1983.

[3] R. E. Blahut. *Principles and practice of information theory*. Addison-Wesley Publ. Co., 1987.

[4] D. Ebert and P. Rheingans. Volume illustration: Non-photorealistic rendering of volume models. In *Proc. of IEEE Visualization '00*, pages 195–202, 2000.

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

[6] J. L. Guynes. Impact of system response time on state anxiety. *Comm. of the ACM*, 31(3):342–347, 1988.

[7] M. Hadwiger, C. Berger, and H. Hauser. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Proc. of IEEE Visualization '03*, pages 301–309, 2003.

[8] L-W. He, M. F. Cohen, and D. H. Salesin. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In *Proc. of SIGGRAPH '96*, pages 217–224, 1996.

[9] G. Kindlmann and J. W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proc. of IEEE Symposium on Volume Visualization '98*, pages 79–86, 1998.

[10] J. J. Koenderink and A. J. van Doorn. The singularities of the visual mapping. *Biological Cybernetics*, 24:51–59, 1976.

[11] J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[12] P. Lacroute and M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. In *Proc. of SIGGRAPH 1994*, pages 451–458. ACM, 1994.

[13] J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. on Information Theory*, 37(1):145–151, January 1991.

[14] N. Max. Optical models for direct volume rendering. *IEEE Trans. on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[15] L. Pessoa, E. Thompson, and A. Noë. Finding out about filling-in: A guide to perceptual completion for visual science and the philosophy of perception. *Behavioral and Brain Sciences*, 21(6):723–748, 1998.

[16] B. Shneiderman. Response time and display rate in human performance with computers. *ACM Computing Surveys*, 16(3):265–285, 1984.

[17] S. Takahashi, I. Fujishiro, Y. Takeshima, and Tomoyuki Nishita. A feature-driven approach to locating optimal viewpoints for volume visualization. In *Proc. of IEEE Visualization '05*, page TBA, 2005.

[18] S. Tenginakai, J. Lee, and R. Machiraju. Salient iso-surface detection with model-independent statistical signatures. In *IEEE Visualization '01*, pages 231–238, 2001.

[19] P. P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *Proc. of Vision, Modelling, and Visualization '01*, pages 273–280, 2001.

[20] P. P. Vázquez, 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.

[21] L. Wong, C. Dumont, and M. Abidi. Next best view system in a 3-d modeling task. In *Proc. of International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 306–311, 1999.