STAR-TM: STructure Aware Reconstruction of Textured Mesh from Single Image

Tong Wu, Lin Gao*, Ling-Xiao Zhang, Yu-Kun Lai, and Hao Zhang.

Abstract—We present a novel method for single-view 3D reconstruction of *textured meshes*, with a focus to address the primary challenge surrounding texture inference and transfer. Our key observation is that *learning* textured reconstruction in a *structure-aware* and *globally consistent* manner is effective in handling the severe ill-posedness of the texturing problem and significant variations in object pose and texture details. Specifically, we perform structured mesh reconstruction, via a retrieval-and-assembly approach, to produce a set of genus-zero parts parameterized by deformable boxes and endowed with semantic information. For texturing, we first transfer visible colors from the input image onto the unified UV texture space of the deformable boxes. Then we combine a learned transformer model for per-part texture completion with a *global consistency loss* to optimize inter-part texture consistency. Our texture completion model operates in a VQ-VAE embedding space and is trained end-to-end, with the transformer training enhanced with retrieved texture instances to improve texture completion performance amid significant occlusion. Extensive experiments demonstrate higher-quality textured mesh reconstruction obtained by our method over state-of-the-art alternatives, both quantitatively and qualitatively, as reflected by a better recovery of texture coherence and details.

Index Terms—Structure-aware single-view 3D reconstruction, textured meshes, texture completion, transformer.

1 Introduction

Single-view 3D reconstruction has been one of the most classical problems in computer graphics and geometric deep learning. Up to now, most efforts have been devoted to *geometry* recovery [1], [2], [3], [4], [5], [6], [7], with more recent attempts aimed at improving the reconstruction of topological [8], [9] and surface details [10]. In comparison, there have been conspicuously few works [11], [12], [13], [14] on the single-view reconstruction of *textured* 3D shapes, despite the ubiquity of textured models in computer graphics.

Existing methods for learning textured shape reconstructions are all limited in terms of the shape topologies or texture details they can handle. In an earlier work, Im2Avatar [11] learns to reconstruct a voxel model and predict a color per voxel. But their voxelization is low-resolution, due to cost constraints, and is unable to represent high-frequency textures. Both the Differentiable Interpolation-Based Render (DIB-R) [13] and Soft Rasterizer (SoftRas) [14] employ deformable mesh templates and rely on neural rendering to set up the supervision signals. However, their template meshes are restricted to having sphere topology, with each corresponding to a single texture image. Flexible shape topologies can be naturally attained via implicit functions, as in Differentiable Volume Rendering (DVR) [12], which can be trained to predict both signed distances and color values. However, both the 3D shapes and textures reconstructed by DVR tend to be "blurry", due to significant variations in the training data, as well as sparse sampling during training to avoid high costs. Another common limitation shared by the above methods is that they all treat a 3D shape and its texture modeling *as a whole*, without accounting for the shape's part structure and its impact on texture inference during reconstruction.

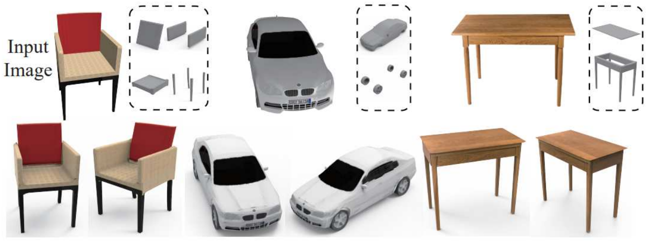


Figure 1. Single-view reconstruction of textured meshes by our *structure-aware* method, STAR-TM. Top: three input images, with the third one taken "in the wild". Bottom: reconstructed meshes in two views, with insets revealing the parts retrieved and assembled into the outputs. Note that the car wheels do not appear in the input image, but are well "hallucinated" by our learned transformer model for texture completion.

In this paper, we present a novel approach for single-view 3D reconstruction of textured meshes to address aforementioned challenges including topology variations and blurry textures. Our approach is data-driven since single-view reconstruction is an ill-posed problem and the ill-posedness becomes even more severe for texture inference and transfer. More importantly, our data-driven reconstruction framework is *structure-aware* and *globally consistent*, to

- * *Corresponding Author is Lin Gao (gaolin@ict.ac.cn).*
- *Tong Wu, Lin Gao, and Ling-Xiao Zhang are with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences. Tong Wu, Lin Gao, and Ling-Xiao Zhang are also with University of Chinese Academy of Sciences, Beijing, China.*
  E-mail: {wutong19s, gaolin, zhanglingxiao}@ict.ac.cn
- *Yu-Kun Lai is with School of Computer Science & Informatics, Cardiff University, UK.*
  E-mail: LaiY4@cardiff.ac.uk
- *Hao Zhang is with School of Computing Science, Simon Fraser University, Canada.*
  E-mail: haoz@sfu.ca
- *Project page: http://geometrylearning.com/STAR-TM/*

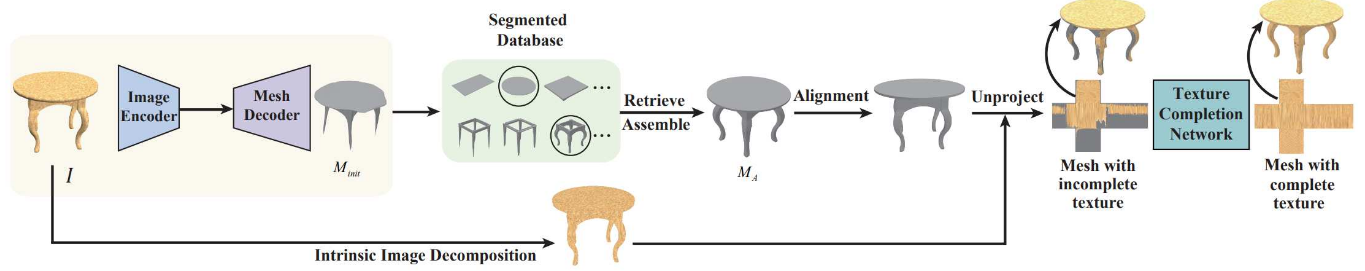


Image Encoder
Mesh Decoder
I
M_{init}
Segmented Database
Retrieve
Assemble
M_A
Alignment
Unproject
Mesh with incomplete texture
Texture Completion Network
Mesh with complete texture
Intrinsic Image Decomposition

Figure 2. Overall pipeline of our method. Given a single image I as input, it predicts an initial mesh M_{init}, then retrieves a set of shape parts with a fixed UV parameterization to assemble a structured mesh M_A. An intrinsic image decomposition network removes lighting effects from the input image I. Texture is then transferred from the input image to the visible regions of M_A, at the proper pose. A texture completion network, based on a transformer, is trained to finally complete missing textures to obtain a fully textured mesh. Note that the table base in the green frame is composed of several deformed boxes.

effectively handle both the ill-posedness of the texturing problem and the significant variations in object pose and texture details from input images. Our main motivation is that there is generally less texture variation across an object part than over an entire object, and shape semantics, e.g., symmetry, can further help facilitate the inference and completion of unseen textures.

Given a single input image of a textured object with arbitrary topology, we perform structured mesh reconstruction, by first predicting a genus-0 mesh using Pixel2Mesh [5] and then performing a *retrieval-and-assembly* approach that utilizes a dataset of segmented 3D objects with consistent semantic labels. The output mesh consists of a set of genus-zero parts parameterized by deformable boxes and endowed with semantic information. For texturing, we first "unproject" the input textures by transferring visible colors from the input image onto the unified UV texture space of the deformable boxes. Then we combine a learned transformer model for per-part texture completion with a global consistency loss to optimize inter-part texture consistency. Specifically, the loss enforces parts belonging to the same semantic group (e.g., all the legs of a chair) to be textured consistently. The overall pipeline is shown in Figure 2.

We have opted to design a transformer [15] for texture completion since it excels at long-range predictions. After the unprojection, there may be large gaps between textured regions on a reconstructed object part due to significant occlusion of the captured object. Our texture completion model operates in an *embedding space* and is trained end-to-end, with the transformer training enhanced with retrieved texture instances to improve texture completion amid significant gaps. The embedding space is constructed via a vector quantization variational autoencoder, specifically a VQ-VAE [16] (Vector-Quantized Variational Autoencoder), which has demonstrated success in learning and generating textured mesh parts with high-frequency details and less blurriness [17], compared to conventional VAEs.

To our knowledge, our method represents the first *part-wise* approach for single-view textured mesh reconstruction. Our main technical contribution lies in the texture prediction module that integrates the transformer model with a global consistency loss to encourage inter-part texture consistency in the reconstruction task. As shown in Figs. 1 and 8, our method works well under the synthetic setting and is applicable to input images "in the wild". Extensive experiments demonstrate higher-quality textured mesh reconstruction obtained by our method, over state-of-the-art alternatives including DIB-R and DVR, both quantitatively and qualitatively, as reflected by a better recovery of texture coherence and details. We also compare our texture completion transformer, which is trained with VQ-VAE encodings, with recent image completion networks [18], [19], including a transformer model which operates in texture space [18].

2 Related work

There has been substantial literature on single-view 3D reconstruction which produces a variety of outputs including point clouds (e.g., [1], [20], [21]), voxels (e.g., [3], [4]), implicit fields (e.g., [6], [7], [8], [10], [22], [23]), and meshes. Our coverage in this section shall focus on single-view *mesh* reconstruction, with more emphasis on structured reconstructions. In addition, we discuss related works on learning textured representations, recent image completion works that are most relevant to our method, as well as the language transformer which inspired our texture completion module.

