CNNs Based Viewpoint Estimation for Volume Visualization

NENG SHI and YUBO TAO, State Key Lab of CAD&CG, Zhejiang University, China

Viewpoint estimation from 2D rendered images is helpful in understanding how users select viewpoints for volume visualization and guiding users to select better viewpoints based on previous visualizations. In this article, we propose a viewpoint estimation method based on Convolutional Neural Networks (CNNs) for volume visualization. We first design an overfit-resistant image rendering pipeline to generate the training images with accurate viewpoint annotations, and then train a category-specific viewpoint classification network to estimate the viewpoint for the given rendered image. Our method can achieve good performance on images rendered with different transfer functions and rendering parameters in several categories. We apply our model to recover the viewpoints of the rendered images in publications, and show how experts look at volumes. We also introduce a CNN feature-based image similarity measure for similarity voting based viewpoint selection, which can suggest semantically meaningful optimal viewpoints for different volumes and transfer functions.

CCS Concepts: • **Human-centered computing → Scientific visualization**; • **Computing methodologies → Neural networks**;

Additional Key Words and Phrases: Viewpoint estimation, convolutional neural networks, volume visualization

ACM Reference format:
Neng Shi and Yubo Tao. 2019. CNNs Based Viewpoint Estimation for Volume Visualization. *ACM Trans. Intell. Syst. Technol.* 10, 3, Article 27 (April 2019), 22 pages.
https://doi.org/10.1145/3309993

1 INTRODUCTION

Viewpoint is one of the important rendering parameters in volume visualization, and it is intentionally selected by users to convey important features clearly and to meet their aesthetic preferences. Poorly chosen viewpoints can lead to an imprecise and misleading analysis of volumetrical features; however, it is not always easy for the general users to choose a good viewpoint from scratch due to the high degree of freedom. Thus, many automatic viewpoint selection methods, such as surface area entropy [37], voxel entropy [2], opacity entropy [11], and gradient/normal variation [48], have been proposed to suggest optimal viewpoints to serve as a starting point of volume exploration. However, relatively little research has considered the viewpoint estimation problem from a rendered image in volume visualization. Viewpoint estimation can help us to

This work was supported by the National Key Research & Development Program of China (2017YFB0202203), National Natural Science Foundation of China (61472354, 61672452, and 61890954), and NSFC-Guangdong Joint Fund (U1611263).
Authors' addresses: N. Shi and Y. Tao (corresponding author), State Key Lab of CAD&CG, Zhejiang University, 866 Yuhangtang Rd., Hangzhou, Zhejiang 310058, China; emails: shineng@zju.edu.cn, taoyubo@cad.zju.edu.cn.

2157-6904/2019/04-ART27 $15.00
https://doi.org/10.1145/3309993

understand how experts select viewpoints for volume visualization and guide users to select better viewpoints based on previously rendered images. It is also the first step in recovering a visual encoding specification from a rendered image, such as the transfer functions [29].

The viewpoint estimation problem can be transformed into a learning problem [14, 21], and there are two main challenges in constructing robust models. The first challenge is the lack of diverse training images with accurate viewpoint annotations. Although there are many rendered images in published papers on volume visualization, their viewpoints are unknown, and it is time-consuming and less accurate to annotate these images manually. In contrast to one feature in a 3D model, there are many features in a volume which are classified by transfer functions, and each image may contain only one or some of the features, such as the skin, bone, and tooth in the head volume. In addition, other rendering parameters, such as projection types, also potentially affect the rendered results and their viewpoint estimation. Therefore, the training images need to be sufficiently diverse to include different features and rendering parameters. The second challenge is to design powerful features specially tailored for viewpoint estimation. SIFT (Scale-Invariant Feature Transform) and HOG (Histogram of Oriented Gradients) are the two most commonly used features, and they have been used for measurement of image similarity in the image-based viewpoint selection model [38]. However, they are designed primarily for image classification and object detection. Recently, Convolutional Neural Networks (CNNs) have been shown to automatically learn better features via task-specific supervision, i.e., the lower layers mostly detect low-level features, such as corners, color patches, and stripes, while the higher layers aggregate these low-level features into high-level task-related features, such as cats and automobiles for image classification. CNNs have been used to determine up front orientations and detect salient views of 3D models [16]. Therefore, CNNs are an attractive choice for extracting specific features for viewpoint estimation in volume visualization.

In this article, we propose a viewpoint estimation method based on CNNs. Since CNN training requires a huge amount of viewpoint-annotated images, we design an overfit-resistant image rendering pipeline, inspired by the "Render for CNN" idea [35], to generate the training dataset. Many volumes are available online in large public volume collections, and they can be classified into several categories, such as the head and tree. Given a category, we take into account different features and rendering parameters to generate diverse training images with accurate viewpoint annotations. After that, we train a category-specific viewpoint classification network to estimate the viewpoint of a rendered image in this category. Our method can achieve good performance on images rendered with different transfer functions and rendering parameters.

We present two applications of our viewpoint estimation method. The first application investigates how visualization experts select viewpoints for volume visualization. For the collected images in the volume visualization literature, we estimate the viewpoints of these images using our viewpoint estimation network, and analyze how visualization experts look at the volume in different categories. Inspired by the image-based viewpoint selection model, our second application suggests an optimal viewpoint with a clear semantic meaning for general users. We introduce a CNN-feature based image similarity measure and apply the measure to the similarity voting based viewpoint selection. Thus, our method can suggest different viewpoints for different volumes and transfer functions based on the similarity between collected images in the volume visualization literature and rendered images.

In summary, our main contributions are as follows:

— A CNN based viewpoint estimation method for volume visualization. We propose an overfit-resistant image rendering pipeline to generate the training dataset considering different transfer functions and rendering parameters which are as diverse as possible, and we design a geometric structure-aware loss function customized for viewpoint estimation.

— Two applications of our viewpoint estimation method: an analysis of the viewpoint preferences of visualization experts, and viewpoint selection based on a CNN-feature based image similarity measure.

2 RELATED WORK

The viewpoint estimation problem can be considered to be an "inverse problem" of data visualization, i.e., given a visualization, can we recover the underlying visual encoding and even data values? This is useful for automated analysis, indexing, and redesign of previous visualizations. This article focuses on recovering the viewpoint from a rendered image. Thus, we review the related work on viewpoint selection, reverse engineering of visualizations, and pose estimation.

2.1 Viewpoint Selection

Viewpoint selection has been widely investigated in computer graphics and visualization, and is a forward problem of viewpoint estimation. Computer-graphics psychophysics provided the "canonical views" [1], a small number of user-preferred viewpoints with the attributes of goodness for recognition, familiarity, functionality, and aesthetic criteria. Thus, the optimal viewpoint often presents the most information about features of interest. Vázquez et al. [40] first applied information theory to search for the optimal viewpoint. Polonsky et al. [27] proposed three principles (view-independent, view-dependent, and semantic meaning) to classify and compare view descriptors for 3D models. Cremanns et al. [3] proposed a search strategy for viewpoint selection by identifying the regions that are very likely to contain best views, referred to as canonical regions, attaining greater search speed and reducing the number of views required. Wu et al. [45] proposed to represent a geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid, and this model is able to predict the next-best-view for an object.

In volume visualization, Takahashi et al. [37] decomposed the volume into features, calculated the optimal viewpoint for each feature using the surface area entropy, and combined these optimal viewpoints to suggest the optimal viewpoint for all features. Bordoloi and Shen [2] proposed the voxel entropy to identify representative viewpoints, and Ji and Shen [11] further presented image-based metrics, including opacity entropy, color entropy, and curvature information. Ruiz et al. [30] introduced the voxel mutual information to measure the informativeness of the viewpoint. A viewpoint suggestion framework presented by Zheng et al. [48] first clusters features based on gradient/normal variation in the high-dimensional space, and iteratively suggests promising viewpoints during data exploration.

A growing body of work focuses on the data-driven or learning-based viewpoint selection methods. Vieira et al. [41] presented intelligent design galleries to learn a classifier based on a large set of view descriptors from the user interaction on viewpoints. Secord et al. [32] collected the relative goodness of viewpoints based on human preferences through a user study, and trained a linear model based on these collected data for viewpoint selection. A web-image voting method proposed by Liu et al. [18] allows each web image to vote its most similar viewpoints based on the image similarity considering the area, silhouette, and saliency attributes, and it performs better than previous view descriptors for 3D models. Since visualization experts generally provide more representative viewpoints for volumes, Tao et al. [38] utilized rendered images in published papers on volume visualization to learn how visualization experts choose representative viewpoints for volumes with similar features. The viewpoint voting is based on the image similarity with SIFT and HOG between the collected image in published papers and the rendered image under the same viewpoint. Our viewpoint selection method is also based on the similarity voting, but the image similarity is evaluated based on learned features from CNNs, not manually designed features.

2.2 Reverse Engineering of Visualizations

Most research on reverse engineering of visualizations focus on static chart images, such as line charts, pie charts, bar charts, and heatmaps, and many methods have been proposed to interactively or automatically extract data values and encoding specifications for visualization interpretation and redesign.

