Gaussians-to-Life: Text-Driven Animation of 3D Gaussian Splatting Scenes

Supplementary Material

S1. Implementation Details

We make our code available under the following URL: https://github.com/wimmerth/gaussians2life. Optimization of dynamics in a single scene takes about 10 minutes on average. In our experiments, video generation is limited to 8 frames to balance computational constraints with the requirements of the video diffusion model. We note that this also limits the amount of motion that can be effectively distilled into the scene.

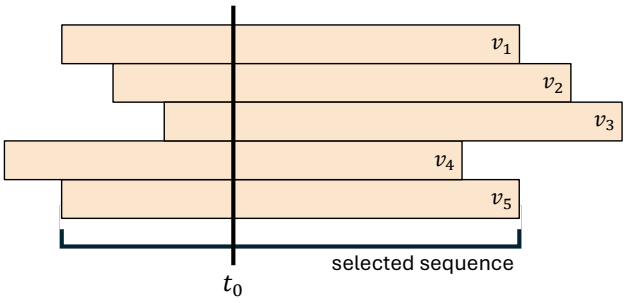

Figure S1. Temporal alignment of predicted trajectories from different viewpoints at t_0. The sequence used as input to the next optimization step is the sequence with the most overlap between the source trajectories.

S1.1. Viewpoint Sampling

Our proposed approach for approximately multi-view consistent video generation is based on the assumption that there are no large viewpoint changes between subsequent optimization steps. The viewpoint sampling strategy has to reflect this constraint while still enabling guidance from views all around the object. To do so, we start at an anchor viewpoint and sample viewpoints on the sphere around the scene center, where we first sample two endpoints lying at the maximum azimuth change $c_{\text{azm,anchor}} \pm m_{\text{azm}}$, with an elevation deviating from $c_{\text{elv,anchor}}$ by at most m_{elv}, and a distance change of at most $m_{\text{dist}} \cdot c_{\text{dist,anchor}}$, both sampled uniformly. We then spread out n_c views uniformly along the path on the sphere from the anchor point to each endpoint. Finally, we disturb each point's azimuth, elevation, and distance with a small noise sampled using standard deviations $\sigma_{\text{azm}}, \sigma_{\text{elv}}, \sigma_{\text{dist}}$.

S1.2. Warping using Optical Flow

Before feeding the previously generated video v_{s-1} through the video diffusion model encoder and using it as motion prior for the generation of a video from the new viewpoint (see Fig. 2), we aim to warp it to resemble the looks of a video taken from the new viewpoint as closely as possible.

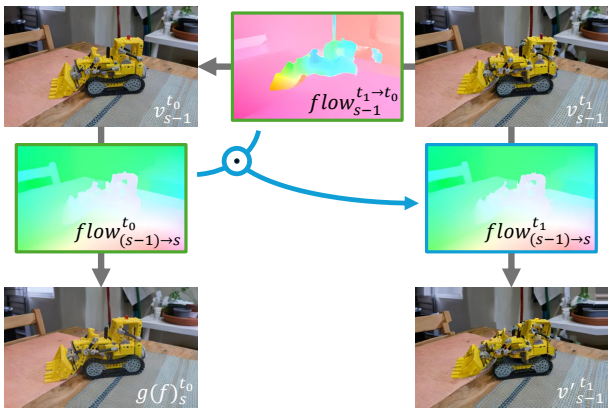

Figure S2. Warping of previously generated video frames v_{s-1} to resemble a video captured from the viewpoint at the next optimization step s. First, optical flow between the video frames of v_{s-1} is computed that is subsequently composed with the optical flow computed between the renderings of the static 3D scene from the two viewpoints at $s-1$ and s. The composed flow is then used to warp the video frames of $v^{t_i}_{s-1}$ to their respective equivalents in the new video $v'^{t_i}_{s-1}$ that is used as motion prior for the video diffusion model.

To do so, one possible approach is to estimate correspondences between the renderings of the given, static 3D scene. Using these correspondences, the optimal homography can be computed, which can subsequently used for warping all frames of the previous video using a perspective projection. While this approach seems reasonable, it comes with the limitation that larger camera pose changes will result in heavy distortions and unrealistic results. Additionally, fore- and background objects are not taken care of separately.

To resolve these issues, we make use of an off-the-shelf optical flow estimation method [48], which we use to compute the optical flow $flow^{t_0}_{(s-1)\to s}$ between the static scene renderings at steps $s-1$ and s, as well as the optical flow $flow^{t_i \to t_0}_{s-1}$ from other frames in the previous video $v^{t_i \neq 0}_{s-1}$ to the static scene rendering $v^{t_0}_{s-1}$. Finally, we remap the optical flow $flow^{t_0}_{(s-1)\to s}$ using $flow^{t_i \to t_0}_{s-1}$ to get $flow^{t_i}_{(s-1)\to s}$, which is used to warp video frame $v^{t_i}_{s-1}$, see Fig. S2.

