GAN-Based 3D Reconstruction from Single-View Images with Occlusion

Abstract

Image-based 3D reconstruction plays an important role in many real-world applications, including virtual reality, robot perception, and autonomous driving. 3D reconstruction can now be achieved using single-view images, but little work has been done in reconstruction from input with occlusions, which brings in more loss of information to this difficult 2D-to-3D ill-posed problem. In this project, we propose a GAN-based paradigm built upon the topology modification network (TMNet) baseline to improve 3D object reconstruction from single view images with occlusion. We use TMNet to generate 3D meshes, PyTorch3D to render images, and a patch-based discriminator to distinguish real and fake samples. Both quantitative and qualitative experiments show that our GAN-based approach achieves better results than the baselines on the chairs category in ShapeNet. The code of our project is at [1].

1. Introduction

Image-based 3D reconstruction refers to the process of using 2D images to infer the 3D geometry and structure of objects. It plays an important role in many applications, including virtual reality, robot perception, and autonomous driving. Traditional approaches, such as stereo-based and shape-from-silhouette methods, require sufficient input images of the same object taken by a well-calibrated camera, which can be unrealistic in many situations [7]. As the presence of 3D shape dataset enables shape encoding in deep neural networks, 3D reconstruction can now be achieved using single-view images. But little work has been done in reconstruction from input with occlusions, which brings in more loss of information to this difficult 2D-to-3D ill-posed problem.

Although existing methods on 3D mesh reconstruction from single view images have achieved impressive results on images with clean background, the reconstruction quality on images with occlusions is still very low. When part of the object is occluded, there is not enough visual cues to guide the reconstruction of the occluded part. Yet the problem of occlusion can easily occur in important applications. For example, in self-driving scenarios, the target car is often blocked by other cars or objects, leading to difficulties in reconstructing the objects and scenes for safe driving. As one learning-based attempt to tackle this, [11] modified [12] by adding a one-hot encoding of the object category to the input feature to the mesh decoder, so that the reconstruction network would make a reasonable "guess" of the occluded part based on the object category, which, however, may not be known in real-life applications.

In this project, we propose a GAN-based paradigm built upon TMNet baseline [12], to improve 3D object reconstruction from single view images with occlusions. The generator uses TMNet to generate 3D meshes from occluded 2D images. To allow the discriminator easily distinguish the ground truth and the generated meshes, we render the 3D meshes into 2D images, and feed them into the discriminator. We then utilize a patch-based discriminator to distinguish on both high and low frequencies components of features. The generator and discriminator are jointly optimized to push the generator to produce more realistic meshes without occlusion. Due to the availability of the dataset and compute resources, we only test our approach on one class (ShapeNet "chair").

The rest of this paper is organized as follows. In Section II, we review related works on 3D reconstruction, and generative adversarial networks. The proposed method is illustrated in Section III. Section IV discusses databases, evaluation metrics, results, and analysis. In the end, Section V summarizes the report and discusses future work.

2. Related Work

2.1. 3D Reconstruction

To reconstruct a 3D object, the object shape and geometry need to be first stored in some form of representation, and various forms have been in use, including voxel, point cloud, and mesh. Voxel representation utilizes a voxel grid with occupancy to represent a shape, but costs a lot of memory [3]. A sparser representation of 3D shape is point cloud,