Single-view mesh reconstruction. Early work by Xu et al. [24] retrieves and deforms a 3D mesh from a dataset, in a structure-preserving manner, to match the silhouette of the captured object. This was followed by Shen et al. [25] to allow parts from different shapes to be retrieved and assembled. Most learning-based methods designed to reconstruct a *single* mesh also rely on deformable mesh templates [13], [14], [26], [27], [28], [29], while in AtlasNet, Groueix et al. [2] decompose a mesh surface into multiple patches and learn a mapping from 2D images to a latent mesh representation for single-view reconstruction. We adopt Pixel2Mesh [5] in our first reconstruction stage. This method learns to deform a mesh template in a coarse-to-fine manner to improve the reconstruction of geometric details.

There have been relatively few works on single-view reconstruction of *structured* meshes. Im2Struct [30] learns to reconstruct an organized arrangement of oriented bounding boxes from an input image, using the generative recursive autoencoders developed by Li et al. in GRASS [31]. GSIR [32] not only predicts oriented bounding boxes from input images but also estimates more detailed geometry by inpainting reconstructed shapes' spherical maps similar to GenRe [33]. BSP-Net [34] learns a set of planes that can best reconstruct a set of training 3D shapes via binary space partitioning, followed by union and merging. Structured single-view reconstruction is possible by utilizing plane correspondences to propagate semantic labels manually provided to one shape. Uy et al. [35] reconstruct structural meshes from 2D images by hierarchically retrieving and deforming shape parts, in a joint learning framework, from the PartNet [36] dataset, which contains fine-grained, hierarchical part annotations. StructureNet [37] and LSD-StructureNet [38] encode shapes into a latent space with recursive geometry and structure encoders and reconstruct structured shapes by learning a mapping network from the input image to its learned latent space. Structural shape reconstruction is also studied in [39], [40].

Learning textured shape representations. Aside from DIB-R [13], Im2Avatar [11] and DVR [12] discussed in Section 1 , TextureFields (TF) [41] reduces the cost of voxel-based representations by learning to implicitly map shape geometry and texture into a high-dimensional latent space. However, TF focused on novel view synthesis from image, point, or depth inputs, rather than single-view reconstruction. There are also works based on implicit representations focusing on reconstructing textured shapes from RGB-D scans for human bodies [42], [43] and scenes [44], [45]. Other works [46], [47] study how to reconstruct textures for a given mesh. Pavllo et al. [48], [49] develop GAN-based generative models for textured meshes, also by deforming a single mesh template and predicting mesh vertex displacements and textures in a latent space. However, a single template mesh is unable to represent shapes with arbitrary topologies and a single UV image is limited by its resolution and expressiveness to generate or reconstruct high-quality and detailed textures. To boost the quality of reconstructed textured meshes, ViewGen [50], UMR [51], and Unicorn [52] introduce semantic consistency loss, cycle consistency loss, and neighborhood consistency loss to the training process.

Built on the autoencoder framework of SDM-Net [53], TM-Net [17] represents the first learned model for structured textured meshes in the form of multiple UV-parameterized deformable boxes. While exhibiting promising results on textured mesh autoencoding and generative tasks, TM-Net was not adapted for single-view reconstruction.

Image completion. Existing literature on image completion and inpainting is rich, e.g., [54], [55], [56], [57]. We only discuss two recent works that are closely related to our texture completion network. In CTSDG, Guo et al. [19] divide the problem into structure completion and texture completion, and develop two encoder-decoder networks to predict complete structures and textures simultaneously. A feature fusion and contextual feature aggregation module is then applied to refine the result. With the success of vision transformers, the Image Completion Transformer (ICT) by Wan et al. [18] first downsamples an incomplete image to a low-resolution patch (e.g., 32×32) and trains a transformer to predict the corresponding complete patch. This is followed by a guided upsampling network to restore the input image resolution. We observe that the completion results by ICT tend to exhibit blurriness and artifacts, likely due to the loss of information during downsampling which is unrecoverable by the upsampling. In contrast, our transformer model operates on a VQ-VAE embedding space which leads to better performance in retaining and recovering texture details, as we demonstrate in Section 4.

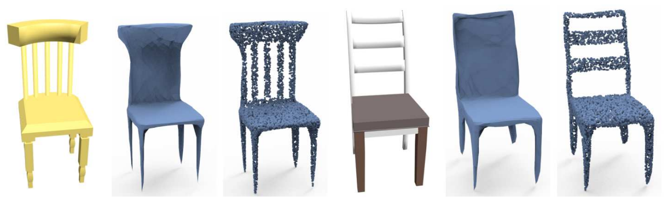


Figure 3. Two results showing that the cropping process operating in image space and utilizing the background turns a genus-0 mesh predicted by Pixel2Mesh into a high-genus point cloud (PC): (c) and (f).

Retrieval-enhanced language models. Recently, large pre-trained models such as GPT-2 [58] have achieved impressive feat on many natural language processing (NLP) tasks. At the same time, *retrieval-enhanced* language models [59], [60], [61], [62], [63] have gained further attention as they learn to search and retrieve training data related to a given test input to better utilize the priors in the training dataset. Our transformer design follows the retrieval-enhanced approach developed in RETRO [64], an NLP model which decomposes sentences into chunks and searches nearest neighbors for each chunk to obtain fine-grained retrieval results. It also introduced a chunked cross attention module to fuse each chunk and its corresponding nearest neighbors.

3 Method

To present our single-view textured mesh reconstruction, we first show how we reconstruct a structured mesh, which provides the unified UV mapping for texture recovery. In the following section, we describe our learning approach to obtain high-quality textures for the mesh.

3.1 Geometry Reconstruction

For geometry reconstruction, we first adopt Pixel2Mesh [5] to predict an initial mesh M_{init} from the input image I.

Topology recovery. It should be noted that the M_{init} produced by Pixel2Mesh is a genus-0 ellipsoid in the camera coordinates. Following the setting of DISN [8] and D2IM-Net [10], we assume that the camera intrinsic parameters K are known. To support arbitrary topology, we project sampled points M_{init} to the image plane using the intrinsic parameters K and remove those points whose projections lie in the background area of the input image I. This "cropping" step would turn M_{init} into a point cloud P_{crop} to allow the reconstructed meshes to have a non-zero genus. Fig. 3 shows two such examples.

Structured mesh construction. To attain geometry quality, we take a *retrieval-and-assembly* approach by utilizing high-quality 3D shapes from existing datasets in a part-wise manner.

To retrieve shapes in the dataset, we need to put the predicted mesh M_{init} which is in the camera coordinates into the same coordinates as the shapes in the dataset. To estimate the extrinsic parameters that represent the transformation between these two coordinates, we first rotate the cropped point cloud P_{crop} sampled from the predicted mesh M_{init} to make its oriented bounding box's faces parallel to

XY, XZ, and YZ planes and translate its center to the origin. Then we rotate P_{crop} along the three axes to find the transformation that minimizes the Chamfer distance between the cropped point cloud P_{crop} and shapes in the dataset. In this way, we estimate the extrinsic parameters and transform the cropped point cloud to the same coordinates as the shapes in the dataset. In the estimation process, we found that randomly sampling 100 shapes from the training set is sufficient to estimate a good transformation. For the detailed algorithm of the camera extrinsics estimation process, please refer to the supplementary material. After transforming the cropped point cloud into the same coordinates with shapes in the dataset, we further scale and translate each shape in the dataset to make sure it fits into the bounding box of the cropped point cloud and perform the Iterative-Closest-Point algorithm [65] to align shapes in the training dataset with the cropped point cloud before retrieval and assembly.

With the predicted mesh M_{init} providing 3D information, the input image I providing 2D information, the estimated extrinsic parameters, as well as the already known intrinsic parameters K, we perform retrieval and assembly at the part level by maximizing the 2D similarity between the input image and the assembled shape's projection and the 3D similarity between the predicted mesh M_{init} and the assembled shape following [25] to obtain an assembled mesh M_A from a dataset, where consistent semantic information is defined and each shape part is approximated by a deformable box [53]. As a result, the assembled shape M_A is composed of multiple deformed template boxes that have a unified UV texture space. For more details of the retrieval-and-assembly process, please refer to the supplementary material.

Structured mesh alignment. For the accurate texture unprojection, we would align the reconstructed mesh M_A with the input image I in both global and local parts manners. Then the visible texture is unprojected from the input image I to the mesh surface of M_A. Finally, a texture completion network with a retrieval-enhanced module and global consistency loss is trained to complete the unseen area in the UV parameterization space. To alleviate misalignments between the mesh M_A and the input image I, we propose to first optimize the global transformation of the assembled shape M_A via a soft neural renderer [14]. Let T_g denote the global transformation for the mesh M_A to be optimized and I_s is the silhouette of the input image. The global optimization process can be formulated as:

$$L_g = L_{IOU}(P(T_g \times M_A, K), I_s), \tag{1}$$

where P represents the neural render and $\times$ denotes matrix product. L_{IOU} is the Intersection-Over-Union(IOU) loss in DIB-R [13] defined as:

$$L_{IOU} = 1 - \frac{\|I_s \odot I_s^{'}\|_1}{\|I_s + I_s^{'} - I_s \odot I_s^{'}\|_1} \tag{2}$$

where $I_s^{'}$ is the silhouette of the assembled mesh M_A.

The global alignment cannot ensure that each shape part is aligned to the input image. Further, we use a similar optimization strategy to optimize the transformation for each shape part in M_A. Let T_l^i denote the local transformations for the i-th shape part M_A^i in the mesh M_A as the optimization target. We minimize the following loss to optimize the local transformations:

$$\begin{aligned} L_l = &\sum_i v_i L_{IOU}\left(P\left(T_l^i \times \hat{T}_g \times M_A^i, K\right), I_s\right) + \\ &\lambda_{sym} \sum_{j \in sym(i)} L_{cd}\left(ref\left(T_l^i \times \hat{T}_g \times M_A^i, n_j, d_j\right), T_l^j \times \hat{T}_g \times M_A^j\right) \end{aligned} \tag{3}$$

where $\hat{T}_g$ denotes the optimized global transformation, v_i denotes the visibility of the i-th part. $v_i = 1$ if the ratio of visible vertices not blocked by other parts to visible vertices is over 0.5. $sym(i)$ is a set of symmetry parts' indices for the i-th part, L_{cd} stands for the Chamfer distance operator. n_j and d_j are the normal vector and offset of the reflection plane between the i-th part and the j-th part. $ref(\cdot,\cdot,\cdot)$ is the reflection function and defined as: $ref(p,n,d) = p - 2\frac{p \cdot n + d}{\|n\|^2} n$. λ_{sym} controls the relative importance of the second term and is set to 0.5 in our experiment. We iterate 20 times with a learning rate 10^{-2} for both global optimization and local optimization. After transformation optimization, the location of each shape part is adjusted to make its projection better fit the input image.