S1.3. Optimization-based Baseline

We model the deformations of the scene by optimizing a neural network f_θ that maps input coordinates $x \in \mathbb{R}^3$ and a time $t \in \mathbb{R}$ to a change in position $\delta x \in \mathbb{R}^3$, and optionally a change in rotation $\delta q \in \mathbb{R}^4$ and scaling $\delta s \in \mathbb{R}^3$. We thus

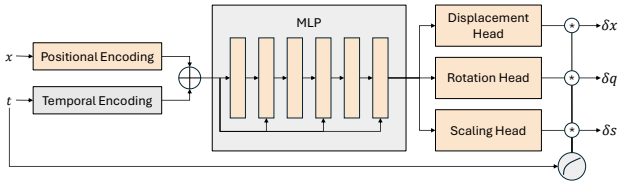

Figure S3. Architecture of the neural deformation field used in our optimization-based baseline. Learnable components in orange.

use the 3D scene initialization as canonical representation, i.e., the displacement, rotation, or scaling of a Gaussian at position $x = \mu_i$ in the static 3D scene at time t is obtained through querying the deformation field $f_\theta(x, t)$.

We employ a multi-resolution hash encoding for spatial input coordinates x [31], as well as random Fourier features as encoding for t [46]. We use a standard MLP with separate heads for the prediction of displacement, rotation, and scale changes, where we multiply outputs with $t^{0.35}$, following previous works [27], to fix the scene initialization at $t = 0$.

We further use a similar deformation technique as presented in Sec. 3.3, where we first sample a set of anchor points using k-Means, that are used to query the deformation field, and motion is subsequently transferred to all 3D Gaussians using the presented techniques in Sec. 3.3 to get smoother deformations.

We use our approximately multi-view consistent videos as guidance, where we employ standard rendering-based losses, i.e. L_1, D-SSIM, and LPIPS, as well as several regularization terms to promote rigid motion [18] and preserve momentum [11], as well as local isometries [29]. Additionally, we make use of an optical flow supervision signal following Gao et al. [12].

S1.4. DreamGaussian4D Baseline

DreamGaussian4D [38] is a video-to-4D method that works by first generating a static 3D asset using multi-view SDS or the feed-forward LGM model [47]. This 3D model is then deformed, guided by SDS with a multi-view image diffusion model that is conditioned on the frames of a guidance video from an anchor viewpoint. To enable a comparison to our method, we use the same 3DGS initialization and the same first sampled diffusion videos for both methods. Further, if a mask for the moving 3D objects is given, we apply it to the 3DGS scene and only use this object as input to DreamGaussian4D, which is necessary as the method is based only on single objects without backgrounds and will otherwise collapse the backgrounds. At test time, we can then add the background Gaussians back into the 3D scene. Importantly, we do not use the second stage of their pipeline, which extracts per-frame meshes and further refines textures on them using SDS with a video diffusion model, as we cannot integrate 3D meshes into the full 3DGS scene.

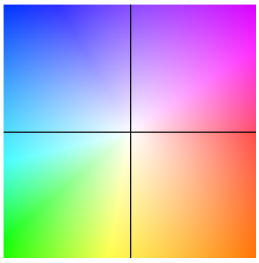

Figure S4. Colormap used for flow visualization in Fig. 7 as proposed by Baker et al. [5].

S1.5. Hyperparameters

We linearly decrease the noise level from 0.75 to 0.2 for the diffusion model inputs and similarly decrease the latent interpolation weight $\lambda_{\text{previous}}$ from 0.6 to 0.0. We use 40 de-noising steps for video generation from new viewpoints. We estimate the motion of every 3D Gaussian from its nearest $n \in [50, 150]$ anchor trajectories, where we increase n over time as more and more anchor trajectories are added. For this, we generally use a high temperature τ (Eq. (4)) but experiment with different weighting strategies. We note that we tracked 1600 points per video originally, but this number is usually reduced to about 600 valid trajectories that lie within the object bounding box at t_0 and that have consistent tracking. Due to the high memory consumption of the used video diffusion model and limited computing resources, we limit our experiments to the generation of $n = 8$ frame guidance videos.

