Supplemental Materials: Neural 3D Video Synthesis from Multi-view Video

Tianye Li[1,2,*] Mira Slavcheva[2,*] Michael Zollhoefer[2]
Simon Green[2] Christoph Lassner[2] Changil Kim[3] Tanner Schmidt[2]
Steven Lovegrove[2] Michael Goesele[2] Richard Newcombe[2] Zhaoyang Lv[2]
[1]University of Southern California [2]Reality Labs Research [3]Meta

1. Supplemental Video

We strongly recommend the reader to watch our *supplemental video*, hosted at the project website: https://neural-3d-video.github.io/, to better judge the photorealism of our approach at high resolution, which cannot be represented well by the metrics.

The *supplemental video* includes:

- 3D video synthesis results on various dynamic scenes including challenging dynamic topology change fast motion, view-dependent effects such as specularity and transparency, varying illuminations and shadows, and volumetric effects such as steam and fire;
- A short presentation on the method (the DyNeRF representation and the efficient training method);
- Video comparisons to baseline methods: NeRF-T, DyNeRF-noIS, LLFF [4], NeuralVolume [3];
- Visualization of the estimated geometry (rendered as depth maps);
- Slow-motion and bullet-time effects by our DyNeRF;
- More results on more challenging indoor scenes;
- Results on immersive video datasets [1];
- Demonstration of interactive playback of our 3D videos in commodity VR headset *Quest 2* using layered meshes distilled from our pretrained DyNeRF model;
- Limitation of our results on more challenging outdoor scenes.

2. Datasets

Details on the capture setup. We build a mobile multi-view capture system using 21 GoPro Black Hero 7 cameras, as shown in Fig. 2. For all results discussed in this paper, we capture videos using the linear camera mode at a resolution of 2028×2704 (2.7K) and frame rate of 30 FPS. The multi-view inputs are synchronized by a timecode system, and the camera intrinsic and extrinsic parameters are obtained by COLMAP [6] and are kept the same throughout the capture.

Our collected data can provide sufficient synchronized camera views for high quality 4D reconstruction of challenging dynamic objects and view-dependent effects in a natural daily indoor environment, which did not exist in public 4D datasets. Our captured data demonstrates a variety of challenges for video synthesis, including objects of high specularity, translucency and transparency. It also contains scene changes and motions with changing topology (poured liquid), self-cast moving shadows, volumetric effects (fire flame), and an entangled moving object with strong view-dependent effects (the torch gun and the pan), various lighting conditions (daytime, night, spotlight from the side), multiple people moving around in open living room space with outdoor scenes seen through transparent windows with relatively dark indoor illumination. We visualize one snapshot of the sequence in Fig. 1. Unless otherwise stated, we use keyframes that are 30 frames apart. In total, we trained our methods on a 60 second video sequence (*flame salmon*) in 6 chunks with each 10 seconds in length, five other 10 seconds cooking videos captured at different time with different motion and lighting, and one 25 seconds video in indoor videos in 5 chunks. We also trained a few additional videos of outdoor scenes in chunks of 5 seconds with denser keyframes, which are 10 frames apart. In the end, we employ a subset of 18 camera views for training, and 1 view for quantitative evaluation for all datasets except one sequence observing multiple people moving, which only uses 14 cameras views for training. We calculate a continuous interpolated spiral trajectory based on the training camera views, which we employ for qualitative novel view evaluation.

We found that the GoPro linear FOV mode sufficiently well compensates for fisheye effects, thus we employ a pin-

∗ Equal contribution. TL's work was done during an internship at Reality Labs Research.

Figure 1. **Frames from our captured multi-view video** *flame salmon* sequence (top). We use 18 camera views for training (downsized on the right), and held out the upper row center view of the rig as novel view for quantitative evaluation. We captured sequences at different physical locations, time, and under varying illumination conditions. Our data shows a large variety of challenges in high quality wide angle 3D video synthesis.

