Neural Fields for Co-Reconstructing 3D Objects from Incidental 2D Data

Supplementary Material

Table 4. Number of parameters required by each model, with instance- and view-specific parameters separated out. Note that our architecture (Ours) includes an additional appearance stream to enable disentangling of shape and appearance, allowing for material editing, at the cost of additional parameters. The remaining architectures have a single stream. Our single-stream entangled architecture (Ours-E) has fewer parameters than the StyleGAN baseline (Ours-E-SG) and the best performing competitor (EG3D).

Method	# model params	# code params per instance	# code/camera params per view
CodeNeRF [22]	714K	512	0/6
EG3D [5]	28.7M	512	0/6
Ours-E-SG [5]	28.7M	512	0/6
Ours-E	25.8M	512	32/6
Ours	37.5M	768	32/6

6. Implementation Details

6.1. Architecture

Our backbone architecture, for the triplane decoders, consists of 6 upsampling blocks with channel widths 648/648/648/648/324/162 that decode a shared, learnable $4 \times 4 \times 648$ tensor into a $256 \times 256 \times 162$ feature map. This feature map is interpreted as three $256 \times 256 \times 54$ feature planes: a triplane. Note that the channel widths are divisible by 3, and that group convolutions are used throughout the decoder in order to process the triplanes independently (*i.e.* without introducing spurious correlations). We use SiLU activations [18] in the residual blocks, and GeGLU activations [41] in the conditioning blocks.

The density, diffuse, and specular networks are implemented as MLPs with 0/1/1 hidden layers of dimension 64/128/128, with a 5-frequency Integrated Directional Encoding (IDE) [51] on the view directions. All latent codes have 256 parameters, except the directional code which has an additional 32 per-frame parameters.

6.2. Optimization

We optimize the network with AdamW [25, 29] with the initial learning rate, beta, and weight decay parameters given in Tab. 5. Note that momentum is not used to optimize the camera or latent code parameters (beta1=0) since they are accessed and receive a gradient signal only infrequently. For the large dataset (4157 instances), our model reaches convergence after 1M iterations on 4 NVIDIA A40 GPUs, which takes 322h with an un-optimized PyTorch Lightning implementation. A recent profiling study suggests that overheads associated with PyTorch Lightning may make it $3.6\times$ slower than a pure PyTorch implementation [46]. We use a learning rate scheduler that decays the learning rate of each parameter group by a factor of 0.5 every 800 epochs.

Table 5. AdamW optimization parameters.

Parameter group	Learning rate	Betas	Weight decay
Model	5×10^{-5}	(0.9, 0.999)	1×10^{-2}
Camera	5×10^{-4}	(0, 0.9)	1×10^{-2}
Latent codes	2.5×10^{-3}	(0, 0.9)	1×10^{-2}

We form a batch by sampling 4 instances (objects) per batch, 8 images per instance, 256 rays (pixels) per image, and 256 points per ray. We train with hyperparameters $\lambda^{\text{m}} = 1$ and $\lambda^{\text{d}} = 0.1$. We evaluate the co-reconstruction performance on a random subset of 1000 instances of the NuScenes dataset.

7. ScanNet Dataset Results

In this section, we evaluate on another real-world dataset, the indoor ScanNet dataset [11]. We focus on the chairs category, which is the most numerous and diverse category. ScanNet was not captured for the purpose of reconstructing chairs; this incidental capture setting is reflected in the chairs being heavily occluded and seen from very limited viewpoints (*i.e.*, not circumnavigated). Moreover, the masks used for the chairs are quite inaccurate, often missing the legs entirely, and that occlusions are extremely prevalent. This severely impedes the reconstruction performance of all models. Despite this, our model is able to produce reasonable reconstructions.

The results are shown in Fig. 9. It is notable that training diverged for the CodeNeRF [22] and EG3D [5] models on this dataset, likely due to the poor mask estimates. In contrast, our model was able to learn to co-reconstruct the chairs, despite the poor quality of the input masks. This shows that our model is able to handle noisy incidental data more gracefully than existing methods.