[1] https://github.com/ZZWENG/CS231A_Project

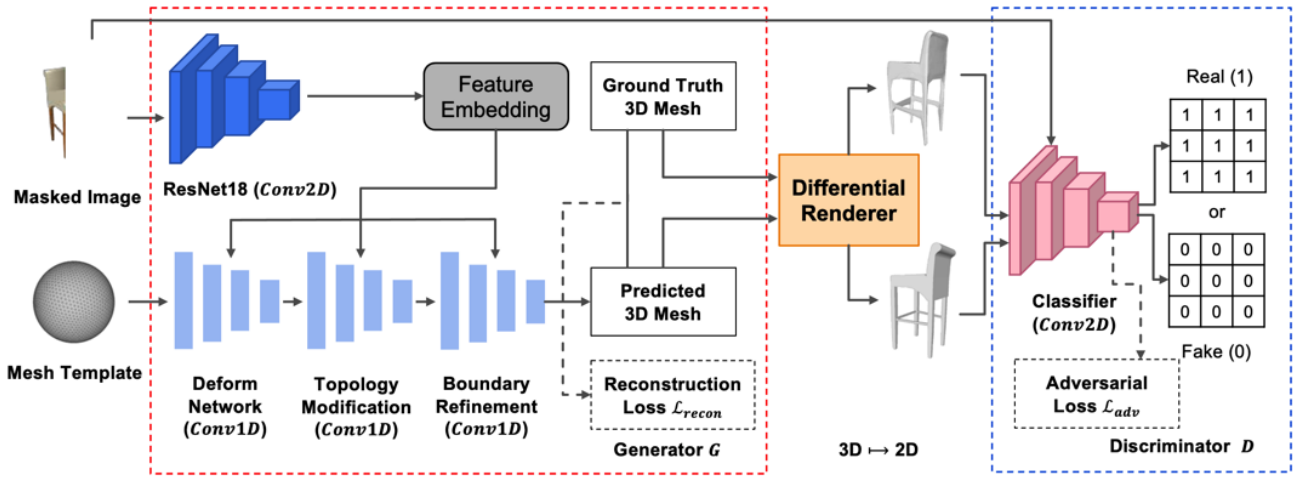

Figure 1. Overview of our method. Red-dashed box shows the TMNet-based generator. Blue-dashed box shows the discriminator. A differentiable render is added in between to produce 2D images from generated and ground truth meshes.

which presents a shape by a set of points in the 3D space [4]. Mesh, in particular, has gained attentions after the success of Pixel2Mesh [14, 15] for its capability to model shape details and its lightweight for easy deformation. Given a single-view image I, the aim of mesh-based reconstruction is to reconstruct the surface S of it. A mesh can be defined as $M = (V, E, T)$, where $V \in \mathbb{R}^3$ represents the mesh vertices, E represents edges, and T represents triangles. To achieve the reconstruction aim, a common method is to deform a mesh template M_0 until it is as close as the target mesh M. The aforementioned Pixel2Mesh method fails in some shape reconstruction cases as it cannot change the topology of the template mesh. But the topology modification network (TMNet) proposed in [12], which our project builds upon, has shown its superior performance in addressing the previoulsy mentioned issue by progressively alternating between mesh deformation and topology modification.

2.2. Generative Adversarial Network

Generative adversarial network (GAN) is able to generate images with desirable features and with strong similarity with target images or objects [5]. GAN consists of two components, which are a generator and a discriminator. As a generative model, GAN aims to learn the distribution p_g over data x with a noise variable z distributed by p_z. The generator $G(z, \theta_g)$ is parameterized by θ_g and the discriminator $D(x, \theta_d)$ is parameterized by θ_d. The discriminator D is optimized so that it can distinguish between real and fake samples and maximize the probability when providing either kind of the samples, while the generator aims to fool the discriminator by minimizing $log(1 - D(G(z)))$. The objective function can be derived by Kullback-Leiber divergence and Jensen-Shanon divergence, where the details can be found in [5]. However, the optimization of GANs is relatively harder than other networks due to mode collapse. To address such problem, Wasserstein GAN is proposed in [1] by introducing the Wasserstein loss. Another problem associated with the original GAN is the lack of control over the generated images, a problem that was tackled by the conditional GAN (cGAN) [10]. cGan uses the class embedding as the condition to make the generative results controllable. The work of Pix2Pix [8] uses 2D image as its condition to enforce the generated images as close as possible to some particular images, an approach that our project would like to similarly adopt.

3. Methodology

Our proposed methodology springs from the observation that objects within a certain category can exhibit large variance in shapes. Therefore, directly training a mesh generation network on masked images with supervised loss, such as as Chamfer Distance (CD), would increase the bias of the mesh generation network. We would like to introduce some guidance on the shape reconstruction process to achieve the de-occlusion effect. Hence, we propose to employ a GAN-based TMNet approach.

