Learning to Generate Dense Point Clouds with Textures on Multiple Categories

Tao Hu, Geng Lin, Zhizhong Han, Matthias Zwicker
University of Maryland, College Park
taohu@cs.umd.edu, geng@cs.umd.edu, h312h@umd.edu, zwicker@cs.umd.edu

This supplementary material provides additional experimental results and technical details for the main paper.

A. Optimization

Our pipeline implements a two-stage reconstruction approach, including 2D-3D transformation by a 2D-3D net, and view completion by either the Multi-view Depth Completion Net (MDCN) or the Multi-view Texture-Depth Completion Net (MTDCN). In Figure 1 we take MDCN as an example. We implement all of our networks in PyTorch 1.2.0.

Training 2D-3D net. We use Minibatch SGD and the Adam optimizer [9] to train 2D-3D net, where the momentum parameters are $\beta_1 = 0.5$, $\beta_2 = 0.999$. We train 2D-3D net for 200 epochs with an initial learning rate of 0.0009, and the learning rate linearly decays after 100 epochs. The batch size is 64.

We train our networks in two stages, as shown in Figure 1. We denote the training input data (single RGB images) of 2D-3D net as X_1, and test data as T_1. The training input data (object coordinate images) of MDCN is X_2, and test data is T_2, which correspond to the output of 2D-3D net given X_1 or T_1 as input respectively.

Let us denote $f(C)$ as the average error of C, like the average L_1 distance to ground truth. In our case, C is a set of object coordinate images. In general, since X_1 is available during training while T_1 is novel input, $f(X_2)$ is smaller than $f(T_2)$ by a large margin, that is, the relative difference $\epsilon = |f(X_2) - f(T_2)|/f(X_2)$ is large. For example, the training input X_2 is often less noisy than the test data T_2, which results in a large ϵ and limits the generalizability of MDCN. In contrast, a smaller ϵ leads to better generalizability.

To decrease ϵ, we train two 2D-3D networks separately, a 'good' net (G) and a relatively 'bad' (B) one such that B's performance on the training set, $f(B(X_1))$, is similar to G's performance on the test set, $f(G(T_1))$. In this way, $X_2 = B(X_1)$ will look similar to $T_2 = G(T_1)$, which improves the generalizability of MDCN. We control the number of training samples to train the two networks. G is trained with 8 random views per 3D object, while B is trained with only 1 for each. Note that we trained our net on both a category-specific task and a multiple-category task, hence we obtained 4 networks in total, 2 for each task.

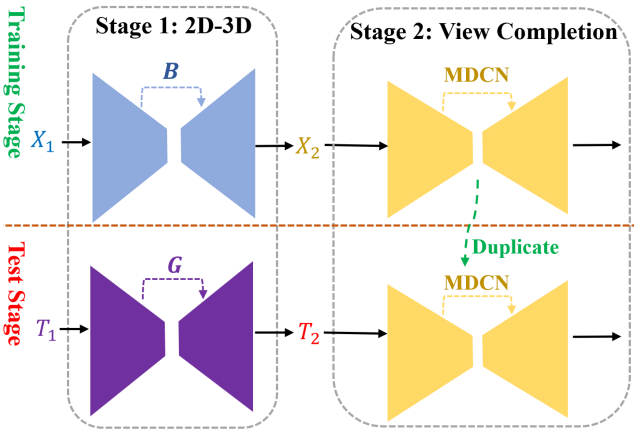

Figure 1: Two-stage Training.

Training multi-view completion net. Different from the training of 2D-3D net, we only need to train one view completion net for both MTDCN and MDCN.

Since MTDCN and MDCN have a similar network structure as 2D-3D net, we use the same optimizer setup. For MDCN, the initial learning rate is 0.0012, and the batch size is 128. For MTDCN, the learning rate is 0.0008, and the batch size is 48. Note that in our network, we concatenate all the 8 views of a 3D object into one image whose size is 2048×256, so that we do not need to store a shape memory or shape descriptor for each object mentioned in [7], because they are generated on the fly on one single GPU, which makes the pipeline more efficient than [7].

B. More Experimental Results and Details

Comparisons with AtlasNet [6] on single view reconstruction. AtlasNet is a mesh based reconstruction method, which learns to generate the surface of 3D shapes. We follow the same setting as AtlasNet and provide the reconstruction results in Table 1, where the results of AtlasNet-25 are reported in [6] and CD is calculated on 1024 points. A qualitative comparison on single view reconstruction is shown in Figure 2.

