Feed-Forward Bullet-Time Reconstruction of Dynamic Scenes from Monocular Videos

Hanxue Liang[1,2]*, Jiawei Ren[1,3]*, Ashkan Mirzaei[1,4]*, Antonio Torralba[1,5], Ziwei Liu[3], Igor Gilitschenski[4], Sanja Fidler[1,4,6], Cengiz Oztireli[2], Huan Ling[1,4,6], Zan Gojcic[1]†, Jiahui Huang[1]†

[1]NVIDIA, [2]University of Cambridge, [3]Nanyang Technological University, [4]University of Toronto, [5]MIT, [6]Vector Institute

https://research.nvidia.com/labs/toronto-ai/bullet-timer/

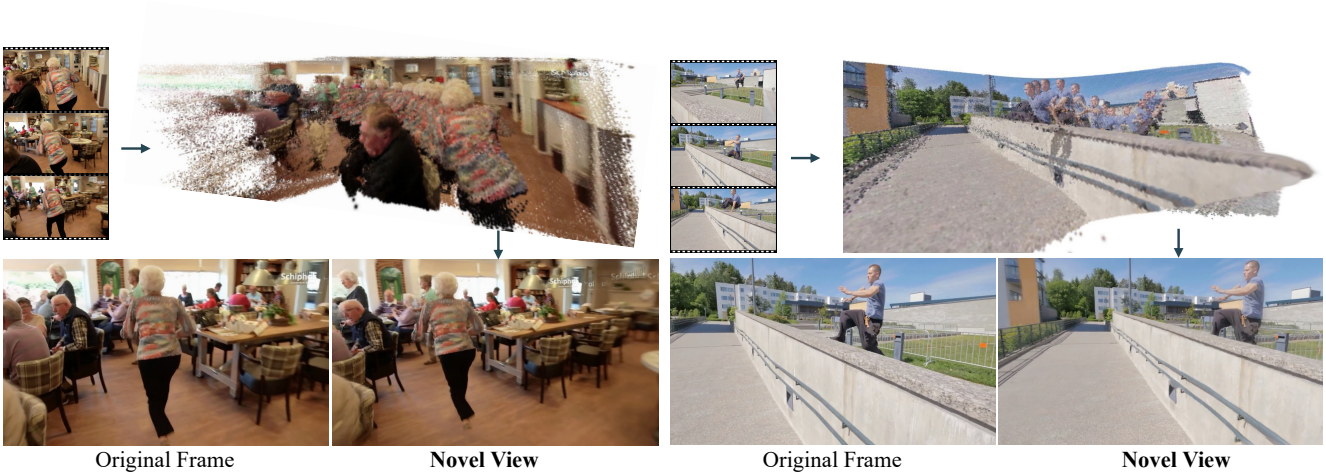

Figure 1. **Our method** takes a monocular video as input and reconstructs a 3D Gaussian Splatting (3DGS) [29] representation at any desired timestamp in a feed-forward fashion.

Abstract

Recent advancements in static feed-forward scene reconstruction have demonstrated significant progress in high-quality novel view synthesis. However, these models often struggle with generalizability across diverse environments and fail to effectively handle dynamic content. We present BTimer (short for BulletTimer), the first motion-aware feed-forward model for real-time reconstruction and novel view synthesis of dynamic scenes. Our approach reconstructs the full scene in a 3D Gaussian Splatting representation at a given target ('bullet') timestamp by aggregating *information from all the context frames. Such a formulation allows BTimer to gain scalability and generalization by leveraging both static and dynamic scene datasets. Given a casual monocular dynamic video, BTimer reconstructs a* bullet-time[1] *scene within 150ms while reaching state-of-the-art performance on both static and dynamic scene datasets, even compared with optimization-based approaches.*

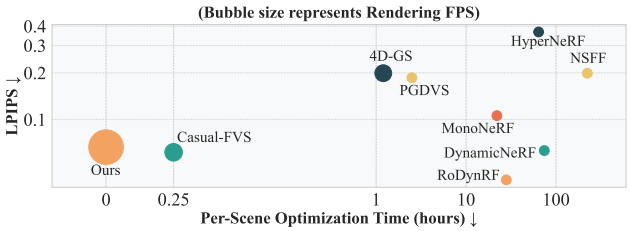

Figure 2. **Rendering quality vs. speed.** Our model can reconstruct and render dynamic scenes at a much faster speed than existing approaches with a competitive quality. Numbers are reported on NVIDIA Dynamic Scene Dataset [75]

1. Introduction

Multi-view reconstruction and novel-view synthesis are long-standing challenges in computer vision, with numerous applications ranging from AR/VR to simulation and content creation. While significant progress has been made in reconstructing static scenes, dynamic scene reconstruc-

*/†: Equal contribution/advising.

[1]In this paper, we define *bullet-time* as the instantiation of a 3D scene *frozen* at a given/fixed timestamp t.

tion from monocular videos remains challenging due to the inherently ill-posed nature of reasoning about dynamics from limited observations [36].

Current methods for static scene reconstruction can be broadly divided into two categories: *optimization-based* [45, 48] and *learning-based* [59, 79] approaches. However, extending both of these to dynamic scenes is not straightforward. To reduce the ambiguities of scene dynamics, many optimization-based methods aim to constraint the problem with data priors such as depth and optical flow [33, 36, 37, 68]. However, balancing these priors with the data remains challenging [34, 63]. Moreover, per-scene optimization is time-consuming and thus difficult to scale.

On the other hand, learning-based approaches [7, 10, 12, 25, 53] are trained on large-scale datasets to directly predict reconstructions in a feed-forward manner, thereby learning strong priors from data. These inherent priors could help resolve ambiguities due to complex motion, but none of these approaches have yet been extended to dynamic scenes. This limitation stems from both the complexity of modeling dynamic scenes and the lack of 4D supervision data. The only feed-forward dynamic reconstruction model [53] is thus trained on synthetic object-centric datasets, requires fixed camera viewpoints and multiview supervision, and cannot generalize to real-world scene scenarios.

In this work, we aim to answer the question: *How can one build a feed-forward reconstruction model that can handle dynamic scenes effectively?* We build upon the recent success of the pixel-aligned 3D Gaussian Splatting (3DGS [29]) prediction models [79] and propose a novel *bullet-time* formulation for feed-forward dynamic reconstruction. The core idea is simple yet effective: we add a *bullet-time* embedding to the context (input) frames, indicating the desired timestamp for the output 3DGS representation. Our model is trained to aggregate the predictions of context frames to reflect the scenes at the *bullet* timestamp, yielding a spatially complete 3DGS scene. This design not only naturally unifies the static and dynamic reconstruction scenarios, but also enables our model to become implicitly motion-aware while learning to capture scene dynamics. The proposed formulation **(i)** allows us to pretrain our model on large amounts of *static* scene data, **(ii)** scales effectively across datasets, without being constrained by duration and frame rates of input videos, and **(iii)** outputs volumetric video representations that inherently support multiple viewpoints. In the presence of fast motions, we introduce an additional Novel Time Enhancer (NTE) module to predict the intermediate frames before feeding them to the main model.

