4DGS360: 360° Gaussian Reconstruction of Dynamic Objects from a Single Video

Supplementary Material

In this appendix, we present further implementation details, more results, and ablations. The additional material is structured as follows:

A Additional Details

A.1 AnchorTAP3D details

AnchorTAP3D employs BootsTAP [7] as the 2D tracking model and TAPIP3D [61] as the 3D tracking model. The BootsTAP model outputs two logits for calculating confidence: occlusion logit o and uncertainty logit u. We convert these raw logits into interpretable probability values as $v = 1 - \sigma(o)$ and $p = 1 - \sigma(u)$, where v represents the visibility probability, p denotes the certainty probability, and $\sigma(\cdot)$ is the sigmoid function.

To compute the final confidence c for each track point, we multiply its visibility and certainty as $c = v \times p$. This calculated confidence represents how certain the 2D tracker is about the given track point. We filter the points by applying a threshold as shown in Eq. (9). We set the default threshold τ to 0.5. This value can be adjusted based on the scene complexity and the reliability of tracking results when adopting new 2D or 3D tracking models in the future, which would yield more stable results. Ablation study on threshold is in Sec. C.

As mentioned in Sec. 3.2, we employ $\mathcal{P}^{3D}$ with a transformer architecture, which simultaneously infers points within a window of fixed length L frames. We set L to 16 throughout our experiments. Each window undergoes 6 inference iterations. After each iteration, if there exist unprojected 2D track points with high confidence, they replace the corresponding points and serve as anchors.

Based on these updated anchors, the 3D tracking model infers other points within the window in the next iteration step. We utilize dynamic object masks to exclude inferred points whose positions fall outside the mask region. We first run the 2D tracking model over all frames. The query points of 2D tracks at each timestep t are then reused as query points for the 3D tracking model after unprojection. Using these query points together with the anchors, AnchorTAP3D generates 3D tracks for initialization. AnchorTAP3D takes 71 minutes for 130-frame video using an NVIDIA RTX A6000 D6 48GB GPU.

A.2 Additional initialization details

As mentioned in the initialization details in Sec. 3.2, we randomly select a subset of tracks across all query times from the 3D tracks obtained by AnchorTAP3D to initialize the Gaussians. Each track stores the position of a point at each frame, which we transform into the temporal trajectory of a single Gaussian in our model. Thus, one track corresponds to one Gaussian and its motion path over time. The initial color of each Gaussian is set by taking the point's color at the query time of the track.

A.3 Optimization details

We use the Adam [17] optimizer. In Eq. (13), D-SSIM [62] loss and LPIPS [49] loss are used together with $\mathcal{L}_{\text{rgb}}$ with weights of 0.2, 0.01, and 0.8, respectively. We apply $\mathcal{L}_{\text{mask}}$ to align rendered and predicted masks using MSE (weight 1.0). $\mathcal{L}_{\text{depth}}$ enforces depth consistency through an MSE term (weight 0.5) and smoothness via gradient regularization (weight 1.0). $\mathcal{L}_{\text{track}}$ ensures rendered trajectories match 2D track observations through two components: a 2D coordinate alignment term $\mathcal{L}_{\text{track-2d}}$ (weight 2.0) and a depth consistency term $\mathcal{L}_{\text{track-depth}}$ (weight 0.1) that matches reprojected and estimated depths.

As mentioned in Eq. (12), $\mathcal{L}_{\text{arap}}$ is computed based on spatial relationships between temporally non-adjacent frames within each batch. In Eq. (12), $\mathcal{N}$ represents node pairs (i, j), which are constructed in two ways. First, for each node i, we select the 5 nearest nodes across all frames. Second, to ensure that nodes aligned along the ray direction maintain consistent distances when representing the same rigid object, we select the 2 nodes with the most similar velocities along the ray direction and incorporate them into $\mathcal{N}$ to preserve their distances across different time frames. The nearest node method is applied throughout all training steps with a weight of 2.0.

