3D Car Shape Reconstruction from a Contour Sketch using GAN and Lazy Learning

Naoki Nozawa[1], Hubert P. H. Shum[2][a], Qi Feng[1], Edmond S. L. Ho[2][b] and Shigeo Morishima[3]
[1]*Department of Pure and Applied Physics, Waseda University, Tokyo, Japan*
[2]*Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne, UK*
[3]*Waseda Research Institute for Science and Engineering, Waseda University, Tokyo, Japan*
s112800563@akane.waseda.jp, hubert.shum@northumbria.ac.uk (corresponding author),
fengqi@ruri.waseda.jp, e.ho@northumbria.ac.uk, shigeo@waseda.jp

Keywords: Generative Adversarial Network, Lazy Learning, 3D Reconstruction, Sketch-based Interface, Car, Contour Sketch

Abstract: 3D car models are heavily used in computer games, visual effects, and even automotive designs. As a result, producing such models with minimal labour costs is increasingly more important. To tackle the challenge, we propose a novel system to reconstruct a 3D car using a single sketch image. The system learns from a synthetic database of 3D car models and their corresponding 2D contour sketches and segmentation masks, allowing effective training with minimal data collection cost. The core of the system is a machine learning pipeline that combines the use of a Generative Adversarial Network (GAN) and lazy learning. GAN, being a deep learning method, is capable of modelling complicated data distributions, enabling the effective modelling of a large variety of cars. Its major weakness is that as a global method, modelling the fine details in the local region is challenging. Lazy learning works well to preserve local features by generating a local subspace with relevant data samples. We demonstrate that the combined use of GAN and lazy learning produces is able to produce high-quality results, in which different types of cars with complicated local features can be generated effectively with a single sketch. Our method outperforms existing ones using other machine learning structures such as the variational autoencoder.

1 INTRODUCTION

3D car models are heavily used across multiple fields such as entertainment (Goedicke et al., 2018), visual effects (Rameau et al., 2016) and automotive designs (Umetani, 2017). The process to generate models that resemble similar features from the real-world ones is usually time-consuming and labour-intensive. Automatic approaches that reconstruct 3D models from a single image input can be served as effective solutions.

Despite significant research (Umetani, 2017) (Dinesh Reddy et al., 2018) in related areas, high-quality reconstruction of complicated 3D car models remains challenging due to several reasons. First, in animation, game, and automotive manufacturing industries, the design process usually involves concept arts of the car, which are typically represented as sketches on predefined viewpoints. Therefore, previous methods with photo inputs (Jiang et al., 2018) are not practical in designing new shapes or modifying existing designs. Second, for learning-based methods, different types of cars such as SUVs and trucks consist of significantly diverse features, making the process of learning complicated data distributions difficult for a single neural network (Nishida et al., 2016). Third, cars modelling with fine details is a distinct problem as cars have common features such as where to place the wheels, but also distinctive parts such as the shape of rear wings and roofs (Dinesh Reddy et al., 2018). Past research (Umetani, 2017) shows that it is challenging to learn a diverse car subspace that represents both common and distinctive car features well.

In this paper, we propose a novel system to reconstruct a 3D car using a single sketch image, enabling an effective car shape creation process. To provide a good user interface for creating car shape, we use contour sketches (Li et al., 2019) that contain car boundaries and salient inner edges as the input.

[a] https://orcid.org/0000-0001-5651-6039
[b] https://orcid.org/0000-0001-5862-106X

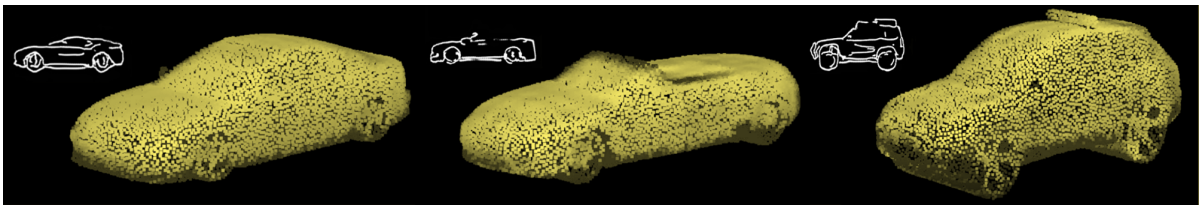


Figure 1: Examples of 3D car shapes generated by our system with side-view sketches. Top-left white lines are input contour sketches, while blue point clouds are corresponding outputs.

Such drawing greatly helps the car designing process as it allows users to directly use their understanding of scene geometry. We propose a data generating system that produces synthetic quadruplets of contour sketches, depth, segmentation masks and shape annotations from 3D car models obtained from ShapeNet (Chang et al., 2015) for facilitating an effective training process, and a feature-preserving car mesh augmentation pipeline to maximize the data variation.

To tackle the challenge of modelling 3D car shapes, we propose a novel 2-stage machine learning framework. In the first stage, We propose a GAN-based network that learns from contour sketches and 3D shapes to ensure a wide variety of car shapes. As the GAN (Goodfellow et al., 2014) has shown success in modelling 3D shapes (Sela et al., 2017) with its strong ability to model complicated data distributions, we adapt GAN to generate global shapes of 3D car models. Since it is computationally inefficient to directly learn 3D shape representation from mesh, we propose to learn an intermediate representation of multiple depth images instead, and reconstruct the 3D car mesh as a post-processing step. As global deep learning-based methods are limited in representing local details such as rear wings (Umetani, 2017; Güler et al., 2018), in the second stage, we introduce a lazy learning method to learn a local subspace from the relevant samples in the database. Compared to traditional approaches, lazy learning postpones the generalization of the database to run-time (Chai and Hodgins, 2005), which reduces the scale of learning by only considering the most relevant data. This facilitates the representation of local features that may be insignificant on a global scale. We further apply Principal Component Analysis (PCA) to improve the search space of point clouds, from which we search for the k-nearest neighbours. We finally perform a low-cost optimization process on the subspace to generate a 3D car shape with fine details.

Experimental results show consistent outputs of 3D car models generated form contour sketches with diverse shapes and topologies (Figure 1). In addition to well resembled global shapes of contour sketches, fine details and local features such as rear bumpers are also effectively preserved in the generated models (see Figure 8). When compared to existing methods, ours out-performs existing ones using other machine learning structures such as the variational autoencoder (VAE) (Nozawa et al., 2020).

The major contributions of this paper are summarized as follows:

- We propose a system to synthesize training data to construct a large database of 56,224 samples with contour sketches, depth, masks and 3d models. With realistic sketch-like features, the contour sketch facilitates real-world designing and editing applications.
- We propose a generative adversarial network to learn the correlation between a 2D contour sketch and the corresponding multi-view depth images that generate a 3D shape. Our GAN-based method out-performs existing learning-based approaches such as VAE with more diverse car topologies and shapes.
- We propose a lazy learning algorithm to learn a local subspace to reconstruct the fine detail features of the car. Such a subspace bases only on the relevant car shapes in the database, and therefore effectively retains detailed features in the samples.(Chang et al., 2015).

The preliminary results of this work have been presented in (Nozawa et al., 2020). In this work, we have made the following technical improvements. First, we represent a new database using contour sketches for a better sketch-like drawing style, while (Nozawa et al., 2020) uses a naive Laplacian filter to generate artificial samples. Second, we propose GAN that can more robustly learn the 3D representation of different car types due to its strong capability to model complicated data distributions, as oppose to (Nozawa et al., 2020) that used VAE with limited capacity on varied car topologies. Third, we evaluate the proposed system with a set of new experiments. We further compare our work with (Nozawa et al., 2020) to demonstrate the improvements.