Figure 2. **Our multi-view capture setup** using synchronized GoPro Black Hero 7 cameras.

hole camera model for all our experiments. For all training, we hold out the top center camera for testing, and use the rest of the cameras for training. For each captured multi-view sequence, we removed a particular camera if the time synchronization did not work. We also notice there are some inconsistent appearance in the some video streams caused by different lighting sources observed from different view angles, which we excluded in training.

Additional Immersive Videos from [1]. We also demonstrate our method using the multi-view captured videos from [1] which have been made publicly available recently. Due to the time constraints, we train DyNeRF models individually on a few 5s video clips from "Welder", "Flames", and "Alexa Meade Face Paint" to validate our algorithm. There are a few differences in their capture setup which pose different opportunities and challenges to our method. First, different from our captured linear camera videos which are front-facing, their videos are captured on a half spherical inside-out rig with heavy distortion in each view. Second, their rig is composed of 46 cameras in each scene, which contains more than two times more numbers of cameras in training. Successfully training on this scene using Dynerf requires us to compress a larger dynamic view space and utilizing all training video pixels more efficiently. During training, we sample the rays directly from the raw resolutions of the distorted multi-view videos and render the novel view video using a pinhole camera. We demonstrate our algorithm can work on this type of data to create an immersive 3D video experience without any change in the representation.

3. Importance Sampling Schemes

Sampling Based on Global Median Maps (DyNeRF-ISG). For each ground truth video, we first calculate the global median value of each ray for all time stamps $\overline{\mathbf{C}}(\mathbf{r}) = \underset{t\in\mathcal{T}}{\text{median}}\ \mathbf{C}^{(t)}(\mathbf{r})$ and cache the global median image. During training, we compare each frame to the global median image and compute the residual. We choose a robust norm of the residuals to balance the contrast of weight. The norm measures the transformed values by a non-linear transfer function $\psi(\cdot)$ that is parameterized by γ to adjust the sensitivity at various ranges of variance:

$$\mathbf{W}^{(t)}(\mathbf{r}) = \frac{1}{3}\left\|\psi\left(\mathbf{C}^{(t)}(\mathbf{r}) - \overline{\mathbf{C}}(\mathbf{r}); \gamma\right)\right\|_1 . \qquad (1)$$

Here, $\psi(x;\gamma) = \frac{x^2}{x^2+\gamma^2}$ is the Geman-McLure robust function [2] applied element-wise. Intuitively, a larger γ will lead to a high probability to sample the time-variant region, and γ approaching zero will approximate uniform sampling. $\overline{\mathbf{C}}(\mathbf{r})$ is a representative image across time, which can also take other forms such as a mean image. We empirically validated that using a median image is more effective to handle high frequency signal of moving regions across time, which helps us to approach sharp results faster during training.

Sampling Based on Temporal Difference (DyNeRF-IST). An alternative strategy, DyNeRF-IST, calculates the residuals by considering two nearby frames in time t_i and t_j. In each training iteration we load two frames within a 25-frame distance, $|t_i - t_j| \leq 25$. In this strategy, we focus on sampling the pixels with largest temporal difference. We calculate the residuals between the two frames, averaged over the 3 color channels

$$\mathbf{W}^{(t_i)}(\mathbf{r}) = \min\left(\frac{1}{3}\left\|\mathbf{C}^{(t_i)}(\mathbf{r}) - \mathbf{C}^{(t_j)}(\mathbf{r})\right\|_1, \alpha\right) . \quad (2)$$

To ensure that we do not sample pixels whose values changed due to spurious artifacts, we clamp $\mathbf{W}^{(t_i)}(\mathbf{r})$ with a lower-bound α, which is a hyper-parameter. Intuitively, a small value of α would favor highly dynamic regions, while a large value would assign similar importance to all rays.

Combined Method (DyNeRF-IS*). We empirically observed that training DyNeRF-ISG with a high learning rate leads to very quick recovery of dynamic detail, but results in some jitter across time. On the other hand, training DyNeRF-IST with a low learning rate produces a smooth temporal sequence which is still somewhat blurry. Thus, we combine the benefits of both methods in our final strategy, DyNeRF-IS*, which first obtains sharp details via DyNeRF-ISG and then smoothens the temporal motion via DyNeRF-IST.

