Supplementary Material for VideoLifter: Lifting Videos to 3D with Fast and Efficient Hierarchical Stereo Alignment

Wenyan Cong[1], Hanqing Zhu[1], Kevin Wang[1], Jiahui Lei[2], Colton Stearns[3], Yuanhao Cai[4], Dilin Wang[5], Rakesh Ranjan[5], Matt Feiszli[5], Leonidas Guibas[3], Zhangyang Wang[1], Weiyao Wang[5*], Zhiwen Fan[1,6*]
[1]UT Austin [2]UPenn [3]Stanford [4]JHU [5]Meta [6]TAMU
Project Website: https://videolifter.github.io

1. 3D Gaussian Splatting Preliminary

3D-GS [4] is an explicit 3D scene representation using a set of 3D Gaussians to model the scene. A 3D Gaussian is parameterized by a mean vector $\boldsymbol{x} \in \mathbb{R}^3$ and a covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$:

$$G(\boldsymbol{p}) = \frac{1}{(2\pi)^{3/2}\,|\Sigma|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{p}-\boldsymbol{x})^T\Sigma^{-1}(\boldsymbol{p}-\boldsymbol{\mu})} \tag{1}$$

To capture view-dependent effects, spherical harmonic (SH) coefficients are attached to each Gaussian, and the color is computed via $\boldsymbol{c}(\boldsymbol{d}) = \sum_{i=1}^{n} \boldsymbol{c}_i \mathcal{B}_i(\boldsymbol{d})$. $\mathcal{B}_i$ is the i^{th} SH basis. The color is rendered via $\boldsymbol{c} = \sum_{i=1}^{n} \boldsymbol{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j)$, where $\boldsymbol{c}_i$ is the color computed from the SH coefficients of the i^{th} Gaussian. α_i is given by evaluating a 2D Gaussian with covariance multiplied by a learned per-Gaussian opacity. The 2D covariance matrix is calculated by projecting the 3D covariance to the camera coordinates. The 3D covariance matrix is decomposed into a scaling matrix and a rotation matrix and is optimized as Gaussian attributes.

2. Implementation Details

2.1. Datasets

When replicating the experiments of CF-3DGS on the CO3D-V2 dataset, we observed inconsistencies between the sequences they claim to use and those for which they report metrics. To ensure clarity and consistency, we base our evaluation on the sequences they used for experiments and reported metrics. Detailed information about each sequence is provided in Table 1.

To evaluate performance on long sequences from the Tanks and Temples dataset, we down-sample the videos to 5 FPS and extract ∼1,000 frames that provide at least 360° coverage of the scenes. These frames are then divided into training and test splits. Experiments on these long videos are conducted under a more challenging and comprehensive setting, with detailed statistics for each sequence reported in Table 2.

Scenes	Type	Seq. length	Frame rate	Max. rotation (deg)
34_1403_4393	indoor	202	30	180.0
46_2587_7531	indoor	202	30	180.0
106_12648_23157	outdoor	202	30	180.0
110_13051_23361	indoor	202	30	71.6
189_20393_38136	indoor	202	30	180.0
245_26182_52130	indoor	202	30	180.0
247_26441_50907	indoor	202	30	180.0
407_54965_106262	indoor	202	30	180.0
415_57112_110099	outdoor	202	30	180.0
429_60388_117059	outdoor	202	30	180.0

Table 1. **Details of selected sequences in CO3D-V2.** *Max rotation* denotes the maximum relative rotation angle between any two frames in a sequence.

Scenes	Type	Seq. length	Frame rate	Max. rotation (deg)
Church	indoor	950	5	180.0
Museum	indoor	900	5	180.0
Barn	outdoor	1100	5	180.0
Family	outdoor	1000	5	180.0
Horse	outdoor	1000	5	180.0
Ballroom	indoor	1050	5	180.0
Francis	outdoor	900	5	180.0
Ignatius	outdoor	900	5	180.0

Table 2. **Details of extracted long sequences in Tanks and Temples dataset.** *Max rotation* denotes the maximum relative rotation angle between any two frames in a sequence.