ReVision [31] automatically identifies the chart type of bitmap images, infers the data by extracting the graphical marks, and redesigns visualizations to improve graphical perception. Harper and Agrawala [9] presented a deconstruction tool to extract the data in a D3 visualization and allow users to restyle existing D3 visualizations. FigureSeer [33] applies a graph-based reasoning approach based on a CNN-based similarity metric to extract data and its associated legend entities to parse figures in research papers. iVoLVER [22] enables flexible data acquisition from bitmap charts and interactive animated visualization reconstruction. Jung et al. [13] introduced ChartSense to determine the chart type using a deep learning–based classifier and to semi-automatically extract data from the chart image. Instead of data values, Poco and Heer [25] automatically recovered visual encodings from a chart image based on inferred text elements. They further contributed a method to extract color mapping from a bitmap image semi-automatically, and presented automatic recoloring and interactive overlays to improve perceptual effectiveness of visualizations [26].

Besides chart images, there are several learning-based methods to recover the viewpoint and transfer function from a rendered image of 3D models and volumes. Liu et al. [19] described a data-driven method for 3D model upright orientation estimation using a 3D CNN. Similarly, Kim et al. [16] applied one CNN on 3D voxel data to generate a CNN shape feature for the upright orientation determination, and the other CNN to encode category-specific information learned from a large number of 2D images on the web for the salient viewpoint detection. Given a target image, Raji et al. [29] combined CNN and evolutionary optimization to iteratively refine a transfer function to match the visual features in the rendered image of a similar volume dataset to the one in the target image. Their CNN is used to compare the similarity between the rendered and target image, not trained specially for the transfer function optimization task. In this article, our CNN is an end-to-end training for viewpoint estimation.

2.3 Pose Estimation

Pose estimation is an active branch of research in computer vision for object detection and scene understanding. For example, the indoor mapping problem is based on estimating the pose of the sensor of each k-th frame and building a map of the environment with the estimated camera pose of each frame [5]. Recently, most methods have been based on CNNs, and these methods can be divided into two categories: keypoint-based method and direct estimation method.

The keypoint-based method usually predicts 2D keypoints from an image, and recovers the 3D pose from these keypoints by solving a perspective-n-point problem. These 2D keypoints can be semantic keypoints defined on 3D object models [24, 44]. Given an image, the CNN trained on semantic keypoints is used to predict a probabilistic map of 2D keypoints and recover the 3D pose by comparsion with pre-defined object models. Instead of semantic keypoints, 2D keypoints can be eight corners of the 3D bounding box encapsulating the object [8, 28]. The CNN is trained by comparing the predicted 2D keypoint locations with the projections of 3D corners of the bounding box on the image under the ground-truth pose annotations.

The direct estimation method predicts the 3D pose from an image without intermediate keypoints, and mostly uses the Euler angle representation of rotation matrices to estimate the azimuth, elevation, and camera-tilt angles separately. The pose estimation problem can be solved by directly regressing the angle with a Euclidean loss [43], or through transformation into a classification

problem by dividing the angle into non-overlapping bins [4, 35, 39]. Massa et al. [21] experimented with multiple loss functions in CNNs based on regression and classification, and concluded that the loss function based on classification outperforms the one based on regression by a considerable margin. They further proposed a joint object detection and viewpoint estimation method for diverse classes in the Pascal3D+ dataset [46]. Besides the Euler angle representation, PoseCNN [47] employs the quaternion representations of 3D rotations, introduces a new loss function for the 3D rotation regression problem to handle symmetric objects, and estimates 6D object pose in cluttered scenes. Mahendran et al. [20] proposed an axis-angle representation in a mixed classification regression framework. This framework can accommodate different architectures and loss functions to generate multiple classification-regression models, and it achieves good performance on the Pascal3D+ dataset.

Similarly, our method is also based on CNNs. However, pose estimation in computer vision mostly focuses on analyzing the localization of the object in the real scene for mobile robotics, navigation, and augmented reality. Our objective is to estimate the viewpoint of a rendered image to recover the rendering parameters. Technically, there are two differences between pose estimation and viewpoint estimation. The first is the camera's intrinsic parameters. Pose estimation is generally based on the assumption that the camera's intrinsic parameters are known. For example, the ground-truth poses in the Pascal3D+ dataset are computed from 2D-3D correspondences assuming the same intrinsic parameters for all images. However, this article attempts to estimate the viewpoint of a rendered image under different intrinsic parameters of the camera, especially different projection types (parallel projection and perspective projection). Thus, the keypoint-based method is not suitable for our problem, since it is difficult to solve the perspective-n-point problem with the unknown intrinsic parameters of the camera. The second is that pose estimation aims to predict the 3D rotation between the object and the camera. Under the Euler angle representation, the 3D pose includes azimuth, elevation, and camera-tilt angle. In this article, we are only concerned about the camera's viewpoint, including only azimuth and elevation under the Euler angle representation, and the camera-tilt angle is less interesting in our viewpoint estimation for volume visualization. As a result, when we apply the direct estimation method in pose estimation to our viewpoint estimation, we need to revise the angle representation at first.

3 VIEWPOINT ESTIMATION

Given an input rendered image, our goal is to estimate the viewpoint. We assume that all viewpoints are on the viewing sphere [11], the center of which is located at the volume center. We can parameterize the viewpoint as a tuple (θ, ϕ) of camera parameters, where θ is the azimuth (longitude) angle and ϕ is the elevation (latitude) angle. The viewpoint estimation problem can be transformed to a regression problem. However, the regression model only returns predicted camera parameters, and may not capture the underlying viewpoint ambiguity, such as similar rendered images of nearby viewpoints for some volumes and symmetrical viewpoints for semi-transparent volumes. On the other hand, we can divide the continuous camera parameter domain into intervals and transform it into a classification problem. Thus, we can obtain the probabilities of each interval after the classification. Experiments also show that the classification performance is better than the regression performance for viewpoint estimation [21].

Previous classification methods for viewpoint estimation [4, 35, 39] divide the azimuth and elevation domain independently, and the loss is simply the sum of the azimuth and elevation misclassification. However, the azimuth and elevation are not uniform units of measure, similar to the longitude and latitude of the earth, and they cannot be directly used to evaluate the distance between the predicated and ground-truth viewpoints. In order to overcome this problem, this article explicitly divides the viewing sphere into N uniform regions and assigns a viewpoint label for

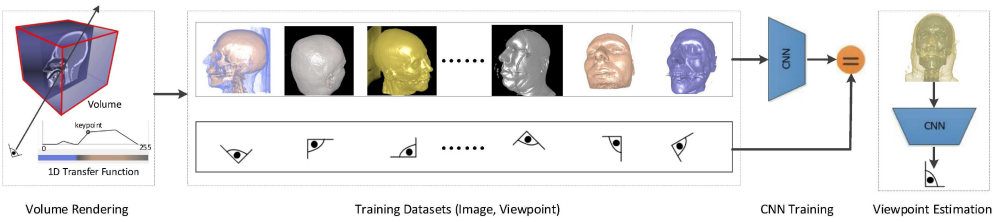


Fig. 1. The viewpoint estimation learning pipeline. For each category (heads in this example), we apply direct volume rendering with the volumes, transfer functions, and other rendering parameters as the input to generate the training datsets (rendered images with annotated viewpoints). The CNN training process takes rendered images as input, to estimate the viewpoint, and the parameters of the CNN are optimized by minimizing the difference between the estimated viewpoint and the annotated viewpoint. Finally, the trained CNN can be used to estimate the viewpoint of an image, rendered with a different volume in this category, different transfer functions, and rendering parameters.

each region. Thus, the viewpoint estimation problem can be formalized as classifying the rendered image into viewpoint labels with probabilities, and the loss is evaluated by the geodesic distance between the predicted and ground-truth viewpoints.

We apply CNNs to the viewpoint classification problem due to their high learning and classification capacities. A large number of viewpoint-annotated images of high variation are required to avoid overfitting of deep CNNs. Since there is no training dataset available for estimating the viewpoints of rendered images, we generate the training dataset through the "Render for CNN" [35] approach on the volumes in large public collections. The rendering process is more complex for volumes than for 3D models, since both the data classification and the rendering parameters have a strong influence on the rendered image. As shown in Figure 1, we first classify volumes available online into several categories, and design an overfit-resistant image rendering pipeline to generate the training dataset. This pipeline should consider both the different data classification and the different rendering parameters. After obtaining these rendered images with viewpoint labels, we train a category-specific viewpoint classification network for each category. With the trained CNN, we can estimate the viewpoint of a new rendered image of volumes in these categories.

3.1 Training Image Generation

The viewpoints are sampled uniformly on the viewing sphere by Hierarchical Equal Area isoLatitude Pixelization (HEALPix) [7], which can effectively discretize a sphere, and these viewpoints are our viewpoint labels. There are several volumes or one volume in each category in the training dataset depending on the volumes available online. For example, the head category has three different volumes, and the engine category has only one volume. We design an overfit-resistant image rendering pipeline to generate rendered images with viewpoint annotations as the training dataset for each category.

For each volume in the category, we render as many images as possible for each viewpoint label according to different data classification and rendering parameters. For each rendered image, the rendered viewpoint is randomly shifted within the region of the viewpoint label to avoid overfitting in the training process.

Data Classification. The transfer function classifies the features in the volume [15]. Different opacities are specified to highlight features of interest and remove unrelated features. Different colors are used to label different features. Thus, the transfer function has a strong influence on the rendered image and its viewpoint classification. During training, various transfer functions are required to generate rendered images with as many different features as possible.