We optimize Node-based motion representation and canonical Gaussians. For Gaussian parameters, we set learning rates as follows: mean (1.6×10^{-4}), opacity (1×10^{-2}), scale (5×10^{-3}), rotation (1×10^{-3}), and color (1×10^{-2}). We incorporate adaptive density control for Gaussians during training proposed in 3DGS [15]. Motion bases are optimized with a learning rate of 1.6×10^{-4}. For node parameters, we configure learning rates at 1.6×10^{-5} for position, 5×10^{-4} for radius, and 1×10^{-2} for motion coefficients. We set the number of first-level nodes to 70, second-level nodes to 10, and the number of node bases to 15

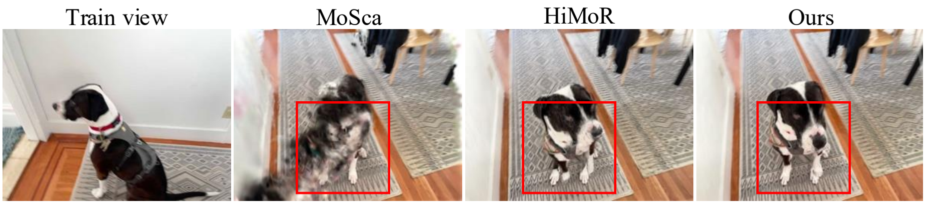

Fig. 9: iPhone dataset [10] *Haru-sit* scene(no test ground truth) qualitative result. It shows that our model reconstructs unseen regions such as the leg area with reasonable accuracy.

for iPhone360 experiments. Since there is no universally optimal value for these hyperparameters, as the ideal setting varies depending on scene characteristics, we report results under fixed values rather than per-scene tuning.

Since our model performs 360° reconstruction using a loss closely related to prior works [24, 47], the overall optimization pipeline does not introduce substantial additional computational overhead compared to existing approaches. Our model requires approximately 3 hours to reconstruct on 140 images using an NVIDIA RTX A6000 D6 48GB GPU.

B Additional Results

B.1 Evaluation details

As mentioned in Sec. 4, we adopt bounding boxes to compare with ground truth at extreme novel-view synthesis. When synthesizing extreme novel views in object-centric scenes, background regions appear mostly white because training cameras rarely observe those areas. Therefore, to exclude unseen background regions from evaluation and focus on the reconstruction of dynamic objects, we compare renders using only the dynamic foreground Gaussians against the corresponding dynamic object regions in the test images. However, naively computing CLIP scores between foreground Gaussian renders and foreground-masked ground truth is ineffective due to excessive white regions in both images. Instead, we create bounding boxes by extending the ground truth mask horizontally and vertically by 1/8 of the image resolution on each side. We apply these bounding boxes to both images and compute CLIP similarity scores within the cropped regions. Visualizations before and after bounding box application are shown in Fig. 10(a).

We also provide ground truth and render image pairs and per-scene quantitative scores in Fig. 10(b), illustrating the correlation between the magnitude of scores and the visual results. As can be observed, the actual extreme novel-view synthesis results of HiMoR [24] and Ours exhibit substantial perceptual differences, which is reflected in the CLIP-I scores. This discrepancy is not clearly captured by pixel-wise evaluation metrics. Furthermore, the absolute values of pixel-wise scores are relatively lower compared to those in conventional tasks,

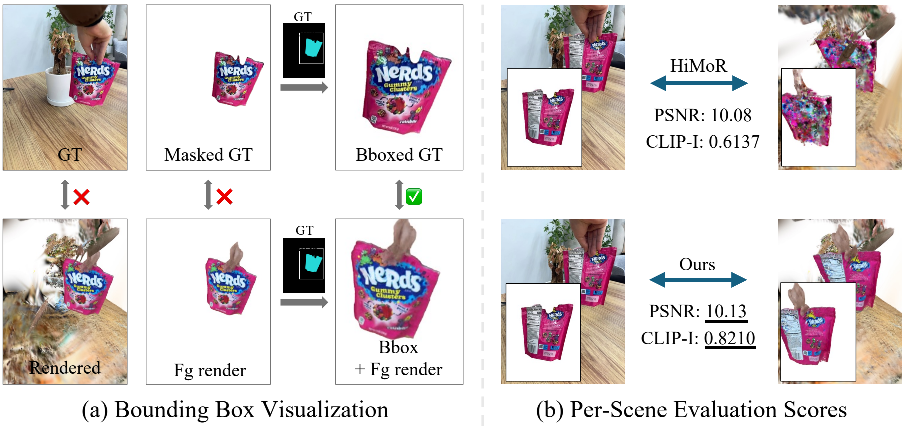

Fig. 10: (a) shows Visualization of bounding box (Bbox) application. For both the GT and rendered images, the bounding boxes are computed from a processed version of the GT mask. Evaluation between GT and rendered images is unreliable due to background ambiguity. Evaluation between masked GT and foreground rendering is also problematic due to excessive white regions. Evaluation between Bbox + GT and Bbox + FG rendering provides the most reliable comparison strategy. (b) denotes per-scene evaluation scores to illustrate the correspondence between quantitative metrics and visual results, where CLIP-I better reflects the perceptual similarity compared to the pixel-wise metric PSNR.

