DeepShapeSketch: Generating hand drawing sketches from 3D objects

Meijuan Ye¹, Shizhe Zhou²,∗, Hongbo Fu³
¹,²College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
¹,²Key Laboratory of High Performance Computing and Stochastic Information Processing, Ministry of Education of China
³City University of Hong Kong, Hong Kong, China
mj@hnu.edu.cn, shizhe@hnu.edu.cn(corresponding author), hongbofu@cityu.edu.hk

Abstract—Freehand sketches are an important medium for expressing and communicating ideas. However creating a meaningful and understandable sketch drawing is not always an easy task especially for unskilled users. Existing methods for rendering 3D shape into line drawings such as Suggestive Contours, only consider the geometry-dependent and view-dependent information thus leads to over-regular or over-perfect results which doesn’t look like a human freehand drawing. In this work we address the challenge of generating freehand line drawing sketches from a 3D object under a given viewpoint automatically. The core solution here is a deep recurrent generative neural network, which learns a functional mapping from the suggestive contours of a 3D shape to a more abstract sketch representation. We drop the encoder of the generator, i.e., use only a decoder to achieve better stability of the sketch structure. Users can tune the level of freehand style of the generated sketches by changing a single parameter. Experiments show that our results are expressive enough to faithfully describe the input shape and at the same time be with the style of freehand drawings created by a real human. We conduct a user study to verify the quality and style of our results over existing methods. We also retrain our network using several different mingled dataset to test the extendibility of our method for this particular application. To the best of our knowledge, it is the first research effort to automate the generation of human-like freehand sketches directly from 3D shapes.

Index Terms—Freehand sketches, Generative recurrent neural network, 3D object

I. INTRODUCTION

Sketching is a common art form used in communication, rendering, and exchanging ideas, especially for expressing the overall shape of a certain 3D object. People in ancient times have already created line arts such as animal figures or buildings on cave rock on shells as a record of their lives. This is basically because we are able to easily envision the actual 3D shape when we observe its corresponding sketch images.

However, not every human being is well-trained for sketching or drawing. It is generally not very easy for an amateur artist to create an accurate sketch according to a given 3D object. Furthermore, the sophisticated skills used by professional artists for creating cartoons or mangas are not quite suitable for daily utility, since usually they take a lot of time to finish. It is more natural to use simple strokes to create relatively abstract patterns to convey shape-wise information or even express new design ideas. However we do not inherently have the ability to draw using just a few sketch lines to correctly describe a 3D shape. On the other hand, existing methods for directly rendering 3D shape into line drawing such as Suggestive Contours [1], only considers the geometry-dependent and view-dependent information thus leads to over-regular or over-perfect results which doesn’t look like a human freehand drawing. The straightforward solution of extracting the edge map of a simple projection from 3D models also suffer from the same problem. Furthermore, such edge map results are often uncontinuous, difficult for vectorization.

In this paper, we propose a novel method of translating three-dimensional objects into freehand human sketches. To sketch the shape of a 3D object is not a simple orthogonal or perspective projection of the 3D object on the image plane. A good shape sketch contains not only the outline of the object but also the lines of the fold detail that can reflect the internal features of the object. This requires the 3D object to be converted into a 2D view with distinct features. Our main idea of this paper is to take the 3D object as input and generate sketches that are quite similar to human hand-drawings through neural network learning. We first build a GMM distribution of sketch line shapes via supervised training of a Generative Adversarial Recurrent Neural network that introduces a certain level of randomness or minor deformation to the sketch lines, and then constrain the overall orientation of the sketch object with a novel sampling method after training the neural network.

Variational Autoencoders (VAEs) [2], [3] and Generative Adversarial Networks (GANs) [4]–[6] are important deep learning models in recent years. Sketch-RNN [7], an existing deep learning method for sketch generation, used the sequence-to-sequence variational-auto-encoder (VAE). However, one drawback of VAE is that the resulting samples are usually blurred due to injected noise and imperfect elemental measurements such as squared errors. Therefore, we remove the encoder and add a discriminative network based on Sketch-RNN [7]. The generative network and the discriminative network together form our generative recurrent adversarial networks. The generative network is used to generate data simulating an unknown distribution, and the discriminative network is used to distinguish between the generated samples and the ones from the true observations. The goal is to find a generator that fits the true data distribution while maximizing the possibility of confusing the discriminator.
The main steps of our work are as follows: We first convey the shape of the 3D object in the form of lines by using a non-photorealistic rendering system. Here we use DeCarlo et al.’s method: Suggestive contours [1]. We render the suggestive contours of each object from many different viewpoints to bulk-produce a two-dimensional line drawing sketch image dataset. Then a vectorization process is performed on the sketch images to obtain two-dimensional coordinate information. Based on the Sketch-RNN model [7], we remove the sketch images to obtain two-dimensional coordinate information. Then a vectorization process is performed on the bulk-produce a two-dimensional line drawing sketch image contours of each object from many different viewpoints to method: Suggestive contours [1]. We render the suggestive photorealistic rendering system. Here we use DeCarlo et al.’s the shape of the 3D object in the form of lines by using a non-