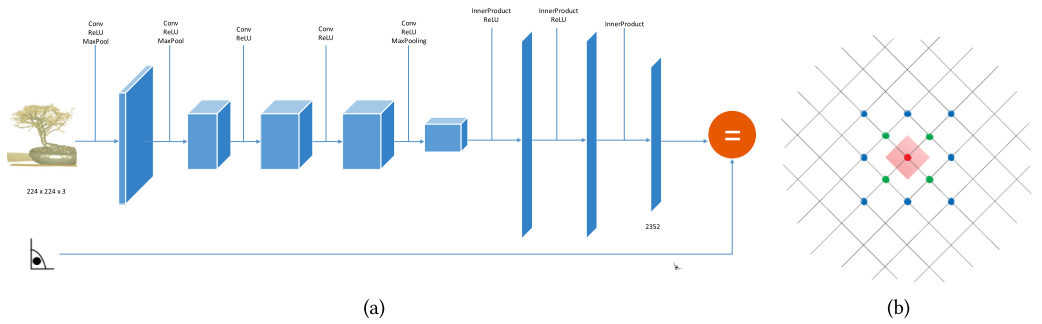
We manually design different opacity transfer functions for each volume to classify different features. For example, the head volume generally has the skin, skull, and tooth, and the opacity transfer functions are designed to show only one feature semi-transparently or opaquely, or some of the features with semi-transparent outer features and semi-transparent or opaque inner features. Opacity transfer functions are not randomly generated in our training dataset, since random opacity transfer functions may easily miss important features completely, and these rendered images may lack features and reduce the performance of the viewpoint estimation. Our model is expected to estimate the viewpoint from the rendered image generated from a manually designed opacity transfer function, such as the rendered image in the visualization paper, instead of a random opacity transfer function. In order to improve the generalization, we still add a small random disturbance to the designed opacity transfer function. For each rendered image, we randomly adjust the opacity slightly for each feature independently, such as moving the keypoint of the 1D transfer function (Figure 1) left or right by the distance $d \sim \mathcal{N}(0, 1)$. For the color transfer function, the color of each feature is randomly sampled for each rendered image, since users may choose different colors for features during data classification. The color is biased toward a high contrast with the background color to mimic a user's intent on emphasizing features. These random color transfer functions would improve the generation of viewpoint estimation.

Rendering Parameters. Besides the viewpoint, there are many other rendering parameters: the camera-tilt angles, scales, projection types, background color, and so on. Since CNNs are not rotation invariant, i.e., if the whole image is rotated then the CNNs' performance suffers, we need to deal with rotation invariance through data augmentation, i.e., the effect of the camera-tilt angle. We randomly rotate *the camera-tilt angle* for each rendered image. There are generally two projection types: parallel projection and perspective projection. Thus, for each rendered image, *the projection type* is randomly selected from the two projection types. The background color also affects features due to alpha blending in direct volume rendering. The most common background colors are black and white, and we randomly choose black or white as *the background color* for each rendered image. It is worth noting that the color transfer functions in different backgrounds are slightly different in order to distinguish the features from the background. Although CNNs are relatively invariant to scaling, we further reduce the influence of *the scale* by rendering volumes with the scale uniformly sampled from 1 to 1.8.

For *the lighting condition*, three lighting modes are used. The first is the environment light only. The second is the environment light and one headlight located at the same position of the camera, and the lighting intensity is uniformly sampled from 0.7 to 1. The third one includes environment light, one headlight, and one scene light. The position of the scene light is uniformly sampled on a sphere with the radius uniformly sampled from three to five times the radius of the viewing sphere, and the lighting intensities of the headlight and scene light are uniformly sampled from 0.35 to 0.5. The coefficients of the Phong reflection model are also randomly sampled. The ambient reflection coefficient is fixed at 1, the diffuse reflection coefficient is uniformly sampled from 0.25 to 0.75, the specular reflection coefficient is uniformly sampled from 0.5 to 1, and the shininess coefficient is uniformly sampled from 20 to 100. Other rendering parameters, such as gradually changed background colors, can be included in our image rendering pipeline to further improve the generalization of viewpoint estimation.

3.2 Network Architecture and Loss Function

In this section, we introduce the network architecture and different loss functions for the viewpoint estimation problem, standard cross-entropy loss function, and geometric structure-aware loss function, respectively.



(a) (b)

Fig. 2. (a) The structure of our viewpoint estimation network based on AlexNet. (b) The illustration of neighbor viewpoints in $V(v_s, n)$ for the geometric structure-aware loss function. The red viewpoint is the ground-truth viewpoint v_s, and its first-order neighbor viewpoints are the four green viewpoints ($n = 1$).

3.2.1 Network Architecture. A CNN is a type of artificial neural network, and has become a hotspot in computer vision and natural language processing due to its satisfactory learning capacity. For instance, it has been applied to question answering (QA) systems [42] and image recognition [23]. It is a multilayer perceptron specifically designed to recognize 2D shapes, and this network structure is invariant to translation, scaling, or other forms of deformation [6].

Recently, many well-designed networks have been proposed, such as AlexNet [17], VGGNet [34], GoogLeNet [36], and ResNet [10], and they all achieve good performance in image classification. We first experiemnt with AlexNet, and then mainly choose the 19-layer VGGNet to implement the viewpoint classification task. The structure of our network based on AlexNet is shown in Figure 2(a). The network ends with an N-way fully connected layer with softmax, according to the assigned viewpoint label.

3.2.2 Geometric Structure-Aware Loss Function. The loss function in the last layer is very important for viewpoint estimation. The widely used loss function, Softmax loss, employs the output probability as the predicted probability and computes the cross-entropy loss based on the ground-truth value. During training, minimizing the cross entropy is equivalent to maximizing the log-likelihood of the ground-truth label. The disadvantage of the Softmax loss function is that it learns to predict the viewpoints without explicitly considering the continuity between neighbor viewpoints. It is obvious that two neighbor viewpoints have a great deal in common, and the geometric information may be particularly important for viewpoint estimation. One solution to the problem is to design a geometric structure-aware loss function customized for viewpoint estimation.

The geometric structure-aware loss function is modified from the Softmax loss function by adding geometric constraints:

$$L_{vp}(\{s\}) = -\sum_{\{s\}} \sum_{v \in V(v_s, n)} q(v) log P_v(s), \tag{1}$$

where v_s is the ground-truth viewpoint for the rendered image s. $V(v_s, n)$ is the neighbor viewpoint set of the ground-truth viewpoint v_s, determined by the relative distance bandwidth parameter n. Since HEALPix can provide a viewpoint's neighbors, the neighbor viewpoint set contains the viewpoint itself for $n = 0$, the viewpoint and its first-order neighbor viewpoints for $n = 1$, and so on. For example, Figure 2(b) shows the ground-truth viewpoint and its neighbor viewpoints on the viewing sphere. The light-red area is the region for the red viewpoint, i.e., all viewpoints in the light-red area have the same label as the red viewpoint. The four green viewpoints are the first-order

neighbor viewpoints of the red viewpoint. $P_v(s)$ is the probability for the image s classified to the viewpoint label v based on the Softmax loss function.

The only difference between the geometric structure-aware loss function and the orginial Softmax loss function is the designed ground-truth distribution $q(v)$:

$$q(v) = e^{-d(v, v_s)}, \tag{2}$$

where $d : V \times V \mapsto \mathbb{R}$ is the geodesic distance between two viewpoints. We substitute an exponential decay weight w.r.t. the viewpoint distance, to explicitly exploit the correlation between neighbor viewpoints. In our experiment, the relative distance bandwidth parameter is 1 ($n = 1$), i.e., we consider only the ground-truth viewpoint and its first-order neighbor viewpoints. In the Softmax loss function, $q(v_s) = 1$ and $q(v) = 0$ for $v \neq v_s$. However, in the geometric structure-aware loss function, $q(v_s) = 0.87$, $q(v) = 0.36$ for $v \neq v_s$ and $v \in V(v_s, 1)$, otherwise $q(v) = 0$ for $v \notin V(v_s, 1)$. We expect the geometric structure-aware loss function with the bandwidth parameter as 1 could improve the prediction accuracy, compared with the original softmax loss function.

4 RESULTS

4.1 Training Datasets

We collected available volumes from public volume databases, such as VolVis and The Volume Library, and extracted six volume categories manually: engine, fish, head, tooth, tree, and vessel. Most of these volumes are widely used in volume visualization research. There are three different volumes in the head and tree categories, and the other categories have only one volume. Since the head and tree categories have more than one volume, the experiment can demonstrate the generalization of our classification model when there are multiple volumes in the category.

In our experiment, the number of viewpoints is $N = 2,352$, and we apply the proposed image rendering pipeline to generate training images with viewpoint annotations for each category. As shown in Table 1, we take into consideration the main features of each category when we manually design these opacity transfer functions. For example, the tree generally has the trunk, branch, and leaf features, and the vessel only has the vessel and aneurism features. Thus, the number of opacity transfer functions is different for each category depending on the number of volumes and features in each volume. For instance, there are seven, four, and one opacity transfer functions for the Chapel Hill CT head, the visual male, and the MRI head in the head category, respectively, and there are five opacity transfer functions for different features for the tooth volume in the tooth category. We rendered 50,000 images at 256×256 size for each opacity transfer function, considering different viewpoints, color transfer functions, camera-tilt angles, projection types, and lighting conditions. The number of training images is from 100,000 to 600,000, as listed in Table 1.

4.2 Training Process

With these rendered images, we train a viewpoint classification network for each category. Because of the abundant labeled training data in ImageNet, pretrained models on ImageNet would generally have a very powerful generalization ability. The low and middle layers contain a massive number of general visual elements, and we only need to fine-tune the last several layers based on our training dataset for the viewpoint classification task. Thus, for both AlexNet and the 19-layer VGGNet, convolution layers from conv1 to conv3 are fixed with the pretrained parameters. The remaining convolutional layers and all the fully connected layers except the last one are fine-tuned during the training process. Only the last fully connected layer is trained from scratch.

