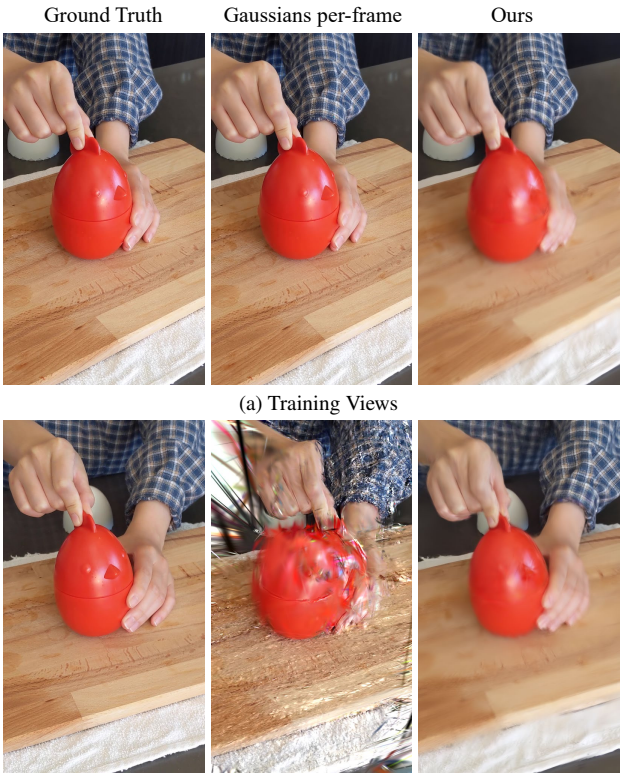
GauFRe 🥯: Gaussian Deformation Fields for Real-time Dynamic Novel View Synthesis

Supplementary Material



(a) Training Views

(b) Test Views

Figure 9. **Per-frame optimization struggles on monocular video.** While this works well on training views (*top center*), the lack of constraints causes serious errors in test views (*bottom center*). Our deformation-based method is better able to handle these sparse constraints. This motivates our approach.

6. Motivation: Per time step vs. deformation-based approaches

One basic option when considering how to implement dynamic Gaussian splatting is to optimize the Gaussian parameters per timestep. Another is to build an initial good first-frame reconstruction and then directly optimize the Gaussian parameters over time in a frame-to-frame fashion. In situations where the Gaussians are well constrained, such as in multi-view video conditions, per timestep optimization works from the training views such that generated intermediate test views are correct. This can provide an initial good first-frame from which to conduct a frame-to-frame approach over time [12].

In the monocular video case, it is tricky to deploy either approach (Fig. 9, middle) and both will fail to generate plausible test views: First, the Gaussians are underconstrained; they are free to move in one spacetime dimension and still appear correct in the training views, so test views will not look correct. Second, as the Gaussians are incorrect, there is *never* a good first-frame (or any t-frame) initialization from which to perform frame-to-frame deformation, unlike in Kerbl *et al.*'s case with multi-view video [12].

Instead, our deformation-based approach with a canonical frame (Fig. 10) makes it easier to constrain the Gaussians over time, producing more plausible test views that maintain their integrity (Fig. 9, right). This is because information on the position and size of the dynamic regions is shared within the canonical frame. For static scene elements, our separate set of Gaussians allows multi-view constraints to be used.