An overview of our proposed workflow is in Figure 1. First, the masked images are fed to the TMNet to generate 3D meshes. Then, given the output mesh and the ground truth mesh, we use a differentiable renderer to render the meshes into images. The rendered images from generated and ground truth meshes are sent into a patch discriminator to classify if they are real or fake images. At inference stage, only the generator will be used to reconstruct 3D meshes.

Details of the proposed network are described in Section 3.1 - 3.4.

3.1. Generator: Topology Modification Network

TMNet is an end-to-end learning pipeline that progressively modifies a template mesh to fit target surface from a given input image [12]. There are two major parts in the network, feature extraction and mesh deformation. For feature extraction, TMNet uses a residual network with 18 layers (ResNet18) to learn a 1024-dimension feature embedding of the input image and passes the embedding to the mesh deformation part. For mesh deformation, TMNet has three sub-modules to achieve the task: a deformation module, a topology modification module, and a boundary refinement module.

The mesh deformation module maps the vertices on template mesh to target surface while maintaining their connectivity. A total of 4 losses are used to supervise the deformation process: a CD loss $\mathcal{L}_{cd}$ to regress the location of mesh vertices, a normal loss $\mathcal{L}_{normal}$ to maintain the consistency of mesh surface normals, a smooth regularization loss $\mathcal{L}_{smooth}$ to enforce relative location among vertices neighbors, and an edge regularization loss $\mathcal{L}_{edge}$ to reduce outliers. CD uses a locally matching strategy to compute the l2-distance with the nearest neighbor for predicted samples S_{pd} and ground truth samples S_{gt} as

$$\begin{aligned}\mathcal{L}_{cd}(S_{pd}, S_{gt}) = &\sum_{p \in S_{pd}} \min_{q \in S_{gt}} ||p-q||_2^2 \\ &+ \sum_{q \in S_{gt}} \min_{p \in S_{pd}} ||p-q||_2^2.\end{aligned} \tag{1}$$

The topology modification module estimates an error between the ground truth using an error estimation network f_e, and then prunes the faces based on predicted error. Faces with large errors are discarded in this process. The error estimation network uses the quadratic loss to estimate error as

$$\mathcal{L}_{error} = \sum_{p \in S_{pd}} |f_e(p) - e_p)|^2, \tag{2}$$

where e_p is the ground truth error, and f_e is the error estimation network.

Finally, the boundary refinement module refines the open boundaries introduced by the previous face pruning step. The module ensures the smoothness of the output mesh based on the rule that boundary vertices are only allowed to move the 2D plane defined by the two boundary edges that intersect at the vertex. The boundary regularization loss is defined as

$$\mathcal{L}_{bound} = \sum_{p \in E} \Big|\Big| \sum_{q \in \mathcal{N}(p)} \frac{p-q}{||p-q||} \Big|\Big|, \tag{3}$$

where $p \in E$ is the set of vertices on the open boundaries, and $q \in \mathcal{N}(p)$ is the set of neighbouring vertices of p on the boundaries.

All neural networks used for these three modules consist of four 1D convolution layers, and subtle changes regarding their activation functions. Each module has a different objective as ground truth vertices, errors, and neighbour vertices respectively. The details of the modules configurations are given in Table 1. The optimal training of TMNet is to train the first two modules individually for two times separately, and then jointly train the three modules in the end.

Therefore, the total reconstruction loss $\mathcal{L}_{recon}$ of TMNet is the sum of the aforementioned losses with λ_i controlling their weights as

$$\begin{aligned}\mathcal{L}_{recon} = &\mathcal{L}_{cd} + \lambda_1 \mathcal{L}_{error} + \lambda_2 \mathcal{L}_{bound} \\ &+ \lambda_3 \mathcal{L}_{normal} + \lambda_4 \mathcal{L}_{smooth} + \lambda_5 \mathcal{L}_{edge}.\end{aligned} \tag{4}$$

In this project, TMNet acts as the generator denoted as $G(x, z)$, where x is the input image and z is a random vector.