The network is implemented in Caffe [12], using an NVIDIA GTX 1080 Ti GPU. For the 19-layer VGGNet, the network is trained by stochastic gradient descent of about two epochs, and the total

Table 1. The Statistical Information for the Training Dataset in Each Category: The Number of Opacity Transfer Functions, the Number of Rendered Images ($\times 10^5$), and Main Features Classified by Opacity Transfer Functions

Category	Transfer Functions	Images	Main Features
engine	9	4.5	surface, gear, other inner structure
fish	6	3.0	bone, skin, gill
head	12	6.0	skull, skin, tooth
tooth	5	2.5	enamel, dentin, pulp chamber, cementum
tree	7	3.5	trunk, branch, leaf
vessel	2	1.0	vessel, aneurism

training time is about 3 days for each category. During the testing, the viewpoint estimation time of each rendered image is about 0.01 seconds based on the proposed network.

4.3 Viewpoint Estimation Evaluation

We first describe how to evaluate the accuracy of viewpoint estimation. The trained CNN generates a probability for each viewpoint label, and for most cases, the classification probability distribution on the viewing sphere approximately subjects to a bivariate Gaussian distribution. Thus, we can model the distribution through $P_v(s) \sim N(v_\mu, v_\sigma^2)$, where v_μ equals

$$\arg\max_{v \in V} P_v(s). \tag{3}$$

The mean v_μ can be used to evaluate the accuracy of our viewpoint classification network. The standard deviation v_σ is small, when the CNN is very confident to estimate the viewpoint with a high probability. For challenging cases, the CNN becomes less confident and the v_σ becomes bigger. Examples in the head category are shown in the first row of Figure 5. The first two are simple cases with a small deviation and the last two are challenging cases with a large deviation.

We define the accuracy metric based on the geodesic distance between the estimated viewpoint v_μ and the ground-truth viewpoint v_s. Our evaluation metric is a viewpoint accuracy with a "tolerance." Specifically, we select five tolerances, 2°, 5°, 8°, 11°, and 15°, respectively. In the evaluation, if the geodesic distance between v_μ and v_s is within the tolerance, we count it as a correct prediction. When the geodesic distance is within 2°, the predicted viewpoint v_μ exactly matches the ground-truth viewpoint v_s.

In our evaluation, we apply the same image rendering method in Section 3.1 to generate the testing dataset. Since we add a random disturbance for each opacity transfer function and randomly sample one color for each feature, the transfer functions in the testing dataset are different from the ones in the training dataset. We generate 3,000 images for each opacity transfer function in each category, for example, 27,000 (3,000 × 9) images for the engine category.

The classification accuracy of the testing dataset is shown in Table 2. For these rendered images, our model can obtain a good performance. First, under the 19-layer VGGNet, we compare our method (uniform division of the viewing sphere and geometric structure-aware loss function) with UD+Softmax (uniform division of the viewing sphere and Softmax loss function) and SD+GS (separate division of the azimuth and elevation and geometric structure-aware loss function) on the six categories under different tolerances. The same training dataset is used to train these classification models, and the classification accuracy under different tolerances is also listed in Table 2. The models using uniform division of the viewing sphere (UD+GS and UD+Softmax) have better performance than the model with a separate division of the azimuth and elevation (SD+GS). The reason is that when dividing the viewing sphere into uniform regions, the CNN can

Table 2. Classification Accuracy Comparison of AlexNet (UD+GS, Uniform Division of the Viewing Sphere and Geometric Structure-Aware Loss Function), VGGNet (UD+GS), VGGNet (UD+Softmax, Uniform Division of the Viewing Sphere and Softmax Loss Function), VGGNet (SD+GS, Separate Division of the Azimuth and Elevation and Geometric Structure-Aware Loss Function), and the Category-Independent Network VGGNet.CI (UD+GS) on the Six Categories Under Different Tolerances

Cat.	Angle Tol.	2°	5°	8°	11°	15°
engine	AlexNet (UD+GS)	0.4664	0.8194	0.9592	0.9860	0.9931
	VGGNet (UD+GS)	**0.8450**	**0.9774**	**0.9987**	**0.9997**	**0.9999**
	VGGNet (UD+Softmax)	0.6692	0.9313	0.9942	0.9985	0.9996
	VGGNet (SD+GS)	0.5371	0.9412	0.9896	0.9948	0.9957
	VGGNet.CI (UD+GS)	0.6845	0.9279	0.9911	0.9982	0.9995
fish	AlexNet (UD+GS)	0.5859	0.8315	0.9532	0.9756	0.9872
	VGGNet (UD+GS)	**0.7946**	**0.9278**	**0.9932**	**0.9969**	**0.9986**
	VGGNet (UD+Softmax)	0.6389	0.8714	0.9798	0.9924	0.9974
	VGGNet (SD+GS)	0.6564	0.9440	0.9812	0.9884	0.9906
	VGGNet.CI (UD+GS)	0.6242	0.8626	0.9745	0.9876	0.9946
head	AlexNet (UD+GS)	0.3888	0.7714	0.9419	0.9792	0.9886
	VGGNet (UD+GS)	**0.7224**	**0.9373**	**0.9912**	**0.9971**	**0.9984**
	VGGNet (UD+Softmax)	0.5893	0.8893	0.9828	0.9951	0.9976
	VGGNet (SD+GS)	0.3675	0.8267	0.9586	0.9848	0.9924
	VGGNet.CI (UD+GS)	0.5835	0.8766	0.9823	0.9938	0.9959
tooth	AlexNet (UD+GS)	0.6130	0.9038	0.9868	0.9958	0.9994
	VGGNet (UD+GS)	**0.8463**	**0.9800**	**0.9983**	**0.9995**	**0.9999**
	VGGNet (UD+Softmax)	0.7291	0.9532	0.9954	0.9990	0.9997
	VGGNet (SD+GS)	0.5963	0.9617	0.9898	0.9927	0.9932
	VGGNet.CI (UD+GS)	0.6731	0.9259	0.9890	0.9984	0.9994
tree	AlexNet (UD+GS)	0.3417	0.7267	0.9268	0.9767	0.9918
	VGGNet (UD+GS)	**0.6679**	**0.8896**	**0.9882**	**0.9970**	**0.9992**
	VGGNet (UD+Softmax)	0.5078	0.7922	0.9678	0.9930	0.9980
	VGGNet (SD+GS)	0.3362	0.8108	0.9588	0.9863	0.9933
	VGGNet.CI (UD+GS)	0.5015	0.8201	0.9585	0.9798	0.9921
vessel	AlexNet (UD+GS)	0.6355	0.9232	0.9950	0.9995	1.0000
	VGGNet (UD+GS)	**0.9495**	**0.9957**	**0.9998**	**1.0000**	**1.0000**
	VGGNet (UD+Softmax)	0.7948	0.9692	0.9975	0.9998	1.0000
	VGGNet (SD+GS)	0.5758	0.9637	0.9953	0.9970	0.9973
	VGGNet.CI (UD+GS)	0.7130	0.9060	0.9725	0.9943	0.9984

optimize a more straightforward problem by the geodesic distance, instead of the angle difference in the azimuth and elevation. Besides, the proposed geometric structure-aware loss function is better than the Softmax loss function. The comparison between VGGNet (UD+GS) and AlexNet (UD+GS) shows the effect of a deeper network, i.e., the 19-layer VGGNet. VGGNet has better performance than AlexNet. Furthermore, we compare our category-specific model with the category-independent network. In the category independent network, all the convolution layers and fully connected layers except the last one are shared by all classes, while class-dependent layers (one fc layer for each class) are stacked over them. The category-independent network can save parameters for the whole system and have similar performance to VGGNet (UD+Softmax), but it would reduce the system's prediction accuracy for each category, as shown in Table 2. Taking

Table 3. Classification Accuracy Comparison of VGGNet (UD+GS), AlexNet (SD+GS) [35], and $\mathcal{M}_G$+ [20] on the Six Categories Under Different Tolerances

Cat.	Angle Tol.	2°	5°	8°	11°	15°
engine	VGGNet (UD+GS)	**0.8450**	**0.9774**	**0.9987**	**0.9997**	**0.9999**
	AlexNet (SD+GS) [35]	0.1571	0.5084	0.7669	0.8764	0.9185
	$\mathcal{M}_G$+ [20]	0.6016	0.9060	0.9672	0.9837	0.9920
fish	VGGNet (UD+GS)	**0.7946**	**0.9278**	**0.9932**	**0.9969**	**0.9986**
	AlexNet (SD+GS) [35]	0.1361	0.4310	0.6454	0.7595	0.8215
	$\mathcal{M}_G$+ [20]	0.3772	0.7883	0.9338	0.9737	0.9891
head	VGGNet (UD+GS)	**0.7224**	**0.9373**	**0.9912**	**0.9971**	**0.9984**
	AlexNet (SD+GS) [35]	0.0691	0.2703	0.5121	0.6969	0.8136
	$\mathcal{M}_G$+ [20]	0.3018	0.6649	0.8655	0.9427	0.9699
tooth	VGGNet (UD+GS)	**0.8463**	**0.9800**	**0.9983**	**0.9995**	**0.9999**
	AlexNet (SD+GS) [35]	0.2123	0.6261	0.8770	0.9547	0.9759
	$\mathcal{M}_G$+ [20]	0.4629	0.8191	0.9368	0.9721	0.9860
tree	VGGNet (UD+GS)	**0.6679**	**0.8896**	**0.9882**	**0.9970**	**0.9992**
	AlexNet (SD+GS) [35]	0.0748	0.2915	0.5344	0.7131	0.8195
	$\mathcal{M}_G$+ [20]	0.2444	0.6264	0.8354	0.9319	0.9674
vessel	VGGNet (UD+GS)	**0.9495**	**0.9957**	**0.9998**	**1.0000**	**1.0000**
	AlexNet (SD+GS) [35]	0.2322	0.6698	0.8988	0.9707	0.9965
	$\mathcal{M}_G$+ [20]	0.3600	0.7787	0.9335	0.9742	0.9888

all the comparisons into consideration, we choose the VGGNet (UD+GS) model for the following sections (Error Analysis and Applications).