Motivation. The reasons why we choose to reconstruct structured meshes in a retrieval-and-assembly manner based on Pixel2Mesh instead of predicting from a latent representation, e.g., through TM-Net [17], are three-fold. Firstly, meshes predicted from latent representations are usually smooth in terms of geometry. By retrieving and assembling the training data, we can produce 3D reconstructions with fine geometry details. Secondly, our *retrieval-and-assembly* approach transforms the reconstructed mesh into a structural representation consisting of several genus-0 parts with a fixed texture mapping to facilitate texture reconstruction. Finally, the projection of the mesh reconstructed by Pixel2Mesh is well aligned to the input image, allowing for accurate texture un-projection from the input image to the mesh surface.

3.2 Texture Reconstruction

Before reconstructing texture, we remove the shadow from the input image and extract the albedo component using the recently proposed intrinsic image decomposition method [66]. Then we unproject texture from the input image I onto each shape part's surface and then to its UV space. To obtain a fully textured mesh, we need to fill the missing texture, which poses several challenges:

- First, texture images often contain high-frequency details, which are hard to be encoded or generated.
- Second, visible areas are quite limited due to occlusions, and missing pixels can be spatially far from those visible pixels in the UV texture space.
- Last but not least, reconstructed textures should be consistent or compatible in the same semantic group. For example, the four legs of a chair are expected to have the same texture.

To solve the above problems, we first apply a vector quantization (VQ) based texture encoding network inspired by TM-Net [17] to encode texture details. With the texture

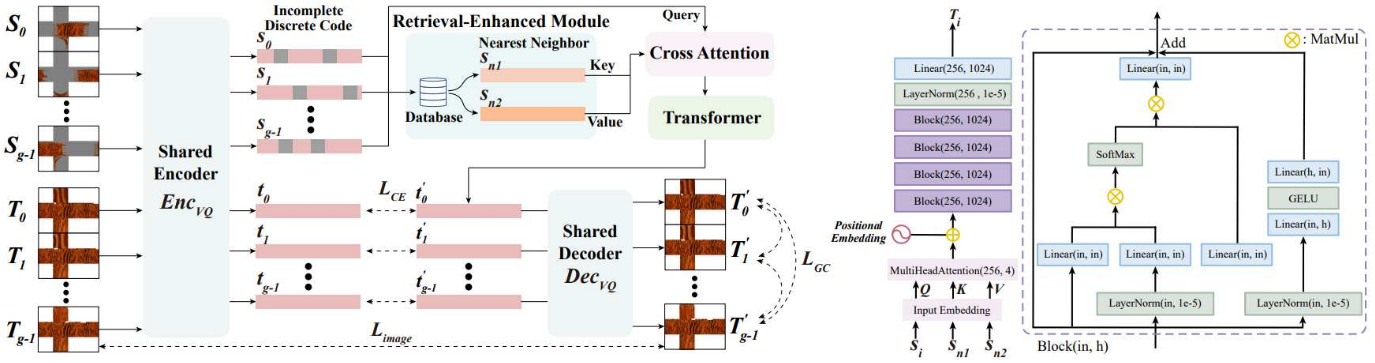


Figure 4. The architecture of our texture completion network. Our texture completion network takes several incomplete textures $S_0, \ldots, S_{g-1}$ in the same semantic group as input and encodes them into discrete codes $s_0, \ldots, s_{g-1}$. Our retrieval-enhanced module searches for the two nearest neighbors s_{n1}, s_{n2} for these discrete codes from the training set, which goes through a cross-attention module and a transformer decoder together with the discrete codes to predict the discrete codes of complete textures $t'_0, \ldots, t'_{g-1}$. At last the pre-trained vector quantization decoder decodes the predicted discrete codes into complete textures $T'_0, \ldots, T'_{g-1}$ for these parts. Our model minimizes the difference between predicted complete textures and ground truth and the perceptual differences between predicted complete images of different parts via a global consistency loss.

encoding network embedding textures into a code vector, the problem is turned to completing texture images in the vector quantization embedding space. To model the long-range relation between missing pixels and visible pixels, it is natural to adopt the transformer here as it is good at learning relationships between elements in a long sequence.

In addition, we apply a retrieval-enhanced module to search in the dataset for information related to the given incomplete texture to better use the knowledge prior in the training set and get complete textures with high quality. To promote texture consistency among different parts, the relationship between different parts in the same semantic group is taken into account with a global consistency loss to minimize perceptual differences among different parts.

Texture encoding. Let S denote the incomplete texture of a shape part and T denote the corresponding ground truth texture for the part; see Section 4 under "Data" for an explanation of how we prepare these textures. As texture images usually contain repetitive patterns, we first train a vector quantization (VQ) based network to encode them. This network consists of an encoder Enc_{VQ}, a codebook storing representative features, and a decoder Dec_{VQ}.

The encoder first encodes the texture image S into a continuous feature map s_e, which passes through VQ to map each feature vector in s_e to its nearest neighbor e in the codebook to obtain a discrete feature map s_q. To capture the data distribution in the training set, the codebook is updated during training as in [16]. The decoder then decodes the discrete feature map s_q to reconstruct the texture image S'.

After the texture encoding network is trained, we use the pre-trained encoder Enc_{VQ} to encode the incomplete texture S and the corresponding ground truth T into their discrete codes $s = Enc_{VQ}(S)$ and $t = Enc_{VQ}(T)$, in which each element stores an entry to the codebook. So we convert the texture completion problem into translating the discrete code s of the incomplete texture to the discrete code t of the ground truth texture.

Retrieval-enhanced texture completion. As unseen pixels on the incomplete texture can be spatially far from visible ones, it is natural to adopt a transformer network here for texture completion which can better capture the relationships between elements in a long sequence or with long-range dependencies. Our texture completion network is illustrated in Fig. 4. Further, we observe that incomplete texture can have a large unseen area or is even totally invisible. To boost the completion quality, we propose a retrieval-enhanced module [64] to make better use of the training set prior. Given the discrete code s of an incomplete texture, the module retrieves two nearest neighbors s_{n1} and s_{n2} from the training set in the discrete code space. We then project the incomplete texture's discrete code s, s_{n1}, and s_{n2} into feature vectors via a learnable embedding and add extra positional embeddings [15] onto it to form the intermediate feature representations E, E_{n1}, and E_{n2} in d dimensions. The positional embedding is extracted by a positional encoding module $\mathbf{PE}$ defined as:

$$\mathbf{PE}(p) = (sin(2^0\pi p), cos(2^0\pi p), ..., sin(2^{L-1}\pi p), cos(2^{L-1}\pi p)) \tag{4}$$

The positional encoding module PE takes each element p in a feature vector as input and outputs its positional embedding feature. Then a cross-attention module takes E, E_{n1} and E_{n2} as query, key, and value respectively to fuse them together. The retrieval-enhanced module can be formulated as:

$$\begin{aligned} head_i &= softmax\left(\frac{E\mathbf{W_i^Q}(E_{n1}\mathbf{W_i^K})^T}{\sqrt{d}}\right)\left(E_{n2}\mathbf{W_i^V}\right), \\ \mathbf{RE}(E, E_{n1}, E_{n2}) &= [head_1; ...; head_h]\,\mathbf{W^O}, 1 \le i \le h, \end{aligned} \tag{5}$$

where h represents the number of heads, $\mathbf{W_i^Q}$, $\mathbf{W_i^K}$, and $\mathbf{W_i^V}$ are three linear projection layers. $\mathbf{W^O}$ is also a learnable fully connected layer, which aggregates the outputs from different heads. The cross-attention module learns to find which values in E, E_{n1} and E_{n2} are most relevant to the incomplete texture's discrete code and deserve the largest attention coefficients.

Globally consistent texture completion. It is common that shape parts in the same semantic group share the same or similar texture patterns, for example, textures for different legs in a chair are usually the same, which reminds us that invisible areas for shape parts in the same semantic group should be filled simultaneously instead of individually. Hence we introduce a global consistency loss in the completion network to ensure the consistency between inpainted results of different parts in the same semantic group. Let $s_0, \dots, s_{g-1}$ be discrete codes of g incomplete textures in the same semantic group. Then we use the retrieval-enhanced module to search for the two closest neighbors in the discrete code space for this semantic group via a weighted sum of distances to these parts where weights are defined by the number of visible pixels in incomplete textures for different parts. The texture completion network then takes $s_0, \dots, s_{g-1}$ and the two nearest neighbors as input to predict complete discrete codes $t_0^{'}, \dots, t_{g-1}^{'}$. These complete discrete codes are decoded by the Vector Quantization network to generate complete texture images $T_0^{'}, \dots, T_{g-1}^{'}$, where $T_i^{'} = Dec_{VQ}(t_i^{'})$ is the complete texture for the i-th part. The global consistency loss is defined as $L_{gc} = \sum_{i=0}^{g-1} \sum_{j=i+1}^{g-1} LPIPS(T_i^{'}, T_j^{'})$, where $LPIPS(\cdot, \cdot)$ is the perceptual similarity operator [67]. Adopting perceptual loss here instead of using L1 or L2 loss is because different parts in the same semantic group are not guaranteed to have exactly the same textures so may not be pixel-wise identical.