The rest of the paper is organized as follows. We review previous work in Section 2. We explain the

construction of our car database with contour sketches in Section 3. We present our generative adversarial network for generating 3D car shapes from 2D sketches in Section 4. We present our lazy learning for constructing fine details for the car in Section 5. We show the results of our system in Section 6. We conclude the paper and discuss possible future directions in Section 7.

2 RELATED WORK

In this section, we first review previous sketch-based applications, followed by related work in the areas of sketch-style image rendering, Data Representations for 3D Shapes, and finally machine learning for 3D shape reconstruction.

2.1 Sketches for 3D Reconstruction

Sketches are powerful representations to capture users' design for computer graphics applications. In particular, we are interested in the problem in using sketches as a cue to reconstruct 3D shapes. Earlier work utilizes sketches with a predefined control interface to capture complicated 3D shape designs (Igarashi et al., 2007). Such a method is further extended to improve the smoothness of the generated 3D shape (Igarashi et al., 2006), as well as building the internal structure of the shapes (Owada et al., 2006). The control scheme was further enhanced to incorporate extra information that represents shape symmetry and angles (Gingold et al., 2009). A major problem for these methods is that the control scheme has to be manually designed, and users have to learn such schemes before using the system.

As an improvement, more general sketch-to-3D methods are proposed by using the input sketches to represent geometric features of the 3D shapes. For example, it is possible to use fit 3D primitive shapes into the input curves (Shtof et al., 2013), to represent 3D inflating surfaces (Joshi and Carr, 2008), or even to estimate the normal map of the surface of a shape (Shao et al., 2012). That said, these methods aim at using artificial intelligence to infer the sketch information provided by the users. With advanced machine learning such as deep learning becoming more and more available, we prefer to learning such a kind of logics automatically.

To create sketch images effectively for machine learning approaches, sketch-style rendering techniques are useful. These techniques use 2D lines to represent 3D shapes. Different visual cues such as image boundaries and edges are usually inferred from a single input image with edge detectors such as (Canny, 1986). However, these methods capture high-frequency signals without understanding the image. Boundary detectors, otherwise, understand the scene and yield semantic segmentation of different objects (Martin et al., 2001). However, object boundaries that only contain outer edges poorly resemble realistic drawing features. Recently, contour sketches (Li et al., 2019) are proposed to provide more sketch-like features while maintaining the ability to convey geometric information, salient edges and occlusion events. Contrary to professional computer-aided design software that requires professional training and has an engineering focus, sketch-based interfaces (Olsen et al., 2009) are more designer-friendly. To accommodate users with diverse drawing skills and artistic styles, we adapt contour sketches as the input of our approach.

2.2 Machine Learning for 3D Shape Reconstruction

3D shapes can be represented in different formats, and such representations affect the performance of machine learning approaches heavily. In general, there are three types of representation. The 3D point cloud has been used heavily for modelling 3D shapes due to its simplicity, enabling applications such as shape reconstruction (Fan et al., 2017) and shape classification (Qi et al., 2017). The disadvantage of the representation is that it does not represent volumetric information. As a solution, voxels are used as they can model the volumetric occupancy of a complex 3D shape, allowing more accurate shape reconstruction (Choy et al., 2016). However, although methods are proposed to relax the computation requirements using octrees (Tatarchenko et al., 2017), 3D operations are still computationally expensive. Depth images that represent different views of a 3D shape using 2D distance images can resolve this problem. As the surface information of a shape can be represented using multiple 2D views, 2D operations can be used to reconstruct 3D shapes (Li et al., 2018), thereby significantly reduces the computational requirements. The combined use of depth images and normal images can further enhance the representation power and give details to the surfaces during a reconstruction process (Isola et al., 2017). In this project, we utilize depth images as an intermediate output, such that we can relieve our system from using computationally expensive and difficult to optimize 3D operations. Multiple views of depth images are then combined to form a 3D shape using an analytical solution.

Traditional methods use multiple views to 3D re-

construct a scene. Reconstructing shapes from a single image is a challenging task but would benefit a wide range of real-world applications. With recent advances in deep learning, data-driven methods have gained increasing attention. Han et al. (Han et al., 2017) propose a convolutional neural network (CNN) based system to generate 3D faces from input sketches. Nishida et al. (Nishida et al., 2016) adapt a CNN to generate building models by adding surface curve information as a style of sketching. In the preliminary work (Nozawa et al., 2020), We also utilize deep learning for constructing the sketch-based interface by adapting the Variational Autoencoder (VAE) (Kingma et al., 2014) for correlating the 2D sketch and the output represented as depth and mask images. Although such a generative model has shown promising results in the translation of image style, its capability in modeling complicated data distribution is limited.

As a global model, GAN was introduced as a generative model to synthesize new instances from multiple predefined classes (Goodfellow et al., 2014). Other than computer graphic tasks such as image inpainting (Pathak et al., 2016) and texture transfer (Li and Wand, 2016), GAN has shown success in recovering the geometric structure from a single given image (Sela et al., 2017) (Jiang et al., 2018). When reconstructing the 3D shapes from image inputs, the adversarial loss of GAN is capable of working as a fidelity regularizer, and ensures that the generated samples share a close shape probability distribution with the training data. Considering the ability to learn complicated data distribution and the flexibility to support different inferences, we adapt GAN in our method to infer and generate global features of car models.

To further complement the expressiveness and represent fine details of the shapes that are specific to a small cluster of samples (Umetani, 2017). In the area of car reconstruction, different categories of cars have different specific details such as side mirrors and rear wings. Past research has shown that local models utilizing lazy learning can help to preserve fine details in different problems. Chai et al. (Chai and Hodgins, 2005) generate a human surface from a sparse input with a large motion database. Shum et al. (Shum et al., 2013) reconstruct noisy 3D human motion captured by Kinect using lazy learning. The main idea is to extract relevant data based on a run-time query and construct a local model during run-time. In this work, we adapt lazy learning to generate the fine details of a car based on the output generated by a deep learning network.

3 DATABASE CREATION

In this section, we present a robust and efficient process to construct a 3D car mesh database with contour sketches that highly resemble human sketches. A synthesis-based approach reduces the cost to acquire expensive paired 2D and 3D training samples, while the generated database can be easily extended with a larger size and scale. We generate contour sketches with a conditional GAN and create a point cloud representation from the 3D meshes. With contour sketch annotations, it is made easier for users with drawing skills to make use of the approach. With novel feature-preserving data augmentation techniques, we create a large variety of logically correct car meshes.

The database contains two sets of representations: 1) 2D contour sketches, depth and mask images for shape reconstruction, and (2) registered 3D point clouds for details synthesis.

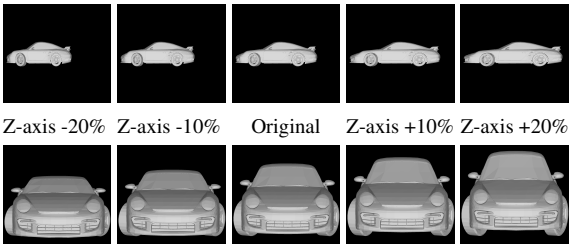


Figure 2: Examples of 3D car meshes synthesized with our feature-preserving data augmentation method.

3.1 Feature-preserving Car Mesh Augmentation

We follow the method in (Nozawa et al., 2020) to augment the 3D car meshes such that we can generate a larger database with more car variations. The key advantage of the method is that it can retain the local features of the car during the augmentation process. For example, while scaling a car, unlike previous methods (Sela et al., 2017) that would distort the proportion of the car wheels, ours can maintain the circular wheel shape.