We further compare our VGGNet (UD+GS) model with two state-of-the-art methods, AlexNet (SD+GS) [35] and $\mathcal{M}_G$+ (Geodesic Bin & Delta Mode) [20] on the six categories under different tolerances. For AlexNet (SD+GS), in our experiment, we ignore the camera-lit angle and only care about the azimuth and elevation, as we did for VGGNet (SD+GS). The $\mathcal{M}_G$+ model predicts the viewpoint label first by classification, then estimates the viewpoint residual by regression, and finally combines the viewpoint label and residual to obtain the final viewpoint. In the experiment, we choose the size of the K-means dictionary $K = 24$ and the importance of the geodesic distance $\alpha = 10$, under which the model achieves the best performance. A modification has been made to these methods, namely, the viewpoint estimation network is now category-specific rather than category-independent in their original experiments, since category-specific networks are proved better in previous experiments. As shown in Table 3, our result is better than the results of the AlexNet (SD+GS), VGGNet (SD+GS), and $\mathcal{M}_G$+ models. Su et al. [35] and Mahendran et al. [20] employed a large angle tolerance (30°) for viewpoint estimation of 3D models. Our angle tolerance is much less than 30°, and this indicates that our method is relatively more accurate for viewpoint estimation for volumes and the estimated viewpoints can be used in the following applications.

4.4 Error Analysis

As shown in Table 2, for the VGGNet (UD+GS) model, except for the vessel category, the angle differences of some of the rendered images are larger than 5°. Thus, we analyzed which kinds of rendered images or features are hard to estimate their viewpoints. We count the number of misclassified images under Acc-5° as the classification error for each ground-truth viewpoint. This results in an error map on a 2D azimuth-elevation plane.

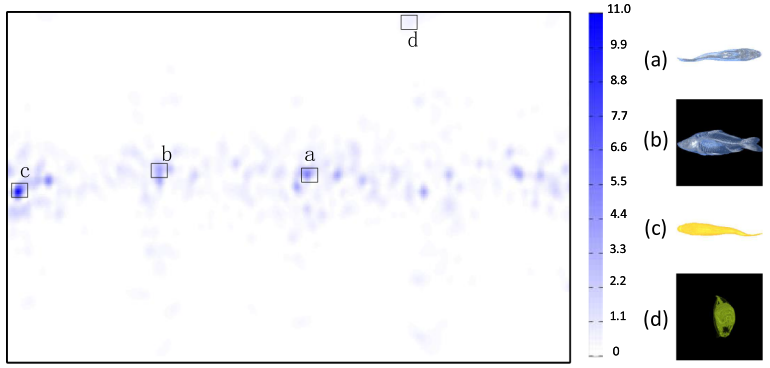


Fig. 3. The classification error map for the fish category. The viewpoints in the blue region are more likely to be misclassified. (a) A side view, (b) a front view, (c) another side view, and (d) a top view from the head.

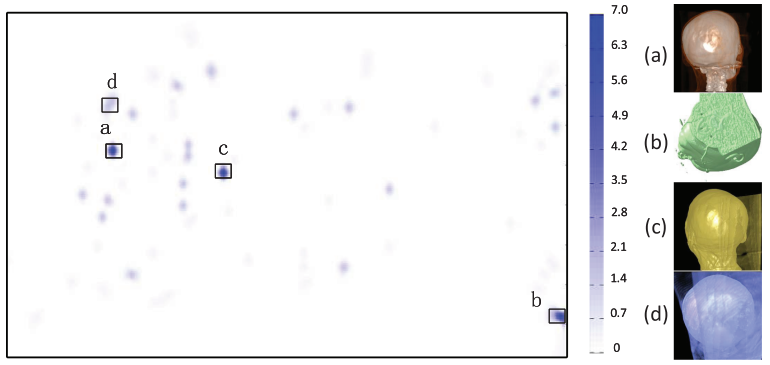


Fig. 4. The classification error map for the head category. The viewpoints in the blue region are more likely to be misclassified. (a)-(d) are four rendered images at representative misclassified viewpoints.

Figure 3 is the error map for the fish category. At the front and the side views (Figure 3(a)–(c)), the classification error is a little higher than most other well-classified viewpoints, although the rendered image of the estimated viewpoint is very similar to the one of the ground-truth viewpoint. The relative higher classification error can be explained by the view stability [2] of these viewing regions, which means a small change occurs when the camera is shifted within a small neighborhood.

Furthermore, the rendered images under the top view and bottom view (Figure 3(d)) are likely to be misclassified into symmetrical viewpoints. This is because their outer contours are similar to each other, but the inner features are not clear enough due to visual clutter. There are no significant gradient changes in the rendered image, so it is confusing for our viewpoint estimation network to identify the right viewpoint.

We also analyzed the classification error of the head category and four misclassified examples, as shown in Figure 4. The misclassified viewpoints are generally distributed among the back, up, and bottom views. These images generally have a lack of distinguishable features, and the viewpoints in these regions are relatively stable. Thus, it is hard to distinguish them from nearby viewpoints due to their featureless rendered images. In the front view, since our network can identify rich facial features, the classification accuracy is relatively high.

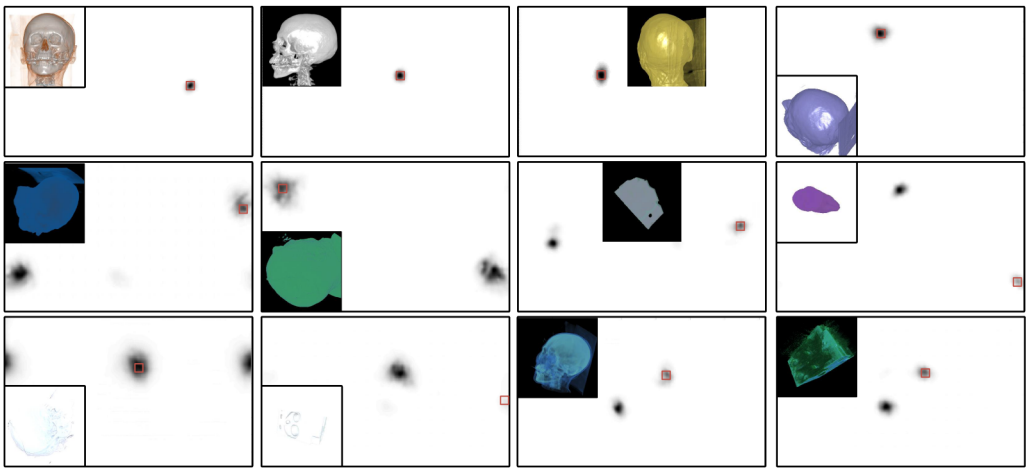


Fig. 5. The classification probability distributions of 12 representative images. On the viewing sphere in 2D, our method classifies the rendered image to the viewpoints in the black region with a high probability. The red box indicates the ground-truth viewpoint. The first row has only one bivariate Gaussian distribution. The middle and bottom rows have two bivariate Gaussian distributions, and the lower peak is around the ground-truth viewpoint.

It is worth noting that some rendered images are obviously misclassified, especially those misclassified under the 50° tolerance. Figure 5 shows eight representative examples. We are interested in what their classification probability distributions look like and why these images are misclassified. We observe an interesting pattern from our representative examples in Figure 5. Although the geodesic distance between the highest peak produced by our method and the ground-truth viewpoint is not within the tolerance, there is still a lower peak around the ground-truth viewpoint. The lower peak can contribute to viewpoint selection in our applications, and this can only be achieved with the classification model, instead of the regression model.

These misclassifications in Figure 5 are due to unrecognizable internal features. We observe some typical misclassification patterns in our results: bad light conditions and bad opacity transfer functions. In the case of the environment light only, some rendered images do not have enough recognizable features for our model. This phenomenon also occurs when the light intensity is too high. In the case of a bad opacity transfer function, it may lead to visual clutter in the rendered image. This usually happens when the opacity is low. Thus, when the inner structure of the volume is complex, such as the Chapel Hill CT Head and the engine, its inner structure will be mixed together due to its transparency. As a result, the rendered image is very similar to the one under a symmetrical viewpoint on the viewing sphere. Thus, these images are likely to be misclassified into symmetrical viewpoints, and this results in an ambiguity of viewpoint estimation. When the opacity is high for the outer feature as the context, it may occlude important inner features, which also makes our model confused and results in the ambiguity of viewpoint estimation.

In summary, some viewpoints are hard to estimate, and this misclassification may be due to the high image similarity between the estimated viewpoint and the ground-truth viewpoint. The rendered images of stable viewpoints are similar to those of nearby viewpoints, and transparent structures may have the same rendered image from the viewpoint and the symmetric viewpoint. In addition, the featureless images are also less distinguishable.