In summary, we present BTimer, the first feed-forward reconstruction model for dynamic scenes using a *bullet-time* formulation. Our method is highly efficient: feed-forward inference with 12 context frames of 256×256 resolution only costs 150ms on a single GPU, and the output 3DGS can be rendered in real-time. BTimer is trained using a large curated dataset comprising both static and dynamic scenes, achieving competitive results on various different reconstruction benchmarks, even surpassing many expensive per-scene optimization-based methods, as illustrated in Fig. 2.

2. Related Work

Dynamic 3D Representations. Depending on the tasks at hand, typical choices of 3D representations include voxels [40, 66], implicit fields/NeRFs [44–46], and point clouds/3D Gaussians [2, 29]. Representing dynamics on top has an even larger design space: One existing line of works directly builds a '4D' representation to enable feature queries at arbitrary positions and timestamps from an implicit field [6, 20] or via marginalization at a given step [17, 72], with the extensibility to higher dimensions such as material [5]. Another line of work first defines a canonical 3D space, and learns a deformation field to warp the canonical space to the target frame. Such a deformation field can be parametrized via embedding graphs [42], control points [26], motion scaffolds [34], rigid transformation modes [63] or bounding boxes [11, 84], explicit scene flows [38], or implicit flow fields [19, 52, 67, 73]. While these methods learn additional information about shape correspondences, their performance heavily relies on the quality and topology of the canonical space.

Novel View Synthesis. For tasks that require a relatively smaller view extrapolation, the problem of novel view synthesis can be tackled without explicit 3D geometry in the loop, using depth warping [75] or multi-plane images [60]. Otherwise, the study of novel view synthesis of dynamic scenes [48, 51] is mainly on (1) effectively optimizing the 3D representation through input images through monocular cues [34, 35, 37, 63] or geometry regularizations [41, 47], and (2) being able to render fast with grids [3], local-planes [32], or dynamic 3D Gaussians [64] formulation. Our method aims to provide a dynamic representation that is fast to build within hundreds of milliseconds while reaching competitive rendering quality as the above optimization-based methods.

Feed-forward Reconstruction Models. In many applications where the reconstruction speed is crucial, most optimization-based reconstruction methods become less preferable. To this end, one line of methods uses data priors to serve as an initialization [18] or guidance [9, 59], but they still need few-shot optimization to refine the results [12, 62]. Another type of work that starts to emerge is fully feed-forward models that directly regress from 2D images to 3D, represented as either triplanes [25], 3D Gaussians [7, 10, 79], sparse voxels [54], or latent tokens [27].

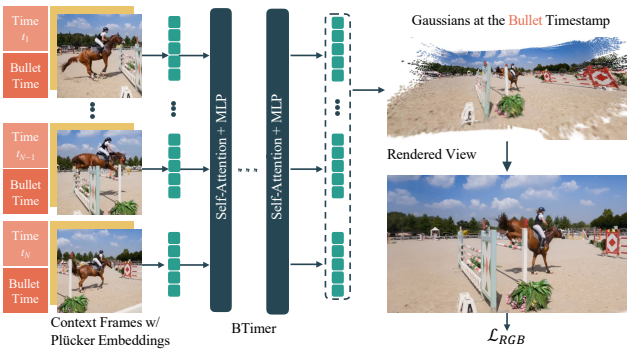

Figure 3. **BTimer.** The model takes as input a sequence of context frames and their Plücker embeddings, along with the context timestamp and target ('bullet') timestamp embeddings. It then directly predicts the 3DGS representation at the bullet timestamp.

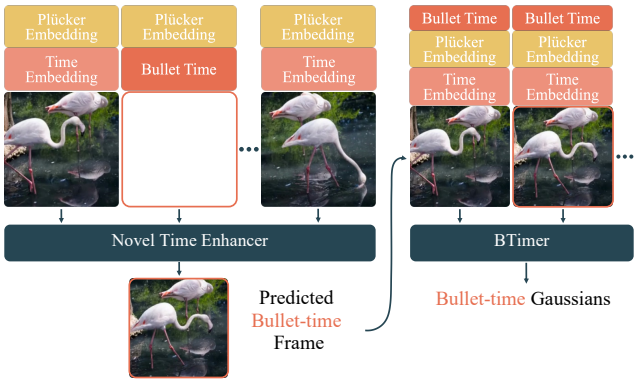

Figure 4. **NTE Module.** It takes as input the target bullet time embedding, target pose, as well as adjacent frames, and directly predicts corresponding RGB values. The predicted frame is then used in BTimer as *bullet* frame for novel time reconstruction.

Crucially, while feed-forward reconstruction models for static scenes have seen development, the extension to dynamic scenes is still challenging. Existing methods either require hard-to-acquire consistent video depth as input [81], do not support rendering [78], or only work on object-scale data [53]. In contrast, our method supports reconstructing from a monocular video containing dynamic scenes in a fully feed-forward manner, and is able to render at arbitrary viewpoints and timestamps.

3. Method

Overview. Given a monocular video (image sequence) represented by $\mathcal{I} = \{\mathbf{I}_i \in \mathbb{R}^{H\times W\times 3}\}_{i=1}^{N}$ with N frames of width W and height H, along with known camera poses $\mathcal{P} = \{\mathbf{P}_i \in \mathbb{SE}(3)\}_{i=1}^{N}$, intrinsics, and corresponding timestamps $\mathcal{T} = \{t_i \in \mathbb{R}\}_{i=1}^{N}$, our goal is to build a feed-forward model capable of rendering high-quality novel views at arbitrary timestamps $t \in [t_1, t_N]$.

The core of our approach is a transformer-based *bullet-time* reconstruction model, named **BTimer**, that takes in a subset of frames $\mathcal{I}_c \subset \mathcal{I}$ (denoted as *context frames*) along with their corresponding poses $\mathcal{P}_c \subset \mathcal{P}$ and timestamps $\mathcal{T}_c \subset \mathcal{T}$, and outputs a complete 3DGS [29] scene frozen at a specified *bullet* timestamp $t_b \in [\min_{\mathcal{T}_c}, \max_{\mathcal{T}_c}]$ (§ 3.1). Iterating over all $t_b \in \mathcal{T}$ results in a full video reconstruction represented by a sequence of 3DGS. We further introduce a Novel Time Enhancer (NTE) module that synthesizes interpolated frames with timestamps $t \notin \mathcal{T}$ (§ 3.2). The output of the NTE module is used along with other context views as input to the bullet-time model to enhance reconstruction at arbitrary intermediate timestamps. To effectively train our model, we carefully design a learning curriculum (§ 3.3) that incorporates a large mixture of datasets containing both static and dynamic scenes, to enhance motion awareness and temporal consistency of our models.