Our contributions are summarized as follows:

- To our best knowledge, for the first time, the problem of generating freehand vectorized sketches from a 3D object at a given viewpoint is addressed using a learned deep model, which enables stroke-level cross-domain visual understanding from a reference 3D shape.
- We use a supervised uncycled translation network to train our model. Due to the absence of the ground truth of our problem, a generative adversarial neural network is applied to verify if the generated sketch has a good balance between structure soundness and the style of an amateur human creator.
- A novel two-phase sampling process to guarantee more correct orientation of the result sketch.
- A middle-size dataset of 3D model snapshot photo and its corresponding vectorized sketches for future study on data-driven sketch generation.

II. RELATED WORK

In essence our problem here is an application of deep generative neural network onto the vectorized line depiction of 3D Shapes. Many research works are related to this problem and we categorize these works into 3 areas listed below.

A. Deep Generative Models

In recent years, there have been many developments in deep generative models. There are generally three themes: VAE [2], [3], GAN [4]–[6] and other neural networks based on GAN [8]–[11]. VAE pairs a programmable network with a decoder/generative network. The Sketch-RNN model of Ha and Eck [7] also used VAE. However, VAE has a disadvantage in which the generated samples are usually ambiguous due to injection noise and imperfect element measurements such as the squared errors. GAN is another popular generative model. It trains two models simultaneously: a generative model to synthesize samples, and a discriminative model to differentiate between natural and synthesized samples. However, the GAN model is difficult to converge in the training phase, and the samples generated from the GAN are often not very natural. Recently, many works have tried to improve the quality of the generated samples, such as Wasserstein GAN (WGAN) [12], McGAN [13], and Loss-Sensitive GAN [14]. There are also methods that attempt to incorporate GAN and VAE, such as VAE/GAN [15], CVAE-GAN [10]. Recurrent neural networks (RNNs) are usually used to simulate data sequences, and now there are many works to combine RNN with GAN, such as C-RNN-GAN [9] and S-LSTM-GAN [16].

B. Recurrent Vector Image Generation

There is relatively little work done using neural networks to generate vector sketch images currently. Earlier, there was a work [17] that used Hidden Markov models to synthesize lines of human sketches. The work of Alex Graves [18] laid the foundation for generating continuous data points using a hybrid density network [19]. They showed how to use the Long Short-term Memory recurrent neural network to generate complex sequences with long-distance structures to approximately generate vectorized hand-written numbers. Similar models were developed for vectorized Kanji characters and freehand human sketches, conditionally and unconditionally modeling them as a series of stroke actions. Sketch-RNN (a kind of recurrent neural network that can construct ordinary objects based on strokes) proposed by Ha and Eck [7] outlines a framework for both unconditional and conditional generation of vector images composed of a series of lines. They implemented training machines to draw and summarize abstract concepts in a human-like manner. Their generation model based on a recurrent neural network can generate sketches of common objects in vector format. Chen et al. [20] made two changes on the basis of Sketch-RNN: One was to use a convolutional neural network (CNN) instead of the bidirectional RNN encoder. The other was to remove the Kullback-Leibler (KL) divergence from the VAE objective function. It had better performance than [7] in learning and generating multiple categories of sketches. A recent work proposed by Muhammad et al. [21] used reinforcement learning to generate sketches with different abstraction level from images. Song et al. [22] used a deep learning model to achieve stroke-level cross-domain visual understanding. The problem of noise supervision caused by subjective and diversified human painting styles was determined, and a new mixed-supervised-unsupervised multi-task learning program was proposed. The unsupervised learning was accomplished more effectively via a shortcut cycle consistency constraint.

