Supplementary Material for Learning 3D Shape Completion from Laser Scan Data with Weak Supervision

David Stutz[1,2] Andreas Geiger[1,3]
[1]Autonomous Vision Group, MPI for Intelligent Systems and University of Tübingen
[2]Computer Vision and Multimodal Computing, Max-Planck Institute for Informatics, Saarbrücken
[3]Computer Vision and Geometry Group, ETH Zürich
david.stutz@mpi-inf.mpg.de,andreas.geiger@tue.mpg.de

1. Overview

This document provides additional details regarding the used datasets and further experimental results complementary to the main paper. In Section 2, we discuss technical details regarding the introduced, synthetic datasets derived from ShapeNet [2], referred to as SN-clean and SN-noisy, the synthetic dataset derived from ModelNet [17], as well as the dataset extracted from KITTI [9]. Then, in Section 3, we provide additional details concerning architecture followed by a discussion of the training procedure in Section 4. In Section 5, we discuss our implementation of the mesh-to-mesh distance used for evaluation. Subsequently, in Section 6 we discuss the evaluation of the approach by Engelmann et al. [7] in more detail and, finally, present additional experiments in Section 7.

2. Data

ShapeNet: For generating SN-clean and SN-noisy, we took a subset of the car models provided by ShapeNet, simplified them, generated several random variants of each model and scaled them to allow signed distance function (SDF) computation. Specifically, we obtained 3253 car models from ShapeNet [2] after manually discarding 262 of the available 3515 car models. The discarded models could not be oriented or scaled automatically and mostly correspond to large, elongated or exotic cars – see Figure 1 (left) for examples. We subsequently simplified the models using the semi-convex hull algorithm described in [10] as illustrated in Figure 1 (middle). This was necessary for two reasons: to reduce the complexity of the models (i.e., reduce the number of faces from up to $\sim 1700k$ to exactly $1k$); and to obtain watertight meshes. While we would expect the majority of the car models to be approximately watertight, this is not always the case – often, we found cars without windows or with open doors. We scaled the simplified models to $[-0.5, 0.5]^3$ (centered at the origin; padded by 0.1 on all sides) in order to randomly scale (by factors in $(0.9, 1.05)$), randomly rotate (by -2.5 to 2.5, -5 to 5 and -2.5 to 2.5 degrees around the x, y and z axes, respectively), randomly translate (by $[-0.1, 0.1] \times [-0.05, 0.05] \times [-0.05, 0.05]$) and obtain 10 variants per model (6 for the validation set). We note that, in our case, x,y and z axes correspond to length, height and breadth of the cars; or x = right, y = up and z = forward. The models are then translated to $(0.5, 0.5, 0.5)$ and scaled by W to fit the voxel grid $H \times W \times D$ (for the voxel grid, x and y axes are flipped).

Based on the pre-processed models, we compute SDFs, occupancy grids and derive observations. To this end, considering watertight meshes, we make the following observation: a ray from the origin (outside of the mesh) to a voxel's center (i.e., $(w + 0.5, h + 0.5, d + 0.5)$ for $0 \leq w < W$, $0 \leq h < H$ and $0 \leq d < D$), intersects the mesh surface an odd number of times if and only if the voxel's center lies inside of the mesh. Thus, for each voxel center, we compute the distance to the surface (using point-to-triangle distance[1]) and assign negative sign if the voxel's center is found to be inside the mesh (using ray-triangle intersections [14][2]). The corresponding occupancy grids are derived by thresholding. To derive observations, we use the OpenGL renderer of [10] and subsequently back-project the pixels to 3D using the known camera calibration matrix; this process is illustrated in Figure 1 (right). To obtain observations from different viewpoints, we randomly rotate the camera around the cars (360 degrees). For SN-noisy, we additionally add exponentially-distributed error (with rate parameter 70) to

[1]We use the code provided by Christopher Batty at https://github.com/christopherbatty/SDFGen.
[2]We use the code provided at http://fileadmin.cs.lth.se/cs/personal/tomas_akenine-moller/code/.

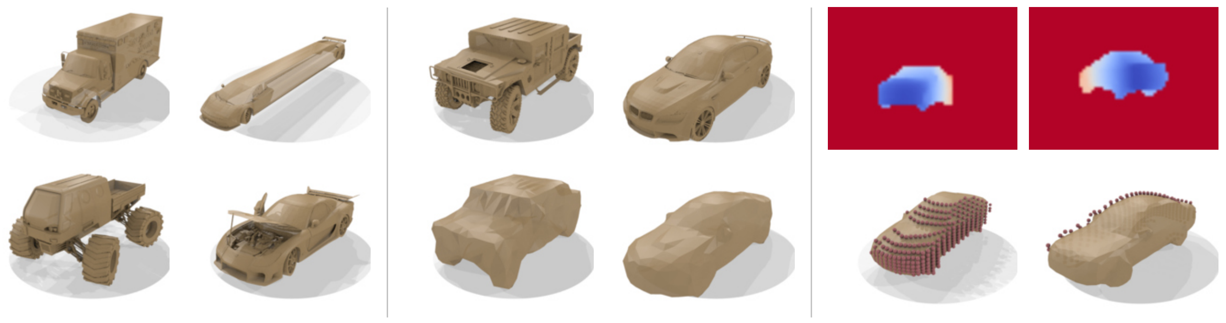

Figure 1: **Left: Examples of Discarded ShapeNet Models.** We manually discarded 262 of ShapeNet's [2] car models by inspecting top-views of the corresponding meshes. Most of the discarded models represent large, elongated or exotic cars such as trucks, limousines or monster trucks. Other models also include opened doors or other unwanted settings. **Middle: Simplification Examples.** The used simplification algorithm [10] reduces all models to exactly $1k$ faces; for some models this results in crude approximations, for others, only minor details are lost. We show the original models (top) and the corresponding simplified models (bottom). **Right: Rendering Examples.** Illustration of our rendering process for SN-clean where a model is rendered into a depth image from a random viewpoint around the car. We show the rendered depth image (top; red is background, blue is closer) and the corresponding model (bottom; beige) including observations (bottom; red).

the depth values or set them to the maximum depth with probability 0.075. The observed points can directly be voxelized; to compute free space, we perform ray tracing (from the camera center to the observed points) by computing ray-voxel intersections [16][3] along the rays. We note that we do not perform ray tracing for background points as we do not have any notion of background on KITTI.