Training Details with the Important Sampling Schemes. We apply global median map importance sampling (DyNeRF-ISG) in both the keyframe training and full video training stage, and subsequently refine with temporal derivative importance sampling only for the full video. For faster computation in DyNeRF-ISG we calculate temporal median maps and pixel weights for each view at $\frac{1}{4}$th of the resolution, and then upsample the median image map to the input resolution. For γ in the Geman-McClure robust norm, we set $1\mathrm{e}{-3}$ during keyframe training, and $2\mathrm{e}{-2}$ in the full video training stage. Empirically, this samples the background more densely in the keyframe training stage than for the following full video training. We also found out that using importance sampling has a larger impact in the full video training, as keyframes are highly different. We set $\alpha = 0.1$ in DyNeRF-IST. In the full video training stage we first train for $250K$ iterations of DyNeRF-ISG with learning rate $1\mathrm{e}{-4}$ and then for another $100K$ iterations of DyNeRF-IST with learning rate $1\mathrm{e}{-5}$.

4. More Results

4.1. Details on Baseline Methods.

- **Multi-View Stereo (MVS)**: We reconstruct the textured 3D meshes using commercial photogrammetry software RealityCapture * and render the novel view with from the textured 3D meshes frame-by-frame. This baseline demonstrates the challenges using traditional geometry based approaches.

- **Local Light Field Fusion (LLFF)** [4]: LLFF is one of the state-of-the-art Multiplane Images based methods tailored to front-facing scenes. We apply the pretrained network in LLFF to produce the multiplane images and render the novel views using default parameters. To work with videos in our datasets, we produce the novel view frame-by-frame by query the inputs at each corresponding time.

- **Neural Volumes (NV)** [3]: NV is one of the state-of-the-art learning based volumetric methods can generate novel view videos. We use the same training videos and apply the default parameters to train the network. We set the bounding volume according to the geometry of the scene. We use 128^3 voxel grid for the RGBα volume and 32^3 for the warping grid. It renders a novel view image via ray marching a warped voxel grid at each timestamps.

- **NeRF-T**: Refers to the version in Eq. 1. in the main paper, which is a straight-forward temporal extension of NeRF. We implement it following the details in [5], with only one difference in the input. The input concatenates the original positionally-encoded location, view direction, and time. We choose the positionally-encoded bandwidth for the time variable to be 4 and we do not find that increasing the bandwidth further improves results.

- **DyNeRF†**: We compare to DyNeRF without our proposed hierarchical training strategy and without importance sampling, i.e. this strategy uses per-frame latent codes that are trained jointly from scratch.

- **DyNeRF with varying hyper-parameters**: We vary the dimension of the employed latent codes (8, 64, 256, 1024, 8192). We also apply ablation studies on different versions of DyNeRF with important

*https://www.capturingreality.com/

Table 1. **Quantitative comparison** of our proposed method to baselines of existing methods and radiance field baselines trained at 200K iterations on a 10-second sequence. DyNeRF-IS* uses both sampling strategies (ISG and IST) and thus runs for *more* iterations: 250K iterations of ISG, followed by 100K of IST; it is shown here only for completeness.

Method	PSNR ↑	MSE ↓	DSSIM ↓	LPIPS ↓	FLIP ↓
MVS	19.1213	0.01226	0.1116	0.2599	0.2542
NeuralVolumes	22.7975	0.00525	0.0618	0.2951	0.2049
LLFF	23.2388	0.00475	0.0762	0.2346	0.1867
NeRF-T	28.4487	0.00144	0.0228	0.1000	0.1415
DyNeRF†	28.4994	0.00143	0.0231	0.0985	0.1455
DyNeRF-ISG	29.4623	0.00113	0.0201	0.0854	0.1375
DyNeRF-IST	**29.7161**	**0.00107**	**0.0197**	0.0885	**0.1340**
DyNeRF-IS*	29.5808	0.00110	**0.0197**	**0.0832**	0.1347

sampling methods: **DyNeRF-ISG**, **DyNeRF-IST**, and **DyNeRF-IS***.

