Global Motion Corresponder for 3D Point-Based Scene Interpolation under Large Motion

Supplementary Material

In this document, we include

- more details about the implementation,
- more details about the SI-MPED metric,
- more results on motion interpolation and extrapolation,
- more results on the application of sparse view refinement,
- and more results on the ablation study.

A. Implementation Details

The input positions and PCA-DINO for the MLPs are normalized using scalars pre-calculated from the start-state 3DGS model. The position input is then scaled by a hyperparameter weight, selected from $\{0.1, 1.0\}$ based on the importance of the positional information. Correspondingly, dropout is applied to the position input to avoid trivial local minima, with a ratio of 0.1 or 0.2 depending on the previously chosen scale. To mitigate the issue of getting trapped in local minima, the Perturb-and-MAP strategy [19] is applied to the total energy, where the perturbations are sampled from a Gumbel distribution. The learning rate of training the MLPs is set to 0.0005, using the Adam optimizer [13] with default parameters. For the RGB loss during the joint refinement, L1 and LPIPS [32] losses are combined with weights 1.0 and 0.1, respectively; gradients from the MLPs are not used to update the 3DGS models. Due to the large size of Gaussian sets, batches of Gaussians are sampled during each iteration when searching for the minimum energy between the two Gaussian sets. For most scenes, the batch size is set to $20,000$, and it can be reduced accordingly if the total number of Gaussians is smaller. The FAISS library [11] is used to perform efficient nearest neighbor searches.

B. SI-MPED Metric

For each interpolation step, the Multiscale Potential Energy Discrepancy (MPED) [27] is calculated between the interpolated point cloud and the ground-truth point cloud from the start and end states, respectively. The MPED is computed by aggregating distances from the neighborhoods comprising 0.1%, 0.5%, and 1% of the total points, summing these values to obtain the overall MEPD. Following PAPR in Motion [20], the Scene Interpolation MPED (SI-MPED) is defined as a weighted sum of the MEPD at each interpolation step, where the weights are proportional to the average distance movement of points compared to the total movement from the start to the end states.

C. Motion Interpolation and Extrapolation

Qualitative results on motion interpolation and extrapolation for global-motion scenes are presented in Figure 10. Additionally, qualitative results on local-motion scenes [20] are shown in Figure 11, and the quantitative interpolation evaluations for these scenes are provided in Table 7. Qualitative interpolation results on two real-world scenes from Dynamic Gaussian [17] are provided in Figure 9, in comparison with the Dynamic Gaussian [17] baseline.

D. Ablation Study

We find the following properties when removing one of the key components of our method:

1. Removing DINO input can result in implausible interpolation (`Ball`), wrong global motion interpolation (`Boat`), or wrong local motion interpolation.
2. Removing position input can result in wrong global matching (`Ball` and `Car`) or wrong local motion interpolation (`Butterfly`).
3. Removing local isometry loss can result in noisy floaters (`Dolphin`) or blurry rendering (`Butterfly` and `Microwave`) during the interpolation.
4. Removing local isometry loss can result in noisy rendering (`Ball` and `Microwave`) during the interpolation or suboptimal end status prediction (`Butterfly`).

E. Sparse View Refinement

In addition to motion interpolation and extrapolation, GMC can also be used to improve reconstruction quality in sparse capture scenarios. Specifically, only five or ten views are available for sparse captures, and we consider two settings: (1) the start state has dense views and the end state has sparse views, and (2) both states have sparse views. When the input views are sparse, the reconstructed 3DGS will have bad geometry and thus will perform poorly in novel view synthesis. While a single state might not have enough views for good 3D reconstruction, we can borrow the information from the other state so that it can refine the self geometry and thus improve the novel view synthesis. Specifically, through the rendering loss $\mathcal{L}_{\text{RGB}}(\boldsymbol{I}_f, \hat{\boldsymbol{I}}_f)$ in Eq. 9, the sparse-view 3DGS can use the training views from the other state, and thus improve itself.

Results. For the sparse-view setting, we set $\beta = 5$, because in this setting, the ground-truth training views are more reliable than the "borrowed" information based on

Sparse + Dense													
		Synthetic Scenes								Real-world Scenes			
Metric	Method	Ball	Boat	Butterfly	Car	Dolphin	Knight	Microwave	Seagull	Box	Shoe	Tapeline	Avg
PSNR ↑	3DGS [12]	30.39	31.64	28.94	24.42	34.50	26.82	31.98	31.12	23.50	26.10	26.39	28.71
	Ours	**38.18**	**36.25**	**31.18**	**33.63**	**37.59**	**33.49**	**37.63**	**35.76**	**26.31**	**26.94**	**26.81**	**33.07**
SSIM ↑	3DGS [12]	0.978	0.970	0.973	0.946	0.992	0.965	0.980	0.964	0.890	0.930	0.959	0.959
	Ours	**0.992**	**0.984**	**0.983**	**0.981**	**0.995**	**0.985**	**0.989**	**0.981**	**0.912**	**0.940**	**0.964**	**0.973**
LPIPS ↓	3DGS [12]	0.045	0.047	0.059	0.086	0.018	0.063	0.042	0.051	0.107	0.087	0.046	0.059
	Ours	**0.006**	**0.011**	**0.013**	**0.016**	**0.005**	**0.008**	**0.012**	**0.013**	**0.064**	**0.066**	**0.033**	**0.022**

Table 5. **Novel View Synthesis for Sparse-Dense View Setting.** For the synthetic scenes, the start state has 100 dense training views, while the end state has 10 sparse training views. For real-world scenes (Shoe, tapeline, and Box), the end state has 5 sparse training views. The results are reported as the mean value of test views for each scene.