KITTI: On KITTI, we obtained observations by extracting the ground truth 3D bounding boxes of KITTI's 3D object detection benchmark. As the bounding boxes are tightly fitted, we first padded them by factor 0.25 on all sides. Then, if l denotes the length of a particular bounding box / car after padding (i.e., its size along the x axis), we scaled the bounding box by factor $\frac{W}{l}$; afterwards, the observed points can directly be voxelized into the voxel grid of size $H \times W \times D$. Based on the average bounding box, we determined $H \times W \times D = 24 \times 54 \times 24$. We also note that the 3D bounding box annotations are in camera coordinates, while the point clouds are in Velodyne coordinates – this can be accounted for using the Velodyne-to-camera transformation as detailed in [8]. To not take points from the street or nearby walls, vegetation or objects into account, we only considered those points lying within the original (i.e., not padded) bounding box. Finally, free space is computed using ray tracing as described above.

For evaluation, we generated ground truth point clouds by accumulating points from multiple frames. In particular, for each ground truth bounding box, we considered the corresponding tracklet in KITTI's raw annotations. Again, we note that the 3D bounding boxes of the 3D object detection benchmark are in camera coordinates, while the 3D bounding boxes from the raw annotations are in Velodyne coordinates which we accounted for as described above. Overall, we considered 10 future and 10 past frames in $0.5s$ intervals (i.e., each 5th frame) going a total of 50 frames / $5s$ into the future and the past, respectively. We found that considering more frames (or smaller intervals) does not contribute to more complete observations of the cars. Finally, the ground truth points are scaled and voxelized as described above. On average, we used 6 frames per sample to obtain 597 points (compared to, on average, 128 points in the observations) on the validation set.

ModelNet: On ModelNet, we largely followed the procedure for SN-clean, considering all provided models (training and test) for object categories bathtub, dresser, monitor, nightstand, sofa and toilet. However, we used a resolution of $32 \times 32 \times 32$ and did not compute SDFs; mainly due to the complexity of the provided models, i.e., fine structures such as the legs of nightstands or dressers or the thin displays of monitors; this can also be seen in Figure 2. We also did not simplify the models beforehand. As the models are not guaranteed to be watertight, however, we first voxelized the models' surface (using triangle-box intersections[4]) and then "filled" the models using flood-filling in a post-processing step[5]. At a low resolution of $32 \times 32 \times 32$ this approach works considerably well even though the models might exhibit minor cracks or holes (as also reported in [11]). For random scaling, rotating and translating we used factors in $(0.9, 1.075)$, -5 to 5/-15 to 15/-5 to 5

[3]We use the code provided at http://www.cs.utah.edu/~awilliam/box/.

[4]We use the code provided at http://fileadmin.cs.lth.se/cs/Personal/Tomas_Akenine-Moller/code/.

[5]Using the connected components algorithm provided by Skicit-Image (http://scikit-image.org/).