4.2. Quantitative Comparison to the Baselines.

Tab. 1 shows the quantitative comparison of our methods to the baselines using an average of single frame metrics. We train all the neural radiance field based baselines and our method the same number of iterations for fair comparison. Compared to the existing methods, MVS, NeuralVolumes and LLFF, our method is able capture and render significant more photo-realistic images, in all the quantitative measures. Compared to the time-variant NeRF baseline NeRF-T and our basic DyNeRF model without our proposed training strategy (DyNeRF†), our DyNeRF model variants trained with our proposed training strategy perform significantly better in all metrics. DyNeRF-ISG and DyNeRF-IST can both achieve high quantitative performance, with DyNeRF-IST slightly more favorable in terms of the metrics. Our complete strategy DyNeRF-IS* requires more iterations and is added to the table only for completeness.

4.3. The Impact of Importance Sampling

In Fig. 3 we evaluate the effect of our importance sampling strategies, DyNeRF-ISG, DyNeRF-IST and DyNeRF-IS*, against a baseline DyNeRF-noIS that also employs a hierarchical training strategy with latent codes initialized from trained keyframes, but instead of selecting rays based on importance, selects them at random like in standard NeRF [5]. The figure shows zoomed-in crops of the dynamic region for better visibility. We clearly see that all the importance sampling strategies manage to recover the moving flame gun better than DyNeRF-noIS in two times less iterations. At 100k iterations DyNeRF-ISG and DyNeRF-IST look similar, though they converge differently with DyNeRF-IST being blurrier in early iterations and DyNeRF-ISG managing to recover moving details slightly faster. The visualizations of the final results upon convergence in Fig. 3 demonstrate the superior photorealism that DyNeRF-IS* achieves, as DyNeRF-noIS remains much blurrier in comparison. We notice that without importance sampling, the system cannot reach an acceptable visual quality within extended training time, indicating the necessity of the importance sampling scheme. In Fig. 4, we compare various settings of the dynamic neural radiance fields. NeRF-T can only capture a blurry motion representation, which loses all appearance details in the moving regions and cannot capture view-dependent effects. Though DyNeRF† has a similar quantitative performance as NeRF-T, it has significantly improved visual quality in the moving regions compared to NeRF-T, but still struggles to recover the sharp appearance details. DyNeRF with our proposed training strategy, DyNeRF-ISG, DyNeRF-IST and DyNeRF-IS*, can recover sharp details in the moving regions, including the torch gun and the flames.

Table 2. **Ablation studies on the latent code dimension** on a sequence of 60 consecutive frames. Codes of dimension 8 are insufficient to capture sharp details, while codes of dimension 8,192 take too long to be processed by the network. We use 1,024 for our experiments, which allows for high quality while converging fast. *Note that with a code length of 8,192 we cannot fit the same number of samples in the GPU memory as in the other cases, so we report a score from a later iteration when roughly the same number of samples have been used.

Dimension	PSNR ↑	MSE ↓	DSSIM ↓	LPIPS ↓	FLIP ↓
8	26.4349	0.00228	0.0438	0.2623	0.1562
64	27.1651	0.00193	0.0401	0.2476	0.1653
256	27.3823	0.00184	0.0421	0.2669	**0.1500**
1,024	**27.6286**	**0.00173**	0.0408	0.2528	0.1556
8,192*	27.4100	0.00182	**0.0348**	**0.1932**	0.1616

Table 3. **Comparison in model storage size** of our method (DyNeRF) to alternative solutions. For HEVC, we use the default GoPro 7 video codec. For JPEG, we employ a compression rate that maintains the highest image quality. For NeRF, we use a set of the original NeRF networks [5] reconstructed frame by frame. For HEVC, PNG and JPEG, the required memory may vary within a factor of 3 depending on the video appearance. For NeuralVolumes (NV), it only accounts the neural network size without counting its dependency on additional input streams. For NeRF, NeuralVolume and DyNeRF, the required memory is constant. All calculation are based on 10 seconds of 30 FPS videos captured by 18 cameras.