In the comparison against our baseline method, we optimize the deformation field for 14,000 steps. We increase the regularization level over the duration of the optimization and use both the rendering- and the optical flow-based losses. In addition, we reduce the latent interpolation from 0.5 to 0.0 and the noise level from 0.75 to 0.4 throughout the optimization. Every 4,000 steps, we sample new guidance videos from 15 viewpoints and alternate between these viewpoints during the optimization.

S2. Additional Results

S2.1. Quantitative Ablation

The quantitative evaluation of generative models is challenging due to missing ground truth. While comparisons with target data distributions can sometimes be computed for 2D or 3D generative models, this is impossible for our method, which would require a collection of ground-truth motions for a given 3D scene. In our case, we believe that a possible metric that can be used is the CLIP similarity score, which we present and extend for measuring coherence among video frames in the next paragraph.

CLIP Similarity Radford et al. [37] presented CLIP, a vision-language model that is trained to align the represen-

Table S1. Quantitative results for several ablated versions of our method on the Mip-NeRF 360 LEGO bulldozer scene. We note that this evaluation mainly shows the insufficiency of any single quantitative metric. The qualitatively best result (see Fig. 7) does not outperform the other versions on any single one of the metrics but does perform the most consistently across all categories.

Metric	Ours	No MV Diffusion	No 2D-3D lifting	Rigid motion estimation	Fewer interpolation anchors	More interpolation anchors
Motion Amount (rank)	3	4	5	2	1	6
Displacement (10^{-4})	1.84	1.83	1.82	2.02	**2.09**	1.54
Geometry / Physics (∅ rank)	3.00	2.50	4.75	4.25	5.25	1.25
Rigidity (10^{-5})	1.70	1.52	5.12	2.46	3.24	**1.15**
Momentum (10^{-5})	3.92	4.02	14.44	4.32	4.62	**3.36**
Isometry (10^{-5})	3.11	2.85	6.92	5.10	5.72	**2.03**
Rotation similarity (10^{-4})	4.00	3.35	**0.01**	4.31	8.04	2.65
Appearance (∅ rank)	2.50	3.50	6.00	3.00	3.00	3.00
$\text{CLIP}_{\text{text}}$ (10^{-2})	33.06	33.02	29.48	**33.14**	33.07	32.65
$\text{CLIP}_{\text{temporal}}$ (10^{-2})	99.68	99.67	96.90	99.63	99.64	**99.72**
Rank over all categories	**1 (2.83)**	4 (3.33)	6 (5.25)	2 (3.08)	2 (3.08)	5 (3.42)

tations of a vision encoder CLIP_V with those of a text encoder CLIP_T. It is possible to use this model as an evaluation method, as we can take the cosine similarity computed between the features from the image and the text encoder for generated 2D outputs given a text prompt.

A straightforward metric to measure text alignment is the averaging over multiple frames taken from multiple viewpoints:

$$\begin{aligned}\text{CLIP}_{\text{text}}(f,s,p) = \mathbb{E}_{v\sim\mathcal{V},t\sim[0,1]} \\ \left[\cos\left(\angle(\text{CLIP}_V(g(f(s,p);v,t)),\text{CLIP}_T(p))\right)\right],\end{aligned} \tag{7}$$

where f is the optimized 4D-scene taking a static 3D scene s and a text prompt p as input, and $g(\cdot;v,t)$ is the rendering of the 4D-scene at timestep t and from viewpoint v.

In the following, we propose a second metric to capture the temporal coherence of rendered videos. For that, we average the similarity of the extracted image embeddings over pairs of successive video frames. Coherency in successive frames can thus be measured, which also is a good measure for the temporal coherence of the full video when averaged over all frame pairs of the generated video:

$$\begin{aligned}\text{CLIP}_{\text{temporal}}(f,s,p) = \mathbb{E}_{v\sim\mathcal{V},t\sim[0,1)} \\ [\cos(\angle(\text{CLIP}_V(g(f(s,p);v,t)), \\ \text{CLIP}_V(g(f(s,p);v,t+\delta t)))],\end{aligned} \tag{8}$$

where δt is the difference between two video frames in practice. We note that this metric is maximized with static scene renderings, which need to be considered during evaluation.

Regularization Terms as Metrics Besides evaluating the visual quality and appearance of dynamic scenes using CLIP losses, we argue that reporting the scores for some of the regularization terms can be used to determine the actual temporal and geometric consistency of scenes. As such, regularization terms are based on the idea of steering the optimization towards animations that are more plausible; they can also be used to validate the optimized results.

In our evaluation in Tab. S1, we thus report the scores for four regularization terms proposed by previous works, that measure local rigidity [29], momentum preservation [11], local isometry preservation [29], and local rotation similarity [29]. We note, however, that these scores are minimized (best possible result) for zero motion. Thus, these metrics are also not sufficient as stand-alone tools to quantify the quality of generated motion.