Model	mean	air.	ben.	cab.	car	cha.	dis.	lam.	lou.	rif.	sof.	tab.	tel.	ves.
AtlasNet	5.11	**2.54**	**3.91**	**5.39**	4.18	6.77	6.71	7.24	8.18	**1.63**	6.76	4.35	3.91	4.91
Ours-S_{d+t}	**4.44**	4.11	3.99	5.67	**4.12**	**4.07**	**4.74**	**6.26**	**6.72**	4.73	**4.64**	**4.08**	**3.76**	**4.15**

Table 1: Comparisons with AtlasNet on single view reconstruction on ShapeNet. The CD reported is multiplied by 100.

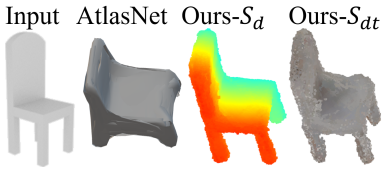

Figure 2: Ours v.s. AtlasNet [6].

Qualitative comparisons with RenGe [17] on single view reconstruction. Besides the quantitative comparisons with RenGe in Table 9 on the manuscript, we also provide qualitative reconstructions on real images of Pix3D dataset [13] in Figure 3, where we show the reconstruction results of objects from three unseen categories, desks, beds and tables. For each real image from Pix3D, we mask the background and extract the objects of interest using the provided information by Pix3D for the reconstructions of both our method and GenRe. Note that the networks of ours and GenRe were trained on ShapeNet [3] cars, chairs, and airplanes.

Qualitative results of novel car objects from ShapeNet. Among the 13 seen categories from ShapeNet [3], car objects generally have more distinct textures. Here we show more qualitative completions of cars in Figure 4, and compare against 3D-R2N2 [4], PSGN [5] and 3D-LMNet [12]. We can generate denser point clouds with reasonable textures given inputs with different colors or shapes. It should be mentioned that for the first car object, our method S_{d+t} generates the correct shape, while other methods fail.

Chamfer Distance (CD) on dense point clouds. We also report the results of taking dense point clouds as ground truth in Table 2. Each ground truth point cloud has 40K points. Our method is the best on both dense and sparse ground truth point clouds (shown in Table 5 of the main paper), compared with existing methods (e.g., PSGN [5], 3D-LMNet [12], OptMVS [14]).

More details about the experiments of 6D Pose estimation. In Section 4.4 of the manuscript, we evaluate the accuracy of 6D pose estimation by calculating ADD-S [15]. ADD-S is an ambiguity-invariant pose error metric for 6D pose estimation. Given the estimated pose $[\tilde{R}|\tilde{T}]$ and ground truth pose $[R|T]$, ADD-S calculates the mean distance from each 3D model point transformed by $[\tilde{R}|\tilde{T}]$ to its closest neighbor on the target model transformed by $[R|T]$ [11, 15]. For object-centered reconstruction methods, we extract the estimated pose $[\tilde{R}|\tilde{T}]$ from the generated shapes via ICP alignment [2] with the ground truth. We then use it to transform the ground truth point cloud (P) by $[\tilde{R}|\tilde{T}]$, which yields a shape (P'). We calculate the ADD-S between P and P' on 1024 points, and scale it by a factor 100.

C. Dataset Processing

We describe how we prepare our data for network training and testing. The dataset we use is ShapeNet [3]. For each model, we render 8 RGB images at random viewpoints as input, and 8 depth/texture image pairs as ground truth in MDCN training. All images have size 256×256.

Scene setup. The camera has a fixed distance, 2.0, to the object center, which coincides with the world origin. It always looks at the origin, and has a fixed up vector $(0, 1, 0)$. What vary among the viewpoints is the location of the camera.

Rendering of RGB images. We use the Mitsuba renderer [8] to render all RGB images.

Rendering of depth images. Unlike previous works [16, 10] which use a graphics engine like Blender to render depth images, ours utilizes a projection method that is similar to the Joint Projection introduced in Section 3.2 of the main paper. However, different from Joint Projection which projects partial shapes, the ground truth shape for each object is denser, which has 100K points sampled from mesh models, and the depth buffer is increased from 5×5 to 50×50 to alleviate collision effects. Because our projection method is mainly based on matrix calculation, it renders depth maps faster than ray tracing of graphics engines.

Rendering of object coordinate images. Following the depth projection pipeline, we also render object coordinate images as the ground truth to train the 2D-3D nets. First, since in our method, the object coordinate images represent the observed parts of objects, we render a depth map from the viewpoint of the input RGB image by projection method. Next, we back-project the depth map into a partial shape $\{P_i = [x_i, y_i, z_i]\}$, which can be represented by an object coordinate image, where RGB values are $[x_i, y_i, z_i]$. It should be mentioned that the input RGB image, the intermediate representation of depth map, and the object coordinate image has the same pose, which means they are aligned in pixel level.