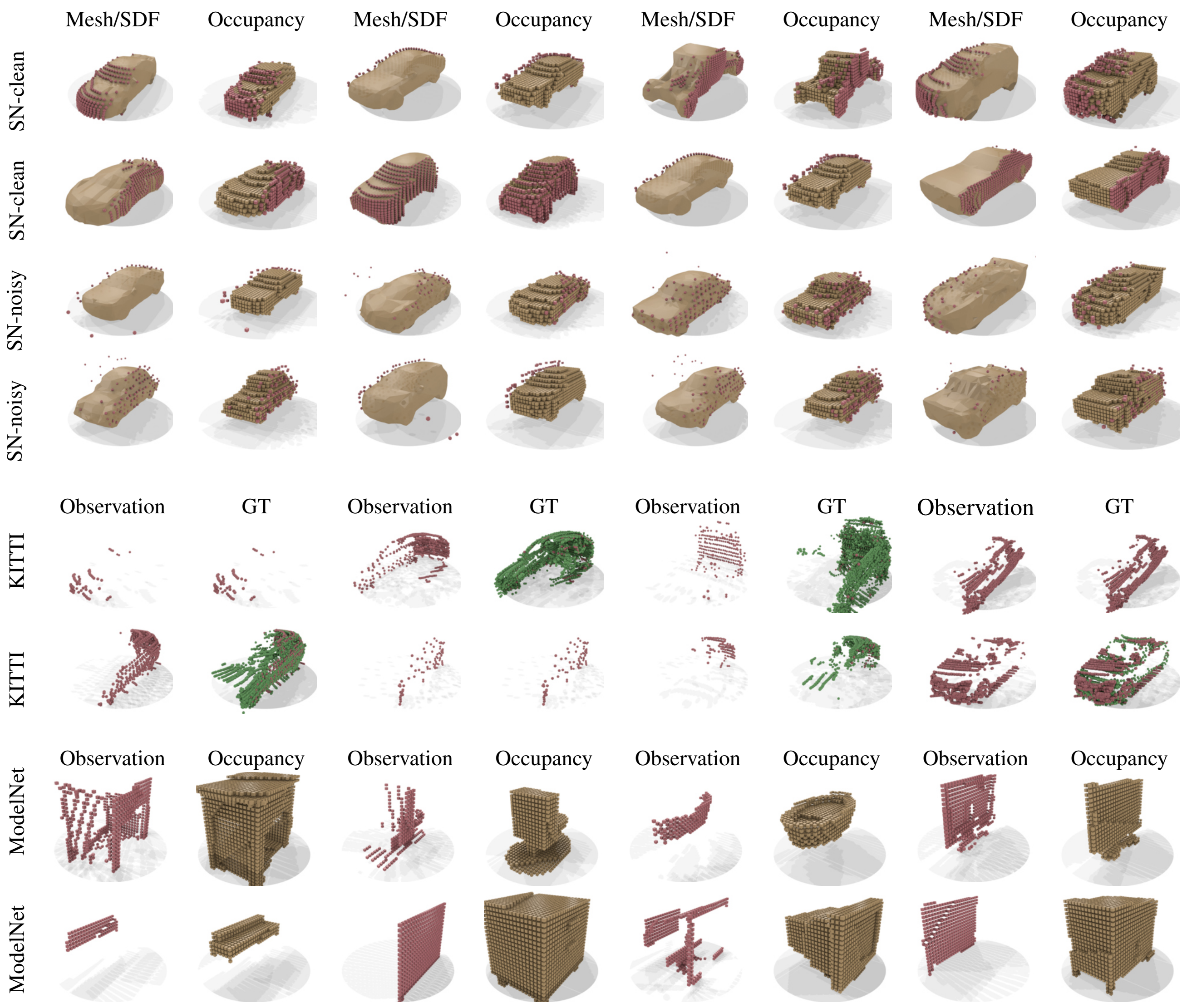

Figure 2: **Samples from our Datasets.** From top to bottom: SN-clean, SN-noisy, KITTI and ModelNet. For our synthetic datasets, we show the ground truth shapes (beige) together with the corresponding observations (red), as meshes (if applicable) and occupancy grids. For KITTI, we show the observations (red) as well as the, over multiple frames accumulated, ground truth points (green).

degrees around x/y/z axis and $[-0.075, 0.075] \times [-0.05, 0.05] \times [-0.075, 0.075]$, respectively.

Development Kit: In the spirit of reproducible research and as stated in the introduction of our main paper, we intend to publicly release the created datasets as shape completion benchmark[6]. Towards this goal, we briefly describe the provided data and tools. For SN-clean, SN-noisy and ModelNet, we provide the ground truth meshes (in OFF format) and the corresponding observations (in TXT format); both are in the same scale and coordinate system as described above. Meshes and observations are also provided using occupancy and SDF representation (if applicable, in HDF5 format). Again, we note that x,y and z axes correspond to length, height and breadth of the cars (i.e., x = right, y = up, z = forward); for the volumes in HDF5 format, x and y axes are flipped. For ModelNet we provide data for bathtub, dresser, monitor, nightstand, sofa, toilet and a combined ("all" Figure 5) dataset. For KITTI, the observations as well as the ground truth points are provided in the same coordinate system (in TXT format) and in voxelized form (in HDF5 format). Overall, the datasets are available in formats supporting both data-driven approaches as well as learning-based approaches and specifically encourage the use of

[6]See https://avg.is.tuebingen.mpg.de/research_projects/3d-shape-completion.