C. Lines to Depict the Shape of 3D objects

The goal of line rendering is to capture key features on the surface of an object and use lines to represent the shape of the object. Rendering methods are generally classified into two categories: view-independent and view-dependent. For view-independent rendering methods, the main research focuses on the extraction of ridges and valleys. Such lines do not take into account the viewing angle and are determined only by analyzing the geometric properties of the surface of the object; the view-dependent rendering methods must determine the viewing angle firstly, and the resulting line is related to the choice of viewing angle. The most commonly used is object contours, which are lines formed by the point perpendicular to the line of sight [23]. The representative method is the
Suggestion Contours method proposed by DeCarlo et al. [1]. This method has been used by Sang et al. [24] and Yoon et al. [25] for several related applications such as retrieval. And it has also been used for many other applications. In addition, Judd et al. [26] proposed the Apparent Ridges method, which differed from the Suggestive Contours in which it projected the normal vector of points on the surface onto the viewing plane and calculated the points on the curved surface along the various directions in the viewing plane. According to Cole’s [27], [28] comparative analysis of existing three-dimensional model line rendering algorithms, the Suggestive Contours method can obtain better model feature line segments. Therefore, this paper chooses the Suggestive Contours method to obtain the visual features of the object.

III. METHODOLOGY

In this section we will introduce the preparation of our dataset, and describe our network structure in detail. In addition, we will describe the method of generating multi-view sketches.

A. Training Data

1) Line drawings of 3D objects: Because our network should be able to translate surface and shape information out of a given 3D object at any view direction to human-like sketches, we need to construct a training dataset which includes “snapshots” of a 3D shape from all possible angles. Furthermore, since the final output is line drawing sketches, we also need to make a linear representation of these “snapshot” images.

At the beginning, we carefully considered the dataset used by Sketch-RNN [7] and sketch-pix2seq [20], from QuickDraw [29], a public sketch database published by Google. However, almost all the sketches in the QuickDraw dataset are taken from a frontal perspective that do not meet our multi-view requirements. Thus we need to synthetically construct a new dataset specifically designed for our problem.

Considering that real human freehand sketches are often casual and spontaneously expressive, it is reasonable to make such a linear representation including not just the outer contours of the object but also the internal detail lines. Using the Suggestive Contours method [1], we can get a variety of perspective views of a 3D object, which constitute the initial form of our dataset. Specifically, we place the normalized 3D object in the central point and then rotate it randomly and uniformly by 2900 times. For each view we capture it into a 256×256 image. Generally it takes less than 20 minutes to finish capturing all images. Among all the images, we randomly select 2300 images for training, 300 images for validation and the rest 300 image for testing.

2) Vectorization and simplification: Next we use the Win-topo software [30] to vectorize all the sketch images into actual vector lines represented as a sequence of vertices (i.e., point coordinate data) and edges (i.e., connectivity information).

Because the above linear representation has a rich set of internal lines, the number of line segments is overlarge, making the subsequent training difficult. Therefore, before vectorization we first filter the line segments and remove short ones (length below 3 pixels). Here we also apply the Ramer-Douglas-Peucke method [31] to do stroke simplification. This method has a controlling parameter ε for deciding whether to discard a certain input point, depending on how aggressively the user wants to simplify the lines. In all of our experiment, we set the parameter ε to be 0.1.

Next we reform all the lines into the same data format used in QuickDraw [29], which sequentially represents each vector line as a list of coordinate offsets: Δx, Δy and the binary...
According to the description of Sketch-RNN [7], when an encoder is incorporated, it is called conditional generation, and conversely, it is unconditional when the encoder is not used. We perform several experiments on conditional and unconditional vector image generation using the network with or without the discriminator \( \mathcal{D} \), i.e., we compare the results of our modified network with their corresponding results of the original Sketch-RNN model, see Table I. As shown in Figure 2, the comparison results visually demonstrate that
<table>
<thead>
<tr>
<th>Viewpoint</th>
<th>Suggestive Contours</th>
<th>ED</th>
<th>D</th>
<th>EDD</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>![3D rendering]</td>
<td>![Corresponding contours]</td>
<td>![Sketch results of ED]</td>
<td>![Sketch results of D]</td>
<td>![Sketch results of EDD]</td>
<td>![Sketch results of DD]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>![3D rendering]</td>
<td>![Corresponding contours]</td>
<td>![Sketch results of ED]</td>
<td>![Sketch results of D]</td>
<td>![Sketch results of EDD]</td>
<td>![Sketch results of DD]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>![3D rendering]</td>
<td>![Corresponding contours]</td>
<td>![Sketch results of ED]</td>
<td>![Sketch results of D]</td>
<td>![Sketch results of EDD]</td>
<td>![Sketch results of DD]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>![3D rendering]</td>
<td>![Corresponding contours]</td>
<td>![Sketch results of ED]</td>
<td>![Sketch results of D]</td>
<td>![Sketch results of EDD]</td>
<td>![Sketch results of DD]</td>
</tr>
</tbody>
</table>