Overall, the texture completion network minimizes the loss:

$$\begin{aligned} L_{completion} &= L_{image} + \lambda_1 L_{CE} + \lambda_2 L_{gc} \\ &= \sum_{i=0}^{g-1} \|T_i^{'} - T_i\|_1 + \lambda_1 CE(t_i^{'}, t_i) + \lambda_2 L_{gc} \end{aligned} \tag{6}$$

where t_i and T_i are the ground truth discrete code and texture for the i-th part in the group. The first term stands for the mean L1 loss and $CE(\cdot, \cdot)$ stands for the cross entropy loss. λ_1 and λ_2 control the relative importance of each loss term. Note that since we know which pixel is invisible in the incomplete image S and which element is unknown in discrete code s by downsampling the input visibility mask, we only calculate losses in invisible areas in Eqn. 6.

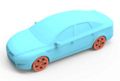
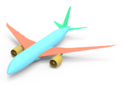
Network architecture. The input image to our vector quantization network is downsampled four times by the encoder Enc_{VQ} and converted into an intermediate feature map of size $24 \times 32 \times 256$. The feature map then goes through quantization and is transformed into the quantized feature map. Finally, the decoder Dec_{VQ} decodes the quantized feature map to reconstruct the input image. The detailed network architecture is depicted in the supplementary material.

The architectural detail of our texture completion transformer network is shown on the right of Fig. 4. The input sequence first goes through a retrieval-enhanced module that finds the two closest neighbors from the training dataset. Then the input sequence and its two neighbors go through a multi-head attention module to turn into a fused feature. After that, the fused feature is fed into a 4-layer self-attention block, a normalization layer, and a fully connected layer and converted to the output.

4 Results and Evaluations

We first introduce the data we used in our experiments and how we generate ground truth textures and incomplete textures for training the texture completion network. Then we present our implementation details including time and hyper-parameters we use. We also give a brief introduction of baseline methods and metrics we use in quantitative and qualitative experiments. Finally, we will present the results of comparisons and ablation studies.

Table 1
Semantic labels for each object category used in our retrieval and assembly procedure. The inset figures provide one example per category with color matching between labels and object parts.

Category	Semantic Labels	Example
Car	'bodywork', 'chassis'	
Chair	'head', 'back', 'arm', 'base', 'seat'	
Plane	'fuselage', 'wings', 'tails', 'power_system'	
Table	'tabletop', 'table_base'	

Data. We perform experiments on the four largest object categories in the ShapeNet dataset [68]: chairs, tables, cars, and planes, as they provide ample texture data to train our texture completion network. Segmentation information for chairs and tables are from PartNet [36], while for cars and planes, we use semantic labels from SDM-Net [53]. Table 1 summarizes the semantic labels used to retrieve and assemble shapes by our method STAR-TM. Each object category is randomly divided into training and test sets, with a ratio of 4:1.

To provide the ground truth textures, we non-rigidly deform a template box with a fixed UV parameterization to each semantic part and project vertices on the box to the ShapeNet mesh to obtain vertex colors following TM-Net [17]. For incomplete textures, we follow Choy et al. [3] to render shapes from 36 different viewpoints and output images of size 1024×1024 with corresponding intrinsic and extrinsic parameters. By projecting deformed boxes onto the image plane, we get partial textures from the rendered images for each deformed box. Both the ground truth texture and incomplete texture for a single part are of size 256×192.

Implementation details. When training the texture encoding network, we set the codebook size to 1024×256 and the discrete code is of size 24×32 and flattened to be a sequence of length 768. For the texture completion transformer network, we set λ_1 to 0.1 and λ_2 to 0.1 in all our experiments. For the texture encoding network, we train for 20 epochs using Adam [69] with learning rate 4.5×10^{-5} and 10 images per batch. The transformer network is trained for 30 epochs and also optimized using Adam, the batch size is set to 2 and the learning rate is set to 9×10^{-6}. We train the network on four V100 GPUs and the whole training process takes about a week for a typical category like table.

Evaluation metrics. To evaluate the quality of the reconstructed textured meshes, we render the meshes into the image space and compare the rendered results by different methods with the corresponding ground truth. For this comparison, we adopt three well-known metrics: structure similarity image metric (SSIM) [70], peak signal-to-noise ratio (PSNR), and Learned Perceptual Image Patch Similarity (LPIPS) [67]. For texture completion evaluation, we compare the inpainted texture images with the corresponding complete images under these three metrics. In addition, we also compare the mesh geometries in the 3D object space using the symmetric Chamfer Distance (CD).

Baselines. For the single-view textured mesh reconstruction task, we compare our method with DIB-R [13], Unicorn [52] (mesh-based), PixelNeRF [71] (implicit representation based), and DVR [12] (implicit representation based). In addition, to prove the effectiveness of the proposed texture completion network for single-view reconstruction, we replace it with current state-of-the-art image completion methods, Image Completion Transformer (ICT) [18] and CTSDG [19] to form two other baselines denoted as ICT and CTSDG.

4.1 Reconstruction results and comparisons

We show single-view textured mesh reconstructions of different object categories and compare them with state-of-the-art methods including DIB-R [13], Unicorn [52], PixelNeRF (PN) [71], DVR [12], TM-Net [17], as well as two recent image completion baselines, CTSDG [19] and ICT [18], which also reconstruct shape geometry via retrieval and assembly from shapes parts. From the qualitative comparisons shown in Fig. 5, it can be seen that the results of DIB-R [13] and Unicorn [52] may have wrong shape topology compared to the inputs as its geometry/topology is constrained by the template deformation. The results of PN [71] have correct topology but their textures are blurry since it fails to accurately reconstruct the density values in its volumetric representation from a single image. DVR [12] can reconstruct correct topology and is able to extract an explicit surface, but it fails to reconstruct the texture details due to its sparse sampling during training to reduce computational cost. Two other image completion baselines ICT [18] and CTSDG [19] do not take the relationship between different parts into account and may fail to produce plausible complete texture images or are unable to reach texture consistency between different parts. In comparison, our method, STAR-TM, is able to recover the shape geometry more faithfully with higher-quality texture.

We also compare geometry reconstruction results with the retrieval and deformation approach [35] in Fig. 6. Uy et al. [35] perform shape-level retrieval in its latent space and later deform the retrieved shape at the part level to adjust its parts' locations and scales to reconstruct the mesh from the input image. However, its shape-level retrieval may output a very different shape from the input image.

Reconstruction quality of the textured meshes is first evaluated by comparing multi-view rendering results to the ground truth renderings using the three image quality metrics SSIM, PSNR, and LPIPS. The comparison results are shown in Table 2. Overall our method outperforms the other methods in all three metrics and over all four object categories. In addition, Table 3 compares the methods

Table 2
Quantitative comparisons with DIB-R [13], PixelNeRF (PN) [71], Unicorn [52], DVR [12], CTSDG [19], and ICT [18] on SSIM, PSNR, and LPIPS metrics.

Metrics	Category	Methods							
		DIB-R	PN	Unicorn	DVR	TM-Net	CTSDG	ICT	Ours
SSIM ↑	Car	0.701	**0.852**	0.726	0.681	0.682	0.731	0.739	0.831
	Chair	0.696	0.854	0.745	0.721	0.675	0.860	0.847	**0.886**
	Plane	0.884	0.915	0.854	0.912	0.881	0.924	0.921	**0.934**
	Table	0.787	0.918	0.793	0.897	0.791	0.899	0.905	**0.926**
PSNR ↑	Car	21.811	23.691	22.616	20.095	20.303	22.212	22.234	**24.216**
	Chair	20.512	23.843	22.491	22.144	19.821	24.281	23.466	**25.348**
	Plane	24.329	28.562	23.868	28.235	24.705	28.887	28.672	**29.196**
	Table	22.736	28.584	22.974	26.838	23.013	27.719	27.925	**28.956**
LPIPS ↓	Car	0.292	0.182	0.264	0.298	0.294	0.247	0.239	**0.177**
	Chair	0.295	0.176	0.256	0.260	0.321	0.174	0.188	**0.154**
	Plane	0.157	0.143	0.151	0.144	0.164	0.139	0.141	**0.127**
	Table	0.225	0.141	0.187	0.151	0.211	0.149	0.147	**0.137**

Table 3
Quantitative comparison of geometric reconstruction quality. Note that Uy et al. [35] do not contain the car or plane category.

Metrics	Methods	Category			
		Car	Chair	Plane	Table
CD ↓	Pixel2Mesh [5]	0.273	0.340	0.198	0.383
	DIB-R [13]	0.362	0.503	0.386	0.569
	Unicorn [52]	0.268	0.328	0.245	0.344
	Uy et al. [35]	—	0.589	—	0.654
	DVR [12]	0.311	0.405	0.297	0.428
	TM-Net [17]	0.542	0.656	0.431	0.745
	Ours	**0.258**	**0.316**	**0.178**	**0.312**

in terms of geometric quality of the mesh reconstructions, using CD. Again, our method comes on top.

Next, we evaluate this method on the texture completion task. Two comparison methods ICT [18] and CTSDG [19] are re-trained on the same texture datasets. We show the qualitative comparison results of texture completion with these two methods in Fig. 7. It can be found that ICT [18] and CTSDG [19] fail to produce texture details (the first row) or output plausible images (the second row) while our method completes the textures with high quality. We also report in SSIM, PSNR, and LPIPS metrics by comparing inpainted textures produced by different methods with corresponding ground truth in Table 4. The results show that STAR-TM outperforms the two baselines.

Finally, we show in Fig. 8 reconstruction results from images in the wild, where the backgrounds are removed using U2Net [72] and the segmented object in the image is then scaled and translated to the center of an input image of size 1024×1024. Additional reconstruction and texture completion results can be found in the supplementary material.

4.2 Ablation studies

In this subsection, we conduct experiments to validate some of our design choices and present ablation studies on several important modules in our reconstruction pipeline.