as the evaluation is performed on extreme novel view. As mentioned in Sec. 4, monocular dynamic scene reconstruction under novel-view synthesis is a highly ill-posed problem. In particular, when it comes to extreme novel-view, most object regions visible from extreme viewpoints are not observed in the training images at the same timestamp, making it difficult to precisely guide object localization. Combined with the sensitivity of pixel-wise metrics to even small positional errors, achieving high scores becomes inherently challenging, and the scores may not always align with human perceptual quality. Pixel-wise evaluations are reported in Tab. 4 for models supporting 1x resolution inference, and all training and evaluation on the iPhone and iPhone360 datasets are conducted at 1x resolution.

B.2 iPhone360 dataset

We provide a **Supplementary Video** at project page for qualitative validation. It includes bullet-time effect 360° results and extreme test view comparisons with ground truth. The bullet-time effect 360° results demonstrate that our model can reconstruct 360° of an object at every dynamic time step if it appears in the video. These results demonstrate that our model surpasses baseline models in 360° reconstruction [22, 24, 47]. In the *jelly*, *jacket*, and *goat* scenes, only our method produces proper geometry. The baselines exhibit abnormal geomet-

Table 4: Pixel-level Evaluation on iPhone and iPhone360 Datasets. All training and evaluations are conducted at 1x resolution.

	iPhone dataset		iPhone360 dataset	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑
HiMoR	16.2441	**0.6329**	10.9883	0.6854
Ours	**16.3431**	0.6309	**11.3364**	**0.7012**

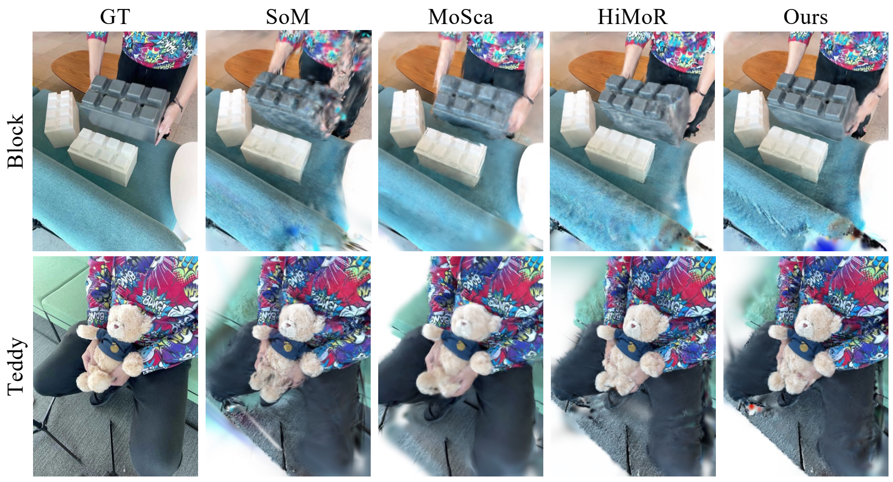

Fig. 11: Qualitative comparison with the baseline models [22, 24, 47] on the iPhone dataset [10] Block and Teddy scenes.

ric deformations, simply overfitting to the training video. In the *walk-around* scene, the baseline videos [22, 24] exhibit clear visual artifacts and abrupt velocity changes. Gaussians originating in regions visible to the training cameras undergo abrupt velocity changes when they pass through occluded areas. Consequently, the person's lower body appears disconnected and exhibits unnatural motion with abrupt velocity changes. These results indicate that when initialization is based on 2D tracks, optimization fails to reliably reconstruct invisible regions and instead induces sharp velocity changes to overfit the training images. As also shown in the bullet-time 360° results on the *goat* scene, the baseline [24] struggles to reconstruct even the training views under complex dynamic motion. In contrast, our method places Gaussians in plausible locations within invisible regions from the outset via AnchorTAP3D, effectively preventing Gaussian drifting even under a similar optimization scheme.