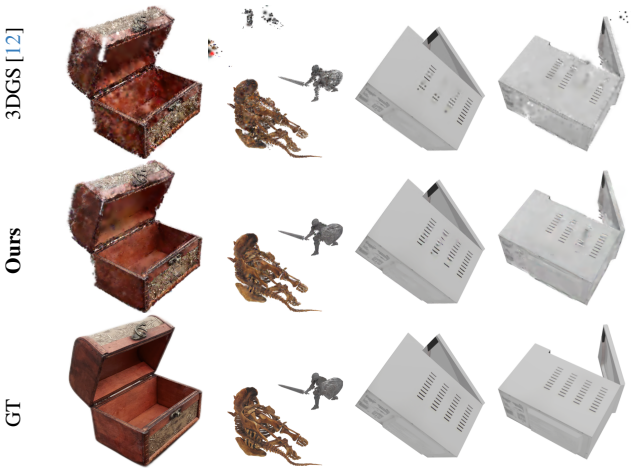

Figure 8. **Novel-View Synthesis in Sparse-View Setting.** This figure demonstrate novel-view syhnthesis results in sparse-view setting. From top to bottom, the rows show results from vanilla 3DGS [12], 3DGS refined by our method, and the ground truth. The first column displays the real-world Box scene, where the start state has 100 dense training views and the end state has only 5 sparse views. The second column shows the synthetic Knight scene, with the start state having 100 dense views and the end state having 10 sparse views. The last two columns present the Microwave scene, where both start and end states have 10 sparse views, showing novel view synthesis for the start state (left) and end state (right). The proposed method effectively transfers information between states, improving texture quality in the Box and Microwave scenes, and enhancing geometry by remvinng floaters in the Knight scene.

Gaussian matching. Otherwise, each scene has the same setting as the dense-dense setting studied for interpolation and extrapolation tasks. We showcase three examples of the application in sparse-view refinement in Figure 8, and we report quantitative results, average PSNR, SSIM [24], and LPIPS [32] of novel-view synthesis of 200 views. Quantitative results on sparse-view refinement with the sparse-sparse view setting are presented in Table 6, and results with the sparse-dense view setting are shown in Table 5.

Sparse + Sparse								
		Synthetic Scenes				Real-world Scene		
Metric	Method	Car		Microwave		Box		Avg
		start	end	start	end	start	end	
PSNR ↑	3DGS [12]	23.54	24.37	26.25	31.94	23.80	23.48	25.56
	Ours	**29.07**	**29.96**	**33.60**	**34.83**	**25.19**	**25.36**	**29.67**
SSIM ↑	3DGS [12]	0.943	0.946	0.962	0.980	0.900	0.890	0.937
	Ours	**0.965**	**0.967**	**0.983**	**0.985**	**0.915**	**0.907**	**0.953**
LPIPS ↓	3DGS [12]	0.097	0.086	0.079	0.042	0.105	0.107	0.086
	Ours	**0.041**	**0.040**	**0.028**	**0.021**	**0.069**	**0.075**	**0.046**

Table 6. **Novel View Synthesis in Sparse + Sparse View Setting.** For the scenes Car and Microwave, both states have 10 training views; for Box, both states have 5 training views. The results are reported as the mean value of test views for each scene.

Both qualitative and quantitative results show that our method significantly improves upon the vanilla 3DGS trained on sparse views. When training views are few and sparse, two significant issues arise: (1) the presence of floaters, and (2) a lack of details in under-observed regions. The qualitative results show that our method is able to reduce (1) and handle (2). Our method reduces floaters because they have a poor match in the other state, and thus, when transformed and rendered, they can be removed by the rendering loss. Our method improves details in under-observed regions because the other state may have more information on appearance details, which can be borrowed to enhance the current state.