Structured mesh alignment. First, we optimize shapes' transformations globally and locally to make the projection

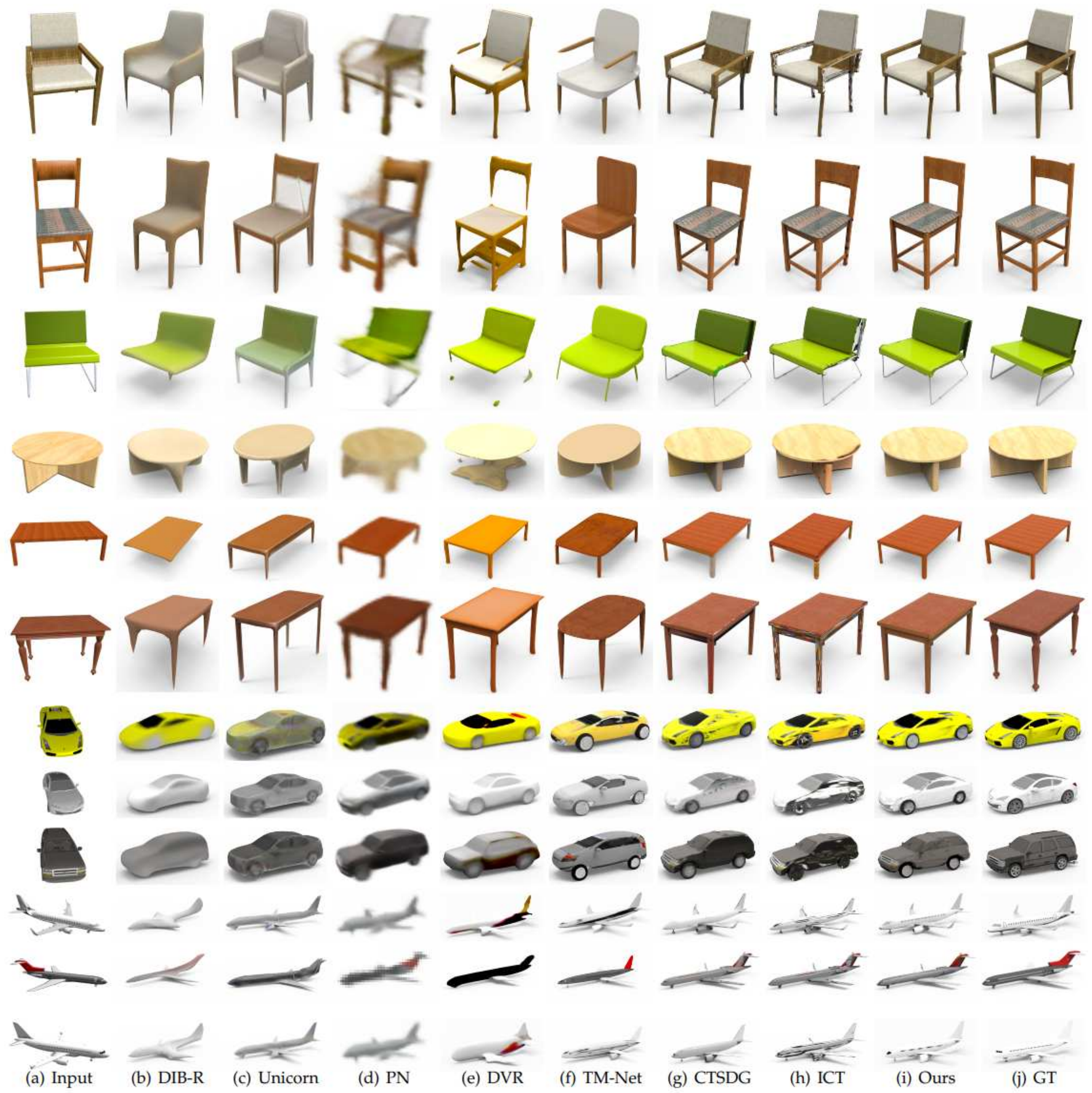


Figure 5. Qualitative comparison between textured shape reconstruction by DIB-R, Unicorn, PixelNeRF (PN), DVR, TM-Net, CTSDG, ICT, and our method STAR-TM. We show reconstructed textured meshes from a novel viewpoint, for examples from all four object categories. Note that PN outputs 2D renderings from different viewpoints instead of reconstructing explicit geometry, while CTSDG and ICT use the same geometry as our method. More reconstruction results can be found in the supplementary material.

of the assembled shape better align with the input image. Fig. 9 shows the qualitative comparison between with and without transformation optimization. We also perform quantitative comparisons in Table 5. It indicates that the structured mesh alignment can alleviate wrong unprojection from the input image to the reconstructed mesh, leading to better textures on reconstructed textured meshes and superior quantitative results.

Intrinsic image decomposition. To reduce lighting effects from the environment, we pass the input image through an intrinsic image decomposition network [66]. This network aims to remove lighting effects from the input image and extract its albedo component. We show albedo maps of both synthetic images and realistic images predicted by the intrinsic image decomposition network in Fig. 10. It can be seen that the network can well remove lighting effects from the image and preserve texture details under both synthetic and real-world settings.

Qualitative comparisons of with and without intrinsic image decomposition are shown in Fig. 11. We also evaluate how the intrinsic image decomposition network influences the reconstructed textured meshes by comparing rendered

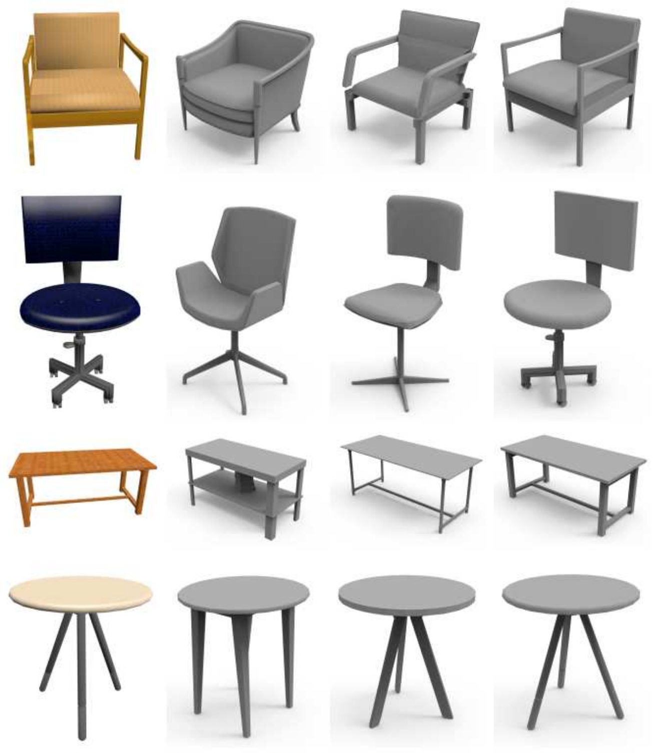


Figure 6. Geometry reconstruction results compared with the retrieval and deformation method Uy et al. [35].

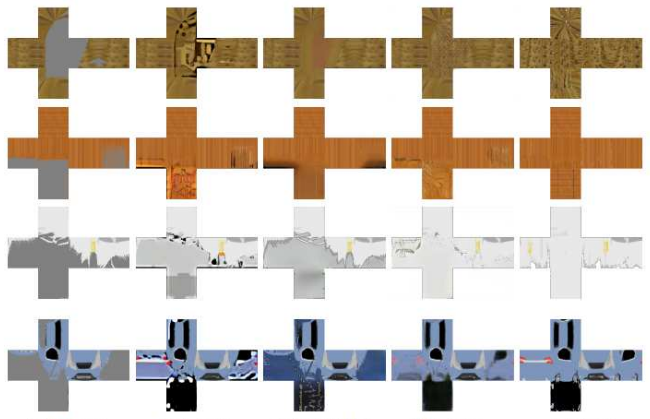


Figure 7. Texture completion results by ICT [18], CTSDG [19], and our method. Both ICT and CTSDG may fail or are unable to generate texture details while our method can produce meaningful results with texture details. We use gray color to denote unseen areas. More texture completion results can be found in the supplementary material.

Table 4
Quantitative comparison with CTSDG [19] and ICT [18] on texture completion task using SSIM, PSNR, LPIPS, and FID metrics.

Metrics	Methods	Category			
		Car	Chair	Plane	Table
SSIM ↑	CTSDG	0.697	0.702	0.681	0.741
	ICT	0.689	0.685	0.672	0.698
	Ours	**0.749**	**0.823**	**0.724**	**0.851**
PSNR ↑	CTSDG	21.759	22.073	20.453	22.438
	ICT	21.625	21.512	20.176	21.977
	Ours	**22.594**	**23.317**	**22.299**	**23.956**
LPIPS ↓	CTSDG	0.279	0.272	0.293	0.233
	ICT	0.281	0.291	0.296	0.278
	Ours	**0.215**	**0.206**	**0.237**	**0.182**
FID ↓	CTSDG	45.601	44.784	46.432	40.085
	ICT	48.962	48.917	49.641	45.051
	Ours	**39.125**	**38.172**	**40.572**	**37.729**

Table 5
Quantitative comparison between with structured mesh alignment (w/ alignment) and without structured mesh alignment(w/o alignment) using SSIM, PSNR, and LPIPS metrics on rendered images.

Metrics	Settings	Category			
		Car	Chair	Plane	Table
SSIM ↑	w/o alignment	0.815	0.853	0.913	0.908
	w/ alignment	**0.831**	**0.886**	**0.934**	**0.926**
PSNR ↑	w/o alignment	23.752	24.396	28.534	27.453
	w/ alignment	**24.216**	**25.348**	**29.196**	**28.956**
LPIPS ↓	w/o alignment	0.190	0.163	0.131	0.150
	w/ alignment	**0.177**	**0.154**	**0.127**	**0.137**

Table 6
Quantitative comparison between with intrinsic image decomposition network (w/ IID) and without intrinsic image decomposition network(w/o IID) using SSIM metric on rendered images of the table category.