	HEVC	PNG	JPEG	NeRF	NV	DyNeRF
Size (MB)	1,406	21,600	3,143	1,080	773	**28**

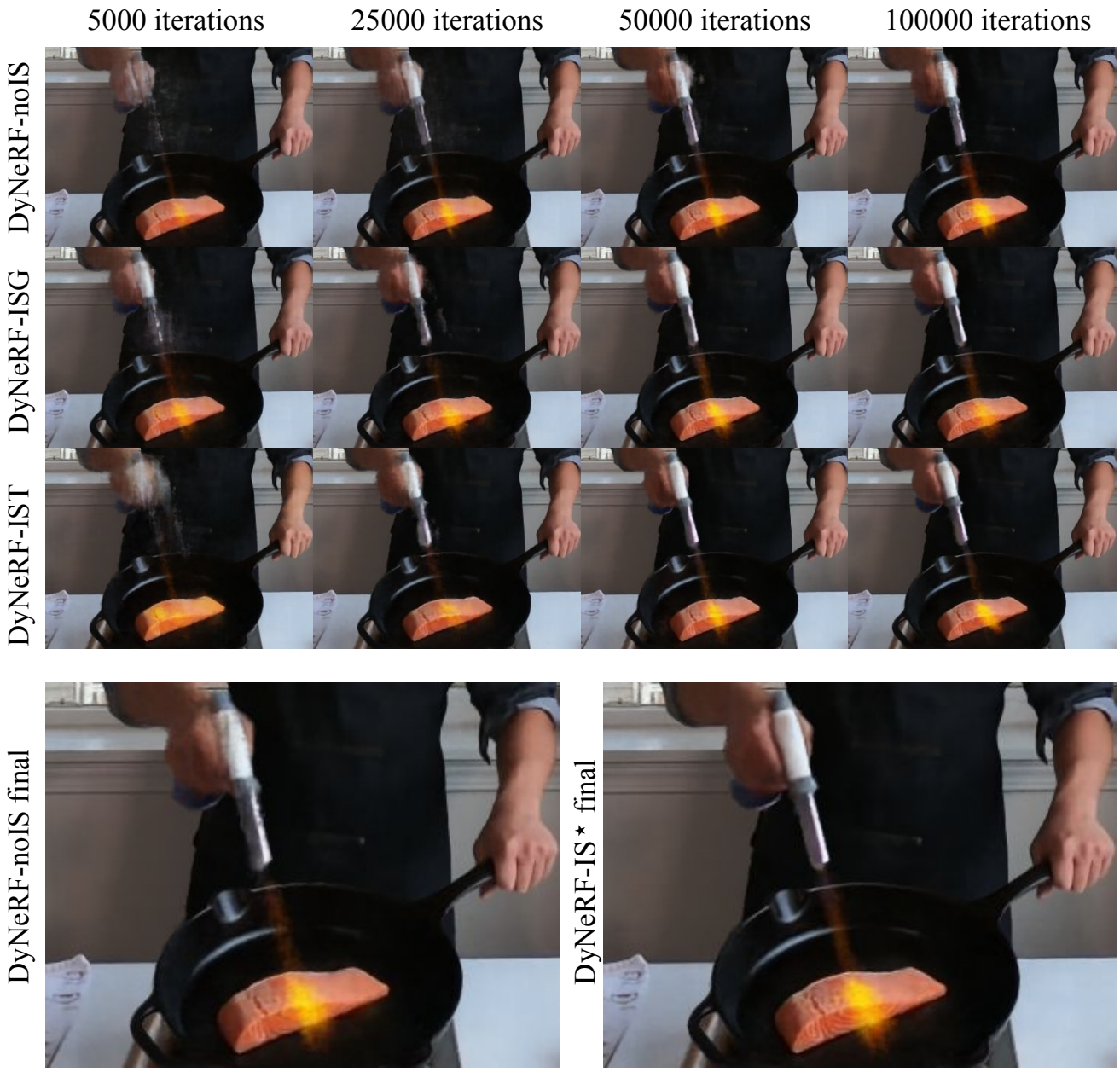

Figure 3. **Comparison of importance sampling strategies over training iterations.**