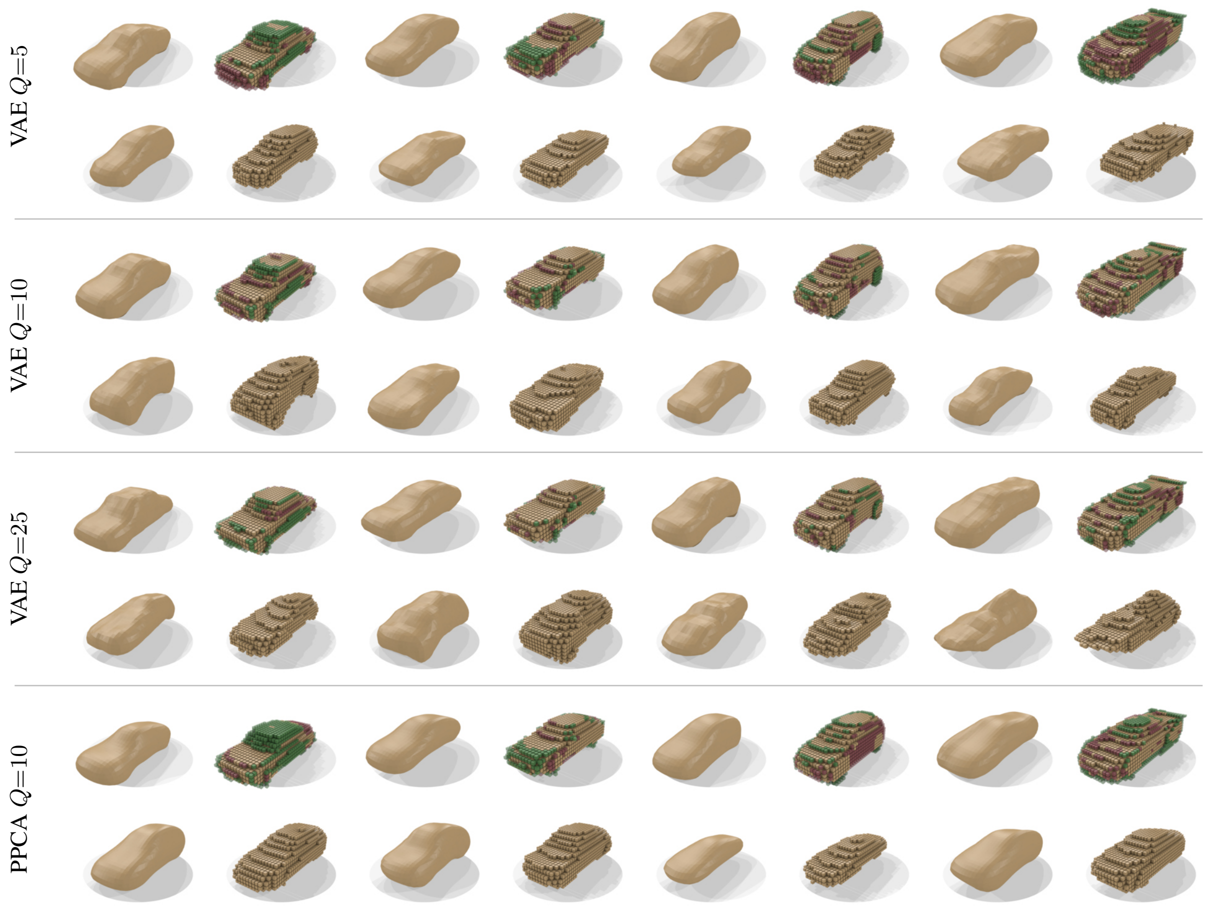

Figure 3: **Additional Qualitative Results.** Reconstruction performance and random samples for our VAE shape prior in comparison with a PPCA shape prior for different sizes of the latent space Q. In each case, we show reconstructions in the top row and random samples in the bottom row; for occupancy grids we also show false positives (red) and false negatives (green).

(convolutional) neural networks.

We also provide a development kit including C++ and Python Code for I/O and evaluation. Specifically, we provide C++ and Python code to read and write files in OFF format[7], TXT format for observed points and HDF5 format. Additionally, we provide Python code to read ShapeNet's and ModelNet's original meshes in OBJ[8] / OFF format, and to convert our TXT format for points into PLY format[9]. Meshes in OFF or OBJ format, and point clouds in PLY format can easily be visualized using MeshLab [3]. For evaluation, we provide a parallel C++ implementation of the mesh-to-mesh and mesh-to-point distance (i.e., for computing accuracy, Acc, and completeness, Comp). We also provide a implementation of the Hamming distance (Ham) in Python.

Discussion: In Figure 2, we show additional qualitative examples of the constructed datasets. As can be seen, SN-clean and SN-noisy show quite some variation in the type of cars included, as well as the generated observations – SN-clean, however, is considerable easier than SN-noisy. This is mainly due to the sparse and very noisy observations generated for SN-noisy. However, when considering KITTI, we see that real world data is very sparse and noisy by nature. We also see that the ground truth derived on KITTI is not perfect: for some samples, ground truth could not be generated; for others, the future / past frames do not contain novel viewpoints; or accumulated points do clearly not belong to the same

[7]See e.g., http://segeval.cs.princeton.edu/public/off_format.html for specifications.
[8]See e.g., http://paulbourke.net/dataformats/obj/ for specifications.
[9]See e.g., http://paulbourke.net/dataformats/ply/ for specifications.

SN-clean and SN-noisy	SN-clean	SN-noisy
#Models (Prior / Inference / Val)	325 / 1464 / 1464	
#Samples (Prior / Inference / Val)	1950 / 14640 / 1464	
Avg. #Points	411	106
% Observed / Free Space Voxels	1.06 / 7.04	0.32 / 4.8
% Invalid Free Space Voxels	0.078	0.44

KITTI	Training	Validation
#Samples	7140	7118
Avg. #Points	135	128
% Observed / Free Space Voxels	0.31 / 3.49	0.29 / 3.2

ModelNet	Prior	Inference	Val
#Models / #Samples			
bathtub	70 / 700	70 / 700	15 / 150
dresser	128 / 1280	128 / 1280	28 / 280
monitor	254 / 1524	254 / 1524	56 / 336
nightstand	128 / 1280	128 / 1280	28 / 280
sofa	351 / 1755	351 / 1755	78 / 390
toilet	199 / 1393	199 / 1393	44 / 308
Avg. #Points	1030	1038	1045
% Observed / Free Space Voxels	1.04 / 7.28	1.04 / 7.19	1.05 / 7.28

Table 1: **Dataset Statistics.** Key statistics of the synthetic datasets, SN-clean, SN-noisy (derived from ShapeNet [2]) and ModelNet [17] as well as the dataset extracted from KITTI [9].

car. On ModelNet, we observe that other object categories exhibit significantly more variations compared to ShapeNet's (simplified) cars; additionally thin structures contribute to the difficulty of ModelNet for 3D shape completion. Finally, Table 1 summarizes key statistics of the created datasets.