8. ShapeNet Dataset Details and Results

Here, we first provide additional details on our re-rendered ShapeNet cars dataset [7]. We use Blender to render medium-resolution (512×512) images against a white background, with cameras randomly distributed on the unit sphere, looking at the object. This contrasts with the dataset generated for Scene Representation Networks (SRN) [43], which contains low-resolution (128×128) images with

Figure 9. Co-reconstruction results on chairs from the real-world ScanNet dataset [11]. Repeats as: ground-truth image, training view render, novel view render. Note that the model is severely under-trained at this point, so fine details are missing. ScanNet was not captured for the purpose of reconstructing chairs; this incidental capture setting is reflected in the chairs being heavily occluded and seen from very limited viewpoints (not circumnavigated).

transparency and specularities disabled. Other than this, we follow the dataset split and rendering protocol of SRN. While it may be less appropriate quick experimentation, our higher-resolution dataset brings the data closer to real conditions and challenges. We also provide the ground-truth depth maps to faciliate geometric evaluation. Since this synthetic dataset provides exact camera poses, has a plain white background, has static objects only, and has constant camera parameters (focal length, exposure, *etc*.), we do not enable camera optimization, the mask loss, or view codes.

For the experiment, we train on the 2458 training instances and evaluate on the reserved test frames for the same instances. The frames are divided into a train–test split such that the train and test frames are taken from a strictly different half-space $sx > 0$ where the sign $s \in \{-1, +1\}$ is chosen at random per instance. In our coordinate system, this corresponds to a left–right split about the car's symmetry axis. This experiment therefore assesses the ability of a model to *extrapolate* to significantly different viewpoints—visualizing a side of the car it has not seen.

The results for the ShapeNet cars and chairs datasets are shown in Tab. 2, with additional qualitative results in Figs. 10 and 11, where we evaluate on a subset of 1000 instances ($\sim$41%). They indicate that our model can reconstruct the training data with high fidelity, despite sharing almost all parameters between the instances, and can accurately extrapolate to views unseen in the training data. In particular, we outperform the baseline CodeNeRF [22] model with respect to all measures, including the perceptual similarity (LPIPS), since our renders are less blurry and perceptually closer to the ground truth, and the mean absolute error of the distance maps (D-MAE)—*i.e*. the geometric error—because the reconstructions are more detailed. Note that EG3D [5] training diverged on the ShapeNet Cars and Chairs datasets.

9. Additional Qualitative Results

In this section, we present additional high-resolution qualitative results on the NuScenes dataset [4], shown in Figs. 12 and 13. We also include high-resolution videos of our novel view synthesis results alongside this document, for co-reconstruction experiment. They show a consistent reconstruction with smoothly varying view-dependent effects and high-frequency specular highlights appearing where appropriate, despite one side of the car being entirely unseen in most of the training instances.

We also present qualitative results on the Woven Planet (Lyft) Level 5 dataset [20] in Fig. 14, where we observe similarly high-quality reconstructions. This is the same setting as for the NuScenes experiments, using incidental data from driving sequences.

Finally, we trained a representative in-the-wild mesh reconstruction method "Shelf-Supervised Mesh Prediction" [54] on the NuScenes dataset and provide the qualitative results in Fig. 15. This approach performs significantly worse than the baselines reported in the main paper, which we attribute to a lower tolerance for occlusion, blur, and noisy masks. In particular, we can see that it struggles to reconstruct from truncated views (where only part of the car is visible in the input image), which are very typical of incidental data like that captured in the NuScenes dataset.

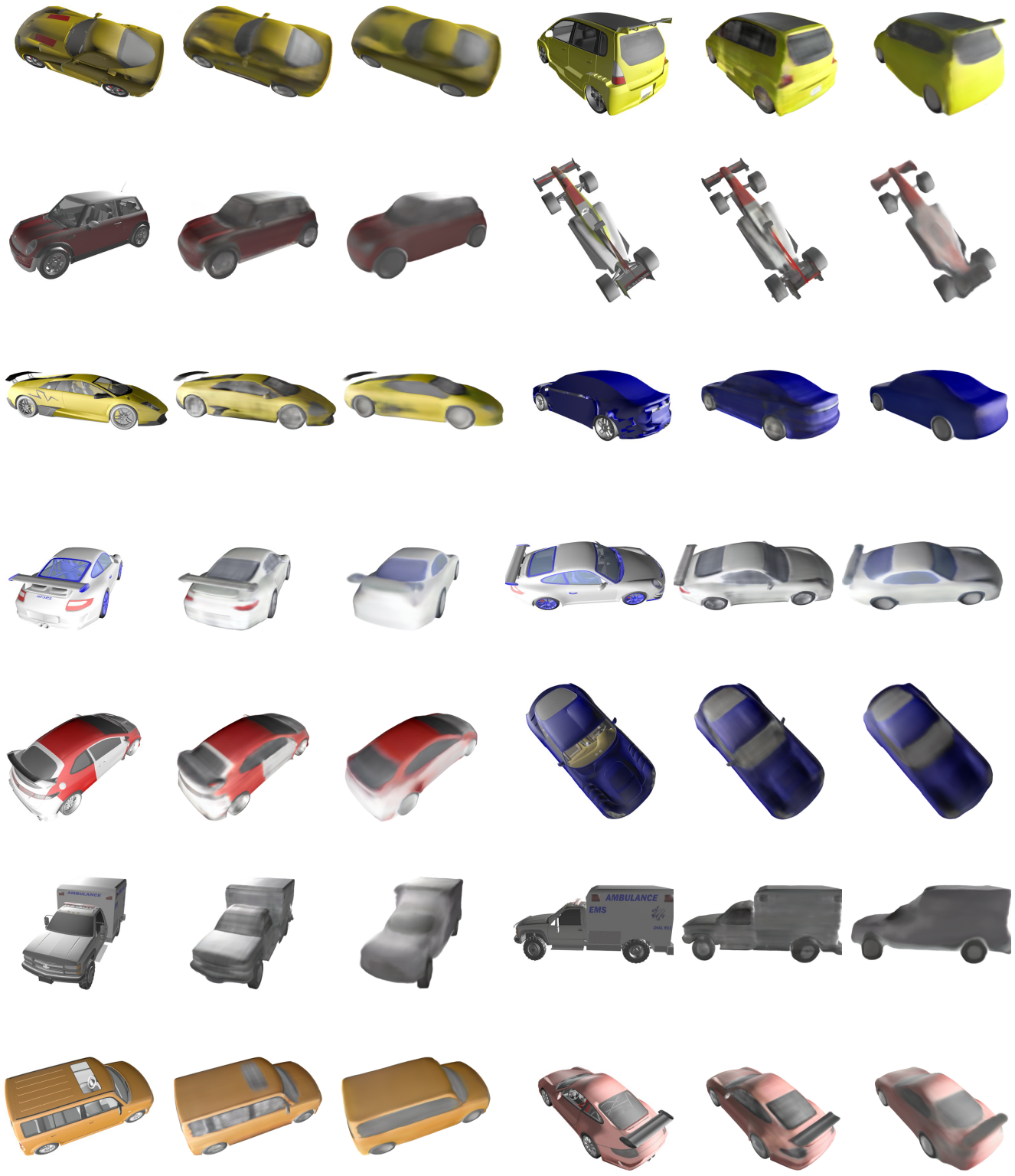

Figure 10. Co-reconstruction results on the synthetic ShapeNet Cars dataset. We display the ground-truth image alongside the rendered images from our model and CodeNeRF [22], for test frames from the *unseen* side of the car, testing the ability of the models to extrapolate and learn shape and appearance priors. Our model produces noticeably sharper and more detailed reconstructions than CodeNeRF [22].

Figure 11. Co-reconstruction results on the synthetic ShapeNet Chairs dataset. We display the ground-truth image alongside the rendered images from our model and CodeNeRF [22], for test frames from the *unseen* side of the car, testing the ability of the models to extrapolate and learn shape and appearance priors. Our model produces noticeably sharper and more detailed reconstructions than CodeNeRF [22].

Figure 12. Comparison of co-reconstruction methods on the NuScenes car dataset. Our model produces sharper reconstructions than EG3D [5] and CodeNeRF [22], which is particularly noticeable at the wheels and handles.

Figure 13. Comparison of co-reconstruction methods on the NuScenes car dataset. Our model produces sharper reconstructions than EG3D [5] and CodeNeRF [22], which is particularly noticeable at the wheels and handles.

Figure 14. Qualitative results for co-reconstructions on the Woven Planet (Lyft) Level 5 dataset [20].

Figure 15. Qualitative results of the reconstructions on the Nuscenes dataset using the mesh-based self-supervised approach [54]. Note it struggles to reconstruct in this challenging setting.Shelf-Supervised Mesh Prediction in the Wild