Here, we summarize the augmentation method. Readers are referred to (Nozawa et al., 2020) for more details.

Our car meshes came from ShapeNet (Chang et al., 2015), as it offers different models of car with different variations. That said, the database is not big enough to effectively train the deep learning system. Therefore, we augment the car meshes following (Kraevoy et al., 2008), which allows us to change the

shape of the car while retaining local features at the same time. The core of the method is to voxelize the mesh and to interpolate the vertices based on the augmented (i.e. scaled) voxel grid. In our system, the resolution of the grid is set as $5 \times 10 \times 15$. We then scale the grid with $\pm 20\%$, $\pm 15\%$, $\pm 10\%$ and $\pm 5\%$ for the height and length directions. With an input of 7,028 cars meshes, with our augmentation method, we can create 56,224 meshes. Examples are shown in Figure 2.

3.2 The 2D contour sketches Depth and Masks Representation

With the 3D meshes created, we produce the corresponding 2D contour sketches and depth representations for training our deep learning system on car shape reconstruction. On the one hand, traditional edge detector such as a Canny edge detector (Canny, 1986) only capture high-frequency signals of an input image, and such samples with excessive details greatly differ from human drawings. On the other hand, segmentation models are usually trained with only the outer boundaries with no salient inner edges, resulting in oversimplified predictions that poorly represent geometric information of the input (Li et al., 2019). To make our method practical when applied to real-world scenarios such as automotive designing processes, we synthesize contour sketches, which contain both the outer boundaries and salient inner edges to represent occlusion events happen in the original photorealistic counterpart, generated 3D meshes.

To achieve high-quality contour sketch inferences from 3D car meshes, we adapt conditional GAN (Mirza and Osindero, 2014). Compared to a traditional GAN structure, we incorporate an additional L1 loss that compares the ground truth contour sketches with the predictions in addition to the adversarial loss of GAN. For generating contour sketches of cars with imperfect alignments, we freeze the weight that is trained with the Contour Sketches database (Li et al., 2019) and predict sketches with rendered images from certain perspectives (front, top, side) with a copper material which empirically best resemble the samples from the previous database, and then downsize the normalized cube faces to a 256×256 resolution for a more efficient generation process.

To create the depth and mask images, we set up a template cube that contains the normalized complete car shapes, and then obtain depth and rendered images from each face of the cube. For efficient calculation, we utilize a pixel shader and store them as floating-point values. We ignore the bottom face of the car and do not produce the corresponding depth and mask. This is because the bottom of the car typically consists of complicated geometry involving mechanical gears, which is unrelated to our application.

Figure 3: From left to right: the template, a car shape, and the flow for mapping them.

Compared to existing databases that poorly resemble realistic sketches such as the Laplacian filter-based approach (Nozawa et al., 2020), our mesh augmentation framework creates high-quality sketch-like annotations and generates a new database that solves the limitation of sub-optimal performance when the trained model is applied to real-world scenarios.

3.3 The Registered 3D Point Cloud Representation

We follow (Nozawa et al., 2020) to generate a registered 3D point clouds format from the car meshes. The importance of the registration process is that it allows machine learning systems to have a uniform input vector for effective learning. In our system, this is particularly helpful when we use lazy learning introduce fine details into the car shape. We will give a summary of the method here. Readers are referred to (Nozawa et al., 2020) for the implementation details.

We fist generate a point cloud format using Poisson sampling (Corsini et al., 2012). As we are aiming for a registered point cloud format, we need to have the same number of points per car. We control the total number of points by iteratively changing the radius of the Poisson sampling process. Once the number is within an acceptable range, we randomly take away points such that the number of point reaches a predefined value, which is set as 10,000 in our system.

We then register the point clouds from different cars by considering this process as an Earth Mover problem (Henry et al., 2014; Shen et al., 2019). This means that we will first select one point cloud randomly as the template, while treating all the rest as targets. For each point in the template point cloud, we will find the best mapping point in the target point cloud, such that the total distance between all template-target point pairs is minimized. Such one to one mappings for all points between the template and the target point clouds are called flow. Figure 3 visualize the flow between the template and the target.

4 GENERATIVE ADVERSARIAL NETWORK FOR CAR SHAPE RECONSTRUCTION

In this section, we present a deep neural network to reconstruct 3D car shapes from 2D contour sketches. With a GAN-based network, our method is capable of modelling complicated and distinctive shapes with an effective training process, and has the ability to generate high-quality shapes from a single contour sketch.

In the first stage of car shape reconstruction, we adapt Generative Adversarial Network (GAN) (Goodfellow et al., 2014) for getting the depth images and reconstruct a rough 3D shape that resembles the 2D contour sketches, as such a generative model has shown promising results in image translation by altering the input with a different style. After the shapes of cars are generated, we introduce the details of the car in the process of the second stage.

4.1 The Design of the Generative Adversarial Network

We propose a GAN-based network that learns from contour sketches and 3D shapes to ensure a wide variety of car shapes. Compared to previous designs such as Variational Autoencoder (VAE) (Kingma et al., 2014), GANs share superior performance (Isola et al., 2017; Chen and Koltun, 2017; Wang et al., 2018) in terms of the appearance of the output. More importantly, our novel network design takes input at the latent vector layer and generate multiple views from random noises, and thus reduce the expensive training cost and allow a larger variety of car shapes. To further highlight the distinctive features among different car shapes during the reconstruction such as spoilers and rear wings, instead of directly outputting the 3D point cloud (Fan et al., 2017; Charles et al., 2017) or the voxels (Delanoy et al., 2018; Choy et al., 2016) of a car, we propose to output a set of depth images from the side, top, front and rear views, and reconstruct the 3D vertices by combining them.

We adapt an encoder-decoder network structure as the generator for creating depth images (Li et al., 2018). We modified the design of the generator to add noise directly to the latent space, as shown in Figure 4. The decoder needs to generate depth images in multiple predefined views. On the one hand, existing research typically prepares multiple decoders, with one decoder generating one output view (Lun et al., 2017). However, such an approach increases computational cost and memory requirement significantly, considering that we need to generate five different views (i.e., front, rear, left, right, and top). On the other hand, traditional cGAN networks add noises to the input through concatenation, resulting in inefficient memory usage with increased input resolutions.

As a solution, our network shares the encoder among multiple views, and we novelly control the input at the latent vector layer to solve both limitations. This design is driven by the observation that there is shared information across different views. By sharing the same encoder, such information can be discovered. Apart from the massive reduction in memory usage and training time, such a setup allows the different output views to be more coherence and produces higher quality results. We justify our choice in the network design by conducting an ablation test in Section 6.3.

4.2 The Loss Function

For the depth images, we implement two loss functions - the mean absolute error (MAE, L1 loss) on the generated depth images, and the MAE on their Laplacian representations as in our pilot study (Nozawa et al., 2020). In particular, minimizing the MAE loss on the generated depth images can preserve the overall shape and structure of the car. On the other hand, including the MAE loss on the Laplacian representation can better preserve the surface appearance. Readers are referred to (Nozawa et al., 2020) for more details and justifications.

An adversarial loss is added to the discriminator. By concatenating all views of depth images and input to the discriminator, the model learns to distinguish if the set of depths is real or not. Masks are one of the sub-task outcomes for depths, so we ignored such mask images when training the discriminator. By training a single discriminator for multiple views, this design can save memory usage and learn the relationship between views simultaneously.