7. Additional results

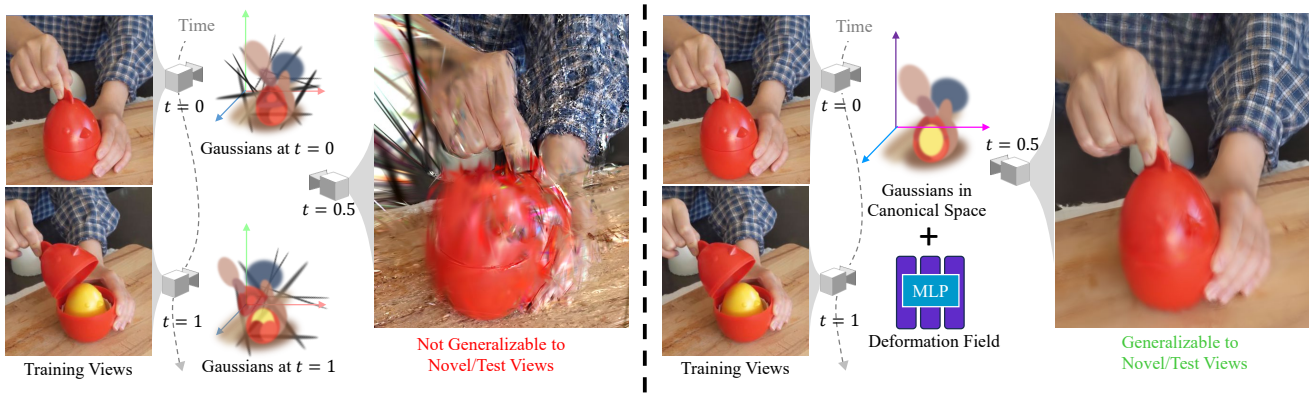
Table 6 contains quantitative results for each sequence in the *NeRF-DS [38]* dataset individually.

HyperNeRF dataset [23] We will additionally use the real-world monocular video HyperNeRF dataset for evaluation. This is formed of real-world dynamic scene sequences captured with a handheld camera. For test views, there are two modes: 1) 'vrig', where a second handheld camera was pointed at the scene, and where these camera poses are somewhat in error given the dynamic scenes; 2) 'interp', where test frames are at unseen times within a single video. This dataset has challenging sequences. With the camera pose error, some methods perform better under certain errors than others (Tab. 5). Please see our qualitative results, too (Fig. 11).

8. Static/dynamic separation and depth

Figure 12 shows more examples of the separation of static and dynamic regions, given that our model explicitly separates static regions. While this aspect of our approach relies on dynamic regions undergoing sufficient motion for high quality, when this is so the results show good-quality static background regions (*Bell*) and reconstruction of objects in motion like the knife in *Lemon*.

Further, Figure 13 shows that our approach produces reasonable results for depth on the evaluation datasets.



(a) Per-frame Gaussians **(b)** Deformable Gaussians

Figure 10. **Optimizing Gaussians per time step from training views leads to low-quality test views for monocular video.** A lack of spacetime constraints causes Gaussians to fail to represent as meaningful images (a). A canoncial space with a deformation field helps improve the integrity of the image in situations with few constraints (b).

HyperNeRF-vrig [23] (real)				
	PSNR↑	MS-SSIM↑	LPIPS↓	Optim. ↓
Nerfies [22]	22.2	0.803	-	∼hours
HyperNeRF [23]	22.5	0.816	-	∼hours
NDVG [9]	23.4	0.777	0.566	35mins
TiNeuVox [5]	24.2	0.791	0.478	20mins
V4D [8]	24.8	0.827	0.417	∼hours
Ours	22.6	0.779	0.353	2hours
HyperNeRF-interp [23] (real)				
	PSNR↑	MS-SSIM↑	LPIPS↓	Optim. ↓
NDVG [9]	23.9	0.785	0.540	35mins
TiNeuVox [5]	27.0	0.889	0.386	20mins
V4D [8]	27.3	0.909	0.322	∼hours
Ours	25.0	0.838	0.309	1.5hours

Table 5. **Quantitative comparison for novel-view synthesis [23].** For HyperNeRF, pose error causes alignment issues to reduce PSNR, but our overall image quality is maintained as shown by the lowest LPIPS metric evaluation. Please see our qualitative results for comparison.