3.1. BTimer Reconstruction Model

Model Design. Inspired by [79], our BTimer model uses a ViT-based [15] network as its backbone, consisting of 24 self-attention blocks with LayerNorms [4] applied at both the beginning and the end of the model. We divide each input context frame $\mathbf{I}_i \in \mathcal{I}_c$ into 8×8 patches, which are projected into feature space $\{\mathbf{f}_{ij}^{\text{rgb}}\}_{j=1}^{HW/64}$ using a linear embedding layer. The camera Plücker embeddings [70] derived from the camera poses $\mathbf{P}_i \in \mathcal{P}_c$ and the time embeddings (detailed later) are processed similarly to form the camera pose features $\{\mathbf{f}_{ij}^{\text{pose}}\}$ and the time features $\{\mathbf{f}_i^{\text{time}}\}$ (shared for all patches j). These features are added together to form the input tokens for the patches of the context frame $\{\mathbf{f}_{ij}\}_{j=1}^{HW/64}$, where $\mathbf{f}_{ij} = \mathbf{f}_{ij}^{\text{rgb}} + \mathbf{f}_{ij}^{\text{pose}} + \mathbf{f}_i^{\text{time}}$. The input tokens from all context frames are concatenated and fed into the Transformer blocks.

Each corresponding output token $\mathbf{f}_{ij}^{\text{out}}$ is decoded into 3DGS parameters $\mathbf{G}_{ij} \in \mathbb{R}^{8\times 8\times 12}$ using a single linear layer. Each 3D Gaussian is paramaterized by its RGB color $\mathbf{c} \in \mathbb{R}^3$, scale $\mathbf{s} \in \mathbb{R}^3$, rotation represented as unit quaternion $\mathbf{q} \in \mathbb{R}^4$, opacity $\sigma \in \mathbb{R}$, and ray distance $\tau \in \mathbb{R}$, resulting in 12 paramaters per Gaussian. The 3D position of each Gaussian $\boldsymbol{\mu} \in \mathbb{R}^3$ is obtained through pixel-aligned unprojection as $\boldsymbol{\mu} = \mathbf{o} + \tau\mathbf{d}$, where $\mathbf{o} \in \mathbb{R}^3$ and $\mathbf{d} \in \mathbb{R}^3$ are the ray origin and direction obtained from $\mathbf{P}_i$.

Time Embeddings. The aforementioned input time feature $\mathbf{f}_i^{\text{time}}$ is obtained from: **(i) context** timestamp t_i that is separate for each context frame $\mathbf{I}_i$, and **(ii) bullet** timestamp t_b that is shared across all context frames i. Both timestamp scalars are encoded using standard Positional Encoding (PE) [61] with sinusoidal functions, and then passed through two linear layers to obtain the features $\mathbf{f}_i^{\text{ctx}}$ and $\mathbf{f}_i^{\text{bullet}}$ respectively. Finally, we set $\mathbf{f}_i^{\text{time}} = \mathbf{f}_i^{\text{ctx}} + \mathbf{f}_i^{\text{bullet}}$.

Supervision Loss. Our model is supervised only by losses

defined in the RGB image space, bypassing the need for any source of 3D ground truth that is hard to obtain for real data. The final loss is a weighted sum of Mean Squared Error (MSE) loss and Learned Perceptual Image Patch Similarity (LPIPS) [80] loss between the images rendered from the 3DGS output and the ground-truth image:

$$\mathcal{L}_{\text{RGB}} = \mathcal{L}_{\text{MSE}} + \lambda \mathcal{L}_{\text{LPIPS}}, \tag{1}$$

with $\lambda = 0.5$.

Careful selection of input context frames and corresponding supervision frames (at the bullet timestamp) during training is essential for stable training and good convergence. In practice, we find the combination of the following two strategies particularly effective: **(i) In-context Supervision** where the supervision timestamp is randomly selected from the context frames, encouraging the model to accurately localize and reconstruct the context timestamps. For multi-view video datasets, images from additional viewpoints can also contribute to the loss. **(ii) Interpolation Supervision** where the supervision timestamp lies between two adjacent context frames. This forces the model to interpolate the dynamic parts while maintaining consistency for the static regions. The interpolation supervision significantly impacts our final performance (*cf*. § 4.4 for details); without it, the model falls into a local minima by positioning the 3D Gaussians close to the context views but hidden from other views.

Inference. Our *bullet-time* formulation makes it straightforward to reconstruct a full video, which only involves iteratively setting the bullet timestamp t_b to every single timestamp in the video, and can be done efficiently in parallel. For a video longer than the number of training context views $|\mathcal{I}_c|$, at timestamp t, apart from including this exact timestamp and setting $t_b = t$, we uniformly distribute the remaining $|\mathcal{I}_c| - 1$ required context frames across the whole duration of the video to form the input batch with $|\mathcal{I}_c|$ frames.

3.2. Novel Time Enhancer (NTE) Module

While our BTimer model can already reconstruct the 3DGS representation for all observed timestamps, we notice that forcing it to reconstruct at a novel intermediate timestamp, *i.e*. performing interpolation at $t_b \notin \mathcal{T}$, leads to suboptimal results. In such cases, the exact bullet-time frame cannot be included in the context frames as it does not exist. Our model specifically fails to predict a smooth transition between adjacent video frames when the motion is complex and fast. This is mainly caused by the inductive bias of pixel-aligned 3D Gaussian prediction. To mitigate this issue, we propose a *3D-free* Novel Time Enhancer (NTE) module that directly outputs images at given timestamps, which are then used as input to our BTimer model, as illustrated in Fig. 4.

NTE module Design. The design of this module is largely inspired by the very recent decoder-only LVSM [27] model. Specifically, NTE copies the same ViT architecture from the BTimer model, but the time features of input context tokens only encode their corresponding context timestamps (*i.e*. we set $\mathbf{f}_i^{\text{time}} = \mathbf{f}_i^{\text{ctx}}$). Additionally, we concatenate extra target tokens to the input tokens, which encode the target timestamp and the target pose for which we want to generate the RGB image. Following [27], we use QK-norm to stabilize training. Implementation-wise we apply an attention mask that masks all the attention to the target tokens, so KV-Cache (*cf*. [50]) can be used for faster inference. From the output of the Transformer backbone, we only retain the target tokens, which we then unpatchify and project to RGB values at the original image resolution using a single linear layer. The interpolation model is trained with the same objective as the main BTimer model (see § 3.1), but the output image is directly decoded from the network and not rendered from a 3DGS representation.