3.2. Differentiable Renderer

The conditional GAN we adopted requires the input image stacked as a condition before sending it to the discriminator. To achieve this, we use a differentiable renderer to first render the 3D meshes into 2D images. Following the tutorial[2], we use Pytorch3D [13] to render the meshes. The renderer consists of a resterizer to project and rasterize the inputs, and a shader to texure, light and blend the image. In each iteration of the training process, the generator takes a masked image of a chair, which is a rendering of the chair model at a pre-defined view angle with occlusion. The renderer takes the output of the generator, which is a reconstructed mesh consisting of predicted vertex locations and faces, and renders the output image at the same view point. The ground truth mesh is also rendered at the same view point. For simplicity, we ignore the texture of the meshes during rendering.

3.3. Discriminator: Patch-based Classifier

The aim of the discriminator D is to classify the generated fake images (i.e. rendering of the reconstructed mesh) and ground truth real images (i.e. rendering of the ground truth mesh). While the reconstruction loss of generator sufficiently pushes for generation of low-frequency components, a patch-based discriminator or PatchGAN in [8] puts more attention on local patches and is able to recover high-frequency components by modeling the image as a Markov random field and assuming that the pixels are independently separated by more than one patch diameter.

Instead of discriminating real and fake images with a single Boolean value, the patch-based discriminator classifies

[2] https://github.com/facebookresearch/pytorch3d/blob/master/docs/tutorials/dataloaders_ShapeNetCore_R2N2.ipynb

Layer	Kernel	Stride, Padding	Output Size
Deformation Module			
Input			1027×2562
Conv1D_1	1	1, 0	1027×2562
BatchNorm_1			1027×2562
Conv1D_2	1	1, 0	513×2562
BatchNorm_2			513×2562
Conv1D_3	1	1, 0	256×2562
BatchNorm_3			256×2562
Conv1D_4	1	1, 0	3×2562
Tanh			3×2562
Error Estimate Network			
Input			1027×10000
Conv1D_1	1	1, 0	1027×10000
BatchNorm_1			1027×10000
Conv1D_2	1	1, 0	513×10000
BatchNorm_2			513×10000
Conv1D_3	1	1, 0	256×10000
BatchNorm_3			256×10000
Conv1D_4	1	1, 0	1×10000
Sigmoid			1×10000
Boundary Refinement Module			
Input			1027×119
Conv1D_1	1	1, 0	1027×119
BatchNorm_1			1027×119
Conv1D_2	1	1, 0	513×119
BatchNorm_2			513×119
Dropout_1			513×119
Conv1D_3	1	1, 0	256×119
BatchNorm_3			256×119
Dropout_2			256×119
Conv1D_4	1	1, 0	2×119
Tanh			2×119

Table 1. Configuration of TMNet modules. Conv1D stands for 1D convolutions; BatchNorm stands for 1D batch normalization operations; Tanh and Sigmoid are activation functions.

the result on a $m \times m$ patch. In detail, the true or false value will be padded to the same size as the output single channel feature map before the adversarial loss is calculated. The discriminator network is a 2D convolutional network. Details of the network used for the discriminator is shown in Table 2.

3.4. Loss Function

The overall loss $\mathcal{L}$ of the proposed GAN-based 3D reconstruction is as follows

$$\mathcal{L} = \lambda_a \mathcal{L}_{adv}(G, D) + \lambda_r \mathcal{L}_{recon}(G), \tag{5}$$

Layer	Kernel	Stride, Padding	Output Size
Input			$224 \times 224 \times 6$
In_Conv2D	1×1	1, 1	$224 \times 224 \times 8$
B1_Conv2D_1	3×3	1, 1	$224 \times 224 \times 16$
B1_Conv2D_2	3×3	1, 1	$224 \times 224 \times 16$
B1_MaxPool	2×2	2, 0	$112 \times 112 \times 16$
B2_Conv2D_1	3×3	1, 1	$112 \times 112 \times 32$
B2_Conv2D_2	3×3	1, 1	$112 \times 112 \times 32$
B2_BatchNorm			$112 \times 112 \times 32$
B2_MaxPool	2×2	2, 0	$56 \times 56 \times 32$
B3_Conv2D_1	3×3	1, 1	$56 \times 56 \times 64$
B3_Conv2D_2	3×3	1, 1	$56 \times 56 \times 64$
B3_BatchNorm			$56 \times 56 \times 64$
B3_MaxPool	2×2	2, 0	$28 \times 28 \times 64$
B4_Conv2D_1	3×3	1, 1	$28 \times 28 \times 128$
B4_Conv2D_2	3×3	1, 1	$28 \times 28 \times 128$
B4_BatchNorm			$28 \times 28 \times 128$
B4_MaxPool	2×2	2, 0	$14 \times 14 \times 128$
Out_Conv2D	3×3	1, 0	$14 \times 14 \times 1$