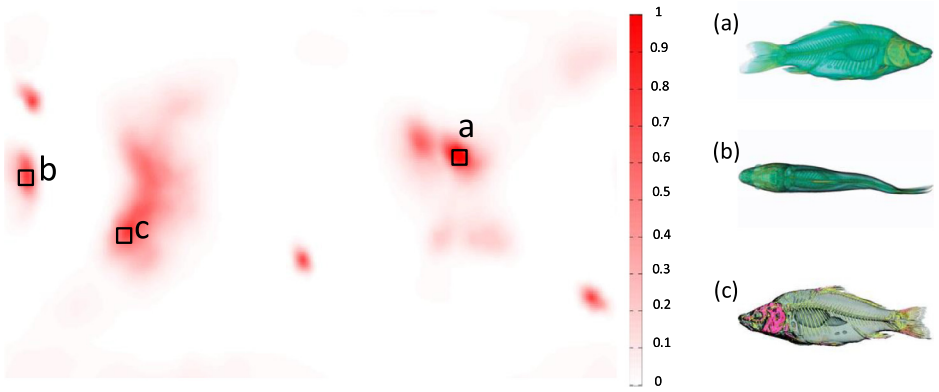


Fig. 6. The viewing map shows the estimated viewpoints of collected images in the fish category. Experts usually have viewpoint preferences for the viewpoints in the red region. (a)–(c) are three collected images at representative viewpoints labeled in the viewing map.

5 APPLICATIONS

Our viewpoint estimation method can be used to support a variety of volume visualization applications, such as extending the transfer function exploration from a rendered image [29]. In this section, we describe two direct applications of viewpoint estimation: viewpoint preference analysis and viewpoint selection based on CNN-feature similarity. We describe two direct applications of viewpoint estimation based on the VGGNet (UD+GS) model.

5.1 Viewpoint Preference Analysis

There are many rendered images in published papers on volume visualization. Most viewpoints for these images are selected by visualization or domain experts to maximize the amount of information about features in the rendered image or to highlight important features. Their viewpoint preferences are expressed in these rendered images. Thus, we can apply our viewpoint estimation method to analyze their preferences.

We utilize the image database [38], collected from visualization journals and conferences. The categories are the same as those in Table 1, and our trained category-specific CNNs can be used to recover the viewpoints from collected images. Since the ground-truth viewpoints of collected images are unknown, we visually compared the rendered images of recovered viewpoints with collected images, and most of them can be classified correctly, with little difference between rendered and collected images. For collected images whose source volume is beyond our training dataset, most of them can also be estimated correctly. Some collected images are misclassified, due to various reasons, such as different projection type and non-uniform background color. Considering the distribution of the estimated viewpoint $N(v_\mu, v_\sigma^2)$, we find that v_σ is very high for misclassified images, which means the probability of the viewpoint v_μ is relatively small.

Since most collected images generate a bivariate Gaussian probability distribution on viewpoints, we accumulate the distribution of each collected image in the category to generate a viewing map to analyze the viewpoint preference for the volume in this category. The distribution in the viewing map is similar to the Gaussian mixture model. Due to the small probability of misclassified images, their influence on viewpoint preference analysis is relatively limited.

Figure 6 shows the viewing map of the fish category. There are three regions with a high probability. Experts tend to select side viewpoints to avoid occlusions. In addition, there is one viewpoint cluster at the fish's back, revealing the swimming pose of the fish.

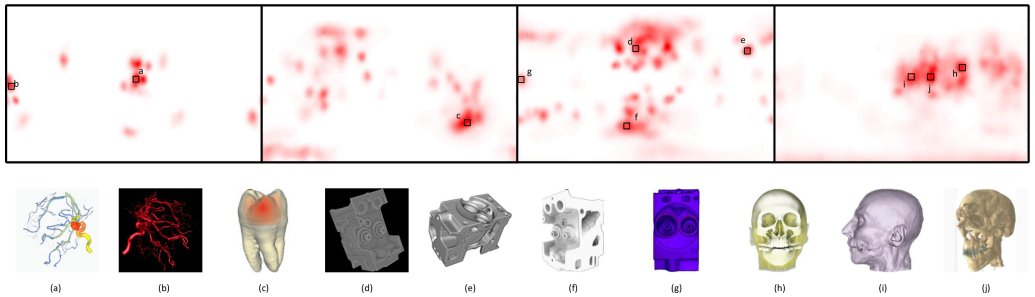


Fig. 7. The viewing map shows the estimated viewpoints of collected images in the foot, tooth, engine, and head categories. Experts usually have viewpoint preferences for the viewpoints in the red regions. (a)-(j) are collected images at representative viewpoints.

We also analyze the other four categories: the vessel, tooth, engine, and head categories in Figure 7. According to the probability distribution estimated from the collected images of the vessel category, experts tend to display the volume in the front view (Figure 7(a)), since they are more interested in the aneurism and would like to avoid any occlusions. Many experts' viewpoint preference for the tooth category would generally focus on the lower right corner of the viewing map, while there are also other viewpoints for the tooth. It is necessary to explain that some images are misclassified to symmetrical viewpoints because they are mirrored, and some images in the image dataset are "worst case" viewpoints. In contrast, the viewpoints are quite diverse for the engine category. As the structure of the engine is quite complex and different viewpoints can reveal different structures, experts use different viewpoints to comprehensively understand the engine. They tend to select the three-quarter views (Figure 7(d)) and also choose side views (Figure 7(g)) as supplements. The preferred viewpoints in the head category are clustered in the front view, side view, and three-quarter view (Figure 7(h)–(j)). There is no clear boundary between these regions. This is due to different experts having slightly different preferences from the front view to the side view.

For the viewpoint preference analysis application, we need only the trained CNN and collected images in this category. The analysis result can tell us how users select the viewpoint for the volume in this category, and find features interesting to most users.

5.2 Viewpoint Selection Based on CNN-Feature Similarity

Given an input volume and a transfer function, the viewpoint selection application suggests the optimal viewpoint for features of interest. The viewpoint preference analysis does not consider the input volume, and suggests the same representative viewpoint for different features of the same volume and different volumes. It would be better to consider the similarity between the input features and the features in the collected images. Inspired by the similarity voting for viewpoint selection [38], we propose a weighted probability voting for viewpoint selection.

We first need to determine the input volume belonging to which category. AlexNet is used to train our category classification network. The volume is classified into seven categories, six of which are listed in Table 1, and the last category is *others*, not belonging to the six categories. This network is trained through the same training dataset with additional images of other volumes. For the input volume, we first render several images randomly using the provided transfer function, and classify these images through the category classification network. The most voted category is considered the category of the volume.

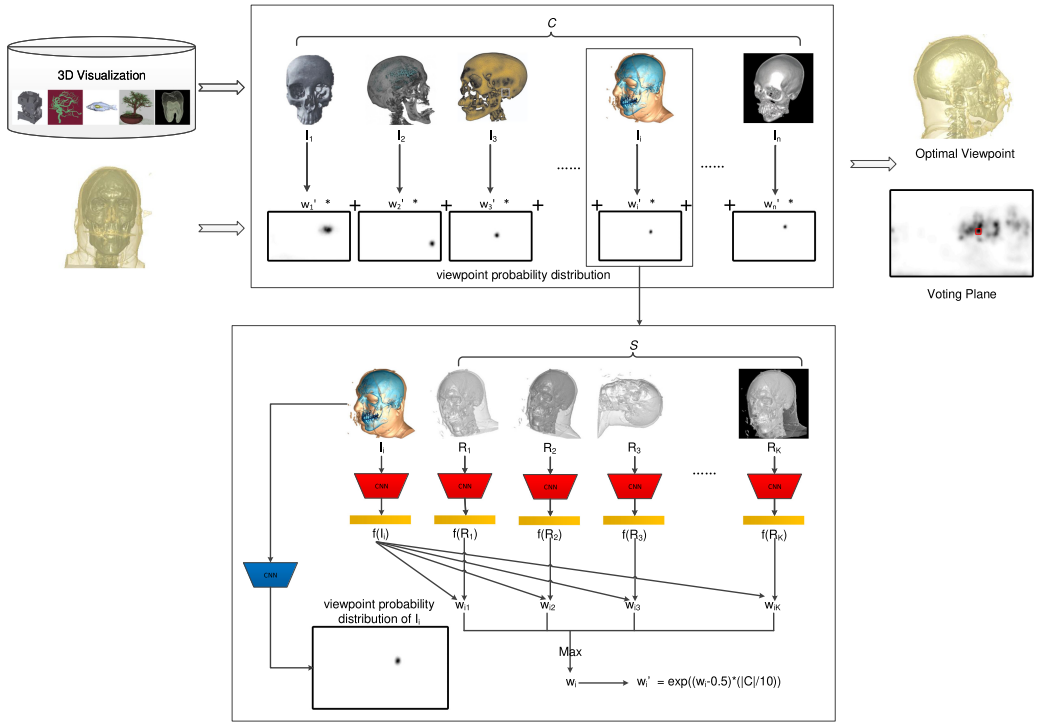


Fig. 8. CNN-feature based viewpoint selection pipeline. The first stage is CNN-feature based similarity computation. The input volume is rendered with the provided transfer function and the estimated viewpoint of the collected image I_i to generate a rendered image set S. The similarity w_i between the input volume and I_i can be computed with the help of the features extracted by the category classification network. Then the viewpoint probability distribution of I_i is weighted by the similarity w_i' to generate the voting probability distribution of I_i. Finally, all weighted voting probability distributions are summed up, and the viewpoint with the largest probability corresponds to the optimal viewpoint.