2.2. Training Details in Hierarchical Alignment

1) Joint Optimization of Camera Poses and Local Gaussians: Local Gaussians, initialized directly from fragments, are optimized for 200 steps to refine both Gaussian parameters and camera poses. 2) Cross-Fragment Alignment: To achieve a more accurate transformation between two local Gaussians, we use the initial transformation from Key Frame Optimization to align the first novel view in the next local Gaussian. This camera pose is optimized for 200 steps. Subsequently, the optimized transformation is applied to the rest of first four novel views, with each view being optimized for an additional 200 steps. 3) Visibility Masking and Pairwise Merging: After optimizing camera poses, visibility masks are generated for the first four novel views through rendering. These views are prioritized because they are less likely to

	Train Time	SSIM ↑	PSNR ↑	LPIPS ↓	ATE ↓
CF-3DGS [3]	10h13min	0.4792	15.21	0.4628	**0.014**
Ours	**2h42min**	**0.8824**	**29.75**	**0.1281**	**0.007**

Table 3. **Quantitative Evaluations on long sequences in Tanks and Temples Dataset.** Note that in training, our method does not require any camera parameters, while CF-3DGS still requires known intrinsics.

Model	Train Time	SSIM ↑	PSNR ↑	LPIPS ↓
MASt3R MVS+3DGS	19min3s	0.6400	18.48	0.5355
CUT3R MVS+3DGS	17min1s	0.6191	18.36	0.5577
VGGT MVS+3DGS	16min57s	0.6541	19.88	0.5392
Ours	24min58s	0.8957	30.02	0.1745

Table 4. **Ablation studies on different MVS initialization methods on CO3D-V2 Dataset.**

be masked and more likely to contain redundant points that need filtering during pairwise merging. Following the merging step, as the local Gaussian expands to include views from one additional fragment, the training iteration is extended by 200 steps.

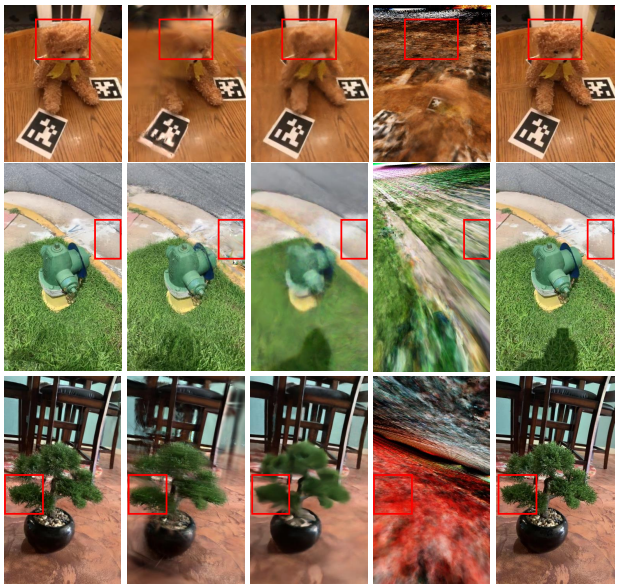

Figure 1. **Visual Comparisons on CO3D-V2 dataset.** `VideoLifter` achieves faithful 3D reconstruction, preserves better details, and alleviates incremental error in progressive learning.

3. More Experimental Results

3.1. Per-Scene Breakdown Results

To better illustrate the effectiveness of `VideoLifter`, we present a breakdown analysis in Table 5 and Table 6, comparing `VideoLifter` with other baselines on individual scenes.

On the Tanks and Temples dataset, `VideoLifter` achieves comparable performance in PSNR and surpasses CF-3DGS in most SSIM and LPIPS metrics. On the CO3D-V2 dataset, the advantage of `VideoLifter` is even more obvious, consistently outperforming CF-3DGS across all scenes. It is worth noting that CF-3DGS is fragile on the CO3D-V2 dataset and can fail intermittently. For fairness, we run CF-3DGS multiple times on the same scene and report its best performance.

3.2. Quantitative Comparison on Long Videos

For long sequences from the Tanks and Temples dataset, we compare against the strongest baseline, CF-3DGS, on both novel view synthesis and pose estimation in Table 3. Our method achieves significantly better performance while requiring only about 26% of the training time.

3.3. More Qualitative Comparisons

We present additional qualitative results for novel view synthesis on Tanks and Temples and CO3D-V2 in Figure 2 and Figure 1, respectively, following the same evaluation procedure described in the main paper. For NeRFmm, which is designed for forward-moving camera motions, the large and diverse camera motions in the CO3D-V2 dataset often lead to reconstruction failures.

3.4. MVS Initialization Comparisons

Since 3D-GS requires a point cloud for initialization and camera poses for optimization, a straightforward strategy is to leverage multi-view stereo (MVS) methods to obtain both, and then directly train 3D-GS. This approach, adopted by InstantSplat [2], has shown effectiveness in sparse-view settings. However, InstantSplat cannot directly handle dense views, as the global optimization in stereo-based methods such as MASt3R often leads to out-of-memory (OOM) issues. To extend it to long-sequence videos, we adopt a chunk-by-chunk variant, but the performance degrades significantly.