Fig. 2. Comparison of the 4 generative models on Table I. From left to right: 3D rendering, corresponding suggestive contours [1], and sketch results of ED, D, EDD, DD with 3 different \( \tau \) value.
our method can generate more reasonable and stable freehand sketches of a given 3D objects.

A. Data Processing

We render the suggestive contours of each model from different viewpoints to get two-dimensional line drawing images with size of 256×256. Then, the vectorization process is performed to obtain two-dimensional coordinate information. We also filter the vector line segments using the method described in the section III-A2. Note here the value of ρ of the stroke-3 format is determined according to the start and end points of the line segment.

B. Implementation Details

For fairly comparison, we use LSTM for the encoder, decoder and discriminator. The encoder RNNs consist of 256 nodes, and the total number of nodes contained in both the decoder D and the discriminator D are set to be 512. We also apply recurrent dropout [32] to maintain a probability of 90%. We optimize using Adam [33] with the learning rate to be 0.001, and the batch size to be 100. We also perform simple data augmentation by multiplying the offset columns (Δx, Δy) by two IID random factors chosen uniformly between 0.90 and 1.10. We conduct comparative experiments on four network models, see Table I. Apart from their structural differences, i.e., with or without encoder or discriminator, all the common parameters used in these four network models are exactly the same.

C. Results and Analysis

For comparison, we first train four different generative network structures (see Table I) using the same training data, i.e., sketches from all 2300 viewpoints. First we test them on the 3D vase, wine glass, mug and key objects. For each of them, we generate three viewpoints to test these four network structure, see Figure 2. All the sketch results are produced with N = 3 (see Section III-C). It can be clearly observed that the sketches generated by the ED, D networks, i.e., with a discriminator, have stronger stability of the sketch structure while at the same time exhibit reasonable freehand drafting styles created by real humans. The sketches generated by the ED and D model without a discriminator are more likely to become cluttered. We conclude that adding a discriminator to the network can effectively help generate sketches with correct overall shapes.

We also conduct a user study to qualitatively analyze the generated sketch results. Then we invite 40 users to participate in the survey. First we prepare 60, i.e., 4×3×5 sketches, which are generated by the four network models in Table I, including three 3D objects: vases, wine glasses, hammers, and randomly select 5 different viewpoints for each 3D object. These 60 sketches are shown in Figure 6. Note that we set the temperature parameter τ for ED and D as 0.1, because even when the parameter is 0.1, the degree of randomness is very high. We set the parameter τ for ED and D as 0.3 to reduce their regularity. For each row of Figure 6, we first randomly shuffle the order of each sketch of the 4 networks, then we ask each user to vote for the sketch which looks more similar to human hand-drawings and more correctly describe the input shape. The statistic results are shown in Table II, which shows that very few people voted for ED, a small number of people voted for D and ED, and most people voted for D. It is easy to see that the user study results comply with our previous results and analysis that the network with a discriminator produces more aesthetically pleasing and acceptable sketches that are more preferred by average users.

We also test the effect of different values of the parameter N during our two-phase sampling step (Section III-C). Let
us recall that the first $N$ generated points are used as the anchor points to constrain the orientation of the generated sketch to be similar to the input condition. The test results are shown in Figure 3, where we show cases of 2 different viewpoints using our DDo model when $N$ equals 0, 1, 2, 3 and 4 respectively. For each viewpoint, 6 sketches are randomly sampled from our DDo model. When $N = 0$, the experimental result of the generation is completely random, and its angle of view is totally unaffected by the input. As $N$ increases, the angle of the generated sketch is closer to the input. We also observe that setting an overlarge $N$ could generate over-constrained sketches without the satisfactory degree of abstraction to express human hand drawing style. Note that we set the temperature parameter $\tau$ for all result of Figure 3 as 0.2.