Finally, we also report the average displacement of the 3D Gaussians between all timesteps. As we realize that bringing motion into the scenes is often challenging, we mark higher values for these categories as better. However, as before, this metric alone can not quantify the quality of the generated motion, as a diverging scene, where 3D Gaussians are moved freely in the space, would have a very high value.

As can be seen in Tab. S1, the qualitatively best result does not top the quantitative rankings for any single metric. It is, however, the best method when averaging performance across all three evaluation categories: motion amount, geometry and physics, and appearance.

Analyzing the results further, we can see parallels with our qualitative analysis. When no multi-view consistent diffusion is used, performance deteriorates slightly, while using an optimization-based baseline leads to significantly worse results. Contrarily, the results when using rigid motion estimation are worse in terms of geometric metrics, i.e., rigidity, than when using linear interpolation, as analyzed in Sec. 4.4.

We also see that using fewer anchor trajectories for motion estimation leads to more average displacement, but with strong losses in rigidity, isometry and also appearance,

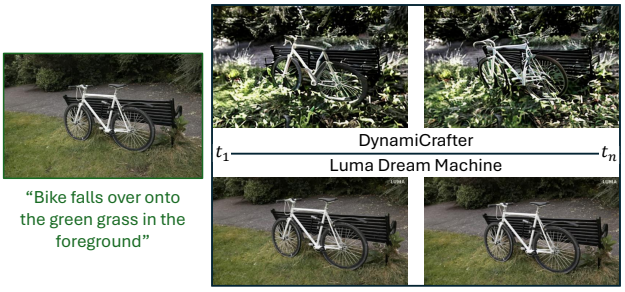

Figure S5. Failed generations of the video diffusion model. While open video diffusion models like DynamiCrafter [58] sometimes result in generation of artifacts when queried with OOD-samples, even more closed-source models like Luma Dream Machine struggle with realistic generation of the desired motion.

indicating a stronger scattering of the 3D Gaussians. Using more anchor trajectories for the deformation estimation leads to opposite behavior, where motion is more rigid, but on average less movement is generated due to the smoothing effect of using too many anchors.

S3. Limitations

As outlined in Sec. 5, our proposed method exhibits several limitations. In this section, we will provide more details on these shortcomings.

While we are able to perform faithful deformations of given 3D scenes, our method is currently mainly limited by the guiding video diffusion model. More specifically, current *open* video diffusion models lack camera control and are often inconsistent in their generations, both for the same view with different random seeds and for multi-view generation with the same seed. As main limitation, we found that the text conditioning often does not succeed and inappropriate motion is generated. Additionally, resulting videos can exhibit artifacts, or heavily diverge from the given image condition to resemble more in-distribution data. In Fig. S5, we show a examples for these video diffusion model failures.

Another limitation of our method is that it is currently unable to compensate for camera motion in the guidance videos. While prompting (e.g., appending "static camera"), or using explicit camera-posed video diffusion models can help alleviate this problem, a more sophisticated solution would be to estimate camera poses before projecting motion from 2D videos to 3D, using similar techniques as, e.g., proposed in monocular dynamic reconstruction works [25].

Finally, as mentioned in Sec. 5, the proposed method only deforms given 3D scenes without adding or removing Gaussians. This can result in the emergence of "holes", as can be seen in Fig. 7 for the toy bulldozer scenes. Another limitation that can be observed in these scenes is the reliance on object masks for deformation, as otherwise neighboring Gaussians can be easily deformed together with the object itself. This is caused by the point cloud nature of the 3DGS representation, where there is no clear notion on which Gaussians belong to the same object. However, as mentioned before, obtaining these masks is possible using off-the-shelf open-world 3D segmentation models [36].

S4. Ethical Considerations

Finally, we would like to briefly address some ethical considerations in connection with our method. While the current results do not yet harbor potential dangers due to the brevity of the generated motion, it should not go unmentioned that the presented method, like all generative models, harbors the danger of deception, e.g., by means of deep fakes. While there are specialized methods for animating people and faces [1, 20, 33, 41] that could be more dangerous in these aspects, our method is still not without danger, as it can be applied to all kinds of objects and scenes. It is therefore necessary that such methods are developed and used responsibly, with clear guidelines and oversight to prevent misuse.

Figure S6. Approximately multi-view consistent video generations, starting from one video generated from an anchor viewpoint. The 3D scene is kept static between all generation steps to demonstrate the effect of the proposed latent interpolation. See the supplementary videos that also contain examples without the proposed latent interpolation for a comparison.