Table 2. Configuration of discriminator's network . B stands for blocks; Conv2D stands for 2D convolutions; MaxPool stands for 2D max pooling operations; BatchNorm stands for 2D batch normalization operations.

where $\mathcal{L}_a$ is an adversarial loss and $\mathcal{L}_{recon}$ is the reconstruction loss defined in Section 3.1. λ_a and λ_r are the weight coefficients for the adversarial and reconstruction parts.

The adversarial loss is defined as a binary cross entropy

$$\begin{aligned}\mathcal{L}_{adv}(G, D) = -\big[&\mathbb{E}_{x,y}[\log D(x, y)] + \\ &\mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))]\big],\end{aligned} \tag{6}$$

where y is the ground truth unmasked image. The randomness is implicitly represented by the dropout in the generator, rather than an explicit input noisy vector z. The optimization of the adversarial loss is a min-max game, in which the discriminator D tries to minimize it by classifying the real and fake images while the generator G tries to maximize it by generating close to "real" images to fool the discriminator.

4. Experiment

4.1. Dataset and Experiment Setup

We evaluate our model on the "chair" category in ShapeNet [2], a large-scale 3D object dataset. ShapeNet contains approximately 50,000 3D CAD models covering about 50 categories. The "chair" category is used by a lot of the previous single-view reconstruction methods such as [12]. It contains over 7000 chairs in the training set and over 1300 chairs in the test set. Similar to [12], we will

use the experiment set up in [6] and use the rendered images in [3]. Each 3D model has 24 rendered RGB images, each representing a different view angle. The model takes the rendered images as input. We will sample 10,000 points uniformly on each 3D shape surface as the ground truth.

To simulate occlusion in input images, we remove the pixel values from either left or right half of the images by setting the RGBA values to $(0, 0, 0, 0)$ for each pixel. Images are randomly selected to have their left or right half removed. Currently we have generated masked images for the ShapeNet "chair" class.

Our implementation is in Python. We use PyTorch and Pytorch3D deep learning frameworks. The TMNet implementation is based on [12]. We develop our models and run experiments on Google Cloud Platform.

4.2. Evaluation

We evaluate the proposed method both qualitatively and quantitatively.

Qualitative Result Demonstration. We render and visualize the reconstructed objects to demonstrate the performance, as discussed in Section 4.5.

Quantitative Result Evaluation Metric. We employ two widely-adopted evaluation metrics for 3D reconstruction, Chamfer distance (CD) and Earth Mover's distance (EMD) [9], which are the same as TMNet in [12]. These metrics require us to sample meshes into sample points. For the details of the calculation of CD, please refer to (1). In terms of EMD, it calculates the minimum distance between points in S_{pd} and bijected points from S_{gt} as

$$EMD(S_{gt}, S_{pd}) = \min_{\phi: S_{gt} \to S_{pd}} \sum_{p \in S_{gt}} ||p - \phi(p)||, \quad (7)$$

where $\phi(p)$ is the closest point in S_{pd} to p.

Following the quantitative evaluation protocol of [12], the evaluation is based on 2,048 ground truth points and 2,048 points uniformly sampled from the generated meshes. The CD is multiplied by 1,000 and the EMD is multiplied by 100.

4.3. Implementation Details

We train the models on Google Cloud Platform and Sherlock with single GPU. The training of a full TMNet model consists of 4 steps: pretraining, train subnet1, train subnet 2, and train subnet 3. Pretraining takes about 4 hours. The training of subnet 1, 2 and 3 takes 6 hours each. In total, training a TMNet from scratch takes about 22 hours. To train the GAN version, we fine-tune the discriminator for 120 epochs where each epoch takes about half an hour, bringing the total training time for the GAN extension to about 60 hours.