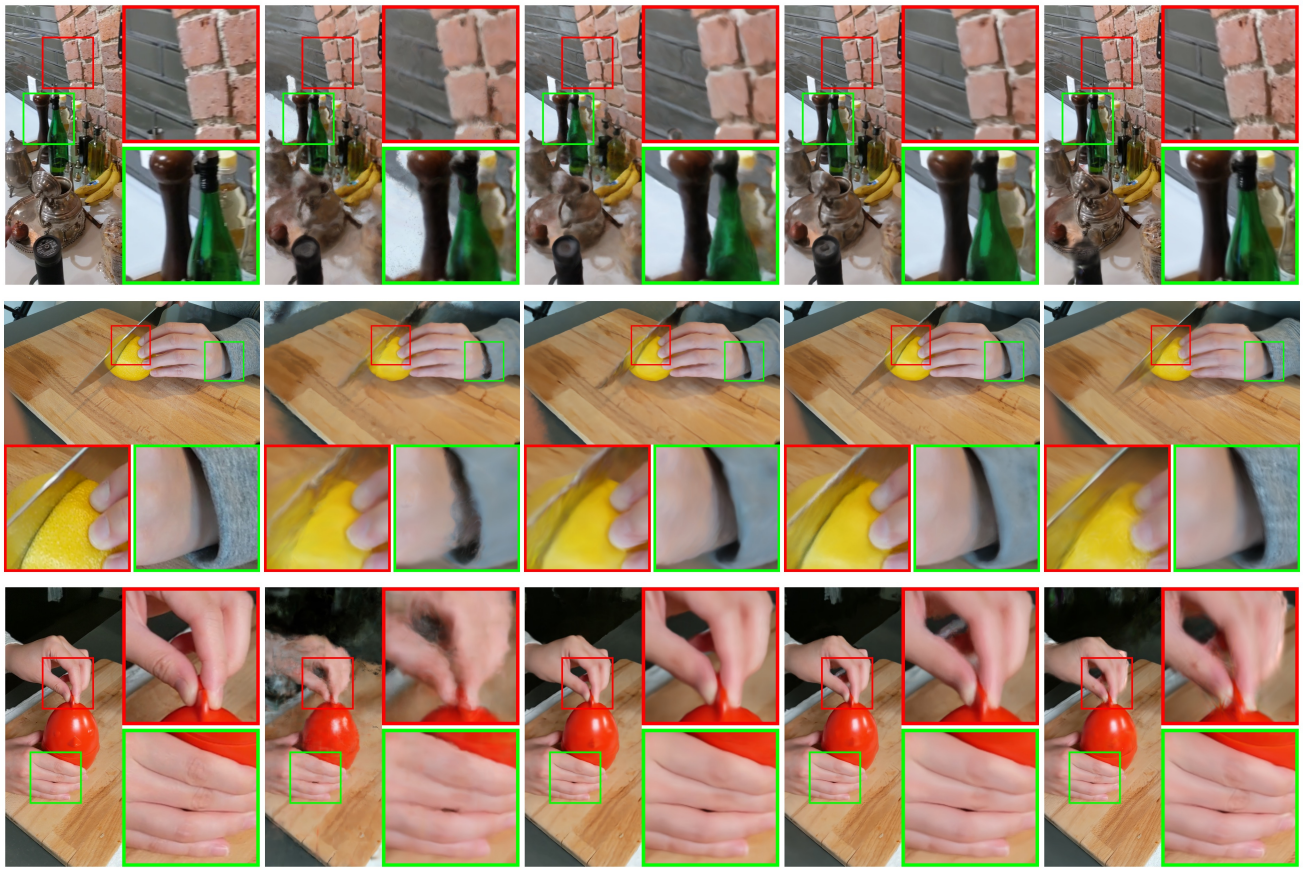


Figure 11. Quantiative comparison of our approach and the baseline methods at test views on the real-world HyperNeRF-vrig [23] dataset. Our approach most closely matches the ground truth in quality, producing sharper images with more detail such as the tiny holes in the bricks (top) or the texture pattern on the sleeve (bottom). Note that, quantitatively, our approach produces lower PSNR and SSIM scores than both TiNeuVox and V4D, which may both be oversmooth, even though qualitatively more fine detail is reproduced in our method.