3. Architecture

Throughout our experiments, we found both the architecture and the shape representation to have significant influence on our results. On ShapeNet and KITTI, we used a shallow architecture where both encoder and decoder consist of three convolutional stages (no bias term) including ReLU non-linearities and batch normalization. We use 3^3 kernels and apply max pooling after each stage; specifically using window sizes $2 \times 3 \times 2$ (first two stages) and 3^3 (final stage). The special windows sizes for the max pooling layers are motivated by our resolution of $24 \times 54 \times 24$. On ModelNet, we used four stages with window size 2^2 for pooling. We also experimented with deeper architectures as well as different pooling schemes. We found that deeper architectures do not necessarily improve performance; specifically we assume that the low-dimensional bottleneck ($Q = 10$ for ShapeNet/KITTI and $Q = 25$ or $Q = 100$ for ModelNet) prevents deeper architectures to result in a significant increase in performance. Overall, the used architecture results in a lightweight but well-performing model.

For cars on ShapeNet, we also found the shape representation to have a significant influence on training. For example, occupancy is considerably easier to learn than SDFs. We assume that both the scale/range and the loss used for predicting SDFs plays an important role; therefore, motivated by work in depth estimation [5, 6, 13], we transformed the SDFs using $\text{sign}(y_i)\log(1 + |y_i|)$ for $y_i \in \mathbb{R}$. We found this transformation and the combination with occupancy (i.e., predicting both in separate channels) to aid training and result in reasonable shape priors. Alternatively, we tried splitting the SDFs into positive and negative parts (in separate channels), and transforming them separately using $\log(\epsilon + y_i)$ for $y_i \in \mathbb{R}_+$ and small $\epsilon \in (0.00001, 0.01)$. However, we assume that the increased scale (e.g., for $\epsilon = 0.00001$ and $\max(y_i) = 12$ a range of roughly $(-11.51, 2.49)$) in combination with the sum-of-squared error makes training harder. Finally, we also considered truncated SDFs, i.e., truncating the values y_i above a threshold (e.g. 5); however, this did not have a significant effect on training. Overall, the combination of transformed SDFs and occupancy was easiest to train.

4. Training

As mentioned in the main paper, we trained all our models using stochastic gradient descent (SGD) with momentum and weight decay. We initialized all weights using a standard Gaussian with standard deviation 0.02. For the shape prior, i.e., training the recognition model $q(z|x)$ and the generative model $p(x|z)$, learning rate, momentum and weight decay were set to 10^{-7}, 0.5 and 10^{-6}, respectively. We trained for 200 epochs with batch size 16 and decayed learning rate and momentum by factors 0.95 and 1.025 every 500 iterations until reaching 10^{-14} and 0.9, respectively. On ModelNet (where only occupancy grids are used), we found that a class-weighted binary cross-entropy error (weights 0.9 and 1.1 for unoccupied and occupied voxels, respectively) to help training. Furthermore, we used an initial learning rate of 10^{-6} and a decay factor of 1.04 for the momentum parameter – every 100 iterations. For shape inference, i.e., training the encoder $z(x; w)$, we used the same strategy, but trained for only 50 epochs with higher learning rate and momentum decay factors (0.9 and 1.1, respectively). The supervised baseline was trained using the same strategy as the shape prior. For the maximum likelihood baseline, we also employed SGD starting with $z = 0$ for a maximum of 5000 iterations (per sample) or until the change in objective drops below 0.001. Initial learning and momentum were set to 0.05 and 0.5, respectively; both were decayed every 50 iterations using factors 0.85 and 1.04, respectively.

	SN-clean (val)			SN-noisy (val)			KITTI (val)	
	Ham	Acc [vx]	Comp [vx]	Ham	Acc [vx]	Comp [vx]	Comp [m]	t [s]
PPCA Q=5	0.03	0.51	0.766					
PPCA (Q=10)	0.025	0.41	0.628					
PPCA Q=15	0.022	0.348	0.535					
PPCA Q=20	0.02	0.312	0.491					
PPCA Q=25	0.018	0.288	0.457					
VAE Q=5	0.021	0.409	0.606					
VAE (Q=10)	0.014	0.283	0.439					
VAE Q=15	0.011	0.245	0.367					
VAE Q=20	0.011	0.24	0.376					
VAE Q=25	**0.01**	**0.231**	**0.328**					
Mean	0.068	1.33	1.576	0.068	1.33	1.576		**0**
ML	0.04	0.733	0.845	0.059	1.145	1.331		30
Sup (on KITTI GT)	**0.022**	**0.425**	**0.575**	**0.027**	**0.527**	**0.751**	**0.176 (0.174)**	0.001
AML occ only	**0.04**	**0.725**	**0.836**	**0.058**	**1.079**	1.301	0.1	
AML sdf only	0.043	0.804	0.916	0.06	1.21	1.294	0.107	
AML Q=5	0.041	0.752	0.877	0.061	1.203	1.39	**0.091**	**0.001**
AML w/o weighted free space	0.043	0.739	0.845	0.061	1.228	1.327	0.117	
AML (on KITTI GT)				0.062	1.161	**1.203**	0.1 (**0.091**)	
[7] (on KITTI GT)		1.164	0.99		1.713	1.211	0.131 (0.129)	0.168*

Table 2: **Additional Quantitative Results.** On SN-clean and SN-noisy, we report Hamming distance (Ham), accuracy (Acc) and completeness (Comp). Both Acc and Comp are expressed in terms of voxels (as multiples of the voxel edge length); **lower is better.** On KITTI, we can only report Comp in terms of meters. All results were obtained on the corresponding validation sets. In contrast to the main paper, we also show results considering probabilistic principal component analysis (PPCA) [15] as shape prior and the VAE's mean prediction for shape completion. * Runtimes on an Intel® Xeon® E5-2690 @2.6Ghz using (multi-threaded) Ceres [1]; remaining runtimes on a NVIDIA$^{\text{TM}}$ GeForce® GTX TITAN using Torch [4].