The final loss function is expressed as:

$$\begin{aligned} E = &\left(\boldsymbol{D}^{ref} - \boldsymbol{D}^{rec}\right) \circ \boldsymbol{M}^{ref}\big)_{L1} + \left(\boldsymbol{M}^{ref} - \boldsymbol{M}^{rec}\right)_{BCE} \\ &+ \left(\left(\Delta \boldsymbol{D}^{ref} - \Delta \boldsymbol{D}^{rec}\right) \circ \boldsymbol{M}^{ref}\right)\big)_{L1} + ADVLoss \end{aligned} \tag{1}$$

$$\begin{aligned} ADVLoss = \arg\min_{G}\max_{D} \{&\log D(\boldsymbol{S}, \boldsymbol{D}^{ref}) \\ &+ \log(1 - D(\boldsymbol{S}, G(\boldsymbol{S}, z))\} \end{aligned} \tag{2}$$

where $\boldsymbol{D}^{ref}$ and $\boldsymbol{M}^{ref}$ are depth and mask images of the ground truths, $\boldsymbol{D}^{rec}$ and $\boldsymbol{M}^{rec}$ are those of reconstructed images, $\boldsymbol{S}$ is input Sketch image, the subscripts *L1* and *BCE (binary cross-entropy)* represent the calculation metrics, Δ means Laplacian filtering

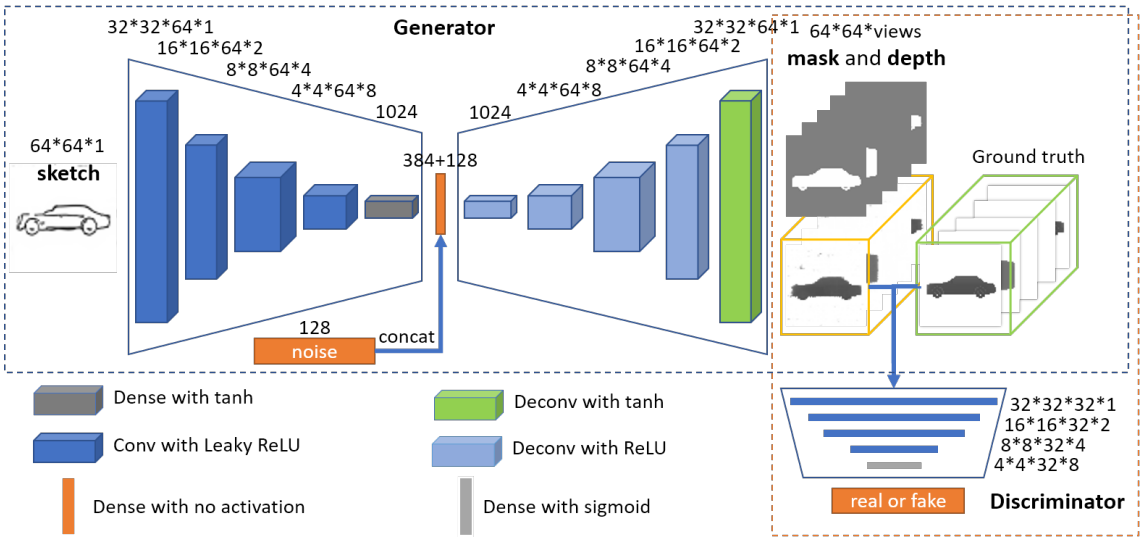


Figure 4: The design of our generative adversarial network.

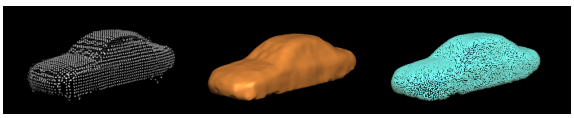


Figure 5: From left to right: the reconstructed shape, the reconstructed surface, and the registered point cloud.

and $\circ$ is the Hadamard product, *ADVLoss* is the adversarial loss function, G is the generator, D is the discriminator, and z is the random noise vector.

4.3 Surface Reconstruction

In this section, the process for reconstructing a rough point cloud from the generated depth and mask images generated by our proposed framework is presented.

As in our pilot study (Nozawa et al., 2020), the mask and depth images pairs from each view can be used for reconstructing the 3D point cloud of a part of the car. By aligning the parts reconstructed from different views, the overall shape of the car can be created (Figure 5 (left)) as a single point cloud. The surface of the entire car (Figure 5 (middle)) can then be reconstructed by applying Poisson surface reconstruction to the point cloud of the car. With the surface of the car, we can uniformly sample points from it and this step is equivalent to the point cloud standardization process as in the database creation (Section 3.3). By this, the register point cloud (Figure 5 (right)) can be directly compared with the example cars in our database and similar cars will be used in the lazy learning stage (Section 5) to add fine details to the car shape.

4.4 Implementation Details

Our framework is implemented with Tensorflow. For optimization, we use Adam solver with a learning rate 1e-5. The decoder has a dropout ratio of 0.5 except for the last layer. Inspired by pix2pix (Isola et al., 2017), We use Leaky ReLU as the activation function for the hidden layers in the encoder, and ReLU for that in the decoder. We use *tanh* as the activation function for output layers. More details regarding the network architecture can be referred to in Figure 5. To achieve high efficiency, the resolution of the images that are inputted in GAN is 64×64. To ensure an accurate evaluation with unaltered data, we train the system using the data generated by data augmentation, and test the system with the original data from ShapeNet (Chang et al., 2015).

5 LAZY LEARNING FOR FINE DETAILS

We follow our preliminary work (Nozawa et al., 2020) in developing lazy learning algorithms to reconstruct local details. Here, we give a summary of the algorithms and highlight the important system designs. We refer the readers to (Nozawa et al., 2020) for the implementation details.

While the main bodies of cars share a lot of common geometric similarities, the fine details such as side mirrors can be different. Learning a universal model from all cars with fine details is therefore highly ineffective. Motivated by the success of lazy learning in mesh processing (Chai and Hodgins,

2005; Shen et al., 2018; Ho et al., 2013), we propose to adapt lazy learning to reconstructed the details.

Unlike traditional approaches that generalize data in the whole database as a preprocess, lazy learning postpones the generalization to run-time (Chai and Hodgins, 2005). As a result, it can utilize run-time information to limit the scale of learning. In particular, given a run-time query, relevant data in the database can be extracted and a small-scale learning process can be performed. By only considering the most relevant data, the common features that may be insignificant on a global scale can be successfully represented. Besides, the similarity of relevant data allows lazy learning to use a much lower dimensional latent space comparing to traditional methods.

5.1 Relevant Data Search

Given a car shape generated in Section 4, we search for the most relevant samples from the database and perform lazy learning. As the point cloud is registered (i.e., it aligns with a pre-defined template car shape), we can effectively calculate the distance using the sum of Euclidean distances from all points between two point clouds. To reduce the high dimensionality of the point clouds during searching, we propose to apply Principal Component Analysis (PCA) onto the position of the point clouds to generate a search space, instead of using the Cartesian space. Searching with the more important components of PCA allows a faster search with less focus of fine details. Following (Nozawa et al., 2020), we set the root mean square distances of the 40 PCA components to find $k = 5$ nearest neighbours to achieve good results.

5.2 Learning and Optimization in Local Space

With the k nearest neighbours selected from the database, we can then learn a small subspace with PCA. Since these neighbours are similar to each other, the details of the shape can be well preserved with a smaller number of components. In such a subspace, we optimize a set of eigenvalues to construct a car shape that is as similar as possible to the one generated by deep learning. We then back project the eigenvalues to formulate a car shape with details such as the headlight, which is served as our final output.