More recently, several end-to-end MVS methods (e.g., CUT3R, VGGT, Fast3R) have been proposed that process multi-view inputs in a feed-forward manner. We therefore conduct additional comparisons following the same strategy as InstantSplat, but replacing the MVS backbone with these methods. As reported in Table 4, while these approaches are faster, their performance remains substantially worse than ours, even when jointly optimizing 3D-GS and poses. The main reason is that the dense predictions from MVS are overly redundant, and the estimated poses lack sufficient accuracy. Optimizing all frames at once with such noisy inputs hinders 3D-GS training, resulting in blur and artifacts. Similar observations have also been reported in GitHub issues [1].

scenes	Ours				CF-3DGS				Nope-NeRF				BARF				NeRFmm			
	SSIM ↑	PSNR ↑	LPIPS ↓	ATE ↓	SSIM	PSNR	LPIPS	ATE	SSIM	PSNR	LPIPS	ATE	SSIM	PSNR	LPIPS	ATE	SSIM	PSNR	LPIPS	ATE
Family	**0.96**	**33.21**	**0.04**	**0.001**	0.96	32.91	0.05	0.003	0.67	23.70	0.49	0.007	0.61	23.04	0.56	0.115	0.64	23.68	0.51	0.075
Horse	**0.96**	33.25	**0.04**	0.006	0.96	**33.74**	0.05	**0.003**	0.82	26.75	0.28	0.008	0.72	24.09	0.41	0.014	0.57	16.91	0.50	0.006
Barn	**0.91**	**30.53**	**0.10**	**0.002**	0.85	29.17	0.12	0.004	0.66	25.22	0.47	0.037	0.64	25.28	0.48	0.050	0.54	18.97	0.55	0.025
Francis	0.91	32.32	**0.11**	**0.004**	**0.92**	**32.72**	0.14	0.006	0.78	28.81	0.40	0.009	0.69	25.85	0.57	0.082	0.63	23.00	0.52	0.053
Ignatius	**0.93**	**30.64**	**0.07**	0.006	0.90	28.11	0.09	**0.005**	0.61	24.05	0.48	0.005	0.47	21.78	0.60	0.029	0.40	18.29	0.59	0.016
Church	**0.93**	29.57	0.11	**0.002**	0.93	**30.08**	**0.09**	0.002	0.72	24.92	0.41	0.010	0.62	23.17	0.52	0.052	0.52	21.45	0.53	0.029
Museum	**0.92**	**30.47**	**0.08**	0.005	0.91	29.90	0.10	**0.004**	0.77	26.48	0.34	0.021	0.61	23.58	0.55	0.263	0.37	15.88	0.65	0.065
Ballroom	**0.96**	**32.75**	**0.04**	0.005	0.96	32.51	0.05	**0.003**	0.67	24.02	0.42	0.006	0.50	20.66	0.60	0.018	0.58	22.00	0.51	0.014
Average	**0.93**	**31.59**	**0.07**	**0.004**	0.92	31.14	0.09	0.004	0.71	25.49	0.41	0.013	0.61	23.42	0.54	0.078	0.53	20.02	0.55	0.035

Table 5. **Per-Scene Quantitative Comparisons on Tanks and Temples**. Each baseline method is trained with its public code under the original settings and evaluated with the same evaluation protocol. The best results are highlighted in bold.