Metric	w/ IID	w/o IID
SSIM on rendered images ↑	**0.926**	0.898

images of the reconstructed textured meshes with that of the ground truth mesh using SSIM metric on the table category. The result in Table 6 shows that the quality of reconstructed textured meshes decreases after removing the intrinsic image decomposition network.

Texture encoding network. Compared to conventional VAE (variational autoencoder), VQVAE (vector quantization-based autoencoder) can better capture texture details. Here, we show qualitative results of textures reconstructed by recent NVAE [73] and our vector quantization-based texture encoding network in Fig. 12. we also compare them using SSIM, PSNR, and LPIPS metrics on reconstructed textures in Table 7. Both qualitative and quantitative results show that the vector quantization-based autoencoder can reconstruct textures with higher fidelity compared to the variational autoencoder (VAE).

Loss functions. As mentioned in Section 3.2, textures reconstructed for different parts are generally not the same, hence LPIPS loss is adopted instead of L_1 or L_2 loss to ensure the consistency between different parts. We evaluate this design choice in Table 8 by comparing the SSIM metric between incomplete textures and complete textures produced by different loss functions. It can be seen that LPIPS loss indeed leads to the best image completion performance over L_1 and L_2.

Loss weights. We evaluate loss weights λ_1 and λ_2 from Equation 6 measured with the SSIM metric in the texture completion task on the chair category. For the hyperpa-

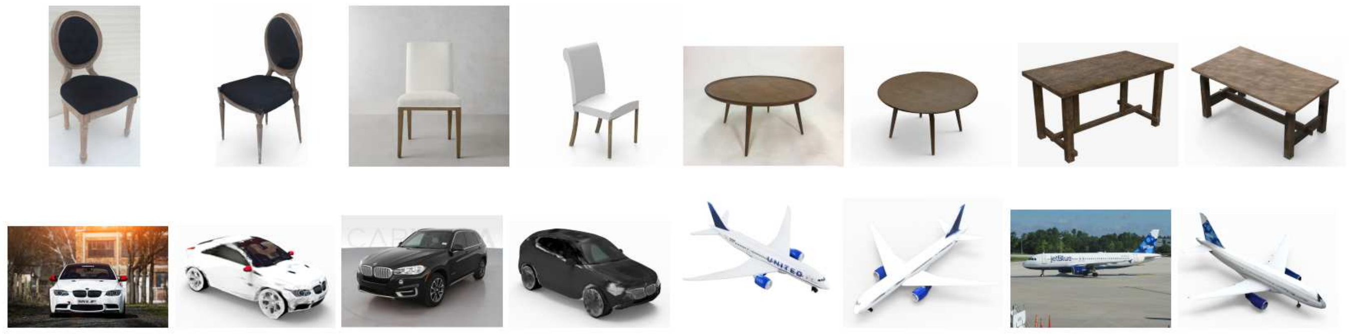


Figure 8. Textured mesh reconstruction from images in the wild by our method. The input image is shown on the left in each pair. We show more reconstruction results from real images in the supplementary material.

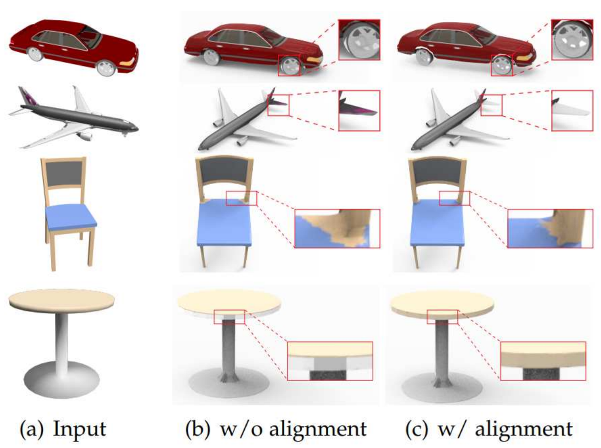


Figure 9. Ablation study of the geometry alignment module. The structured alignment module can encourage better texture unprojection and texture reconstruction. Holes are simulated using the alpha channel of the texture as in TM-Net [17].

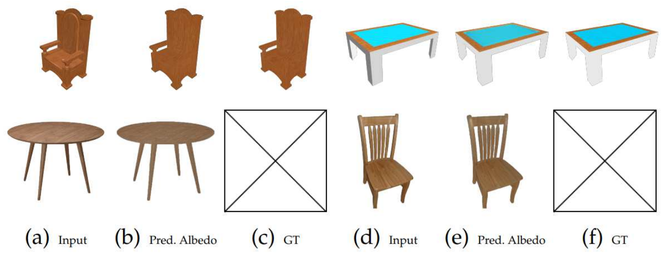


Figure 10. Predicted albedo maps by our intrinsic image decomposition network. Images in the first row are synthetic while in the second row input images are real internet photos so there are no ground truth albedo maps for them.

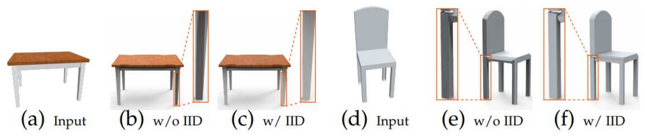


Figure 11. Qualitative comparisons of reconstructed textured meshes between with and without the intrinsic image decomposition network (IID). When IID is not applied, the texture of the reconstructed textured meshes may contain shadows or reflections.

rameter λ_1, we fix λ_2 to 0.1 and conduct an experiment on how its value influences the SSIM score between the predicted complete texture and the corresponding ground truth texture. As shown in Table 9, the completion network reaches its best performance when λ_1 is set to 0.1. Then we fixed λ_1 to 0.1 and performed an ablation study on Table 10. The results in Table 10 show that $\lambda_2 = 0.1$ gives the best performance.

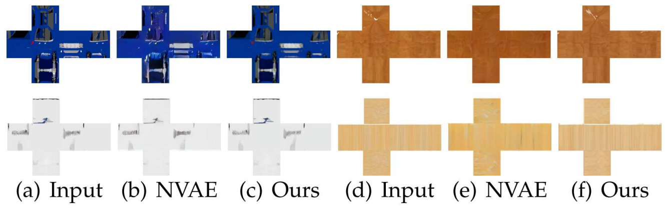


Figure 12. Texture reconstruction results by NVAE [73] and our method, which shows that our method can reconstruct texture details more faithfully.

Table 7
Quantitative comparison with NVAE [73] using SSIM, PSNR, and, LPIPS metrics on reconstructed textures.

Metrics	Methods	Category			
		Car	Chair	Plane	Table
SSIM ↑	NVAE	0.802	0.862	0.783	0.880
	Ours	**0.889**	**0.919**	**0.895**	**0.923**
PSNR ↑	NVAE	21.095	26.805	20.623	28.148
	Ours	**27.826**	**32.979**	**26.323**	**35.136**
LPIPS ↓	NVAE	0.131	0.114	0.165	0.072
	Ours	**0.093**	**0.057**	**0.109**	**0.049**

Table 8
Quantitative comparisons using different loss functions with the SSIM metric on textures of the table category.

Loss Function	L1	L2	LPIPS
SSIM on textures ↑	0.802	0.794	**0.851**

Table 9
Quantitative comparisons using different λ_1 with the SSIM metric on the chair category.

λ_1	0.05	0.1	0.5	1	2
SSIM on textures ↑	0.804	**0.823**	0.808	0.791	0.785

Retrieval-enhanced module. To make better use of the texture prior knowledge from the database, we introduce a retrieval-enhanced module into our image completion network. This module fuses features of the incomplete tex-

Table 10
Quantitative comparisons using different λ_2 using the SSIM metric on the chair category.

λ_2	0.05	0.1	0.5	1	2
SSIM on textures ↑	0.812	**0.823**	0.796	0.782	0.759

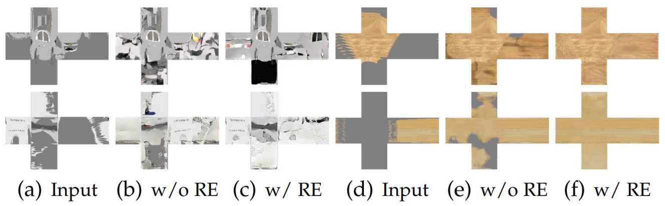


Figure 13. Comparison between with and without (w/o) the retrieval-enhanced (RE) module. We use gray color to represent unseen areas. It can be seen that completed textures may contain artifacts when RE is not applied.

Table 11
Quantitative comparison of with and without the retrieval-enhanced(RE) module using SSIM, PSNR, LPIPS, and FID metrics on completed textures.

Metrics	Settings	Category			
		Car	Chair	Plane	Table
SSIM ↑	w/o	0.657	0.764	0.682	0.789
	w/ RE	**0.749**	**0.823**	**0.724**	**0.851**
PSNR ↑	w/o	19.785	21.080	22.104	21.664
	w/ RE	**22.594**	**23.317**	**22.299**	**23.956**
LPIPS ↓	w/o	0.251	0.249	0.263	0.223
	w/ RE	**0.215**	**0.206**	**0.237**	**0.182**
FID ↓	w/o	42.292	42.611	31.849	41.240
	w/ RE	**39.125**	**38.172**	**40.572**	**37.729**

ture with features of the two nearest complete textures. We evaluate how this module influences texture completion in Fig. 13 and Table 11. It can be seen that completion results contain fewer artifacts and reach a higher image quality when we apply the retrieval-enhanced module.

Global consistency loss. While training the texture completion network, we take the relationship among different parts in the same semantic group into account and penalize the differences of perceptual features. We remove this loss from the completion network and complete each texture image individually for textures in the car's 'bodywork' group, chair's 'chair base' group, plane's 'power_system', and table's 'table base' group and compare the reconstructed textured shapes' quality with those produced by the pipeline with global consistency loss in Fig. 14. The results show that the parts in reconstructed shapes may have completely different textures if we do not utilize the relationship among different parts in the same semantic group. We also evaluate this loss function quantitatively by calculating SSIM, PSNR, and LPIPS metrics on rendered images of reconstructed textured shapes in Table 12. All metrics see a performance drop when the global consistency loss is abandoned.

