3D Video Loops from Asynchronous Input
Supplementary Material

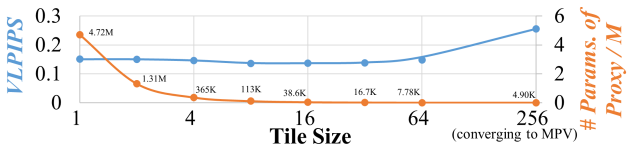


Figure 1. Ablation on the tile size. We plot *VLPIPS* and *# Params.* of the geometry proxy under different tile sizes.

	PSNR ↑	*SSIM* ↑
Ours	**26.62**	**0.8053**
VBR	24.39	0.7657
loop2D + MTV	26.16	0.7658
loop2D + MPV	26.12	0.7722
loop2D + DyNeRF	26.54	0.7756

Table 1. *PSNR* and *SSIM* for static part.

config name	spatial size	spatial stride	temporal size	temporal stride
cfg1	5	2	7	1
cfg2	11	4	5	1
cfg3	17	6	3	1

Table 2. Three patch configurations used when computing metrics.

1. Supplementary Video

We provide a supplementary video with more visual comparisons and ablation results. We strongly recommend readers to watch our supplementary video, where our method produces results with better scene dynamism than other baselines.

2. Real-Time Rendering

In order to render an MTV in real time, we convert the reconstructed MTV into a textured mesh to be compatible with the graphics rendering pipeline. We use quad geometry for the tiles. We also pack the RGBA patches of looping and static tiles into dynamic and static texture atlases. We pack the tiles by iteratively arranging each patch into a regular rectangular grid. We render the textured quads with additive alpha blending in WebGL. Our method can render at real-time rates on all tested platforms. It achieves 60fps in the iPhone X.

3. Quantitative Comparison in Static Region

Apart from evaluating the quality of the whole scene, we additionally measure the masked PSNR and SSIM in static regions for our method and the four baselines. The static regions are identified using the estimated loopable mask. The results are shown in Tab. 1. Ours achieves the best performance because the VBR produces blurry and ghosting results due to the blending operation and inaccurate geometry estimation, and the 2D looping baselines fail to consider view inconsistency.

4. Ablation on Tile Size

We conduct an ablation on the tile size in Fig. 1. It shows that our method is robust to the tile size within a certain range. When the tile size is too small, the *# Params.* of the geometry proxy increase exponentially. When the tile size is too large, the MTV converges to the MPV representation, which is inefficient in rendering and has poor visual quality as shown in the experiment in the paper.

5. Implementation Details of Metrics

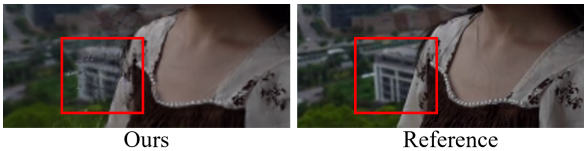
When computing *Com.*, *Coh.* and *loopQ* scores, we extract 3D patches from a source video and find the nearest neighbor 3D patches in the target video. The patches are extracted with fixed patch size and stride in both the spatial and temporal axes. We repeat the same process three times, each time with a different patch configuration (see Tab. 2). We use a mixture of small and large patch sizes in both spatial and temporal dimensions to ensure diversity of the 3D patches when computing metrics. We individually report the value of each metric under three patch configurations in Tab. 3. Our method outperforms other baselines and ablation results under all patch configurations.

	Com. w/ *cfg1*	*Com.* w/ *cfg2*	*Com.* w/ *cfg3*	*Coh.* w/ *cfg1*	*Coh.* w/ *cfg2*	*Coh.* w/ *cfg3*	*loopQ* w/ *cfg1*	*loopQ* w/ *cfg2*	*loopQ* w/ *cfg3*
Ours	**10.47**	**10.76**	**10.73**	**8.648**	**9.483**	**9.677**	**8.647**	**9.478**	**9.665**
loop2D + MTV	11.75	11.90	11.84	9.288	10.13	10.34	9.296	10.13	10.35
loop2D + MPV	11.80	11.88	11.785	9.200	10.02	10.24	9.218	10.04	10.26
loop2D + DyNeRF	11.84	12.03	11.91	9.616	10.46	10.62	9.643	10.50	10.66
w/o pad	**10.47**	10.77	**10.73**	8.651	9.487	9.683	8.736	9.605	9.845
w/o 2stage	11.60	11.79	11.68	9.309	10.20	10.44	9.416	10.34	10.64
w/o pyr	10.70	10.97	10.92	8.927	9.776	9.963	8.859	9.682	9.854
w/o tv	10.93	11.23	11.20	9.137	9.979	10.18	9.082	9.896	10.09

Table 3. *Com.*, *Coh.* and *loopQ* scores of comparisons and ablations under different patch configurations. The suffix w/ *cfg#* indicates the score with some specific patch configuration. (**best** in bold, and second best underlined)

(a) Small Baseline (b) Large Baseline

Figure 2. Failure case when synthesizing a novel view that has a large baseline.



Ours Reference

Figure 3. One failure case when the scene motion is not textural.

6. More Discussion and Limitations

While our method improves the Multiplane Image (MPI) representation by sparsifying the geometry, in inherits some limitations of the MPI representation. Our method can only synthesize novel views from a small baseline. Fig. 2 shows a failure case where our method produces holes in the synthesized novel view. Furthermore, it cannot model complex view-dependent effects, such as non-planar specular. These limitations may possibly be resolved by introducing neural textures into our representation as in NeX [1], which could be an interesting future direction to explore.

Our algorithm also makes some assumptions about the input. Firstly, we assume scenes undergo periodic motions that can form a loop. It may generate noisy results for non-loopable motions, as shown in Fig. 3. We implicitly model the geometry as alpha values in each tile, and the tile geometry remains fixed throughout the video. Therefore, we cannot model large movements that result in significant geometry changes. In addition, our method also assumes known camera poses, which may require the scene to contain some non-dynamic parts covered in each frame for camera pose estimation.

7. Gallery of Results

We show a gallery of the dataset and our results in Fig. 4. Please visit our project website for more results and real-time demos.

References

[1] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534–8543, 2021. 2

Figure 4. A Gallery of the dataset and our results.