We utilize the 3D Morphable Model (Blanz and Vetter, 1999) to optimize the eigenvalues of the components with a non-linear optimization process. Since the point clouds are registered, we use a simple point-to-point Euclidean distance to evaluate the distance between the optimizing shape and target shape in the Cartesian space. To obtain the Cartesian representation of the optimizing shape, we simply back-project the optimized eigenvalues to the Cartesian space.

We propose a simple pre-process that construct a more representative local PCA space with the k nearest neighbours to further improve the optimization process. Based on the observation that there are still small variations in car shapes within the k nearest neighbours, which distracts the system from the main objective of obtaining the detailed shape features, we pre-optimize these shapes individually using the same morphable model-based optimization process described above, such that they share a similar shape before we construct the local PCA space. This way, the significant components of the local PCA space can be more representative on the detailed shape features.

6 EXPERIMENTAL RESULTS

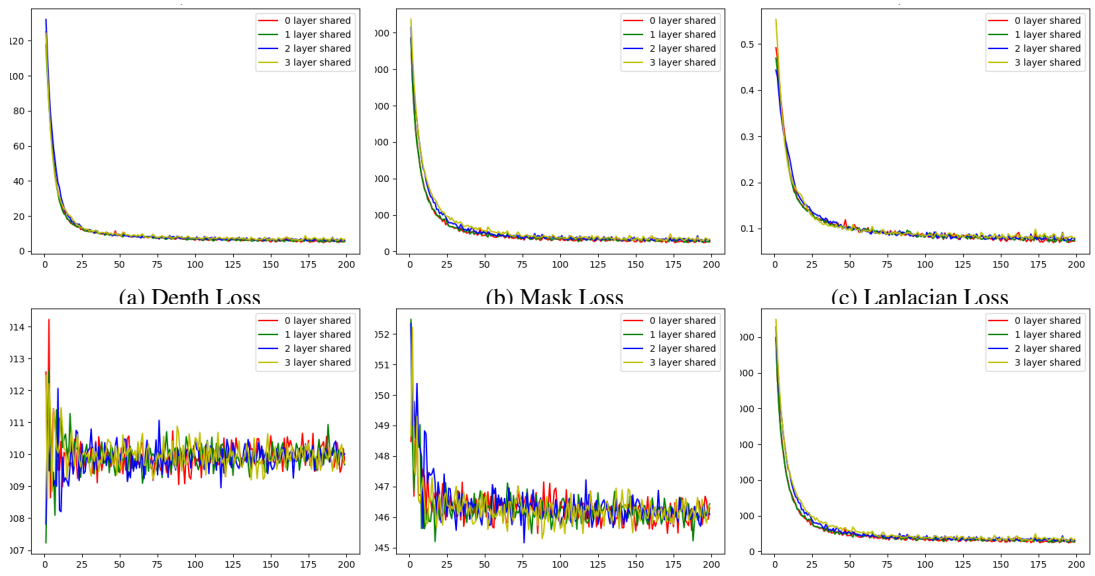
We will first present the experimental results on reconstructing 3D car shape from input sketches. Next, we quantitatively analyze the training loss during the training process to show the convergence of the proposed framework. Finally, a comparison with the state-of-the-art method (Nozawa et al., 2020) will be presented to demonstrate the results obtained from different network architectures and justify our choice.

The training of the deep learning system is performed with an NVIDIA GeForce RTX 2080 Ti GPU that has 11GB VRAM. With the batch size of 32, the training finishes within a few days. The run-time system is performed on a lower-end computer with an NVIDIA GeForce 1050 Ti GPU that has 4GB VRAM. The reconstruction of a car takes approximately 15 seconds to finish, with 5 seconds on car shape reconstruction (i.e., deep learning) and 10 seconds on reconstruction detail features (i.e., lazy learning).

6.1 Reconstructing 3D Shape from Contour Sketches

Since different users may have different drawing styles (e.g. more cartoon-like), real-world sketches are not objective to evaluate the performance of the proposed system. As a result, we utilize synthetic contour sketches for testing.

Examples of the output yielded by every major step of the proposed framework are illustrated in Figure 7. Stating with the input sketch (top row), depth images are computed for reconstructing the 3D meshes (2nd row). Next, a smooth surface is reconstructed from the mesh (3rd row) and a point cloud



(a) Depth Loss (b) Mask Loss (c) Laplacian Loss

(d) Generator Loss (e) Discriminator Loss (f) Total Loss

Figure 6: Losses across epoch during the training stage.

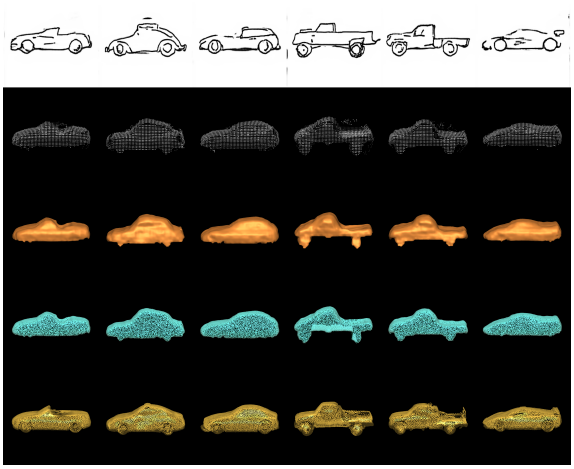


Figure 7: Intermediate outputs. From top to bottom: sketches, meshes from generated depth images, reconstructed surfaces, sampled point clouds on surfaces, and point clouds with details.

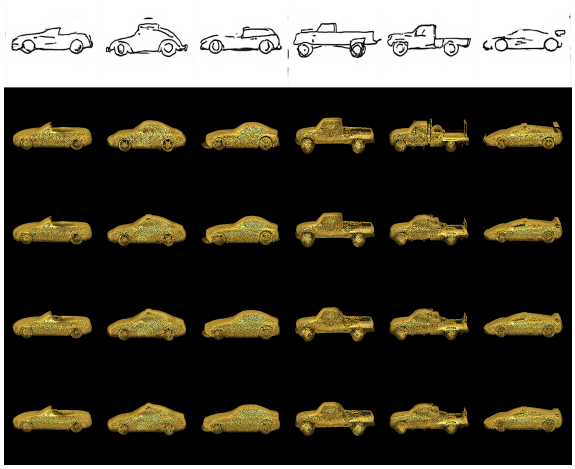


Figure 8: Results generated with different k. From top to bottom: sketches, point clouds with $k = 1$, $k = 3$, $k = 5$ and $k = 7$.

(4th row) is sampled from the surface for retrieving similar car shapes from the database for details refinement. Finally, the refined car model (bottom row) is created using the proposed lazy learning module. It can be seen that the meshes (Figure 7 2nd row) reconstructed from the proposed GAN framework can already resemble the car shape specified in the abstract input sketch. The proposed lazy learning module further enhances the quality of the 3D models by adding details such as side mirrors and spoilers (Figure 7 2nd row). This highlights the effectiveness of the proposed framework. Readers are referred to Figure 1 and the video demo accompanied for more examples.