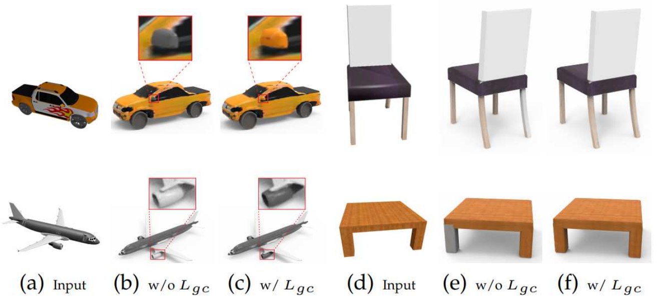


Figure 14. Comparisons of with and without global consistency loss L_{gc}. It can be seen that the reconstructed textured shapes may have inconsistent textures without the global consistency loss L_{gc}.

Table 12
Quantitative comparison of with and without global consistency loss L_{gc} using SSIM, PSNR, and LPIPS metrics on rendered images.

Metrics	Settings	Category			
		Car	Chair	Plane	Table
SSIM ↑	w/o L_{gc}	0.779	0.802	0.890	0.828
	w/ L_{gc}	**0.831**	**0.886**	**0.934**	**0.926**
PSNR ↑	w/o L_{gc}	19.418	21.797	24.833	22.153
	w/ L_{gc}	**24.216**	**25.348**	**29.196**	**28.956**
LPIPS ↓	w/o L_{gc}	0.252	0.185	0.174	0.186
	w/ L_{gc}	**0.177**	**0.154**	**0.127**	**0.137**

5 Conclusion, Discussion, and Future Work

In this paper, we propose a part-wise single-view textured mesh reconstruction method. We first reconstruct structured meshes by retrieving and assembling deformed boxes from the existing database given the single view reconstructed meshes. Then visible textures from the input image are unprojected to the structured mesh with the unified UV texture space, resulting in multiple incomplete texture images in the UV space for these deformed boxes. After that, we propose the retrieval-enhanced image completion network to complete the missing areas on the UV space for texture recovery. A global consistency loss is introduced in the texture completion process to enforce the reconstructed textured shapes to have consistent textures. This method is able to reconstruct meshes with high-quality textures from single-view images and outperforms the state-of-the-art methods in the quantitative and qualitative evaluations.

Our approach has the following limitations: Firstly, it is hard to apply our method to transparent objects like mirrors, see (a-b) in Fig. 15. Secondly, even though we introduced a structured alignment module, it can not ensure perfect unprojection, and misalignment between the shape and the input image may still occur, which can cause texture distortions or texture of a semantic part being unprojected to another part, see (c-d) in Fig. 15.

For future exploration, currently our structured mesh alignment cannot ensure perfect texture unprojection, it is possible to introduce a part-level understanding of the input image and apply non-rigid deformation on the recovered mesh to improve the performance. In addition, compared with previous convolutional models [12], [13], [51], [52] that predict geometry and texture from images with or without 3D supervision, our method can reconstruct textured meshes with higher quality but requires a segmented 3D shape dataset for textured mesh reconstruction. How to reduce the data requirement for textured mesh geometry

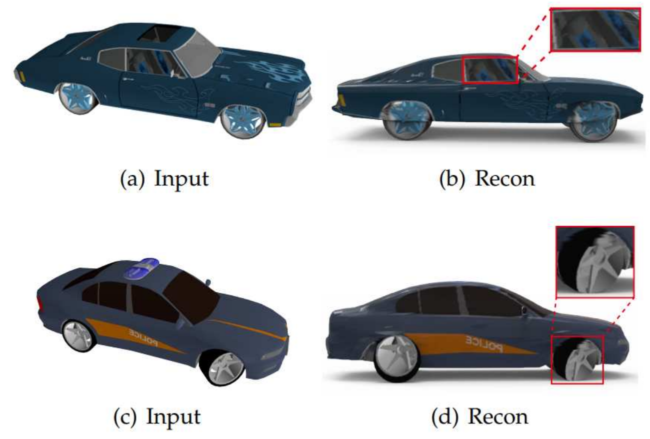


Figure 15. Some failure cases. (a-b): Our method is unable to handle input with transparent objects like windscreens. Textures that belong to objects inside the car are mapped to the car body. (c-d): Local misalignment may still occur, which can cause unexpected unporojected textures. Wheel textures are unprojected to the car body.

reconstruction is also a promising future direction. We also plan to adopt a hybrid geometry representation including both implicit fields with flexible topology and deformable templates which have good texture parameterization similar to NeuralParts [40] to jointly learn geometry and texture prediction with a differentiable neural renderer supervising rendered images of reconstructed textured meshes. Finally, We would like to build the connection between textured meshes with current neural radiance fields [74] to reconstruct shapes at the scene level.

Acknowledgments

This work was supported by grants from the National Natural Science Foundation of China (No. 62061136007), the Beijing Municipal Natural Science Foundation for Distinguished Young Scholars (No. JQ21013), the Royal Society Newton Advanced Fellowship (No. NAF\R2\192151), and the Youth Innovation Promotion Association CAS.

References

[1] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3D object reconstruction from a single image,” in *CVPR*, 2017, pp. 2463–2471.
[2] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, “Atlasnet: A papier-mâché approach to learning 3D surface generation,” *CoRR*, vol. abs/1802.05384, 2018.
[3] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3D-R2N2: A unified approach for single and multi-view 3D object reconstruction,” in *ECCV*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., vol. 9912, 2016, pp. 628–644.
[4] H. Xie, H. Yao, X. Sun, S. Zhou, and S. Zhang, “Pix2Vox: Context-aware 3D reconstruction from single and multi-view images,” in *ICCV*, 2019, pp. 2690–2698.
[5] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y. Jiang, “Pixel2Mesh: Generating 3D mesh models from single RGB images,” in *ECCV*, vol. 11215, 2018, pp. 55–71.
[6] Z. Chen and H. Zhang, “Learning implicit fields for generative shape modeling,” in *CVPR*, 2019, pp. 5939–5948.
[7] R. Wu, Y. Zhuang, K. Xu, H. Zhang, and B. Chen, “PQ-NET: A generative part seq2seq network for 3D shapes,” in *CVPR*, 2020, pp. 826–835.
[8] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann, “DISN: deep implicit surface network for high-quality single-view 3D reconstruction,” in *NeurIPS*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 490–500.
[9] J. Tang, X. Han, J. Pan, K. Jia, and X. Tong, “A skeleton-bridged deep learning approach for generating meshes of complex topologies from single rgb images,” in *CVPR*, 2019.
[10] M. Li and H. Zhang, “D2IM-Net: Learning detail disentangled implicit fields from single images,” in *CVPR*, 2021, pp. 10 246–10 255.
[11] Y. Sun, Z. Liu, Y. Wang, and S. E. Sarma, “Im2Avatar: Colorful 3D reconstruction from a single image,” *CoRR*, vol. abs/1804.06375, 2018.
[12] M. Niemeyer, L. M. Mescheder, M. Oechsle, and A. Geiger, “Differentiable volumetric rendering: Learning implicit 3D representations without 3d supervision,” in *CVPR*, 2020, pp. 3501–3512.
[13] W. Chen, H. Ling, J. Gao, E. J. Smith, J. Lehtinen, A. Jacobson, and S. Fidler, “Learning to predict 3d objects with an interpolation-based differentiable renderer,” in *NeurIPS*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 9605–9616.
[14] S. Liu, T. Li, W. Chen, and H. Li, “Soft rasterizer: A differentiable renderer for image-based 3D reasoning,” *International Conference on Computer Vision (ICCV)*, 2019.
[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *NeurIPS*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5998–6008.
[16] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, “Neural discrete representation learning,” in *NeurIPS*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 6306–6315.
[17] L. Gao, T. Wu, Y.-J. Yuan, M.-X. Lin, Y.-K. Lai, and H. Zhang, “TM-NET: Deep generative networks for textured meshes,” *ACM Trans. Graph.*, vol. 40, no. 6, pp. 263:1–263:15, 2021.
[18] Z. Wan, J. Zhang, D. Chen, and J. Liao, “High-fidelity pluralistic image completion with transformers,” *ICCV*, 2021.
[19] X. Guo, H. Yang, and D. Huang, “Image inpainting via conditional texture and structure dual generation,” in *ICCV*, 2021, pp. 14 134–14 143.
[20] L. Jiang, S. Shi, X. Qi, and J. Jia, “GAL: geometric adversarial loss for single-view 3D-object reconstruction,” in *ECCV*, vol. 11212, 2018, pp. 820–834.
[21] E. Insafutdinov and A. Dosovitskiy, “Unsupervised learning of shape and pose with differentiable point clouds,” in *NeurIPS*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2018, pp. 2807–2817.
[22] J. J. Park, P. Florence, J. Straub, R. A. Newcombe, and S. Lovegrove, “DeepSDF: Learning continuous signed distance functions for shape representation,” in *CVPR*, 2019, pp. 165–174.
[23] L. M. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy networks: Learning 3D reconstruction in function space,” in *CVPR*, 2019, pp. 4460–4470.
[24] K. Xu, H. Zheng, H. Zhang, D. Cohen-Or, L. Liu, and Y. Xiong, “Photo-inspired model-driven 3D object modeling,” *ACM Transactions on Graphics (Special Issue of SIGGRAPH)*, vol. 30, no. 4, pp. 80:1–80:10, 2011.
[25] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu, “Structure recovery by part assembly,” *ACM Trans. Graph.*, vol. 31, no. 6, pp. 180:1–180:11, 2012.
[26] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, “Category-specific object reconstruction from a single image,” in *CVPR*, 2015, pp. 1966–1974.
[27] G. Gkioxari, J. Johnson, and J. Malik, “Mesh R-CNN,” in *ICCV*, 2019, pp. 9784–9794.
[28] J. Gao, W. Chen, T. Xiang, C. F. Tsang, A. Jacobson, M. McGuire, and S. Fidler, “Learning deformable tetrahedral meshes for 3D reconstruction,” in *Advances In Neural Information Processing Systems*, 2020.
[29] Y. Zhang, W. Chen, H. Ling, J. Gao, Y. Zhang, A. Torralba, and S. Fidler, “Image GANs meet differentiable rendering for inverse graphics and interpretable 3D neural rendering,” in *International Conference on Learning Representations*, 2021.
[30] C. Niu, J. Li, and K. Xu, “Im2Struct: Recovering 3d shape structure from a single rgb image,” in *CVPR*, 2018.