Integration with BTimer. While the NTE module can be used on its own to generate novel views, we empirically find the novel-*view*-synthesis quality to be inferior (§ 4.4). We hence propose to integrate it with our main BTimer model. To reconstruct a bullet-time 3DGS at $t_b \notin \mathcal{T}$, we first use NTE to synthesize $\mathbf{I}_b$ at the timestamp t_b, where the target pose $\mathbf{P}_b$ is linearly interpolated from the nearby context poses in $\mathcal{P}$, and the context frames are chosen as the nearest frames to t_b. To accelerate the inference of the interpolation model, we use the KV-Cache strategy. In practice we observe that the interpolation model adds negligible overhead to the overall runtime.

3.3. Curriculum Training at Scale

One important lesson people have learned from training deep neural networks is to scale up the training [1, 56], and the model's generalizability is largely determined by the data diversity. Since our bullet-time reconstruction formulation naturally supports both static (by equalizing all elements in $\mathcal{T}$) and dynamic scenes, and requires only RGB loss for weak supervision, we unlock the potential of leveraging the availability of numerous static datasets to pretrain our model. We hence aim to train a *kitchen-sink* reconstruction model that is *not specific* to any dataset, making it generalizable to both static and dynamic scenes, and capable of handling objects as well as both indoor and outdoor scenes. This is in contrast to, *e.g*., GS-LRM [79] or MVSplat [10] where one needs different models in different domains.

Notably, we apply the following training curriculum to BTimer and the NTE module separately, but during inference they are used jointly as explained in § 3.2.

Stage 1: Low-res to High-res Static Pretraining. To obtain a more generalizable 3D prior as initialization, we first pretrain the model with a mixture of *static* datasets.

Time embedding will not be used in this stage. The collection of datasets covers object-centric (Objaverse [14]) and indoor/outdoor scenes (RE10K [83], MVImgNet [77], DL3DV [39]). The datasets cover both the synthetic and real-world domains and consist of 390K training samples. We normalize the scales of different datasets to be bounded roughly in a 10^3 cube. Due to the complex data distribution, our training starts from a low-resolution few-view setting that reconstructs on 128×128 resolution from $|\mathcal{I}_c| = 4$ context views. To further increase the reconstruction details, we fine-tune the model from 128×128 by first increasing the image resolution to 256×256, and then fine-tune to 512×512.

Stage 2: Dynamic Scene Co-training. After the training on static scenes, we start fine-tuning the model along with time embedding projection layers on dynamic scenes with available 4D data that contains monocular or multi-view synchronized videos. We leverage Kubric [23], PointOdyssey [82], DynamicReplica [28] and Spring [43] datasets for training. Due to the scarcity of 4D data, during this stage we keep the static datasets for co-training which provides more multi-view supervision and stabilizes the training. Additionally, we build a customized pipeline to label the camera poses from Internet videos (detailed below), and add them to our training set to further enhance the model's robustness towards real-world data.

Stage 3: Long-context Window Fine-tuning. Including more context frames is vital when reconstructing long videos. Therefore, as a final stage, we increase the number of context views from $|\mathcal{I}_c| = 4$ to $|\mathcal{I}_c| = 12$ to cover more frames. Note that this stage does not apply to NTE as it only takes nearby frames as input.

Annotating Internet Videos. We randomly select a subset from the PANDA-70M [8] dataset, and cut the videos into short clips with $\sim$20 s duration. We mask out the dynamic objects in the videos with Segment Anything Model [30] and then apply DROID-SLAM [58] to estimate the camera poses. Low-quality videos or annotated poses are filtered out by measuring the reprojection error. The final dataset contains more than 40K clips with high-quality camera trajectories.

4. Experiments

In this section we first introduce necessary implementation details in § 4.1. We evaluate the performance of BTimer extensively on available dynamic scene benchmarks § 4.2, and demonstrate its *backward* compatibility with static scenes § 4.3. Ablation studies are found in § 4.4.

4.1. Implementation Details

Training. Our backbone Transformer network is implemented efficiently with FlashAttention-3 [13] and FlexAttention [24]. We use `gsplat` [74] for robust and scalable 3DGS rasterization since the total number of 3D Gaussians generated by our model can be very large. For BTimer, the numbers of training iterations are fixed to 90K, 90K, and 50K for **Stage 1** training on 128^2, 256^2, and 512^2 resolutions, and are 10K and 5K for **Stage 2** and **Stage 3** dynamic scene training respectively. We use the initial learning rates of 4×10^{-4}, 2×10^{-4} and 1×10^{-4} for the three stages, and apply a cosine annealing schedule to smoothly decay the learning rate to zero. The full training takes $\sim$4 days on 64 NVIDIA A100 GPUs. We use the same training strategy for NTE. The numbers of iterations are 140K, 60K, and 30K for the progressive training in **Stage 1**, and are 20K for **Stage 2**, with the same learning rate schedule as above. As introduced in § 3.3, we use a mixture of multiple datasets for training [14, 23, 28, 39, 43, 77, 82, 83] along with our 40K annotated dataset on PANDA-70M [8]. Note that we make sure that none of the testing scenes we show below is included in the training datasets.

Timing. Our model can be flexibly applied to different resolutions and numbers of context views. BTimer takes 20 ms for 4-view 256^2 reconstruction, 150 ms for the same resolution with 12 views, and 4.2 s for 12-view 512×896 reconstruction. NTE takes 0.44 s seconds for 4-frame 512×896 reconstruction without KV caching.

Model	Rec. Time	PSNR↑	SSIM↑	LPIPS↓
TiNeuVox [19]	0.75 h	14.03	0.502	0.538
NSFF [36]	24 h	15.46	0.551	0.396
T-NeRF [22]	12 h	16.96	0.577	0.379
Nerfies [47]	24 h	16.45	0.570	0.339
HyperNeRF [48]	72 h	16.81	0.569	0.332
PGDVS [81]	3 h†	15.88	0.548	0.340
Depth Warp	–	7.81	0.201	0.678
BTimer (**Ours**)	0.98 s	16.52	0.570	0.338

†: Video-consistent depth estimation step included.

Table 1. **DyCheck iPhone dataset [22] comparison**. 'Rec. Time' is the per-scene reconstruction/processing time needed for each method. We highlight the best, second best and third best.

Model	Rec. Time	Render FPS	PSNR↑	LPIPS↓
HyperNeRF [48]	64 h	0.40	17.60	0.367
DynNeRF [21]	74 h	0.05	26.10	0.082
NSFF [36]	223 h	0.16	24.33	0.199
RoDynRF[41]	28 h	0.42	25.89	0.065
MonoNeRF[59]	22 h	0.05	25.62	0.106
4D-GS [67]	1.2 h	44	21.45	0.199
Casual-FVS [33]	0.25 h	48	24.57	0.081
PGDVS [81]	3 h†	0.70	24.41	0.186
Depth Warp	–	–	12.63	0.564
BTimer (**Ours**)	0.78 s	115	25.82	0.086

†: Video-consistent depth estimation step included.

Table 2. **NVIDIA Dynamic Scene dataset [75] comparison**. The rendering speed is tested on 480×270 resolution. We highlight the best, second best and third best methods.