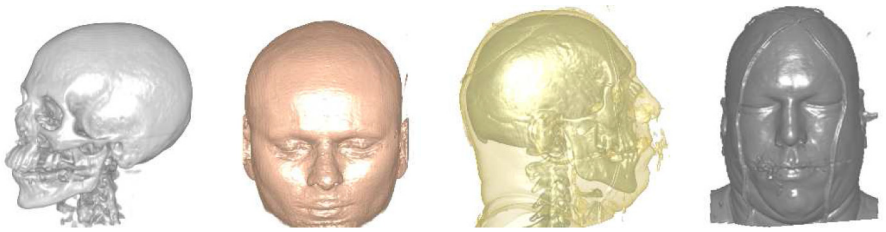
After obtaining the category of the input volume, the weighted probability voting is illustrated in Figure 8. We denote the collected image set of this category as $C = \{I_1, I_2, \ldots, I_n\}$, where n is the number of collected images. For each collected image $I_i \in C$, we can estimate its viewpoint v_i with our viewpoint estimation network. We then render the volume with the provided transfer function under the estimated viewpoint v_i considering different camera-tilt angles and background colors. Thus, we have $k = 40$ images denoted as the rendered image set $S = \{R_1, R_2, \ldots, R_k\}$. We then calculate the similarity between the rendered image set and the collected image I_i as the voting weight for the collected image I_i. Since the category classification network extracts features for volume classification, we can employ the features of the last hidden layer when classifying the rendered image $R_j \in S$ and the collected image I_i, denoted by $f(R_j)$ and $f(I_i) \in R^{4096}$, to measure the similarity w_i between the collected image I_i and the input volume together with the transfer function using the cosine distance as follows:

$$\max_{R_j \in S}(cos(f(R_j), f(I_i))). \tag{4}$$

For each collected image, its probability is weighted by its similarity with the input volume and transfer function, and we can further design the exponential similarity by $w_i' = exp((w_i - 0.5) * |C|/10)$, where $|C|$ is number of images in the collected image set. All weighted probability

(a) (b) (c)

Fig. 9. The viewpoint selection results for the engine, the vessel, and the bonsai tree from (a) to (c), respectively.



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

Fig. 10. The viewpoint selection results of the heads. (a) The optimal viewpoint for the bone of the Chapel Hill CT head. (b) The optimal viewpoint for the skin of the MRI head. (c) The optimal viewpoint for the bone and skin of the visual male. (d) The optimal viewpoint for the skin of the visual male.

distributions are summed up, and the viewpoint with the largest probability corresponds to the optimal viewpoint.

We first evaluate our method by three volumes: the engine, the vessel, and the bonsai tree, as shown in Figure 9. The optimal viewpoint for the engine is quite close to the three-quarter view, which is preferred by a lot of users. For the vessel, since users are generally more interested in the aneurism, our selected viewpoint avoids the occlusion on it and shows the aneurism clearly. For the bonsai tree, our result is not only concerned with the clearness of the semantically important features, such as the trunk, tree branches, and leaves, but it also shows other meaningful features, such as the soil, the grass, and the base plane by a lightly oblique shift.

Our method can suggest different optimal viewpoints for different features of the same volume and different volumes. For exmaple, the head category has three different volumes and each volume has different features. Our method can choose the optimal viewpoint for currently visible features in this category as shown in Figure 10. For the MRI head, we can clearly observe the facial features from the front view, thus the front view is regarded as the optimal viewpoint in Figure 10(b). It is the same for the head of the visual male with only the skin in Figure 10(d). However, for the Chapel Hill CT head with the bone, the side view is selected to show more structures of the bone in Figure 10(a). For the head of the visual male with the bone and the skin, the front view shows visual clutter, and we can display clearer features on the side view. In this situation, our method suggests the side view as the optimal viewpoint in Figure 10(c).

The image-based viewpoint selection model [38] applies one single similarity measure to viewpoint selection, and mixes two separate stages: viewpoint estimation and similarity calculation. Our method separates these two stages and solves them by CNNs. In the following, we compare our method with the image-based viewpoint selection model by the viewpoint estimation accuracy and viewpoint selection result.

Table 4. The Viewpoint Estimation Accuracy of Images with General Viewpoints Using the Image-based Viewpoint Selection Model [38]

Category	Size	Acc-19°	Acc-22°	Acc-25°	Acc-29°	Acc-8°(Our)
engine	450	0.354	0.383	0.406	0.422	**0.870**
fish	300	0.187	0.200	0.235	0.252	**0.769**
head	350	0.310	0.356	0.393	0.417	**0.842**
tooth	250	0.303	0.329	0.342	0.355	**0.871**
tree	150	0.236	0.284	0.308	0.340	**0.755**
vessel	100	0.342	0.387	0.423	0.456	**0.875**

Table 5. The Viewpoint Estimation Accuracy of Images with Manually Selected Representative Viewpoints Using the Image-based Viewpoint Selection Model [38]

Category	Size	Acc-19°	Acc-22°	Acc-25°	Acc-29°	Acc-8°(Our)
engine	513	0.487	0.527	0.555	0.575	**0.859**
fish	324	0.376	0.425	0.465	0.498	**0.791**
head	399	0.387	0.430	0.465	0.488	**0.867**
tooth	250	0.379	0.398	0.417	0.436	**0.880**
tree	150	0.330	0.368	0.400	0.430	**0.727**
vessel	100	0.464	0.518	0.548	0.569	**0.898**

As the viewpoints of the collected images in published papers are unknown, we randomly generate some viewpoint-annotated images as the collected images for each category for viewpoint estimation evaluation. In [38], every collected image votes on the viewpoints of its 12 most similar images with the same weight when the number of viewpoints is 2,352. Thus, the similarity between the voted viewpoints and the ground-truth viewpoint is the key to viewpoint selection. We apply Acc-n° as the average probability of the 12 selected viewpoints within the n° neighbor region of the ground-truth viewpoint, instead of the probability of the optimal estimated viewpoint in Section 4.3. For each collected image, we evaluate the 12 voted viewpoints under Acc-19°, Acc-22°, Acc-25°, and Acc-29°, and the average accuracy for images in each category is shown in Table 4. Since SIFT and HOG are designed primarily for image classification and object detection, the performance on viewpoint estimation of randomly generated images is not very good, especially for the image without too many features. For further validation, we manually generate several rendered images under the viewpoints of images from published papers [38]. The evaluation result in Table 5 is better than the one for randomly generated images in Table 4. However, compared with our estimation result for the 12 selected viewpoints Acc-8° in Table 4 and Table 5, our method has better performance on viewpoint estimation.

The viewpoint selection results of the tooth, the MRI head, and the bonsai tree of our method and the image-based viewpoint selection model are shown in Figure 11. All three volumes have clear semantic meanings, especially the up direction. For the tooth volume, instead of a completely front view, our method selects the viewpoint with a slightly oblique shift to reveal the crown structure more clearly without occlusion. For the MRI head, both methods choose the front view, but the viewpoint of our method is more consistent with the aesthetic criterion. For the bonsai tree, our method can capture the up direction from collected images. However, the trunk, branch, and soil cannot be easily separated under the viewpoint from the image-based viewpoint selection model. As a result, our CNN-feature based image similarity can suggest more semantically meaningful viewpoints than the SIFT and HOG based image similarity.

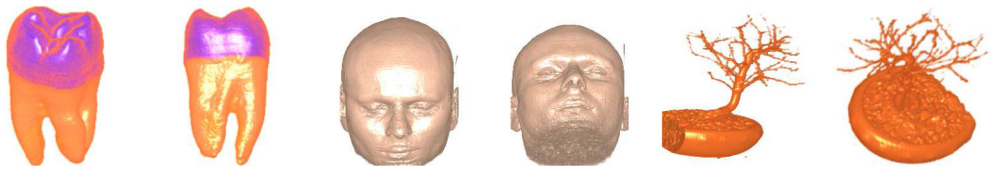


Fig. 11. Comparison of the optimal viewpoints for the tooth, the MRI head, and the bonsai tree by two methods: our method (left) and the similarity voting based viewpoint selection method [38] (right).

6 CONCLUSION

In this article, we propose a CNN based viewpoint estimation method. Inspired by the "Render for CNN" approach, an overfit-resistant image rendering pipeline was designed to generate training images with viewpoint annotations considering different transfer functions and rendering parameters. These images are used to train a category-specific viewpoint classification network. The proposed method was tested on six categories based on available online volumes. We can achieve a classification accuracy of at least 0.89 in the maximum angle difference 5°. Our viewpoint estimation model is better than previous methods due to its better viewing sphere division and the geometric structure-aware loss function. We successfully applied our method on two applications: the viewpoint preference analysis of collected images in publications, and a CNN-feature similarity based viewpoint selection.

In the future, we would like to replace the manually designed opacity transfer functions with image-driven or data-driven opacity transfer functions to improve the richness of features and the training efficiency when training a new category. When more volumes are available, we will add them as training volumes in the corresponding category to improve the generalization, especially for categories with only one volume. We also plan to recover the transfer function of a rendered image based on the estimated viewpoint, or jointly learn the viewpoint and transfer function estimation from a rendered image.

ACKNOWLEDGMENT

This work was supported by the National Key Research & Development Program of China (2017YFB0202203), National Natural Science Foundation of China (61472354, 61672452 and 61890954), and NSFC-Guangdong Joint Fund (U1611263).

REFERENCES

[1] Volker Blanz, Micjael J. Tarr, and Heinrich H. Bülthoff. 1999. What object attributes determine canonical views? *Perception* 28, 5 (1999), 575–599.