	Sheet			*Basin*			*Bell*			*Cup*		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ref-NeRF [33]	21.1	0.673	0.296	18.0	0.643	0.319	18.5	0.564	0.420	20.5	0.705	0.318
Nerfies [22]	23.6	0.834	0.183	18.1	0.635	0.368	19.9	0.696	0.389	20.7	0.750	0.293
HyperNeRF [23]	24.3	0.874	0.148	20.2	0.829	0.168	24.0	0.884	0.159	24.1	0.896	0.138
NeRF-DS [38]	24.7	0.887	0.141	20.1	0.835	0.149	23.9	0.887	0.138	24.4	0.912	0.122
NDVG [9]	18.3	0.462	0.461	16.9	0.585	0.409	17.1	0.422	0.625	23.4	0.825	0.2å57
TiNeuVox [4]	21.3	0.740	0.315	20.6	0.860	0.142	23.1	0.878	0.160	19.9	0.773	0.269
V4D [8]	24.8	0.901	0.133	20.1	0.848	0.140	24.9	0.913	0.128	24.6	0.906	0.116
Ours	25.6	0.899	0.125	19.6	0.852	0.145	25.4	0.917	0.124	24.3	0.922	0.117
	Plate			*Press*			*Sieve*			Average (Mean)		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ref-NeRF [33]	15.3	0.513	0.464	21.3	0.679	0.341	22.1	0.815	0.220	19.5	0.656	0.340
Nerfies [22]	15.4	0.489	0.599	21.3	0.720	0.377	21.8	0.823	0.232	20.1	0.707	0.349
HyperNeRF [23]	18.1	0.714	0.359	25.4	0.873	0.164	25.0	0.909	0.129	23.0	0.854	0.181
NeRF-DS [38]	20.5	0.855	0.196	26.1	0.894	0.142	26.0	0.928	0.112	23.7	0.885	0.143
NDVG [9]	17.4	0.629	0.391	23.3	0.772	0.282	17.2	0.393	0.492	19.1	0.584	0.417
TiNeuVox [4]	20.5	0.820	0.224	25.2	0.863	0.171	21.1	0.793	0.253	21.7	0.818	0.219
V4D [8]	19.0	0.788	0.221	26.4	0.907	0.135	24.9	0.922	0.118	23.5	0.884	0.142
Ours	20.0	0.818	0.238	25.8	0.894	0.147	26.0	0.904	0.114	23.8	0.887	0.144

Table 6. **Quantitative image quality results on the dataset from *NeRF-DS [38].*** Our approach on average shows improved PSNR, SSIM (here the multi-scale variant), and LPIPS metric performance than most methods, and comparable image quality metric performance to NeRF-DS [38] and V4D [8]. However, in contrast to these two methods that each take hours to optimize a representation for a scene and seconds to render each frame, our approach takes minutes to optimize per scene and can render frames in real time (Tab. 3).

Figure 12. **Static/dynamic separation visualization.** *Top to bottom*: Results on scenes from the *Peel* sequence from HyperNeRF-vrig [23] dataset, the *Lemon* sequence from HyperNeRF-interp [23] dataset, and the *Bell* sequence from NeRF-DS [38] dataset. With sufficient motion in the sequence, such as in *Bell*, our approach well separates the dynamic object and produces a sharp static background. In sequences with inaccurate camera poses, such as in *Peel*, the separation is more difficult as static regions do not appear consistently static to the optimization and so there is more noise in the separation. However, the broad motions are still separated.

Figure 13. *Top to bottom*: The novel-view rendering and depth from our method for scenes from the NeRF-DS [38], D-NeRF [26] and HyperNeRF [23] datasets.