However, details like grilles or wheels are not encoded well for practical use of games or movies. The EMD registration process can cause such low-quality appearances because the EMD is based on the theory of optimal transport with global distribution, which can ignore small features. Besides, the converting process into point clouds can reduce mesh resolution that is closely related to details. We will consider landmarks on 3D mesh in the sampling and hierarchical registration process for encoding such small features. Furthermore, the input sketches can affect

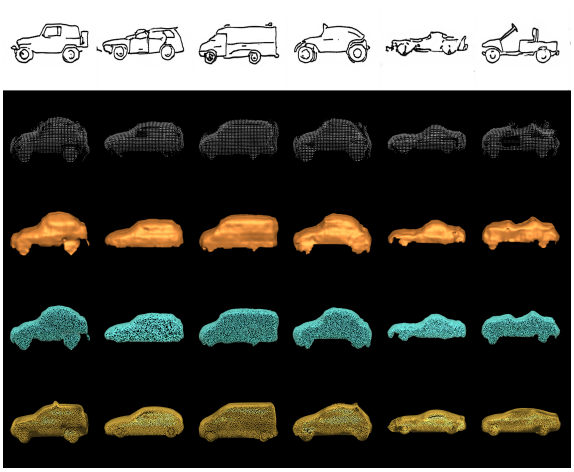


Figure 9: Lower-quality results for sketches that has few similar samples in the database.

appearance because of sparse information comparing with photorealistic images. Feature extraction from sketches is still an open problem in the field of deep learning, so we will update our network structure. An interactive sketch-based system will improve appearance as well.

While the point clouds generated from the proposed framework is highly realistic in term of the overall 3D shape, the system is less effective in reconstructing meshes with sharp edges. The underlying problem is related to the 3D point sampling process during the 3D mesh registration. The 3D point sampling tends to sample points around sharp edges instead of along the edges, which is a well-known problem in 3D point sampling (Huang et al., 2013). As a result, the sharp edges may be lost when the 3D surface is reconstructed from the sampled points using triangulation. While this is an interesting topic to explore, this is out of the scope of this work. Exploring the feasibility of using more advanced sampling methods such as (Gauthier and Poulin, 2009) or (Hanocka et al., 2020) can be an interesting future direction to further enhance the quality of the reconstructed 3D meshes.

We further conducted an experiment to have an in-depth analysis of the proposed lazy learning module. The proposed lazy learning module plays an important role in adding fine details such as the wheels and grills to the reconstructed 3D car shape. Here, we focus on the k-value that defines how many nearest-neighbor will be selected for learning the local space (Section 5). Different k-values are being used in this experiment and the results are presented in Figure 8. From the results, it can be seen that the points clouds tend to be more noise and contain more holes when $k = 1$ and $k = 3$ (Figure 8 2nd and 3rd rows). In addition, wrong details can be added to the final 3D point clouds as the lazy learning process is biased to a small number of samples when the k-value is small. For example, the shape of the bed of the truck on the 5th column (from left to right) in Figure 8 is different from the input sketch when $k = 1$ and $k = 3$. On the other hand, the 3D point clouds generated using $k = 7$ (Figure 8 bottom row) do not have the aforementioned artefacts. However, car shapes tend to be over-smoothed. Finally, using $k = 5$ (Figure 8 4th row) can generate a smooth surface while resembling the shape specified in the input sketch.

While our proposed framework can generate realistic car shapes from sketch images, some of the low-quality results are presented in Figure 9 for further analysis. We found that low-quality results are usually associated with the car shapes that are uncommon in the dataset. The underlying issue is related to the lack of similar car shapes for the refinement in the lazy learning step. For example, the 3D mesh (Figure 9 2nd row) generated by the proposed GAN framework has similar shapes as in the input sketch. However, the refined point clouds (Figure 9 bottom row) changed the shape of the cars, especially the leftmost and rightmost columns in Figure 9.

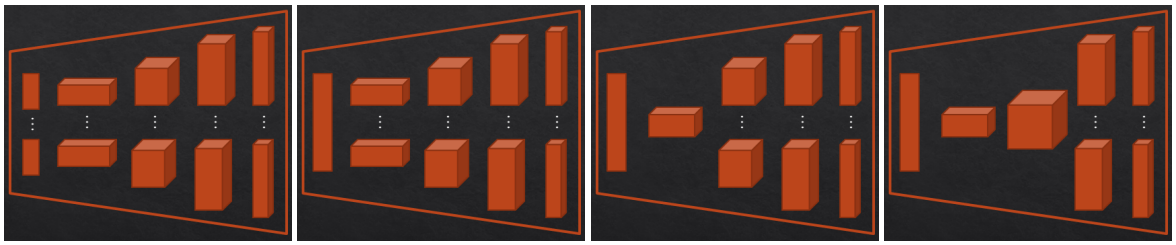
6.2 Training Loss

Here, we present the results of a quantitative evaluation of the performance of the training process in the proposed framework. A wide range of training loss plots are illustrated in Figure 6. The plots also contain the training losses obtained using 4 variants of the proposed decoder network as an ablation study and more detailed will be given in Section 6.3.

In particular, the depth loss, mask loss, Laplacian loss, and total loss are reduced stably as training progress. This highlights the proposed deep learning framework converges and can effectively improve the quality of the generated mesh in the training process. For the Generator and Discriminator Losses (Figure 6 (d) and (e)), the oscillations are mainly caused by the competitive nature between the Generator and Discriminator, which is a typical pattern in GAN frameworks and the losses show a decreasing trend in general.

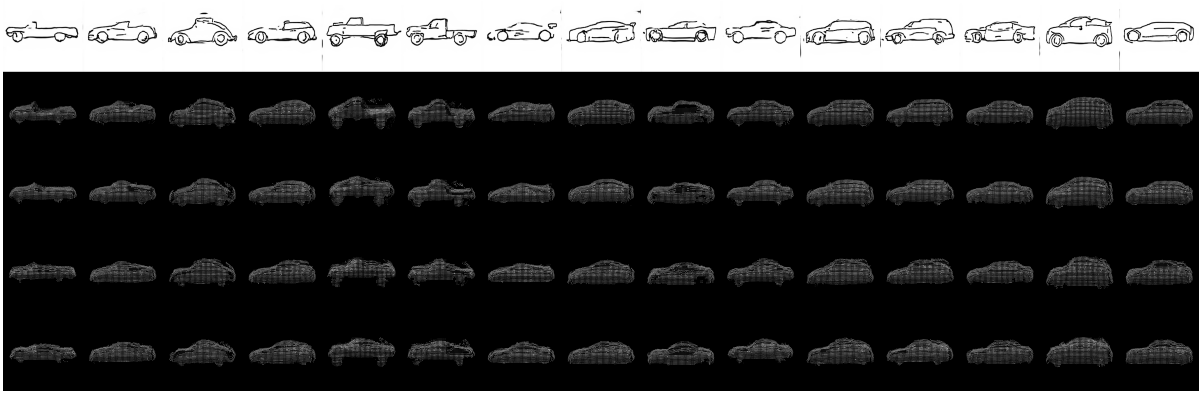
6.3 Comparing with the State-of-the-art and Ablation Tests on Different Decoder Network Architectures

In this section, we present the result of an ablation study to justify our network design and followed by

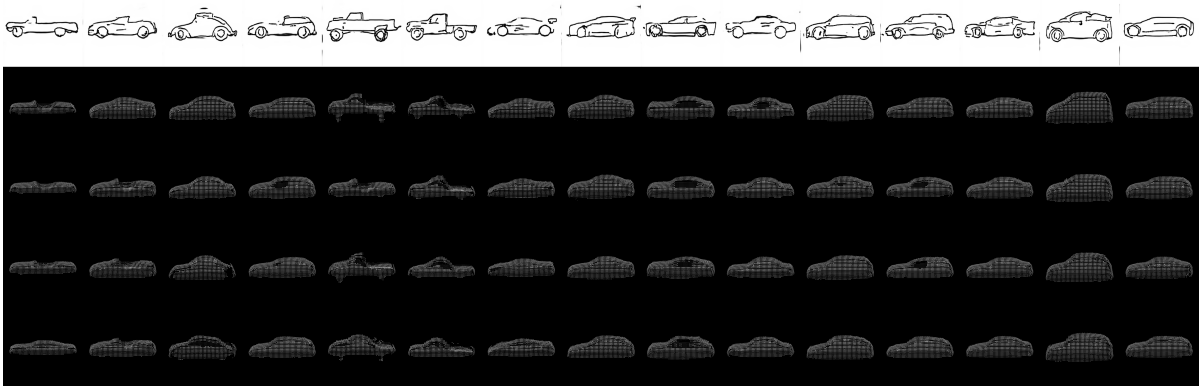


(a) No shared layer (b) A one-layer shared (c) Two layers shared (d) Three layers shared

Figure 10: Different decoder network architectures for the ablation test.