5. Evaluation

For evaluation we consider mesh-to-mesh or mesh-to-point distances to compute accuracy (Acc) and completeness (Comp). For computing the distance of a reconstructed mesh to a target mesh (i.e., Acc), we randomly sample 10k points on the reconstructed mesh. This is done by first computing the area of each face and then sampling points proportionally to a face's area. This ensures that the sampled points cover the full mesh, i.e. the points are uniformly distributed over the reconstructed mesh. We found that $10k$ points provide sufficient accuracy compared to deterministically sampling the mesh (cf. [12]). We then compute the distance of each point to the nearest face of the target mesh and average these distances. Similarly, we can compute completeness. On KITTI, completeness can be obtained by directly computing point-to-face distances for the ground truth points.

6. Baselines

In the following, we briefly discuss related work, in particular the work by Engelmann et al. [7] in more detail. Specifically, we note that Engelmann et al. jointly optimize shape and pose. For our experiments on KITTI, we neglected pose optimization as we can directly provide the true pose (of the corresponding bounding box), or equivalently transform the observations to the origin. For results on ShapeNet, in contrast, we also optimized the pose. This was necessary as the voxel grid of the pre-trained principal component analysis (PCA) shape prior is centered at 1m height and assumes the ground plane to lie at 0m height. For SN-clean and SN-noisy, this is not the case as we consider randomly translated models such that the ground plane is not consistent. Therefore, we also needed to optimize the pose. This however results in a comparison to our approach that is not entirely fair. First of all, the approach by Engelmann et al. can directly work with the observed points (i.e., in 3D), while our approach is bound to the voxelized observations. Second, the pose optimization actually accounts for the random permutations (i.e., translations and rotations) which we deliberately integrated into SN-clean and SN-noisy and our shape prior needs to learn explicitly.

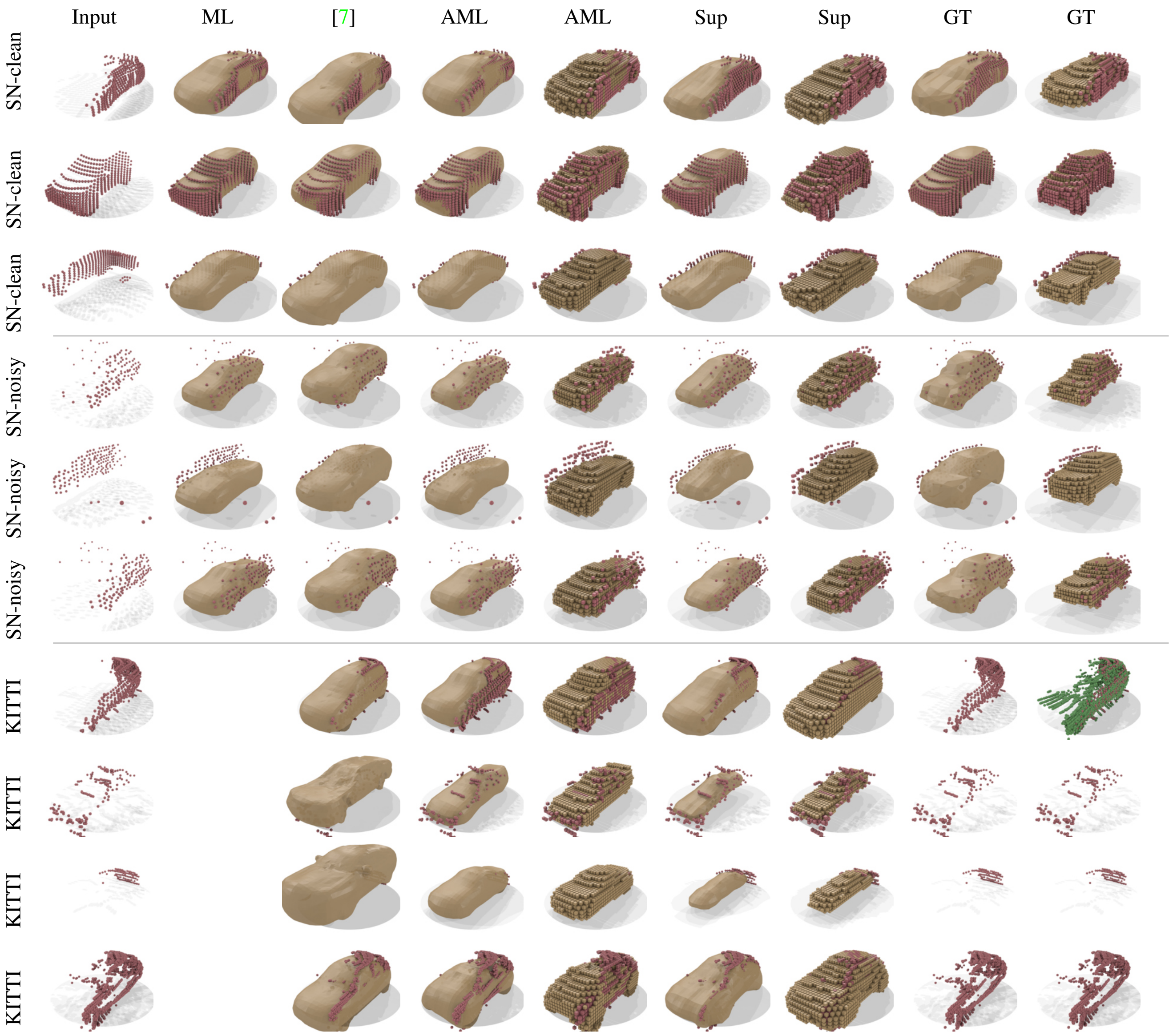

Figure 4: **Additional Qualitative Results.** On SN-clean and SN-noisy we present results for ML, [7], AML and Sup; as reference, we show the ground truth shapes including observations. On KITTI, we show results for [7], AML and Sup. As ground truth shapes are not available, we show the accumulated ground truth points instead. In all cases, we show predicted shapes (meshes and occupancy, beige) and observed points (red).