Figure 5. **Visualizations on DAVIS dataset [49].** We show our renderings on novel combinations of view poses and timestamps, with the correspondending references shown on the left. The lower-left/right corner shows the rendered depth map for each example.

Figure 6. **Qualitative results on DyCheck [21] (left) and NVIDIA dynamic scene [75] (right) benchmarks**.

4.2. Dynamic Novel View Synthesis

4.2.1 Quantitative Analysis

We provide quantitative evaluations on two of the largest dynamic view synthesis benchmarks available. **DyCheck Benchmark [22].** The benchmark includes a DyCheck iPhone dataset that contains 7 dynamic scenes recorded by 3 synchronized cameras. Following the protocol in [22], we take images from the iPhone camera as our context frames and use the frames from the 2 other stationary cameras (totaling 3928 images) for evaluation. Our baselines include per-scene optimization-based methods, *i.e.*, TiNeuVox [19], NSFF [36], T-NeRF [22], Nerfies [47] and HyperNeRF [48]. To the best of our knowledge there is no published method that can be applied without per-scene optimization. We hence compare to direct depth warping and

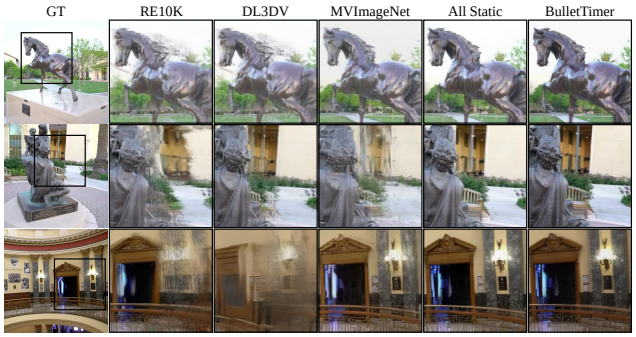

Figure 7. **Qualitative comparison of models trained on different datasets** and evaluated on the out-of-distribution Tanks & Temples benchmark [18].

PGDVS [81]. The latter requires consistent depth estimation as input, which takes hours of GPU hours to estimate.

We report masked Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [65], and LPIPS following the benchmark protocol [22] in Tab. 1, and show visualizations in Fig. 6. Note that since multi-frame inference can run in parallel, for our model we report single-frame reconstruction time regardless of video lengths. It is encouraging to observe that even without per-scene optimization, BTimer achieves a very competitive performance compared to the baselines, ranking 2nd in both SSIM and LPIPS scores. Our model surpasses PGDVS across all 3 metrics without the need of consistent depth estimate. This demonstrates our model's efficiency and strong generalization capability, being capable of providing sharper details and richer textures.

NVIDIA Dynamic Scene Benchmark [75]. NVIDIA Dynamic Scene dataset contains 9 scenes captured by 12 forward-facing synchronized cameras. Following the protocol in DynNeRF [21], we build the input by selecting the frames at different timestamps in a 'round-robin' manner. Then we evaluate the novel view synthesis quality at the first camera view but at different timestamps. We compare against HyperNeRF [48], DynNeRF [21], NSFF [36], RoDynRF [41], MonoNeRF [59], 4D-GS [67], CasualFVS [33] as per-scene optimization baselines.

Our results are shown in Tab. 2 and Fig. 6. Our model demonstrates performance that is competitive or exceeds that of previous optimization-based methods, ranking 3rd among all baselines in terms of PSNR. Compared to the explicit 3DGS-based representation [33, 67], our approach outperforms their performance by 5% on PSNR (25.82dB vs. 24.57dB). In terms of training and rendering speed, NeRF-based methods [21, 36, 59] require multiple GPUs and/or $>$1 day for optimization. Compared to [33, 67], our feed-forward bullet-time formulation is significantly faster, requiring no optimization time and rendering in real-time.

4.2.2 Qualitative Analysis

To assess the performance of our method in real-world scenarios, we select multiple monocular videos from the DAVIS dataset [49] for testing. Camera poses for the videos were estimated using the same annotation technique as detailed in § 3.3. Fig. 5 shows a visualization of the results. Our model demonstrates strong generalization capabilities in real-world captures, producing high-quality, sharp renderings across a variety of objects with complex motions while maintaining robust temporal and multiview consistency. The depth map also demonstrates the correct geometry that we can recover, enabled by the bullet-time reconstruction formulation that allows input frames from large baselines.

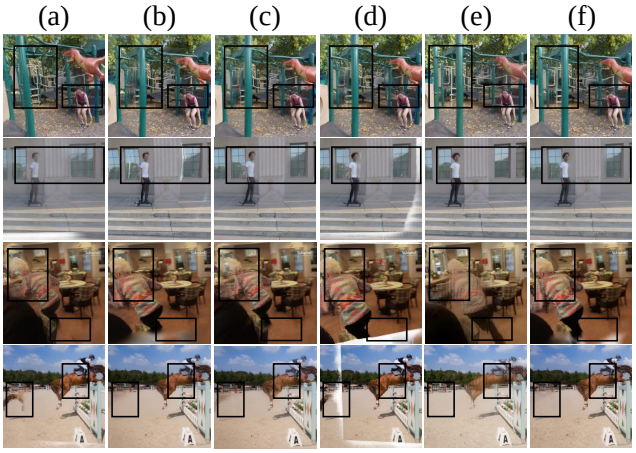

Figure 8. **Ablation results.** (a) model w/o 3D Pretrain, (b) model w/ Re10K only 3D Pretrain, (c) model w/o static Co-train in **Stage 2**, (d) model w/o interpolation supervision, (e) Novel Time Enhancer model, (f) our full model. The upper two scenes are from NVIDIA dataset, and lower two scenes are from DAVIS dataset.

Model	LPIPS↓
GPNR [55]	0.250
PixelSplat [7]	0.142
MVSplat [10]	0.128
GS-LRM [79]	0.114
Ours-Static	0.070
Ours-Full	0.089

(a)

Model	Datasets	LPIPS↓
GS-LRM* [79]	RE10K	0.310
Ours-Static	Objaverse	0.668
	MVImageNet	0.343
	DL3DV	0.278
	All Static	0.093
Ours-Full	+Dynamic	0.093

(b)

Table 3. **Quantitative comparisons on static datasets.** (a) results on the RE10K benchmark [83]; (b) results on the Tanks and Temples benchmark [18]. We highlight the best, second best and third best models. *: Our reproduced results.

4.3. Compatibility with Static Scenes

Although our model is primarily designed to handle dynamic scenes, the formulation and the training strategy enable it to be still backward compatible with static scenes. In this section, we show that the *same* model achieves competitive results on static scenes.