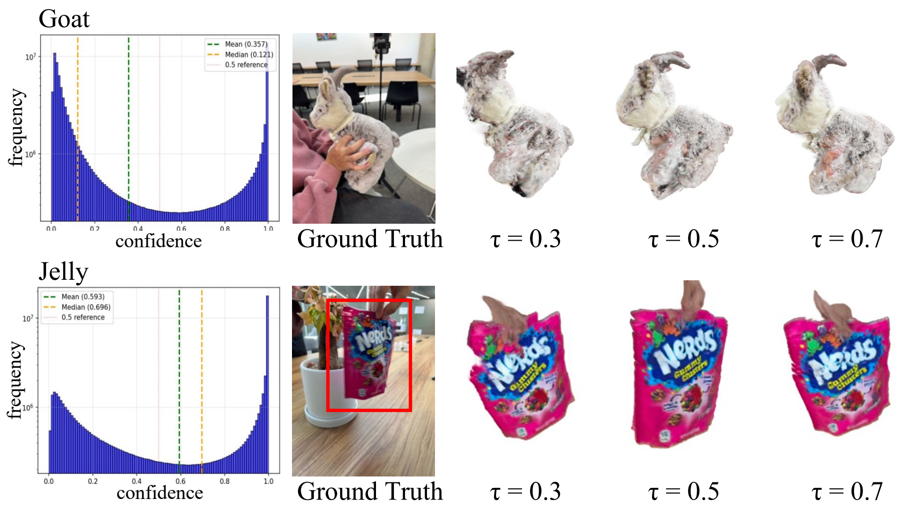

Fig. 12: AnchorTAP3D Threshold Ablation. The left graph shows the confidence distribution of 2D tracking points for each scene, with the mean and median confidence values displayed at the top. The right side presents the final 4DGS360 rendering results with varying τ values applied during the AnchorTAP3D process. This allows us to observe the correlation between the confidence distribution of 2D tracking results and the choice of threshold.

B.3 iPhone dataset

The *Haru-sit* scene in the iPhone dataset [10] captures various parts of the object through a moving camera. Since *Haru-sit* scene lacks novel view ground truth, we report qualitative performance comparisons with existing models on a new pseudo test view in Fig. 9, demonstrating our model's capability in reconstructing regions far from the training view. Comparisons on the *Block* and *Teddy* scenes are shown in Fig. 11. As the iPhone dataset involves relatively smaller view discrepancy compared to iPhone360, baseline models also produce competitive results in these scenes.

C Additional Ablation

As noted in Sec. A.1, all experiments are conducted with a fixed threshold of $\tau = 0.5$ to evaluate the general performance of our model. Here, we provide threshold ablations in Fig. 12 to further investigate how the choice of threshold interacts with the confidence distribution of 2D tracking results across different scenes. The *goat* scene has a mean confidence of 0.357, while the *jelly* scene has a mean of 0.593, indicating that the 2D tracker finds the *goat* scene more challenging and the *jelly* scene more tractable. More reliable 2D tracking results suggest that setting a lower threshold to incorporate more 2D tracking results as anchors can still be beneficial in certain scenes. Consequently, as reflected in the final rendering results, the *jelly* scene achieves its best performance at $\tau = 0.5$, while the *goat* scene benefits from a higher threshold of $\tau = 0.7$. Note that excessively low thresholds such as $\tau = 0.3$ also lead to degraded results for both scenes. For practitioners seeking to reconstruct new scenes, we suggest setting $\tau \approx 1 - \text{ratio}(\text{confidence} > 0.5)$ as a reasonable initial threshold to begin the search for an optimal value.

D iPhone360 Dataset Overview

The supplementary video provides an overview of the scenes comprising the iPhone360 dataset. The *goat*, *jelly*, and *walk-around* scenes are captured with one training camera and two test cameras, while *block2*, *jacket*, and *pull-up* are captured with one training camera and one test camera. In the *goat*, *jacket*, *jelly*, and *walk-around* scenes, the training and test cameras consistently maintain an angular difference of over $90°$. For *block2* and *pull-up*, the training camera moves more actively, resulting in a varying angular difference with respect to the test camera. The six scenes cover a diverse range of motions, including rigid object movement, deformable objects, and human motion. Furthermore, all scenes are captured with a hand-held moving camera, faithfully reflecting real-world capture scenarios. The iPhone360 dataset is the first dataset designed to evaluate monocular dynamic reconstruction under extreme novel-view synthesis in real-world capture scenarios.