For all experiments, we use the following hyperparameters:

- Number of epochs for TMNet: 420 epochs each for pretrain and subnet 1; 120 epochs each for subnet 2 and 3.
- Number of epochs for GAN: We fine-tune subnet3 with adversarial loss for additional 120 epochs.
- Batch size: 32.
- Learning rate: 0.001. Decay learning rate after 300 and 400 epochs when training subnet1.
- Discriminator learning rate: 0.001.

4.4. Quantitative Evaluation

To showcase the effectiveness of our approach as compared to the vanilla TMNet, we conducted 4 experiments. We report the quantitative scores for each experiment in Table 3.

For the first and second experiment, we train TMNet on unmasked images and evaluated on unmasked and masked images respectively. We see that when tested on masked images, the CD and EMD scores drop significantly, as expected.

In experiment 3, we train and evaluated the model on masked images. For experiment 4, we further fine-tune subnet3 with the adversarial loss as described previously. Note that during the adversarial training, the ground truth information about the meshes is not used at all. We rely on the discriminator to refine the output of the generator. Hence, we choose to start training from the end model (i.e. all the modules in TMNet) such that the generator has a decent initialization. From there, the discriminator can pick up the learning instantly.

We show in Table 3 that our GAN-based approach achieves the best CD and EMD scores when evaluated on masked images.

For reference, we also did an experiment where we train the discriminator using the generator trained up to subnet2, and then we train subnet3 using solely the generator loss as in GAN. Evaluating the trained model on masked images, this version achieved 13.387 CD and 4.853 EMD, which is worse than the best reported numbers in Table 3.

Another thing to note is that both CD and EMD are calculated in terms of the locations of the 3D vertices and do not entail any evaluation of the quality of the reconstructed faces. We will showcase the quality of the generated faces in qualitative evaluation by comparing the mesh output from different experiments.

4.5. Qualitative Evaluation

Both CD and EMD are quantitative measures of the quality of the reconstructed vertices. We turn to the qualitative results for the quality of the reconstructed faces. We show

Train on unmasked images				
Experi. No.	**Model**	**Test images**	**CD**	**EMD**
1	TMNet	unmasked	6.201	4.320
2	TMNet	masked	26.860	9.336

Train on masked images				
Experi. No.	**Model**	**Test images**	**CD**	**EMD**
3	TMNet	masked	8.300	4.570
4	Ours	masked	**8.284**	**4.521**

Table 3. Quantitative results on *chairs* category. Best results when testing on masked images are in **bold**. For both metrics, lower values indicate better vertex reconstruction quality.

qualitative reconstruction results of a few chair examples in Figure 2 and Figure 3.

In Figure 2 we show the output of 4 experiments. The setup of experiment 1 gives quite reasonable reconstruction result. However, when we take a pretrained model (trained on unmasked images) and evaluate on masked images as in experiment 2, the reconstruction gets much worse qualitatively. In experiment 3, we train on masked images. The setup of experiment 3 is similar to utilizing data augmentation during training. The training loss is still the same as in experiment 1 and 2. We see that the reconstruction is much better as compared to experiment 2. In our approach (experiment 4), the output is slightly better than the output of experiment 3, as we observe smoother edges of the chair (e.g. the bottom edge of the second chair).

In conclusion, we see that our approach outperforms the naïve baseline in experiment 2. When compared to experiment 3, the difference is subtle, but we do see smoother edges of the chair with our GAN-based approach.

5. Conclusion and Future Work

Single-view reconstruction from RGB images with occlusion is a very challenging task. Directly applying state-of-the-art models (e.g. TMNet) on images with occlusion gives bad reconstruction results. In this project, we proposed a GAN-based approach for single-view reconstruction for masked images. Through quantitative experiments and qualitative evaluation, we showed that our approach can be effective in guiding the reconstruction of the occluded part.