7. Experiments

In the following, we provide additional experimental results. In the main paper, We showed quantitative and qualitative results for the proposed amortized maximum likelihood (AML) approach in comparison to regular maximum likelihood (ML) and a supervised baseline (Sup) as well as the work by Engelmann et al. [7]. Complementary to the provided experiments we discuss the shape prior in more detail, present a brief ablation study for AML and discuss additional results on all three datasets.

7.1. Shape Prior

In Table 2 and in Figure 3, we present additional quantitative and qualitative results for our shape prior; for brevity we concentrate on SN-clean– on ModelNet, a similar analysis applies to $Q = 25$ ($Q = 100$ for category-agnostic model). In our experiments, we chose to use a VAE with latent space dimensionality $Q = 10$. As can be seen in Table 2, in comparison

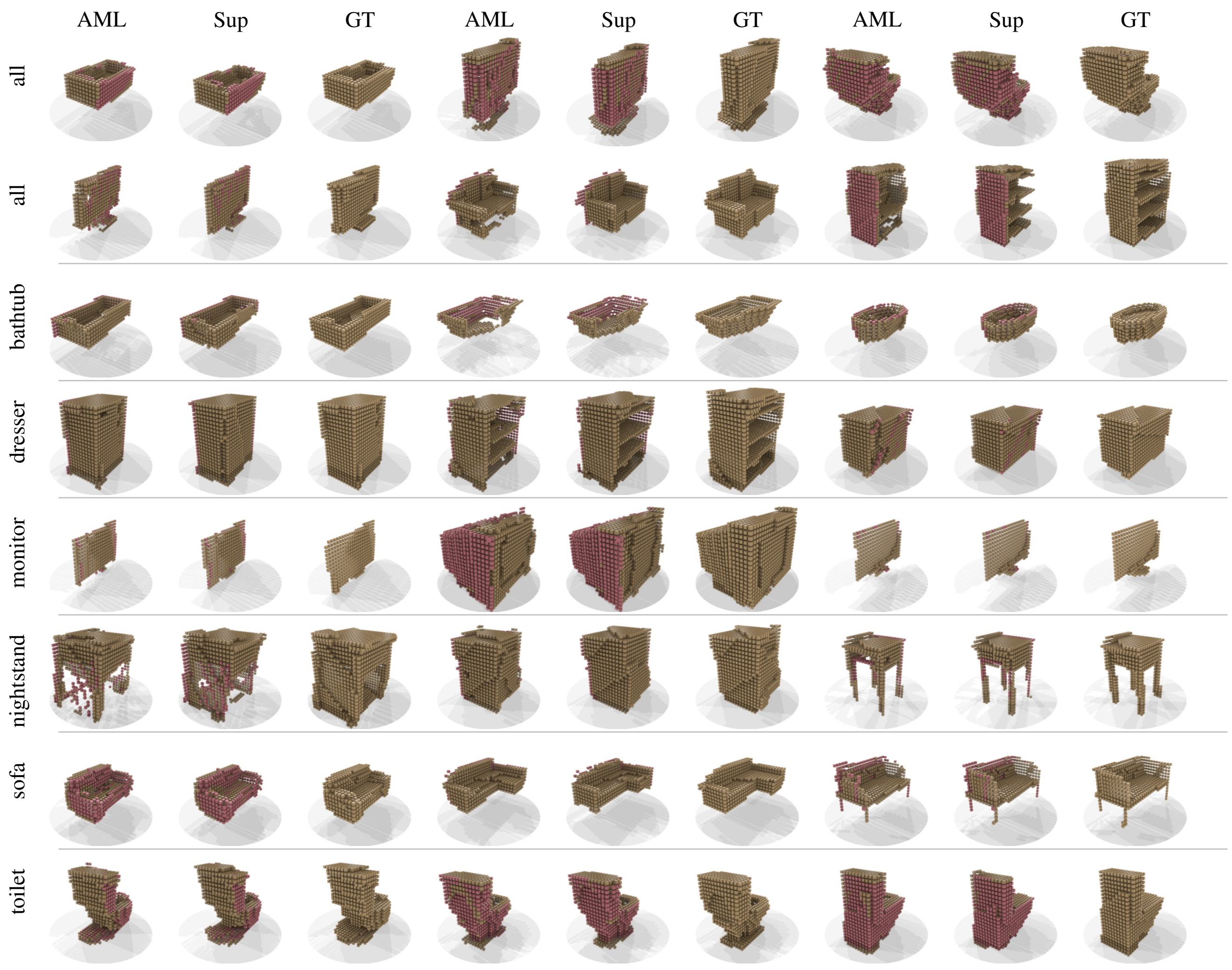

Figure 5: **Additional Qualitative Results on ModelNet.** On ModelNet, we present results for AML and Sup compared to the ground truth; we show shapes (occupancy grids, beige) and observations (red).

to a probabilistic PCA (PPCA) baseline, a VAE clearly demonstrates better reconstruction performance – independent of Q. However, increasing $Q \geq 15$ results in diminishing returns; meaning that reconstruction performance does not increase significantly while the model gets more complex (the number of weights in the fully connected layers scale with $768 \cdot Q$). Additionally, we are also interested in the quality of the learned latent space and found that random samples give a good summary of the latent space. Therefore, in Figure 3, we illustrate that the learned latent space is more likely to contain exotic shapes for larger Q. As we want to constrain the shape prior to generate reasonable cars, we chose a VAE shape prior with $Q = 10$.