scenes	Ours				CF-3DGS				Nope-NeRF				NeRFmm			
	SSIM ↑	PSNR ↑	LPIPS ↓	ATE ↓	SSIM	PSNR	LPIPS	ATE	SSIM	PSNR	LPIPS	ATE	SSIM	PSNR	LPIPS	ATE
34_1403_4393	**0.90**	**30.02**	**0.17**	0.014	0.78	25.19	0.25	**0.009**	0.79	29.02	0.44	0.052	0.53	14.77	0.63	0.058
46_2587_7531	**0.86**	**27.91**	**0.18**	**0.007**	0.76	23.50	0.24	0.009	0.73	26.66	0.42	0.055	0.23	11.74	0.72	0.066
106_12648_23157	**0.82**	**25.04**	**0.20**	0.010	0.34	16.17	0.48	**0.008**	0.42	19.82	0.59	0.065	0.20	11.01	0.73	0.069
110_13051_23361	**0.90**	**29.55**	**0.16**	0.042	0.66	21.35	0.35	**0.031**	0.71	26.58	0.49	0.054	0.48	16.18	0.68	0.045
189_20393_38136	**0.93**	**32.54**	**0.21**	0.007	0.91	32.22	0.26	**0.005**	0.84	29.32	0.54	0.067	0.71	16.43	0.60	0.056
245_26182_52130	**0.89**	**29.77**	**0.21**	**0.007**	0.82	26.23	0.31	0.017	0.79	25.73	0.49	0.046	0.45	13.76	0.70	0.068
247_26441_50907	**0.78**	**25.67**	**0.32**	**0.006**	0.68	21.44	0.41	0.006	0.72	23.99	0.53	0.036	0.33	11.39	0.78	0.051
407_54965_106262	**0.87**	**29.13**	**0.31**	**0.009**	0.76	21.67	0.44	0.009	0.82	27.78	0.57	0.066	0.69	14.69	0.70	0.072
415_57112_110099	**0.80**	**27.47**	**0.23**	0.016	0.69	24.92	0.33	**0.004**	0.60	24.33	0.55	0.049	0.29	12.04	0.73	0.069
429_60388_117059	**0.76**	**26.62**	**0.25**	**0.006**	0.42	17.15	0.45	0.046	0.61	22.21	0.57	0.058	0.47	12.30	0.79	0.056
Average	**0.85**	**28.37**	**0.22**	**0.012**	0.68	22.98	0.35	0.014	0.70	25.54	0.52	0.055	0.44	13.43	0.71	0.061

Table 6. **Per-Scene Quantitative Comparisons on CO3D-V2**. Each baseline method is trained with its public code under the original settings and evaluated with the same evaluation protocol. The best results are highlighted in bold.

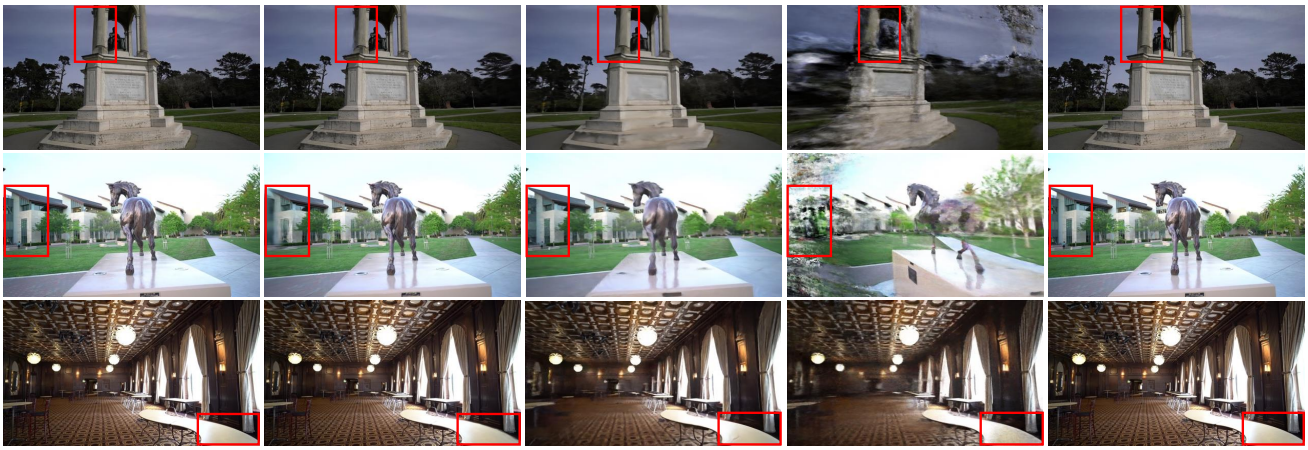

Figure 2. **Visual Comparisons on Tanks and Temples dataset.** `VideoLifter` achieves faithful 3D reconstruction, preserves better details, and alleviates incremental error in progressive learning.

References

[1] facebookresearch. Github issue #310: Training loss keeps increasing. `https://github.com/facebookresearch/vggt/issues/310`, 2025. Accessed: 2025-08-21. 2

[2] Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, et al. Instantsplat: Unbounded sparse-view pose-free gaussian splatting in 40 seconds. *arXiv preprint arXiv:2403.20309*, 2024. 2

[3] Yang Fu, Sifei Liu, Amey Kulkarni, Jan Kautz, Alexei A Efros, and Xiaolong Wang. Colmap-free 3d gaussian splatting. *arXiv preprint arXiv:2312.07504*, 2023. 2

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4): 1–14, 2023. 1