Fusion of depth maps. We fuse the 8 completed depth maps into a point cloud with the Joint Fusion techniques introduced in the main paper. We also use voting algorithm to remove outliers as mentioned in [7]. We reproject each point of one view into the other 7 views, and if this point falls on the shape of other views, one vote will

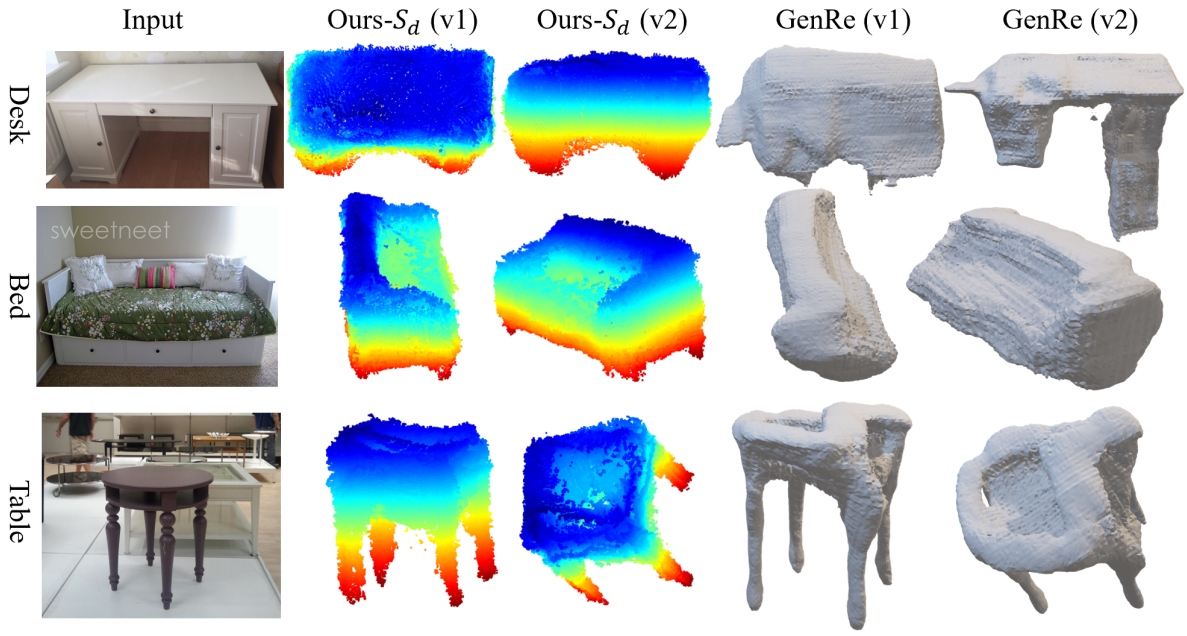

Figure 3: Qualitative comparisons with RenGe [17] on single view reconstruction on Pix3D dataset.

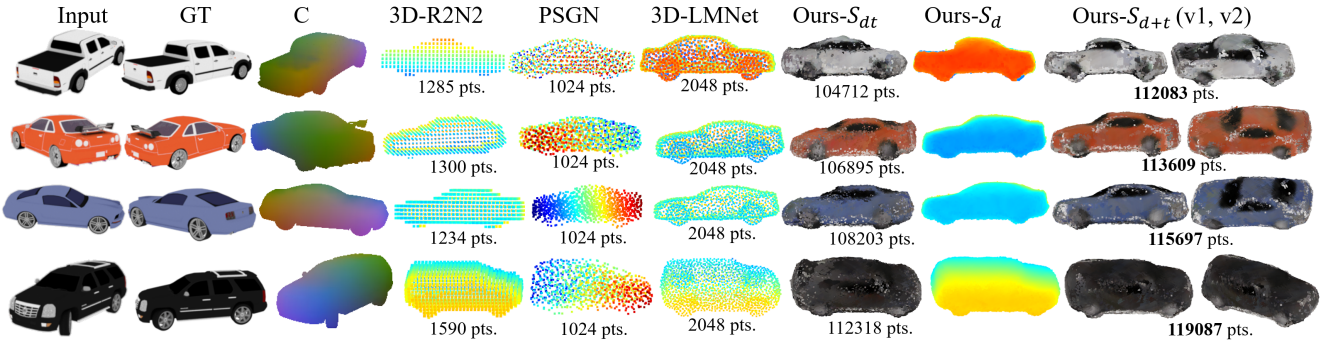

Figure 4: Reconstructions of car objects on ShapeNet dataset. 'C' is the generated object coordinate image, and 'GT' is another view of the target object. Ours-S_{dt} is generated by MTDCN, Ours-S_d and Ours-S_{d+t} are generated by MDCN.

be added. The initial vote number for each point is 1, and we set a vote threshold of 5 to decide whether one point is valid or not. In addition, radius outlier removal method is used to remove noisy points that have less than 6 neighbors in a sphere of radius 0.012 around them. However, according to our experimental results, these post-processing methods have little effect on the quantitative results. For example, for single-category task (shown in Table 1 on the manuscript), the Chamfer Distance decreases from 3.09 to 3.04 after these post-processing steps.

References

[1] Harry G. Barrow, Jay M. Tenenbaum, Robert C. Bolles, and Helen C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *IJCAI*, 1977. 4

[2] Paul Besl and H.D. McKay. A method for registration of 3-d shapes. ieee trans pattern anal mach intell. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14:239–256, 03 1992. 2, 4

[3] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. 2

[4] Christopher Bongsoo Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. *ArXiv*, abs/1604.00449, 2016. 2

[5] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2463–2471, 2016. 2

[6] Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. A papier-mache approach to

Category	PSGN	3D-LMNet	OptMVS	Ours-S_{dt}	Ours-S_{d+t}
chair	(8.36)	8.90 (**4.72**)	8.20 (6.54)	7.76 (6.78)	**6.66** (5.37)
sofa	(6.33)	**6.84** (**5.53**)	7.24 (7.05)	7.61 (6.19)	7.47 (5.84)
table	(8.07)	12.88 (**6.79**)	8.24 (8.06)	8.87 (8.37)	**7.82** (7.20)
mean-seen	(7.84)	9.02 (**5.23**)	7.98 (6.89)	7.86 (6.83)	**7.03** (5.76)
bed*	(8.47)	12.39 (8.24)	11.91 (8.19)	**10.21** (7.67)	10.31 (**7.41**)
bookcase*	(7.49)	7.49 (**5.77**)	**7.17** (7.44)	9.54 (8.86)	8.18 (7.61)
desk*	(7.70)	11.06 (6.98)	8.15 (7.73)	7.97 (7.45)	**6.91** (**6.42**)
misc*	(9.36)	12.98 (10.92)	13.28 (11.96)	12.10 (10.53)	**10.97** (**8.80**)
tool*	(10.92)	13.39 (8.80)	14.69 (10.98)	12.97 (11.39)	**11.89** (**8.71**)
wardrobe*	(6.96)	8.52 (**5.95**)	**5.89** (6.27)	8.25 (7.89)	7.52 (7.43)
mean-unseen	(8.08)	10.95 (7.45)	9.65 (8.03)	9.48 (8.12)	**8.85** (**7.31**)

Table 2: Average Chamfer Distance (CD) [1] on both seen and unseen category on Pix3D [13] dataset with 40K points as ground truth point cloud. All numbers are multiplied by 100, and '*' indicates unseen category. Numbers beyond '()' are the CD before ICP alignment [2], and in '()' are after ICP.

learning 3d surface generation. In *CVPR*, pages 216–224, 06 2018. 1, 2

[7] Tao Hu, Zhizhong Han, Abhinav Shrivastava, and Matthias Zwicker. Render4completion: Synthesizing multi-view depth maps for 3d shape completion. *ArXiv*, abs/1904.08366, 2019. 1, 2

[8] Wenzel Jakob. Mitsuba renderer. In *https://www.mitsuba-renderer.org/*, 2010. 2

[9] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 1

[10] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 2

[11] Cheng long Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3338–3347, 2019. 2

[12] Priyanka Mandikal, K L Navaneet, Mayank Agarwal, and R. Venkatesh Babu. 3d-lmnet: Latent embedding matching for accurate and diverse 3d point cloud reconstruction from a single image. *ArXiv*, abs/1807.07796, 2018. 2

[13] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B. Tenenbaum, and William T. Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018. 2, 4

[14] Yi Wei, Shaohui Liu, Wang Zhao, Jiwen Lu, and Jie Zhou. Conditional single-view shape generation for multi-view stereo reconstruction. In *CVPR*, 2019. 2

[15] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *ArXiv*, abs/1711.00199, 2017. 2

[16] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. *2018 International Conference on 3D Vision (3DV)*, pages 728–737, 2018. 2

[17] Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Josh Tenenbaum, Bill Freeman, and Jiajun Wu. Learning to reconstruct shapes from unseen classes. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2257–2268. Curran Associates, Inc., 2018. 2, 3