Comparisons on of Model Compression. Our model is compact in terms of model size. In Tab. 3, we compare our model DyNeRF to the alternatives in terms of storage size. Compared to the raw videos stored in different images, e.g., PNG or JPEG, our representation is more than two orders of magnitude smaller. Compared to a highly compact 2D video codec (HEVC), which is used as the default video codec for the GoPro camera, our model is still 50 times smaller. It is worth noting that these compressed 2D representations do not provide a 6D continuous representation as we do. Though NeRF is a compact model for a single static frame, representing the whole captured video without dropping frames requires a stack of frame-by-frame reconstructed NeRF networks, which is more than 30 times larger in size compared to our single DyNeRF model. Compared to the convolutional model used in NeuralVolume, DyNeRF is more compact in size and can represent the dynamic scene with better quality.

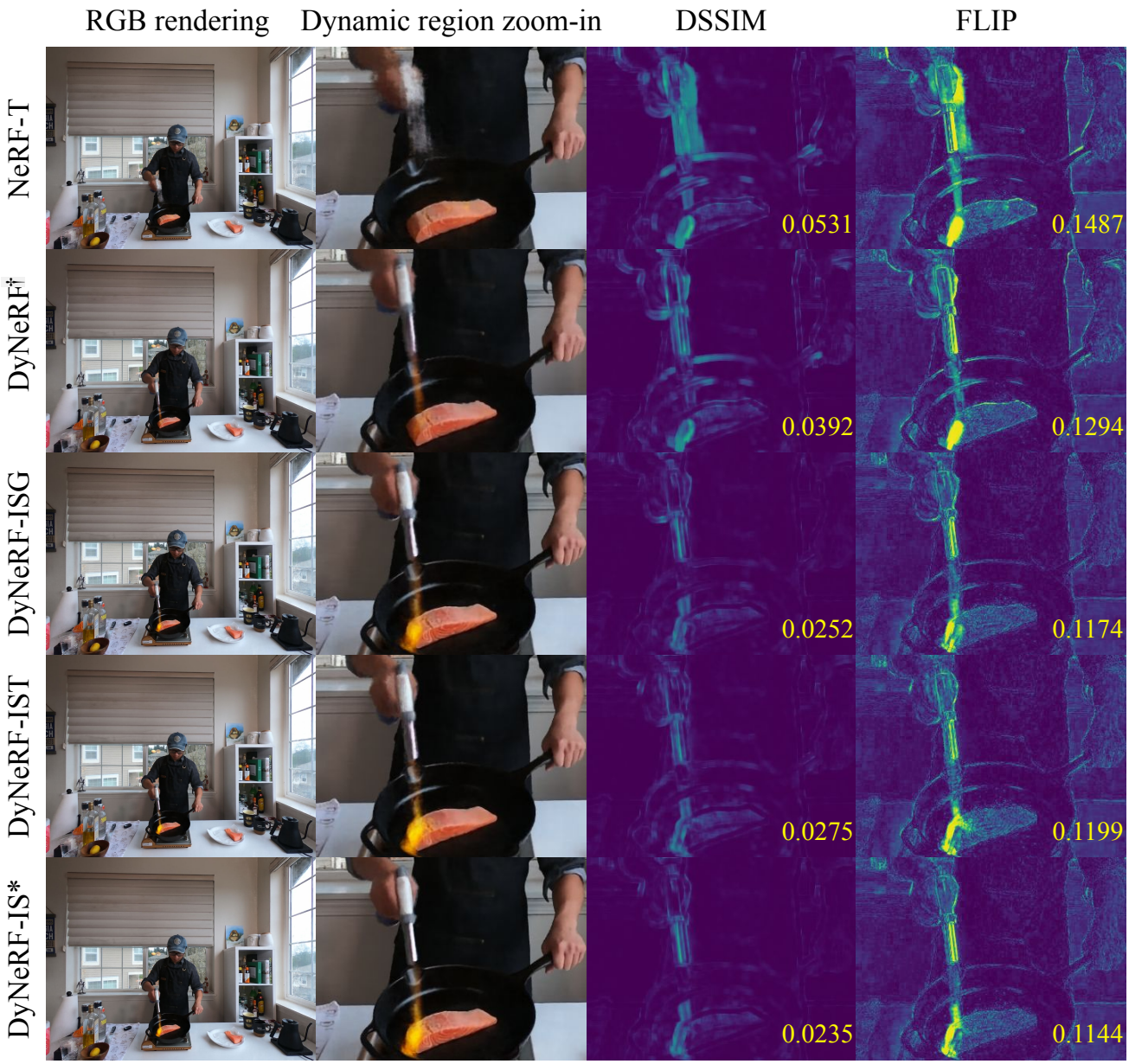

Figure 4. **Qualitative comparisons** of DyNeRF variants on one image of the sequence whose averages are reported in Tab. 1. From left to right we show the rendering by each method, then zoom onto the moving flame gun, then visualize DSSIM and FLIP for this region using the *viridis* colormap (dark blue is 0, yellow is 1, lower is better). The three hierarchical DyNeRF variants outperform these baselines: DyNeRF-ISG has sharper details than DyNeRF-IST, but DyNeRF-IST recovers more of the flame, while DyNeRF* combines both of these benefits.

Impact of Latent Embedding Size on DyNeRF We run an ablation on latent code length on 60 continuous frames and present the results in Table 2. In this experiment, we do not include keyframe training or importance sampling. We ran the experiments until 300K iterations, which is when most models are starting to converge in rendering qualities. Note that with a code length of 8,192 we cannot fit the same number of samples in the GPU memory as in the other cases, so we report a score from a later iteration when roughly the same number of samples have been used. We use $4\times$ 16GB GPUs and network width 256 for the experiments with this short sequence. From the metrics we clearly conclude that a code of length 8 is insufficient to represent the dynamic scene well. Moreover, we have visually

observed that results with such a short code are typically blurry. With increasing latent code size, the performance also increases respectively, which however saturates at a dimension of 1024. A latent code size of 8192 has longer training time per iteration. Taking the capacity and speed jointly into consideration, we choose 1024 as our default latent code size for all the sequences in this paper and the supplementary video.