7.2. Shape Completion

Ablation Study: In Table 2 we present results – in terms of Hamming distance (Ham), accuracy (Acc) and completeness (Comp) – for different variants of AML. On SN-clean, SN-noisy and KITTI, we first consider the occupancy only case, i.e., training with supervision on the occupancy channel only. As the shape prior is trained on both occupancy and SDFs, however, we are still able to predict SDFs. Similarly, we consider the SDF only case. In both cases, the shape prior is able to perform well on both modalities. Still, only using occupancy performs slightly better – supporting our assumption that occupancy is, in general, easier to learn. We also conducted experiments using a shape prior with $Q = 5$, in comparison to [7] also using $Q = 5$, and found that performance does not decrease significantly. Overall, the combination of occupancy and SDFs with $Q = 10$ performs best on KITTI's real observations.

SN-clean and SN-noisy: We consider additional qualitative results in Figure 4. On SN-clean, we notice that [7] can better reconstruct the wheels of the observed cars; our approach, including the baselines ML and Sup, have difficulties reconstructing wheels. We assume this to be caused by the different shape priors, especially regarding the models used for training. Unfortunately, Engelmann et al. do not provide their models – preventing further investigations. However, [7] has difficulties estimating the correct size of the car, often predicting cars that are too large which might also be caused by the pose optimization. On SN-noisy, we explicitly show failure cases of ML and AML. Especially in the third row, [7] predicts a more reasonable shape, while ML and AML are not able to recover the shape of the car. We assume this to be mainly due to the noise incorporated in the observations – especially ignored rays going through the car. AML is particularly influenced by invalid free space, while [7] is not. In general, however, our shape prior is biased towards thin cars (especially in terms of height) while the shape prior of [7] is more biased towards station wagons and SUV-like cars. Overall, these experiments demonstrate the importance of a good shape prior when less observations are available.

KITTI: On KITTI, again considering Figure 4, we also show failure cases. The first row shows a case were all approaches, [7], AML and Sup, predict reasonable shapes. The ground truth, however, contains invalid points – this illustrates the difficulty of objectively evaluating shape completion without proper ground truth shapes. In the second row, [7] overfits the noisy observations resulting in a "crooked" shape; AML, instead, shrinks the predicted shape to alleviate the impact of noise. The last two rows show additional failure cases for [7] and AML; specifically, [7] sometimes overestimates the size of the observed car (row three), while AML might predict unreasonable car shapes (row four). This also reflects the employed shape priors; Engelmann et al. used less car models for training the shape priors resulting in more details being recovered, but the diversity being constrained. In contrast, our shape prior includes more diversity and, as a result, allows the prediction of more exotic and sometimes unreasonable shapes. Finally, Sup has difficulties predicting reasonable car shapes (specifically rows two and three), even though we put significant effort into modeling KITTI's noise characteristics in SN-noisy.

ModelNet: On ModelNet, we consider Figure 5 showing additional qualitative results for the category-agnostic (i.e., "all") as well as category-specific models. For the former model, the first row shows examples where both AML and Sup perform well; while Sup has access to object category information during training (in the form of completed shapes), AML is able to distinguish object categories even without this information. The second row, in contrast, presents more challenging examples where Sup sometimes outperforms AML. For example, the dresser (last column, second row) is difficult to reconstruct for AML as the observations do not cover the interior structure. The remaining rows show results for category-specific models. Here, we also present some hard cases; for example the second bathtub where both AML and Sup have significant difficulties. Other examples – e.g., the second dresser – demonstrate that AML is able to perform better when knowledge about the object category is given (i.e., in contrast to the category-agnostic model). We also note that AML is able to reconstruct fine details such as legs of nightstands or sofas in spite of the sparse observations. Overall, the presented experiments demonstrate that AML generalizes to complex object categories – even without explicit knowledge about the object category during training.

References

[1] S. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org, 2012. 6

[2] 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. *arXiv.org*, 1512.03012, 2015. 1, 2, 5

[3] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. Meshlab: an open-source mesh processing tool. In *Eurographics Italian Chapter Conference*, pages 129–136, 2008. 4

[4] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011. 6

[5] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, pages 2650–2658, 2015. 5

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 5

[7] F. Engelmann, J. Stückler, and B. Leibe. Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors. In *Proc. of the German Conference on Pattern Recognition (GCPR)*, 2016. 1, 6, 7, 8, 9

[8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, 32(11):1231–1237, 2013. 2

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1, 5

[10] F. Güney and A. Geiger. Displets: Resolving stereo ambiguities using object knowledge. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2

[11] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. *arXiv.org*, 1704.00710, 2017. 2

[12] R. R. Jensen, A. L. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs. Large scale multi-view stereopsis evaluation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 6

[13] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *Proc. of the International Conf. on 3D Vision (3DV)*, pages 239–248, 2016. 5

[14] T. Möller and B. Trumbore. Fast, minimum storage ray-triangle intersection. *Journal of Graphics Tools*, 2(1):21–28, 1997. 1

[15] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society (JRSS)*, 61:611–622, 1999. 6

[16] A. Williams, S. Barrus, R. K. Morley, and P. Shirley. An efficient and robust ray-box intersection algorithm. In *ACM Trans. on Graphics (SIGGRAPH)*, number 9, 2005. 2

[17] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 5