In addition to the parameter $N$, we also use a temperature parameter $\tau$ to control the degree of randomness in results. We test the effect of different $\tau$ and the results are shown in Figure 4. Here we sample on two 3D objects: wine glass and hammer, where we increase $\tau$ from 0.1 to 0.8 by 0.1 interval. The sketch results are color-coded according to $\tau$, i.e., $\tau = 0.1$ is shown in blue, and $\tau = 0.8$ in red. The valid range of value of $\tau$ is between 0 and 1. The results clearly suggest that with a increasing $\tau$ the degree of randomness of the sketch increases gradually, which complies with our prediction here.

Finally we test the replicability of our approach by using different training datasets. Specifically we retrain our DDo network model using 3 different mixed datasets, see Figure 5. Here Dataset A consists of 50% Quickdraw sketches and 50% sketches from our synthetic dataset; Dataset B consists of 66% Quickdraw sketches and 34% sketches from our synthetic dataset; Dataset C consists of 100% Quickdraw sketches. Unfortunately we discover that even though the Quickdraw dataset clearly captures freehand sketch style, the mixed training fails to generate better results as universally exhibited in Figure 5. Even though all the results here are produced using our two-phase sampling method with $N = 3$ and the smallest possible $\tau = 0.1$, the retained network cannot correctly produce sketches with the same angle as the input. Please compare Figure 5 with Figure 2. The underlying reason for this is that most of the Quickdraw data are from those popular viewpoints that the average users prefer to select when they are asked to make a draft sketch of a certain object. For instance, most of the sketches of the wineglass in Quickdraw are from an upright and frontal viewpoint. This means the Quickdraw dataset does not provide enough information describing a 3D shape from any viewpoints.

Our approach has several limitations: Firstly, our method needs to train a separate neural network for each different 3D object, so it will be much powerful if we can achieve cross-modal generation in the future. Secondly, even though we specifically optimize for the balance between structural correctness and level of abstraction and distortion for our results, the actual human freehand drawings can still express much larger degree of diversity and abstraction than our results. Augmenting our dataset using real human hand drawing style. Note that the wrong orientation of the sketches.

Finally we test the replicability of our approach by using different training datasets. Specifically we retrain our DDo network model using 3 different mixed datasets, see Figure 5. Here Dataset A consists of 50% Quickdraw sketches and 50% sketches from our synthetic dataset; Dataset B consists of 66% Quickdraw sketches and 34% sketches from our synthetic dataset; Dataset C consists of 100% Quickdraw sketches. Unfortunately we discover that even though the Quickdraw dataset clearly captures freehand sketch style, the mixed training fails to generate better results as universally exhibited in Figure 5. Even though all the results here are produced using our two-phase sampling method with $N = 3$ and the smallest possible $\tau = 0.1$, the retained network cannot correctly produce sketches with the same angle as the input. Please compare Figure 5 with Figure 2. The underlying reason for this is that most of the Quickdraw data are from those popular viewpoints that the average users prefer to select when they are asked to make a draft sketch of a certain object. For instance, most of the sketches of the wineglass in Quickdraw are from an upright and frontal viewpoint. This means the Quickdraw dataset does not provide enough information describing a 3D shape from any viewpoints.

Our approach has several limitations: Firstly, our method needs to train a separate neural network for each different 3D object, so it will be much powerful if we can achieve cross-modal generation in the future. Secondly, even though we specifically optimize for the balance between structural correctness and level of abstraction and distortion for our results, the actual human freehand drawings can still express much larger degree of diversity and abstraction than our results. Augmenting our dataset using real human freehand drawing sketches database consisting of all possible viewpoints of various 3D models. We will also try to avoid the pre-processing process to enable full end-to-end sketch generation. Another direction is the exploration of other new neural network structure and vectorial representations of sketch lines for this topic.

**ACKNOWLEDGMENT**

Shizhe Zhou are supported by the grant of National Science Foundation of China (No. 61303147 ), Science Foundation...
of Creative Media, CityU. We gratefully acknowledge the Applied Computing and Interactive Media (ACIM) of School of Hong Kong (Project No. 7005176 (SCM)), and the Centre for partially supported by grants from the City University of Laboratory,Ministry of Education, China. Hongbo Fu was of Hunan Province (No. 2018JJ3064 ) and HPCSIP Key Laboratory,Ministry of Education, China. Hongbo Fu was partially supported by grants from the City University of Hong Kong (Project No. 7005176 (SCM)), and the Centre for Applied Computing and Interactive Media (ACIM) of School of Creative Media, CityU. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES