OD-NeRF: Efficient Training of On-the-Fly Dynamic Neural Radiance Fields

Supplementary Material

1. Qualitative Result Videos

We include a video rendered by our model and baselines for both synthetic scenes and real-world scenes in the supplementary zip file. The scenes are rendered with changing camera angles to better demonstrate the 3D structure recovered with our method.

2. Implementation Details

We implement our model on top of the TiNeuVox[1] for the synthetic dataset and the InstantNGP[6] for the real-world dataset. We describe the changes we have made to the models in this section.

2.1. Synthetic Dataset Model

As we have mentioned in the main paper, we remove the temporal components of the model because of its poor extrapolation capability. More specifically, the temporal deformation model and the temporal information enhancement are removed. Instead, the mean and variance projected color of the sampled point is concatenated with its spatial feature as the input to the NeRF Multi-Layer Perceptron(MLP). We also replace the multi-scale voxel used in TiNeuVox[1] with the hash voxel used in InstantNGP[6], implemented with tiny-cuda-nn[5]. We observe that this hashed voxel can better capture the details, but converge slower than the original multi-scale voxel. Hence we added the 2-second warm-up for the first frame as mentioned in the main paper. The rest of the model structure is following the TiNeuVox-S version published.

Since the original TiNeuVox sample uniformly along the ray instead of sampling based on the occupancy grid, we implement a rejection sampling based on our transited and updated occupancy grid. The rejection sampling filters the uniform samples based on the occupancy grid and a fixed interval. The ith sample $\mathbf{x}_i$ along the ray is rejected if the occupancy value from the occupancy grid $\sigma_{occ}(\mathbf{x}_i)$ is smaller than a density threshold σ_{min} and it is not on a fixed interval R:

$$reject\ \mathbf{x}_i\ if \quad (\sigma_{occ}(\mathbf{x}_i) < \sigma_{min})\ and\ \neg(i\ mod\ R \equiv R//2). \tag{1}$$

We fix the interval R with a value of 20, and gradually decrease the threshold σ_{min} over the optimization process from 1(at the frame 1) to 0.05(at frame 10) for better convergence at the start.

2.2. Real-world Dataset Model

The NerfAcc[3] implementation of InstantNGP[6] is used as the code base for our implementation. We concatenate the mean and variance of the projected colors of the sampled point with its spatial feature queried from the hash voxel grid, before inputting them into the NeRF MLP. We also notice that the NerfAcc[3] implementation of InstantNGP[6] does not converge very well on the forward facing dynamic dataset of MeetRoom[2] and DyNeRF[4], even for the first frame static scene. It could be because of the large planer background with constant colors, like walls and tables. Hence, we implement a simple depth smoothness regularization based on patch sampling. For any 3×3 patch of ray sampled, the depth regularization loss is calculated as:

$$\mathcal{L}_{depth} = \frac{\mathrm{std}(d_{far}/d_{3x3})}{\mathrm{std}(c_{3x3})}, \tag{2}$$

where std represents the standard deviation, d_{3x3} represents the depth values of the patch, c_{3x3} represents the ground truth color of the patch and d_{far} represents the far plane depth. This depth smoothness loss penalizes local large inverse depth variation when the color variation is small. The loss is added to the total loss with a weight of $1e^{-4}$ for MeetRoom[2] dataset and $1e^{-6}$ for DyNeRF[4] dataset.

Since the InstantNGP[6] model already has a sampling strategy based on the occupancy grid, we only update the occupancy grid itself during the training and do not change the sampling strategy itself.

3. Modifications to D-NeRF Dataset

As we have mentioned in main paper, we use a multi-view forward facing camera version of the D-NeRF[7] dataset. Since the original D-NeRF dataset does not release the blender file, we use the publicly available models and animations to render with multi-forward facing cameras. However, we could not find the scene "BouncingBalls" used in the dataset, hence train and render with a TiNeuVox-B[1] instead.

Similar to the real-world forward facing dataset, this multi-view forward facing version of D-NeRF dataset has 12 static training cameras and 1 static test cameras. The test camera is in the center and the training camera arranged in 3 rows and 4 columns with a 40° spread. The cameras all look at the center of the synthetic model.

4. Qualitative Ablation Results

To better demonstrate the effectiveness of our proposed components qualitatively, we present the qualitative abla-

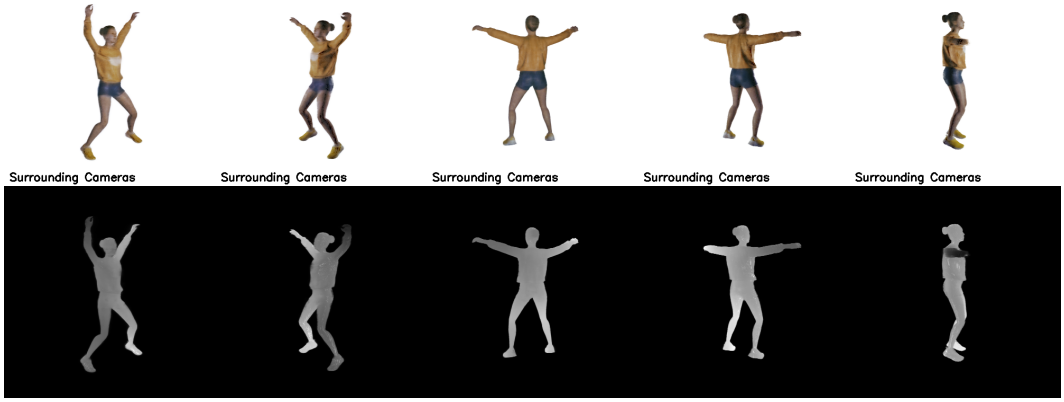

Figure 1. Qualitative results of our method trained on 360 degree surrounding cameras.

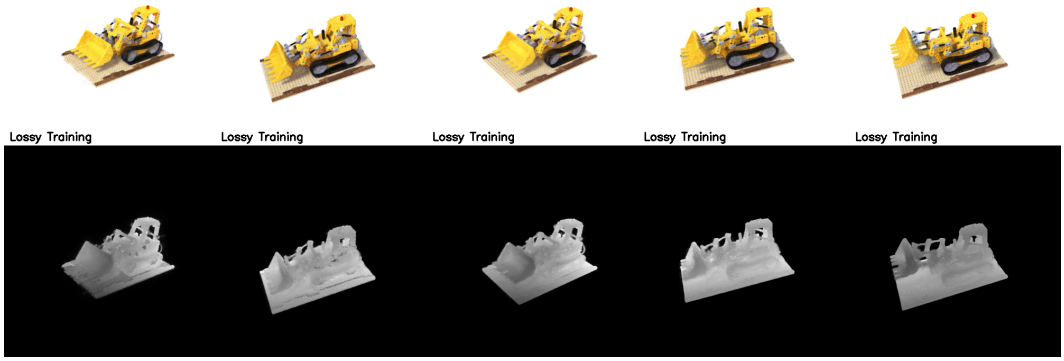

Figure 2. Qualitative results of our model under 50% randomly missing training views.

tion study results in this section. It involves the comparison of our full method and the model without projected color module, occupancy transition module, gradient scaling based on difference region, and dynamic-static fusion based on difference region.

As shown in Fig. 3, the model without the projected color input produces more blurry results. Without the guidance of the generalizable projected color input, the model is not well generalized across frames. Even though the model at the first frame captures the fine details of the scene during warmup, it will quickly be lost when the object moves. Our full model using projected color input to improve cross frame generalizability preserve the fine details and produce high quality rendering with minimal training.

As shown in Fig. 4, the model without the occupancy transition often miss out some part of the scene during sampling. It is shown as the missing square cubes on the rendered images. This is caused by the incorrect updates to the occupancy grid when the object moves. The occupancy grid in the moving region may not be updated fast enough so that the model can sample in these regions after motion. Our full model with occupancy transition mitigates this problem by increasing the occupancy value of the voxels that might be involved in object motion. This effectively reduces the chance of sampling misses especially with a small number of optimization iteration per frame.

As shown in Fig. 5, the model without the gradient scaling is more likely to produce floaters around the moving object. With minimal number of optimization iterations, it is very difficult for the model to be multi-view consistently updated after the scene changes. Floaters are formed to reduce the rendering loss from some training views, but not other training views and the novel view. Our gradient scaling strategy reduces the changes in the regions without multi-view color changes. This multi-view consistent information suppresses the formation of undesirable floaters effectively.

As shown in Fig. 6, the model without the dynamic static fusion often renders artifacts in the background. With the shared MLP decoder of all points in the scene, even

the updates to the moving foreground can influence the static background. The introduction of dynamic-static fusion based on the difference region mitigates this issue by borrowing the rendering results of the static background from the previous frames. This not only accelerates the rendering process, but also reduces the instability in the static background caused by the foreground updates.

5. Additional Experiment Setups

We include a few additional experiment setups to showcase the application of our proposed method in different scenarios.

360 degree surrounding cameras Although our method focuses on the forward-facing camera setup due to its common usage in the streaming industry, our method can be easily extended to 360 degree surrounding camera. We modify the calculation of the mean and variance color guidance of our model to be weighted by the relative position of the rendering and the training camera. The training images closer to the rendering camera contributes more to the mean and variance calculated. As shown in the qualitative results in Fig. 1, our method can still achieve a high quality rendering under this setup.

Incremental learning Despite our focus on training and rendering the scene on-the-fly, the user might want to playback the reconstructed dynamic scene. This is similar to the incremental learning setup where the model is trained with continuously acquired data and tested with the full data. The main challenge is the forgetting problem, and we are interested in whether our model can overcome it.

We implemented a small playback memory, which is one of the simplest approach used in incremental learning. The memory contains 1% of the training rays of the previous frames(1 key frame every 10 frames, 10% rays stored for each key frame). These rays are trained jointly with the latest frame on-the-fly. Even with such small overhead in storage and training, our model can maintain a good rendering quality on all frames instead of just the latest frame. On D-NeRF dataset, our model can produce test views with 29.96dB PSNR for time steps after on-the-fly training, which is only a small drop compared to the 32.87dB PSNR during on-the-fly evaluation. Although the forgetting problem is not the primary concern for streaming applications, this result suggests that the forgetting problem of our model can be easily circumvented. We believe that more sophisticated methods used for other incremental models can further mitigate the forgetting problem.

Random missing training views Our model relies heavily on the projected colors from the training images, which can be dropped or delayed due to network conditions during streaming applications. To verify the robustness of our model under such challenging condition, we setup an experiment with randomly missing training views during training and rendering. As shown in Fig. 2, our model can still produce high quality novel view synthesis results with 50% of the training views randomly dropped during training and rendering. This suggests a strong robustness of our model against lossy training images.

6. Qualitative Results with Depth

As shown in Fig. 7, we demonstrate the RGB and depth rendering of our model, compared with the StreamRF[2] model. Our model captures the 3D geometry of the scene well as illustrated by the depth map.

Figure 3. Qualitative ablation comparison of our full method and the model without the projected color input.

Figure 4. Qualitative ablation comparison of our full method and the model without the occupancy grid transition.

Figure 5. Qualitative ablation comparison of our full method and the model without the gradient scaling based on the difference region.

Figure 6. Qualitative ablation comparison of our full method and the model without the dynamic-static fusion.

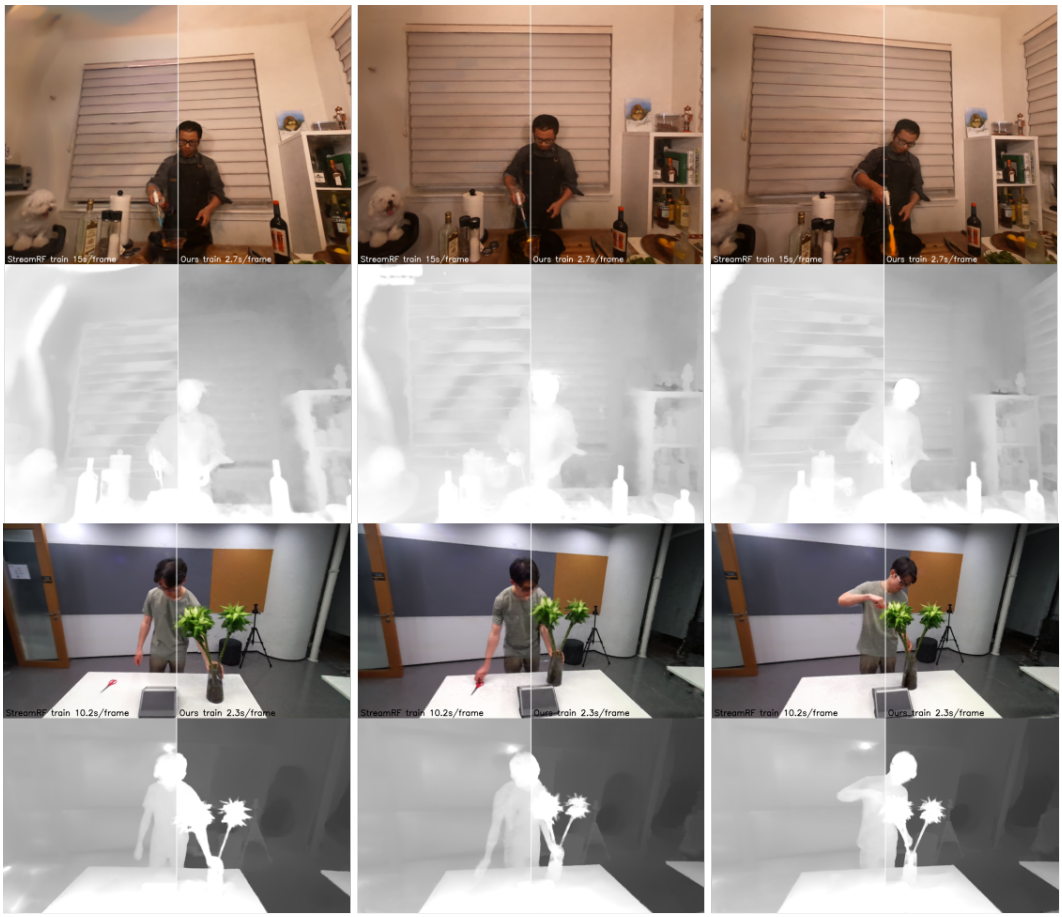

Figure 7. Qualitative comparison between our model with the StreamRF model with both RGB and depth map rendering.

References

[1] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 2022. 1

[2] Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping Tan. Streaming radiance fields for 3d video synthesis. *arXiv preprint arXiv:2210.14831*, 2022. 1, 3

[3] Ruilong Li, Hang Gao, Matthew Tancik, and Angjoo Kanazawa. Nerfacc: Efficient sampling accelerates nerfs. *arXiv preprint arXiv:2305.04966*, 2023. 1

[4] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 1

[5] Thomas Müller. tiny-cuda-nn, 2021. 1

[6] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 1

[7] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. pages 10318–10327, 2021. 1