RealEstate10K (RE10K) Benchmark [83]. We evaluate our model on the RE10K dataset and compare with several state-of-the-art models [7, 10, 55, 79]. To ensure comparability with baseline models, we train and test our model using 256 $\times$ 256 resolution. Tab. 3a presents a quantitative comparison on LPIPS, where our static model outperforms all the baselines. Please refer to the Supplement for more comparisons on other metrics and visualizations.

Tanks & Temples Benchmark [18]. We further evaluate our model on an unseen test dataset, the Tanks & Temples [31] subset from the InstantSplat [18] benchmark, which consists of 10 scenes. We use the state-of-the-art

novel view synthesis model [79] as our baseline, reproducing their model since the original code and weights are not publicly available. Additionally, we include our pretrained static model from **Stage 1** as an additional baseline.

To analyze the impact of our mixed-dataset pretraining strategy, we also train single-dataset models using the same training schedule as further baselines. All models utilize 4 context views. Tab. 3b presents quantitative results, demonstrating that our pretrained static model with mixed-dataset training substantially outperforms the single-dataset models, highlighting the crucial role of multi-dataset training for generalization to unseen domains. Even when incorporating the dynamic scene datasets, BTimer achieves performance comparable to our best static models. Fig. 7 provides a qualitative comparison, showing that BTimer consistently generates sharper outputs that closely align with the ground truth.

4.4. Ablation Study

Effect of Context Frames. We visualize the reconstruction results as we progressively add 3DGS predictions from more context frames across multiple different timestamps in Fig. 9, where increasing the number of context frame leads to progressively more complete scene reconstruction. This demonstrates the flexibility of our bullet-time reconstruction formulation: during the inference stage, we can arbitrarily select spatially-distant frames that contribute to a more complete view coverage of the scene.

Effect of Curriculum Training. We show in Fig. 8 the effect of our curriculum training strategy. Without **Stage 1** of pre-training on 3D static scenes, the model struggles to produce results of correct geometry and sharp details. Pretraining on multiple diverse datasets is also crucial, which we demonstrate by *just* training on RE10K dataset, and non-negligible distortions are observed in the results. Similarly, even in **Stage 2** of our curriculum, we still need to co-train on static scenes which provide more multi-view supervisions, thus maintaining the rich details and reasonable geometries.

Effect of Interpolation Supervision. Shown in Fig. 8, the effect of interpolation supervision (introduced in § 3.1) is significant, without which the model tends to produce *white-edge* artifacts. This occurs because in the absence of interpolation loss, the model often generates 3DGS that are positioned too close to the camera with low depth values to *cheat* the loss. In contrast, adding the interpolation supervision requires the model to account for scene dynamics and encourages consistency across multiple views.

Effect of NTE. As demonstrated in Fig. 10, our NTE module enhances the bullet-time reconstruction model's ability to handle scenes with fast or complex motions, largely reducing the ghosting artifacts. Additional video results are provided in the supplementary material. Although 3D-free design enables NTE to handle complex dynamics and produce smooth transitions between adjacent frames, the model struggles to produce novel views that are far from the input camera trajectory (As illustrated in Fig. 8).

Figure 9. **Illustration of bullet-time reconstruction from multiple context frames.** Increased number of frame predictions leads to progressively more complete scene reconstruction.

Figure 10. **Ablation on the NTE module**. The middle frame is in between the 1st frame and the 2nd frame. Results are rendered from the view of the 1st frame.

5. Conclusion

Limitations. Although our method provides competitive novel view synthesis results, the recovered geometry (hence the depth map) is usually not as accurate as the most recent depth prediction models (*e.g.* [71]). Our model also has limited support for view *extrapolation*. Incorporating a generative prior in the loop is a promising direction to pursue in the future.

In this paper we present BTimer, the first feed-forward dynamic 3D scene reconstruction model for novel view synthesis. We present a bullet-time formulation that allows us to train the model in a more flexible and scalable way. We demonstrate through extensive experiments that our model is able to provide high-quality results at arbitrary novel views and timestamps, outperforming the baselines in terms

of both quality and efficiency.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 4

[2] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 696–712. Springer, 2020. 2

[3] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhoefer, Johannes Kopf, Matthew O'Toole, and Changil Kim. Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16610–16620, 2023. 2

[4] Jimmy Lei Ba. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[5] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 2

[6] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2023. 2

[7] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19457–19467, 2024. 2, 7, 13, 14

[8] Tsai-Shien Chen, Aliaksandr Siarohin, Willi Menapace, Ekaterina Deyneka, Hsiang-wei Chao, Byung Eun Jeon, Yuwei Fang, Hsin-Ying Lee, Jian Ren, Ming-Hsuan Yang, et al. Panda-70m: Captioning 70m videos with multiple cross-modality teachers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13320–13331, 2024. 5, 13

[9] Yun Chen, Jingkang Wang, Ze Yang, Sivabalan Manivasagam, and Raquel Urtasun. G3r: Gradient guided generalizable reconstruction. In *European Conference on Computer Vision*, pages 305–323. Springer, 2025. 2

[10] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision*, pages 370–386. Springer, 2025. 2, 4, 7, 13, 14

[11] Ziyu Chen, Jiawei Yang, Jiahui Huang, Riccardo de Lutio, Janick Martinez Esturo, Boris Ivanovic, Or Litany, Zan Gojcic, Sanja Fidler, Marco Pavone, et al. Omnire: Omni urban scene reconstruction. *arXiv preprint arXiv:2408.16760*, 2024. 2

[12] Wenyan Cong, Hanxue Liang, Peihao Wang, Zhiwen Fan, Tianlong Chen, Mukund Varma, Yi Wang, and Zhangyang Wang. Enhancing nerf akin to enhancing llms: Generalizable nerf transformer with mixture-of-view-experts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3193–3204, 2023. 2

[13] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023. 5

[14] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 5, 13

[15] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[16] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4970–4980, 2023. 14

[17] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d-rotor gaussian splatting: towards efficient novel view synthesis for dynamic scenes. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2

[18] 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*, 2, 2024. 2, 6, 7, 14

[19] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022. 2, 5, 6

[20] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 2

[21] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 5, 6, 7, 13

[22] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *Advances in Neural Information Processing Systems*, 35:33768–33780, 2022. 5, 6, 7

[23] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3749–3761, 2022. 5, 13

[24] Horace He, Driss Guessous, Yanbo Liang, and Joy Dong. FlexAttention: The Flexibility of PyTorch with the Performance of FlashAttention. https://pytorch.org/blog/flexattention/, 2024. 5

[25] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023. 2

[26] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4220–4230, 2024. 2

[27] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. Lvsm: A large view synthesis model with minimal 3d inductive bias. *arXiv preprint arXiv:2410.17242*, 2024. 2, 4

[28] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. *CVPR*, 2023. 5, 13

[29] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 3

[30] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 5

[31] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 7

[32] Yao-Chih Lee, Zhoutong Zhang, and Kevin Blackburn-Matzen. Fast view synthesis of casual videos with soup-of-planes. 2023. 2