(a) Our proposed GAN network



(b) Nozawa et al. (Nozawa et al., 2020)

Figure 11: The 3D point clouds reconstructed with different decoder architectures. From top to bottom: input sketches, results of the decoder with no shared layer, sharing the first layer, sharing the first two layers, and sharing the first three layers.

comparing our results with those obtained using the State-of-the-art method (Nozawa et al., 2020). As explained in Section 4.1, decoders are used for generating depth and masks in different views for reconstructing the 3D shape of the car from input sketches. While the images in different views have a different appearance, they are associated with the same underlying 3D shape. As a result, Nozawa et al. (Nozawa et al., 2020) proposed sharing a common layer among the decoders in the network design to preserve the underlying structure and improve the consistency among all synthesized views. In our proposed encoder-decoder network (see Figure 4), each decoder consists of five layers. In the ablation test, we vary the number of shared layers in the decoder from 0 (i.e., not sharing any layer) to 3. The different decoder architectures are illustrated in Figure 10.

A wide range of 3D car shapes are reconstructed using different decoder network architectures and the results are illustrated in Figure 11 (a). It can be seen that our proposed decoder architecture without sharing any layer (2nd row in Figure 11) produces the best results in terms of reproducing the car shape with a smooth surface. On the other hand, sharing

layers (3rd, 4th and 5th rows in Figure 11) result in 3D point clouds with less distinguishable shape and noisy/rough surface, which can be caused the loss of balance between preserving the underlying structure among decoders and refining each view. On the other hand, Nozawa et al. (Nozawa et al., 2020) reported that results can be obtained using a decoder network with 1 shared layer. This highlights the differences between our proposed GAN framework and the VAE framework presented in (Nozawa et al., 2020). Unlike the VAE network, noise is added when encoding features from an input sketch in our GAN network. As in typical GANs, the noise works as the latent vector and represents the underlying 3D information of the original shape. As a result, the concept of reconstructing the underlying 3D shape by having a common layer for different depth views in the decoder is not needed. Without such an explicit constraint among different views, it can be observed that our proposed framework can still generate high-quality results.

As presented in Section 6.3, we evaluated the performance of our proposed method quantitatively. In Figure 6, the losses of the 4 variants (i.e. sharing 0-3 layers) of the decoder network are plotted. It can be seen the losses obtained by *0-layer shared* (colored in red) are the lowest in general. This further justifies our decoder network design.

Finally, we compare our results with those generated by the State-of-the-art method (Nozawa et al., 2020) and the results are presented in Figure 11. The results highlight that our proposed framework generated more realistic results that are closer to the car shape as in the input sketch and contains more fine details on the mesh. In contrast, the meshes generated by (Nozawa et al., 2020) are having less distinctive shapes and a lot of artefacts such as holes and noise on the surface.

7 CONCLUSION AND DISCUSSIONS

In this paper, we present a system to reconstruct detailed 3D car shapes with a single 2D contour sketch. To effectively learn the correlation between contour sketches and 3D cars, we propose a generative adversarial network (GAN) with an intermediate multi-view depth image representation as to the output, and construct the 3D cars as a post-processing step. To ensure the volume and diversity of the training data, we propose a feature-preserving augmentation pipeline to synthesize more car meshes with realistic sketch-like annotations while keeping the shape of important features. Finally, since deep learning has limited capability in representing fine details, we propose a lazy learning algorithm to construct a small subspace-based only on a few relevant database samples for optimizing a car shape with fine-detail features. We show that the system performs robustly in creating cars of substantially different shape and topology, with realistic detailed features included.

Our main focus in this work is to produce a 3D car models from a single sketch image given by the user. As a result, the proposed framework mainly resembles the exterior shape of the car without considering the configurations of the internal mechanical parts. One of the interesting future direction is to include additional constraints in the 3D car shape generation module to reserve space for the internal car parts.

We use multi-view depth images as an intermediate representation in the generative adversarial network. The two major advantages are that we do not need to deal with 3D deep learning, which is memory hungry and complicated to train, as well as we can have a more explicit 2D to 2D correlation. Currently, we combine the depth images as a post-processing step. However, it is possible to consider them as a means of rectifying the output space, and construct extra layers to learn the regression between multi-view depth images and 3D shapes. One future direction is to explore network architectures for this purpose, and introduce more views of depth images in a middle layer of the network for supervision.

The proposed methodology is generic to product design. It is expected that the framework can be applied to producing 3D shapes of other types of products from sketch images. This requires a new dataset with paired sketch images and 3D model for the new product type. In the future, we will explore in this direction such as producing 3D furniture models from sketch images with the IKEA dataset (Lim et al., 2013).

ACKNOWLEDGEMENTS

This project was supported in part by the Royal Society (Ref: IES\R2\181024 and IES\R1\191147), JST ACCEL (JPMJAC1602), JST-Mirai Program (JPMJMI19B2) and JSPS KAKENHI (JP19H01129).

Conflict of Interest: The authors declare that they have no conflict of interest.

REFERENCES

Blanz, V. and Vetter, T. (1999). A morphable model for the synthesis of 3d faces. In Proc. of the 26th Annual

Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99, pages 187–194, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

Canny, J. (1986). A computational approach to edge detection. IEEE Trans. on Pattern Analysis and Machine Intelligence, (6):679–698.

Chai, J. and Hodgins, J. K. (2005). Performance animation from low-dimensional control signals. ACM Trans. Graph., 24(3):686–696.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. (2015). ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Charles, R. Q., Su, H., Kaichun, M., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 77–85.

Chen, Q. and Koltun, V. (2017). Photographic image synthesis with cascaded refinement networks. In Proc. of the IEEE International Conference on Computer Vision, pages 1511–1520.

Choy, C. B., Xu, D., Gwak, J., Chen, K., and Savarese, S. (2016). 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In The European conference on computer vision (ECCV). Springer.

Corsini, M., Cignoni, P., and Scopigno, R. (2012). Efficient and flexible sampling with blue noise properties of triangular meshes. IEEE Trans. on Visualization and Computer Graphics, 18(6):914–924.

Delanoy, J., Aubry, M., Isola, P., Efros, A., and Bousseau, A. (2018). 3d sketching using multi-view deep volumetric prediction. Proc. of the ACM on Computer Graphics and Interactive Techniques, 1(21).

Dinesh Reddy, N., Vo, M., and Narasimhan, S. G. (2018). Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1906–1915.