Additional Discussions on the Latent Codes Besides all the above findings, we also observe some failure cases to manipulate the latent codes. Extrapolating the latent codes in time cannot directly create high quality extrapolated views. We have extensively investigated latent code optimization with various combinations of parameter learnability for latent codes (keyframe / remaining frames) and the network. With frozen keyframe latent codes, we observe blurrier results than the all-learnable case. Therefore learning both latent codes (keyframe / remaining frames) and the network is necessary for producing sharp and high-quality renderings.

View-dependent Effects in Dark Indoor Scenes. DyNeRF can represent view-dependent effects as well as motion in one continuous representation. When input cameras streams have slightly different appearance differences in observation, we find DyNeRF will model this difference as part of the view-dependent effects when generating novel views. We can observe this artifact in all of our dark indoor scenes where there are more obvious color inconsistency from wide-angle input video streams. Incorporating more careful color calibration and learning color calibration may address this problem, which we leave for future work.

3D Video Editing via Manipulating the Latent Codes DyNeRF represents a continuous spatial-temporal dynamic scene which supports rendering any view within the interpolation boundary of space and time. We can create a latent code at a sub-frame time via interpolation and render a "slow motion" 3D video with any given FPS rate. DyNeRF can enable smooth interpolation from 30fps to 60fps or even 150fps. Furthermore our method can render "bullet-time" effect by by freezing the latent code at any arbitrary time and manipulating camera views in space. We include the video effects of "slow motion" and "bullet-time" from arbitrary time in our supplementary videos.

Rendering Time The rendering time of our dynamic neural radiance fields is on par with NeRF due to the structural similarity of the approaches. Our current, not fully optimized version achieves a rendering time of 45 seconds for one 1080p frame using two V-100 GPUs with 16 GB memory.

References

[1] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020. 1, 2

[2] Stuart Geman and D McClure. Bayesian image analysis: An application to single photon emission tomography. *Amer. Statist. Assoc*, pages 12–18, 1985. 3

[3] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019. 1, 3

[4] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 1, 3

[5] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3, 4

[6] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1