[33] Yao-Chih Lee, Zhoutong Zhang, Kevin Blackburn-Matzen, Simon Niklaus, Jianming Zhang, Jia-Bin Huang, and Feng Liu. Fast view synthesis of casual videos with soup-of-planes. In *European Conference on Computer Vision*, pages 278–296. Springer, 2025. 2, 5, 7

[34] Jiahui Lei, Yijia Weng, Adam Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. *arXiv preprint arXiv:2405.17421*, 2024. 2

[35] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 2

[36] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2, 5, 6, 7, 13

[37] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 2

[38] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21136–21145, 2024. 2

[39] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024. 5, 13

[40] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 2

[41] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13–23, 2023. 2, 5, 7

[42] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023. 2

[43] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4981–4991, 2023. 5, 13

[44] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[45] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[46] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2

[47] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2, 5, 6

[48] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-

Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 2, 5, 6, 7

[49] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016. 6, 7

[50] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5: 606–624, 2023. 4

[51] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2

[52] Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023. 2

[53] Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, et al. L4gm: Large 4d gaussian reconstruction model. *arXiv preprint arXiv:2406.10324*, 2024. 2, 3

[54] Xuanchi Ren, Yifan Lu, Hanxue Liang, Zhangjie Wu, Huan Ling, Mike Chen, Sanja Fidler, Francis Williams, and Jiahui Huang. Scube: Instant large-scale scene reconstruction using voxsplats. *arXiv preprint arXiv:2410.20030*, 2024. 2

[55] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Generalizable patch-based neural rendering. In *European Conference on Computer Vision*, pages 156–174. Springer, 2022. 7, 14

[56] Richard Sutton. The bitter lesson. *Incomplete Ideas (blog)*, 13(1):38, 2019. 4

[57] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2025. 13

[58] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 5

[59] Fengrui Tian, Shaoyi Du, and Yueqi Duan. Monodynerf: Learning a generalizable dynamic radiance field from monocular videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17903–17913, 2023. 2, 5, 7

[60] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 551–560, 2020. 2

[61] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 3

[62] Peihao Wang, Xuxi Chen, Tianlong Chen, Subhashini Venugopalan, Zhangyang Wang, et al. Is attention all that nerf needs? *arXiv preprint arXiv:2207.13298*, 2022. 2

[63] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. 2

[64] Shizun Wang, Xingyi Yang, Qiuhong Shen, Zhenxiang Jiang, and Xinchao Wang. Gflow: Recovering 4d world from monocular video. *arXiv preprint arXiv:2405.18426*, 2024. 2

[65] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 7

[66] Francis Williams, Jiahui Huang, Jonathan Swartz, Gergely Klar, Vijay Thakkar, Matthew Cong, Xuanchi Ren, Ruilong Li, Clement Fuji-Tsang, Sanja Fidler, Eftychios Sifakis, and Ken Museth. fvdb: A deep-learning framework for sparse, large-scale, and high-performance spatial intelligence. *ACM Transactions on Graphics (TOG)*, 43(4):133:1–133:15, 2024. 2

[67] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. 2, 5, 7

[68] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9421–9431, 2021. 2

[69] Haofei Xu, Anpei Chen, Yuedong Chen, Christos Sakaridis, Yulun Zhang, Marc Pollefeys, Andreas Geiger, and Fisher Yu. Murf: Multi-baseline radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20041–20050, 2024. 14

[70] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *arXiv preprint arXiv:2311.09217*, 2023. 3

[71] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv preprint arXiv:2406.09414*, 2024. 8

[72] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023. 2

[73] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20331–20341, 2024. 2

[74] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey

Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for Gaussian splatting. *arXiv preprint arXiv:2409.06765*, 2024. 5

[75] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020. 1, 2, 5, 6, 7

[76] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4578–4587, 2021. 14

[77] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9150–9161, 2023. 5, 13

[78] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024. 3

[79] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 1–19. Springer, 2025. 2, 3, 4, 7, 8, 14, 15

[80] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 4

[81] Xiaoming Zhao, R Alex Colburn, Fangchang Ma, Miguel Ángel Bautista, Joshua M Susskind, and Alex Schwing. Pseudo-generalized dynamic view synthesis from a video. In *The Twelfth International Conference on Learning Representations*, 2024. 3, 5, 7

[82] Yang Zheng, Adam W. Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J. Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *ICCV*, 2023. 5, 13

[83] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 5, 7, 13

[84] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21634–21643, 2024. 2

Supplementary Material

In this supplementary material, we provide additional details on the datasets used in our experiments (Sec. A), qualitative results on dynamic scenes (Sec. B), and more results on static scenes (Sec. C).

A. Dataset Details

The static datasets used in our training are as follows: OBJAVERSE [14] is a synthetic object-centric dataset, and we use the 80K-object subset from [57]. MVIMGNET [77] is a real-world object-centric dataset that has 220K objects. RE10K [83] is a real-world scene dataset that has 80K video clips. DL3DV [39] is a real-world scene dataset that has 10K video. We sample DL3DV 10 times more frequently than other datasets to balance the number of training samples. We use a spatial scale of 8 for Objaverse and scale 1 for all other datasets.

The dynamic datasets used in our training are as follows: KUBRICMV is a synthetic multi-vew video dataset that has 3K scenes. We rendered this dataset using the Kubric [23] engine. The scene setup follows Movi-E [23] and videos are rendered from all camera poses in the camera trajectory so it produces a multi-view video. POINTODYSSEY [82] is a synthetic monocular dataset with 131 scenes. DYNAMICREPLICA [28] is a synthetic stereo video dataset with 484 training sequences. SPRING [43] is a synthetic stereo video dataset with 37 scenes. PANDA-70M [8] is a real-world monocular video dataset. We use around 40K clips filtered from a random subset. We use scale 6 for Spring and DynamicReplica and 1 for other datasets. More details can be found in Tab. S1

Dataset	Dynamic	Subject	Domain	#Views	#Frames	#Scenes	#Multiplies	#Scale
RE10K [83]		*S*	Real	-	10M	80K	1	1
MVImgNet [77]		*O*	Real	-	6.5M	220K	1	1
Objaverse [14]		*O*	Synthetic	-	4M	80K	1	8
DL3DV [39]		*S*	Real	-	51M	10K	10	1
PointOdyssey [82]	✓	*O*+*S*	Synthetic	1	6K	131	3e3	1
Kubric-MV [23]	✓	*O*+*S*	Synthetic	24	70K	3K	2e2	1
DynamicReplica [28]	✓	*O*+*S*	Synthetic	2	145K	484	8e2	6
Spring [43]	✓	*O*+*S*	Synthetic	2	200K	37	1e4	6
PANDA-70M [8]	✓	*O*+*S*	Real	1	19M	40K	10	1

Table S1. **Datasets.** **Dynamic** indicates if the dataset is dynamic or static. **Subject** indicates if the dataset is object-centric (*O*) or scene-centric (*S*). **Domain** indicates if the dataset is captured from the real world or is synthesized. **#Views** denotes the number of synchronized views for a dynamic video. **#Frames** and **#Scenes** are the numbers of image frames and unique scenes in the dataset respectively. **#Multiplies** denotes the number of multiplies we sample the dataset (by scene) in training for balance. **#Scale** is the scale we applied to the dataset so that all datasets have approximately the same metric scale.