[31] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. R. Zhang, and L. J. Guibas, "GRASS: generative recursive autoencoders for shape structures," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 52:1–52:14, 2017.

[32] J. Wang and Z. Fang, "GSIR: Generalizable 3D shape interpretation and reconstruction," in *European Conference on Computer Vision*, 2020, pp. 498–514.

[33] X. Zhang, Z. Zhang, C. Zhang, J. Tenenbaum, B. Freeman, and J. Wu, "Learning to reconstruct shapes from unseen classes," in *NeurIPS*, 2018, pp. 2263–2274.

[34] Z. Chen, A. Tagliasacchi, and H. Zhang, "BSP-NET: Generating compact meshes via binary space partitioning," in *CVPR*, 2020.

[35] M. A. Uy, V. G. Kim, M. Sung, N. Aigerman, S. Chaudhuri, and L. Guibas, "Joint learning of 3D shape retrieval and deformation," in *CVPR*, 2021.

[36] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in *CVPR*, 2019, pp. 909–918.

[37] K. Mo, P. Guerrero, L. Yi, H. Su, P. Wonka, N. J. Mitra, and L. J. Guibas, "Structurenet: hierarchical graph networks for 3d shape generation," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 242:1–242:19, 2019.

[38] D. Roberts, A. Danielyan, H. Chu, M. G. Fard, and D. A. Forsyth, "Lsd-structurenet: Modeling levels of structural detail in 3d part hierarchies," in *IEEE/CVF International Conference on Computer Vision, ICCV*. IEEE, 2021, pp. 5816–5825.

[39] S. Han, J. Gu, K. Mo, L. Yi, S. Hu, X. Chen, and H. Su, "Compositionally Generalizable 3D Structure Prediction," *arXiv preprint*, 2020.

[40] D. Paschalidou, A. Katharopoulos, A. Geiger, and S. Fidler, "Neural parts: Learning expressive 3D shape abstractions with invertible neural networks," in *CVPR*, 2021, pp. 3204–3215.

[41] M. Oechsle, L. M. Mescheder, M. Niemeyer, T. Strauss, and A. Geiger, "Texture fields: Learning texture representations in function space," in *ICCV*, 2019, pp. 4530–4539.

[42] J. Chibane, T. Alldieck, and G. Pons-Moll, "Implicit functions in feature space for 3D shape reconstruction and completion," in *CVPR*, 2020.

[43] J. Chibane and G. Pons-Moll, "Implicit feature networks for texture completion from partial 3D data," in *ECCV Workshops*, 2020.

[44] Y. Siddiqui, J. Thies, F. Ma, Q. Shan, M. Nießner, and A. Dai, "RetrievalFuse: Neural 3D scene reconstruction with a database," in *ICCV*, 2021, pp. 12 548–12 557.

[45] A. Dai, Y. Siddiqui, J. Thies, J. Valentin, and M. Nießner, "SPSG: self-supervised photometric scene generation from RGB-D scans," in *CVPR*, 2021, pp. 1747–1756.

[46] Q. Zhou and V. Koltun, "Color map optimization for 3D reconstruction with consumer depth cameras," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 155:1–155:10, 2014.

[47] J. Huang, J. Thies, A. Dai, A. Kundu, C. M. Jiang, L. J. Guibas, M. Nießner, and T. A. Funkhouser, "Adversarial texture optimization from RGB-D scans," in *CVPR*, 2020, pp. 1556–1565.

[48] D. Pavllo, G. Spinks, T. Hofmann, M. Moens, and A. Lucchi, "Convolutional generation of textured 3D meshes," in *NeurIPS*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[49] D. Pavllo, J. Kohler, T. Hofmann, and A. Lucchi, "Learning generative models of textured 3D meshes from real-world images," in *ICCV*, 2021.

[50] A. Bhattad, A. Dundar, G. Liu, A. Tao, and B. Catanzaro, "View generalization for single image textured 3d models," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2021, pp. 6081–6090.

[51] X. Li, S. Liu, K. Kim, S. De Mello, V. Jampani, M.-H. Yang, and J. Kautz, "Self-supervised single-view 3D reconstruction via semantic consistency," in *ECCV*, 2020.

[52] T. Monnier, M. Fisher, A. A. Efros, and M. Aubry, "Share With Thy Neighbors: Single-View Reconstruction by Cross-Instance Consistency," in *ECCV*, 2022.

[53] L. Gao, J. Yang, T. Wu, Y. Yuan, H. Fu, Y. Lai, and H. Zhang, "SDM-NET: deep generative network for structured deformable mesh," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 243:1–243:15, 2019.

[54] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: a randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009.

[55] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 107:1–107:14, 2017.

[56] G. Liu, K. J. Shih, T.-C. Wang, F. A. Reda, K. Sapra, Z. Yu, A. Tao, and B. Catanzaro, "Partial convolution based padding," in *arXiv preprint arXiv:1811.11718*, 2018.

[57] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *ECCV*, 2018.

[58] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[59] E. Grave, A. Joulin, and N. Usunier, "Improving neural language models with a continuous cache," in *International Conference on Learning Representations (ICLR)*, 2017.

[60] U. Khandelwal, O. Levy, D. Jurafsky, L. Zettlemoyer, and M. Lewis, "Generalization through Memorization: Nearest Neighbor Language Models," in *International Conference on Learning Representations (ICLR)*, 2020.

[61] D. Yogatama, C. de Masson d'Autume, and L. Kong, "Adaptive semiparametric language models," *Trans. Assoc. Comput. Linguistics*, vol. 9, pp. 362–373, 2021.

[62] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "REALM: retrieval-augmented language model pre-training," *CoRR*, vol. abs/2002.08909, 2020.

[63] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models for open domain question answering," in *EACL*, P. Merlo, J. Tiedemann, and R. Tsarfaty, Eds., 2021, pp. 874–880.

[64] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. van den Driessche, J. Lespiau, B. Damoc, A. Clark, D. de Las Casas, A. Guy, J. Menick, R. Ring, T. Hennigan, S. Huang, L. Maggiore, C. Jones, A. Cassirer, A. Brock, M. Paganini, G. Irving, O. Vinyals, S. Osindero, K. Simonyan, J. W. Rae, E. Elsen, and L. Sifre, "Improving language models by retrieving from trillions of tokens," *CoRR*, vol. abs/2112.04426, 2021.

[65] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

[66] J. Luo, Z. Huang, Y. Li, X. Zhou, G. Zhang, and H. Bao, "NIID-Net: Adapting surface normal knowledge for intrinsic image decomposition in indoor scenes," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 12, pp. 3434–3445, 2020.

[67] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[68] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3D model repository," *CoRR*, vol. abs/1512.03012, 2015.

[69] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

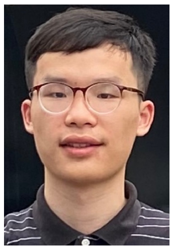
[70] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 13, no. 4, pp. 600–612, 2004.

[71] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "PixelNeRF: Neural radiance fields from one or few images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4578–4587.

[72] X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. Zaiane, and M. Jagersand, "U2-net: Going deeper with nested u-structure for salient object detection," vol. 106, 2020, p. 107404.

[73] A. Vahdat and J. Kautz, "NVAE: A deep hierarchical variational autoencoder," in *Neural Information Processing Systems (NeurIPS)*, 2020.

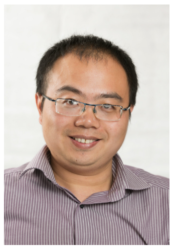
[74] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.



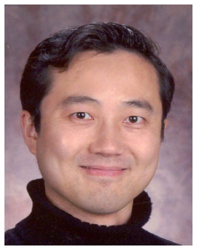
Tong Wu received his bachelor's degree in computer science from Huazhong University of Science and Technology in 2019. He is currently a PhD candidate at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer graphics and computer vision.

Lin Gao received the bachelor's degree in mathematics from Sichuan University and the PhD degree in computer science from Tsinghua University. He is currently an Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences. He has been awarded the Newton Advanced Fellowship from the Royal Society and the Asia Graphics Association young researcher award. His research interests include computer graphics and geometric processing.

Ling-Xiao Zhang received his Master of Engineering's degree in Computer Technology from Chinese Academy of Sciences in 2020. He is currently an assistant engineer at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer graphics and geometric processing.



Yu-Kun Lai received his bachelor's degree and PhD degree in computer science from Tsinghua University in 2003 and 2008, respectively. He is currently a Professor in the School of Computer Science & Informatics, Cardiff University. His research interests include computer graphics, geometry processing, image processing and computer vision. He is on the editorial boards of IEEE Transactions on Visualization and Computer Graphics and The Visual Computer.



Hao Zhang is a professor in the School of Computing Science at SFU and he directs the GrUVi(Graphics U Vision) Lab. He holds a Distinguished SFU Professorship for a five-year term(2020-25). Currently, He is also the Associate Director, Research and Industrial Relations, of his school. His research is in computer graphics and visual computing in general, with special interests in geometric modeling, shape analysis, 3D vision, geometric deep learning, as well as computational design and fabrication.