[2] Udeepta D. Bordoloi and Han-Wei Shen. 2005. View selection for volume rendering. In *Proceedings of the Conference on Visualization'05*. IEEE Computer Society, 487–494.

[3] Robert Cremanns and Friedrich Otto. 1994. Constructing canonical presentations for subgroups of context-free groups in polynomial time (extended abstract). In *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC'94)*. ACM, New York, 147–153. DOI: https://doi.org/10.1145/190347.190395

[4] Mohamed Elhoseiny, Tarek El-Gaaly, Amr Bakry, and Ahmed Elgammal. 2016. A comparative analysis and study of multiview CNN models for joint object categorization and pose estimation. In *International Conference on Machine Learning*. 888–897.

[5] Nadia Figueroa, Haiwei Dong, and Abdulmotaleb El Saddik. 2015. A combined approach toward consistent reconstructions of indoor spaces based on 6D RGB-D odometry and KinectFusion. *ACM Transactions on Intelligent Systems and Technology* 6, 2 (March 2015), Article 14, 10 pages. DOI: https://doi.org/10.1145/2629673

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[7] Krzysztof M. Gorski, Eric Hivon, A. J. Banday, Benjamin D. Wandelt, Frode K. Hansen, Mstvos Reinecke, and Matthia Bartelmann. 2005. HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal* 622, 2 (2005), 759.

[8] Alexander Grabner, Peter M. Roth, and Vincent Lepetit. 2018. 3D pose estimation and 3D model retrieval for objects in the wild. *CoRR* abs/1803.11493 (2018). arxiv:1803.11493 http://arxiv.org/abs/1803.11493

[9] Jonathan Harper and Maneesh Agrawala. 2014. Deconstructing and restyling D3 visualizations. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. 253–262.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 770–778.

[11] Guang-Feng Ji and Han-Wei Shen. 2006. Dynamic view selection for time-varying volumes. *IEEE Transactions on Visualization and Computer Graphics* 12, 5 (2006), 1109–1116.

[12] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*. ACM, 675–678.

[13] Daekyoung Jung, Wonjae Kim, Hyunjoo Song, Jeong-in Hwang, Bongshin Lee, Bohyoung Kim, and Jinwook Seo. 2017. ChartSense: Interactive data extraction from chart images. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 6706–6717.

[14] Alex Kendall, Matthew Grimes, and Roberto Cipolla. 2015. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV'15)*. 2938–2946.

[15] Naimul Mefraz Khan, Riadh Ksantini, and Ling Guan. 2018. A novel image-centric approach toward direct volume rendering. *ACM Transactions on Intelligent Systems and Technology* 9, 4 (Jan. 2018), Article 42, 18 pages. DOI: https://doi.org/10.1145/3152875

[16] Seong-Heum Kim, Yu-Wing Tai, Joon-Young Lee, Jaesik Park, and In-So Kweon. 2017. Category-specific salient view selection via deep convolutional neural networks. *Computer Graphics Forum* 36 (2017), 313–328.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems* 25 (2012), 1097–1105.

[18] Hong Liu, Lei Zhang, and Hua Huang. 2012. Web-image driven best views of 3D shapes. *The Visual Computer* 28, 3 (2012), 279–287.

[19] Zishun Liu, Juyong Zhang, and Ligang Liu. 2016. Upright orientation of 3D shapes with convolutional networks. *Graphical Models* 85, C (May 2016), 22–29.

[20] Siddharth Mahendran, Haider Ali, and René Vidal. 2018. A mixed classification-regression framework for 3D pose estimation from 2D images. In *Proceedings of the British Machine Vision Conference 2018 (BMVC'18)*, 72. http://bmvc2018.org/contents/papers/0238.pdf.

[21] Francisco Massa, Renaud Marlet, and Mathieu Aubry. 2016. Crafting a multi-task CNN for viewpoint estimation. *The British Machine Vision Conference*, 91.1–91.12.

[22] Gonzalo Gabriel Méndez, Miguel A. Nacenta, and Sebastien Vandenheste. 2016. iVoLVER: Interactive visual language for visualization extraction and reconstruction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 4073–4085.

[23] Xinyu Ou, Hefei Ling, Han Yu, Ping Li, Fuhao Zou, and Si Liu. 2017. Adult image and video recognition by a deep multicontext network and fine-to-coarse strategy. *ACM Transactions on Intelligent Systems and Technology* 8, 5 (July 2017), Article 68, 25 pages. DOI: https://doi.org/10.1145/3057733

[24] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis. 2017. 6-DoF object pose from semantic keypoints. In *IEEE International Conference on Robotics and Automation*. 2011–2018.

[25] Jorge Poco and Jeffrey Heer. 2017. Reverse-engineering visualizations: Recovering visual encodings from chart images. *Computer Graphics Forum* 36, 3 (2017), 353–363.

[26] Jorge Poco, Angela Mayhua, and Jeffrey Heer. 2018. Extracting and retargeting color mappings from bitmap images of visualizations. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 637–646.

[27] Oleg Polonsky, Giuseppe Patané, Silvia Biasotti, Craig Gotsman, and Michela Spagnuolo. 2005. What's in an image: Towards the computation of the "best" view of an object. *The Visual Computer* 21, 8–10 (2005), 840–847. Proc. Pacific Graphics'05.

[28] Mahdi Rad and Vincent Lepetit. 2017. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV'17)*, 3848–3856.

[29] Mohammad Raji, Alok Hota, Robert Sisneros, Peter Messmer, and Jian Huang. 2017. Photo-guided exploration of volume data features. *arXiv:1710.06815v1*.

[30] Marc Ruiz, Imma Boada, Miquel Feixas, and Mateu Sbert. 2010. Viewpoint information channel for illustrative volume rendering. *Computers & Graphics* 34 (Aug. 2010), 351–360.

[31] Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. 2011. ReVision: Automated classification, analysis and redesign of chart images. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software & Technology (UIST'11)*. 393–402.

[32] Adrian Secord, Jingwan Lu, Adam Finkelstein, Manish Singh, and Andrew Nealen. 2011. Perceptual models of viewpoint preference. *ACM Transactions on Graphics* 30, 5 (Oct. 2011), Article 109, 12 pages.

[33] Noah Siegel, Zachary Horvitz, Roie Levin, Santosh Divvala, and Ali Farhadi. 2016. FigureSeer: Parsing result-figures in research papers. In *European Conference on Computer Vision (ECCV'16)*. 664–680.

[34] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556 (2014). arxiv:1409.1556 http://arxiv.org/abs/1409.1556

[35] Hao Su, Charles R. Qi, Yangyan Li, and Leonidas J. Guibas. 2015. Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'15)*. 2686–2694.

[36] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of Computer Vision and Pattern Recognition (CVPR'15)*. 1–9.

[37] Shigeo Takahashi, Issei Fujishiro, Yuriko Takeshima, and Tomoyuki Nishita. 2005. A feature-driven approach to locating optimal viewpoints for volume visualization. In *Proceedings of the Conference on Visualization'05*. 495–502.

[38] Yubo Tao, Qirui Wang, Wei Chen, Yingcai Wu, and Hai Lin. 2016. Similarity voting based viewpoint selection for volumes. *Computer Graphics Forum (EuroVis)* 35, 3 (2016), 391–400.

[39] Shubham Tulsiani and Jitendra Malik. 2015. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1510–1519.

[40] Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. 2001. Viewpoint selection using view entropy. In *Proceedings of Vision Modeling and Visualization Conference (VMV'01)*. 273–280.

[41] Thales Vieira, Alex Bordignon, Adelailson Peixoto, Geovan Tavares, Hélio Lopes, Luiz Velho, and Thomas Lewiner. 2009. Learning good views through intelligent galleries. *Computer Graphics Forum* 28, 2 (2009), 717–726.

[42] Pengwei Wang, Lei Ji, Jun Yan, Dejing Dou, Nisansa De Silva, Yong Zhang, and Lianwen Jin. 2018. Concept and attention-based CNN for question retrieval in multi-view learning. *ACM Transactions on Intelligent Systems and Technolology* 9, 4 (Jan. 2018), Article 41, 24 pages. DOI : https://doi.org/10.1145/3151957

[43] Yumeng Wang, Shuyang Li, Mengyao Jia, and Wei Liang. 2016. Viewpoint estimation for objects with convolutional neural network trained on synthetic images. In *Proceedings of Advances in Multimedia Information Processing*. 169–179.

[44] Jiajun Wu, Tianfan Xue, Joseph J. Lim, Yuandong Tian, Joshua B. Tenenbaum, Antonio Torralba, and William T. Freeman. 2016. Single image 3D interpreter network. In *Proceedings of the European Conference on Computer Vision*. 365–382.

[45] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao. 2014. 3D ShapeNets for 2.5D object recognition and next-best-view prediction. *CoRR* abs/1406.5670 (2014). http://dblp.uni-trier.de/db/journals/corr/corr1406.html#WuSKTX14

[46] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. 2014. Beyond Pascal: A benchmark for 3D object detection in the wild. In *Proceedings of the 2014 IEEE Winter Conference on Applications of Computer Vision (WACV'14)*. 75–82.

[47] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. 2018. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *Proceedings of Robotics: Science and Systems (RSS'18)*.

[48] Ziyi Zheng, Nafees Ahmed, and Klaus Mueller. 2011. iView: A feature clustering framework for suggesting informative views in volume visualization. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (2011), 1959–1968.

Received July 2018; revised December 2018; accepted January 2019