B. Qualitative Results on Dynamic Scenes

We have provided video results on the DyCheck Benchmark [21] and NVIDIA Dynamic Scene Benchmark [36] in the supplementary material. Additionally, we include novel view synthesis videos for the DAVIS dataset, DyCheck iPhone dataset, and SORA scenes. We also showcase a video demonstrating the effects of the NTE module, along with our video results on the Tanks & Temples static scenes. For access to these video results, please refer to our website by opening the `index.html` file into a modern browser.

C. More Results on Static Datasets

We provide a comprehensive qualitative comparison of our method against the baselines, MVSplat [10] and PixelSplat [7], on the RE10K dataset, as shown in Fig. S1. For each scene, the figure also displays the input views provided to the networks. Compared to the baselines, our method produces sharper outputs and more closely aligns with the ground-truth renderings. Note that all the methods used for the evaluation in this figure are trained exclusively on RE10K. Additionally, we use two views as context for all methods to ensure fairness in evaluation and to align with the setup of the baselines. Tab. S2 presents a quantitative evaluation against the baselines under the same settings. While our static model achieves the best performance

among the baselines, our dynamic *BTimer* model, trained for the dynamic task, also demonstrates strong performance on the static task, ranking third on the static benchmark.

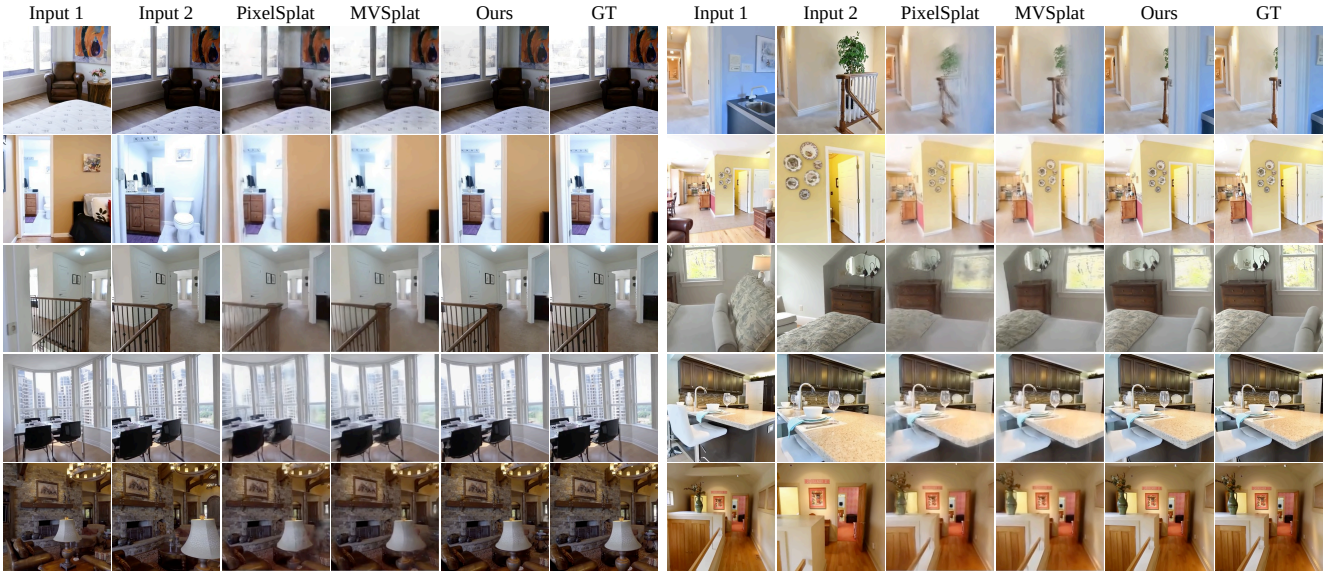

Figure S1. **Comparison on static scene dataset.** We compare our renderings with the baseline models, trained and tested on the RE10K dataset.

Model	PSNR↑	SSIM↑	LPIPS↓
PixelNeRF [76]	20.43	0.589	0.550
GPNR [55]	24.11	0.793	0.250
AttnRend [16]	24.78	0.820	0.213
MuRF [69]	26.10	0.858	0.143
PixelSplat [7]	25.89	0.858	0.142
MVSplat [10]	26.39	0.869	0.128
GS-LRM [79]	28.10	0.892	0.114
Ours-Static	28.91	0.920	0.070
Ours-BTimer	26.82	0.891	0.089

Table S2. **Quantitative comparison of models performance on RE10K test set.** To be consistent with the baselines, we adopt the 256×256 resolution. Our Bullet Timer has been trained on both static and dynamic scenes, while the other model is only trained on RE10K training set. We highlight the best, second best and third best models.

Our complete static model, trained across all datasets, is capable of reconstructing a highly diverse set of environments. Fig. S2 showcases our model's reconstructions across a wide variety of scenes, including outdoor forward-facing, outdoor drone shots, outdoor 360-degree views, indoor 360-degree views, and indoor forward-facing scenes, as well as object-centric synthetic scenes. Notably, all these reconstructions are achieved using a shared set of weights, demonstrating that our model, trained across multiple datasets, generalizes effectively to different scenarios.

To further demonstrate the importance of training on multiple datasets for generalization to unseen datasets, we conduct an ablation study on the datasets used to train our static model. Tab. S3 compares the performance of our model when trained individually on a single dataset—RE10K, MVImageNet, DL3DV, or Objaverse—against its performance when trained on all these datasets simultaneously. The evaluation is conducted on a completely unseen dataset, the Tanks & Temples split from the InstantSplat [18] paper. Our model, whether static or dynamic, trained on all datasets significantly outperforms the single-dataset models.

Figure S2. A diverse set of scenes reconstructed using our static model, trained on multiple datasets and capable of generalizing to various scenarios.

Model	Datasets	PSNR↑	SSIM↑	LPIPS↓
GS-LRM* [79]	RE10K	17.56	0.546	0.310
Ours-Static	Objaverse	7.00	0.363	0.668
	MVImageNet	17.75	0.530	0.343
	DL3DV	17.92	0.566	0.278
	All Static	24.22	0.807	0.093
Ours-Full	+Dynamic	24.13	0.806	0.093

Table S3. **Baseline comparisons on the Tanks & Temples dataset (InstantSplat split).** Test views are 512×512. LPIPS are computed on 256×256. We highlight the best , second best and third best models. *: Our reproduced results.