For future work, we plan to experiment with more realistic occlusion scenarios (e.g. randomly masking out a small part of the image, occlusion in real-world images). For instance, in reality, a chair is often blocked by part of the table such that its lower half is missing. Simulating such occlusion and test on real-world images could be an interesting direction to explore.

In addition, we will also extend the method to cover more categories than just chairs. Since there is not much shape variation within the chair category, our improvement over experiment 3 (train on masked images with supervised loss) is marginal. However, since the generator trained with adversarial loss is more generalizable, we expect that our approach could make significant improvement when we consider reconstruction of multiple categories.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A unified approach for single and multi-view 3d object reconstruction. In *Proceedings of the European Conference on Computer Vision*, pages 628–644. Springer, 2016.

[4] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 605–613, 2017.

[5] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[6] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation, 2018.

[7] Xianfeng Han, Hamid Laga, and Mohammed Bennamoun. Image-based 3d object reconstruction: State-of-the-art and trends in the deep learning era. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[8] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, et al. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[9] Andrey Kurenkov, Jingwei Ji, Animesh Garg, Viraj Mehta, JunYoung Gwak, Christopher Choy, and Silvio Savarese. Deformnet: Free-form deformation network for 3d shape reconstruction from a single image. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 858–866. IEEE, 2018.

[10] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[11] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, et al. Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 55–64, 2020.

[12] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single rgb images

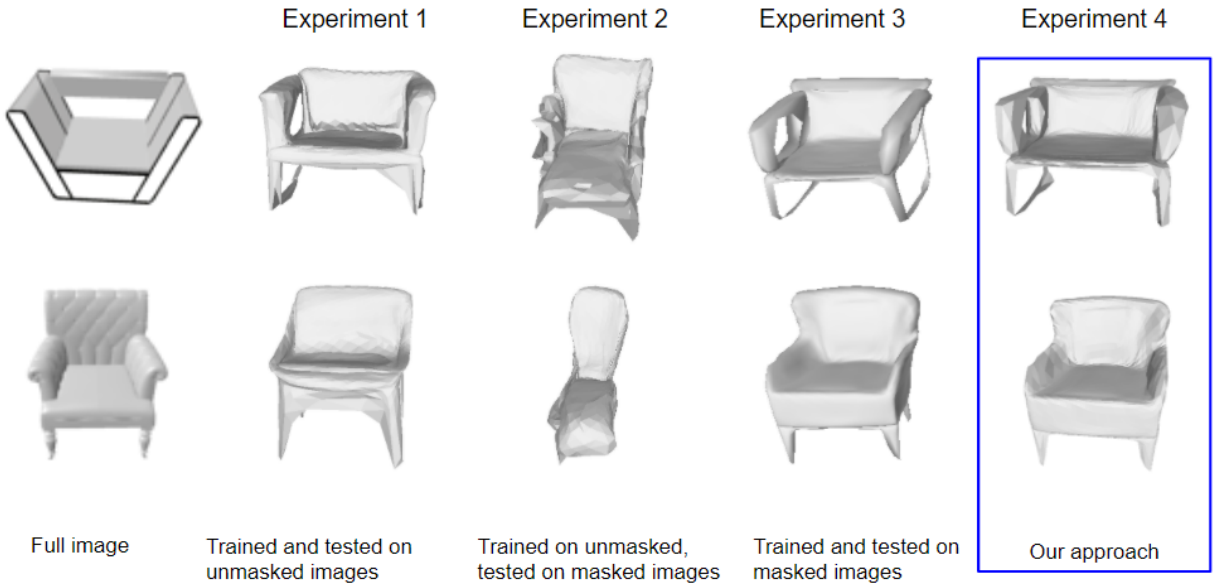

Figure 2. Qualitative results. The four experiments correspond to the ones listed in Table 3.

Figure 3. Additional qualitative results. From top to bottom: full image, result of experiment 2, result of experiment 4 (Ours).

via topology modification networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9964–9973, 2019.

[13] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

[14] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision*, pages 52–67, 2018.

[15] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Hang Yu, Wei Liu, Xiangyang Xue, and Yu-Gang Jiang. Pixel2mesh: 3d mesh model generation via image guided deformation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.