Fan, H., Su, H., and Guibas, L. J. (2017). A point set generation network for 3d object reconstruction from a single image. In Proc. of the IEEE conference on computer vision and pattern recognition, pages 605–613.

Gauthier, M. and Poulin, P. (2009). Preserving sharp edges in geometry images. In Proc. of Graphics Interface 2009, GI '09, pages 1–6, Toronto, Ont., Canada, Canada. Canadian Information Processing Society.

Gingold, Y., Igarashi, T., and Zorin, D. (2009). Structured annotations for 2d-to-3d modeling. ACM Trans. Graph., 28(5):148:1–148:9.

Goedicke, D., Li, J., Evers, V., and Ju, W. (2018). Vr-oom: Virtual reality on-road driving simulation. In Proc. of the 2018 CHI Conference on Human Factors in Computing Systems, pages 1–11.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680.

Güler, R. A., Neverova, N., and Kokkinos, I. (2018). Densepose: Dense human pose estimation in the wild. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7297–7306.

Han, X., Gao, C., and Yu, Y. (2017). Deepsketch2face: a deep learning based sketching system for 3d face and caricature modeling. ACM Trans. Graph. (TOG), 36(4):126.

Hanocka, R., Metzer, G., Giryes, R., and Cohen-Or, D. (2020). Point2mesh: A self-prior for deformable meshes. arXiv preprint arXiv:2005.11084.

Henry, J., Shum, H. P. H., and Komura, T. (2014). Interactive formation control in complex environments. IEEE Trans. on Visualization and Computer Graphics, 20(2):211–222.

Ho, E. S. L., Shum, H. P. H., Cheung, Y.-m., and Yuen, P. C. (2013). Topology aware data-driven inverse kinematics. Computer Graphics Forum, 32(7):61–70.

Huang, H., Wu, S., Gong, M., Cohen-Or, D., Ascher, U., and Zhang, H. (2013). Edge-aware point set resampling. ACM Trans. Graph. (TOG), 32(1):1–12.

Igarashi, T., Igarashi, T., and Hughes, J. F. (2006). Smooth meshes for sketch-based freeform modeling. In ACM SIGGRAPH 2006 Courses, SIGGRAPH '06, New York, NY, USA. ACM.

Igarashi, T., Igarashi, T., Matsuoka, S., and Tanaka, H. (2007). Teddy: A sketching interface for 3d freeform design. In ACM SIGGRAPH 2007 Courses, SIGGRAPH '07, New York, NY, USA. ACM.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proc. of the IEEE conference on computer vision and pattern recognition, pages 1125–1134.

Jiang, L., Shi, S., Qi, X., and Jia, J. (2018). Gal: Geometric adversarial loss for single-view 3d-object reconstruction. In Proc. of the European Conference on Computer Vision (ECCV), pages 802–816.

Joshi, P. and Carr, N. A. (2008). Repoussé: Automatic inflation of 2d artwork. In SBM, pages 49–55. Citeseer.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised learning with deep generative models. In Advances in neural information processing systems, pages 3581–3589.

Kraevoy, V., Sheffer, A., Shamir, A., and Cohen-Or, D. (2008). Non-homogeneous resizing of complex models. In ACM SIGGRAPH Asia 2008 Papers, SIGGRAPH Asia '08, pages 111:1–111:9, New York, NY, USA. ACM.

Li, C., Pan, H., Liu, Y., Tong, X., Sheffer, A., and Wang, W. (2018). Robust flow-guided neural prediction for sketch-based freeform surface modeling. ACM Trans. Graph., 37(6):238:1–238:12.

Li, C. and Wand, M. (2016). Precomputed real-time texture synthesis with markovian generative adversarial networks. In European conference on computer vision, pages 702–716. Springer.

Li, M., Lin, Z., Mech, R., Yumer, E., and Ramanan, D. (2019). Photo-sketching: Inferring contour drawings from images. In 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 1403–1412. IEEE.

Lim, J. J., Pirsiavash, H., and Torralba, A. (2013). Parsing IKEA Objects: Fine Pose Estimation. ICCV.

Lun, Z., Gadelha, M., Kalogerakis, E., Maji, S., and Wang, R. (2017). 3d shape reconstruction from sketches via multi-view convolutional networks. In 2017 International Conference on 3D Vision (3DV), pages 67–77. IEEE.

Martin, D., Fowlkes, C., Tal, D., and Malik, J. (2001). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proc. of the Eighth IEEE International Conference on Computer Vision. ICCV 2001, volume 2, pages 416–423. IEEE.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

Nishida, G., Garcia-Dorado, I., Aliaga, D. G., Benes, B., and Bousseau, A. (2016). Interactive sketching of urban procedural models. ACM Trans. Graph. (TOG), 35(4):130.

Nozawa, N., Shum, H. P., Ho, E. S., and Morishima, S. (2020). Single sketch image based 3d car shape reconstruction with deep learning and lazy learning. In VISIGRAPP (1: GRAPP), pages 179–190.

Olsen, L., Samavati, F. F., Sousa, M. C., and Jorge, J. A. (2009). Sketch-based modeling: A survey. Computers & Graphics, 33(1):85 – 103.

Owada, S., Nielsen, F., Nakazawa, K., Igarashi, T., and Igarashi, T. (2006). A sketching interface for modeling the internal structures of 3d shapes. In ACM SIGGRAPH 2006 Courses, SIGGRAPH '06, New York, NY, USA. ACM.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In Proc. of the IEEE conference on computer vision and pattern recognition, pages 2536–2544.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proc. of the 31st International Conference on Neural Information Processing Systems, NIPS'17, pages 5105–5114, USA. Curran Associates Inc.

Rameau, F., Ha, H., Joo, K., Choi, J., Park, K., and Kweon, I. S. (2016). A real-time augmented reality system to see-through cars. IEEE Trans. on Visualization and Computer Graphics, 22(11):2395–2404.

Sela, M., Richardson, E., and Kimmel, R. (2017). Unrestricted facial geometry reconstruction using image-to-image translation. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 1585–1594.

Shao, C., Bousseau, A., Sheffer, A., and Singh, K. (2012). Crossshade: Shading concept sketches using cross-section curves. ACM Trans. Graph. (SIGGRAPH Conference Proceedings), 31(4).

Shen, Y., Henry, J., Wang, H., Ho, E. S. L., Komura, T., and Shum, H. P. H. (2018). Data-driven crowd motion control with multi-touch gestures. Computer Graphics Forum, 37(6):382–394.

Shen, Y., Yang, L., Ho, E. S. L., and Shum, H. P. H. (2019). Interaction-based human activity comparison. IEEE Trans. on Visualization and Computer Graphics.

Shtof, A., Agathos, A., Gingold, Y., Shamir, A., and Cohen-Or, D. (2013). Geosemantic snapping for sketch-based modeling. Computer Graphics Forum, 32(2pt2):245–253.

Shum, H. P. H., Ho, E. S. L., Jiang, Y., and Takagi, S. (2013). Real-time posture reconstruction for microsoft kinect. IEEE Trans. on Cybernetics, 43(5):1357–1369.

Tatarchenko, M., Dosovitskiy, A., and Brox, T. (2017). Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In The IEEE International Conference on Computer Vision (ICCV).

Umetani, N. (2017). Exploring generative 3d shapes using autoencoder networks. In SIGGRAPH Asia 2017 Technical Briefs, SA '17, pages 24:1–24:4, New York, NY, USA. ACM.

Wang, T., Liu, M., Zhu, J., Tao, A., Kautz, J., and Catanzaro, B. (2018). High-resolution image synthesis and semantic manipulation with conditional gans. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8798–8807.