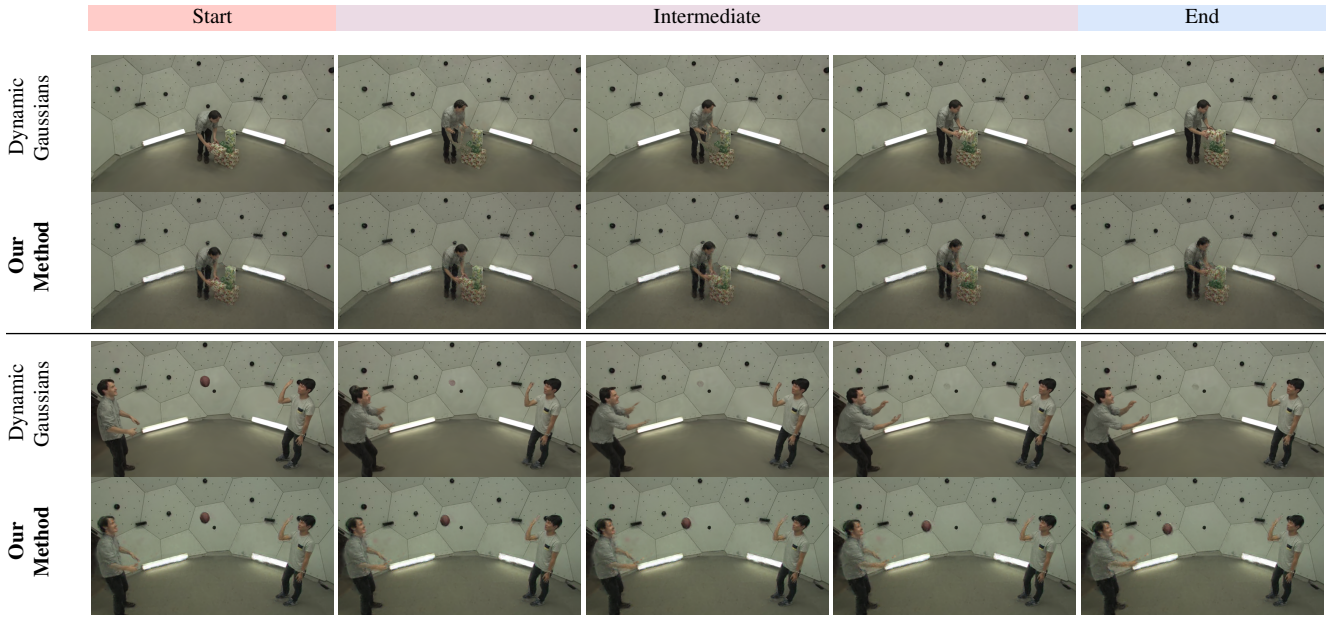

Figure 9. **Additional Interpolation Results.** The figure presents interpolation results using our method on the `Bxoes` and `Football` scene from Dynamic Gaussian [17]. The five columns correspond to five timesteps: 0.00, 0.25, 0.50, 0.75, 1.00.

		Synthetic Scenes							Real-world Scenes		
Metric	Method	`Butterfly`	`Crab`	`Dolphin`	`Giraffe`	`Lego Bulldozer`	`Lego Man`	Avg	`Stand`	`Lamp`	Avg
SI-FID ↓	4DGS [25]	130.98	131.34	**107.55**	**166.66**	229.14	136.82	150.42	431.64	380.52	406.08
	Deformable 3DGS [28]	569.21	179.67	365.49	412.67	254.60	645.14	404.46	-	-	-
	Dynamic Gaussian [17]	328.69	129.52	165.78	215.83	197.68	330.94	228.07	302.40	248.49	275.45
	PAPR in Motion [20]	90.89	73.86	112.92	174.73	**103.34**	151.89	117.94	203.56	265.08	234.32
	Ours	**87.13**	**65.24**	112.34	169.65	110.37	**131.01**	**112.62**	**165.36**	**181.22**	**173.29**
SI-EMD ↓	4DGS [25]	22.94	12.99	**1.81**	5.72	37.60	**11.06**	15.35	97.71	88.11	92.91
	Deformable 3DGS [28]	45.84	29.34	5.96	22.22	24.50	36.24	27.35	-	-	-
	Dynamic Gaussian [17]	104.57	10.19	50.47	11.13	62.60	146.65	64.27	84.69	103.58	94.14
	PAPR in Motion [20]	34.93	**9.87**	2.17	**5.03**	13.34	12.61	12.99	29.77	63.12	46.45
	Ours	**32.59**	13.26	2.78	5.40	**9.52**	14.23	**12.96**	**17.21**	**56.92**	**37.07**
SI-MPED ↓	4DGS [25]	140.40	127.8	30.71	50.61	500.84	89.47	156.64	620.66	769.12	694.89
	Deformable 3DGS [28]	81.87	32.26	12.53	14.78	47.27	53.59	40.38	-	-	-
	Dynamic Gaussian [17]	143.99	44.60	79.09	24.26	260.85	136.67	114.91	260.68	166.58	213.63
	PAPR in Motion [20]	11.57	**7.16**	**4.05**	5.89	19.13	8.50	9.38	**26.72**	22.00	**24.36**
	Ours	**11.02**	7.85	5.07	**4.70**	**18.55**	**8.40**	**9.27**	30.27	**18.95**	24.61

Table 7. **Scene Interpolation Evaluation on Local-Motion Scenes [20].** The table compares our method with the baseline methods on scenes with local motion [20], where "-" represents failure of a method. Rendering quality is evaluated using Scene Interpolation FID (SI-FID), while geometry quality is assessed using Scene Interpolation Earth Mover's Distance (SI-EMD) and Scene Interpolation Multiscale Potential Energy Discrepancy[27] (SI-MPED).

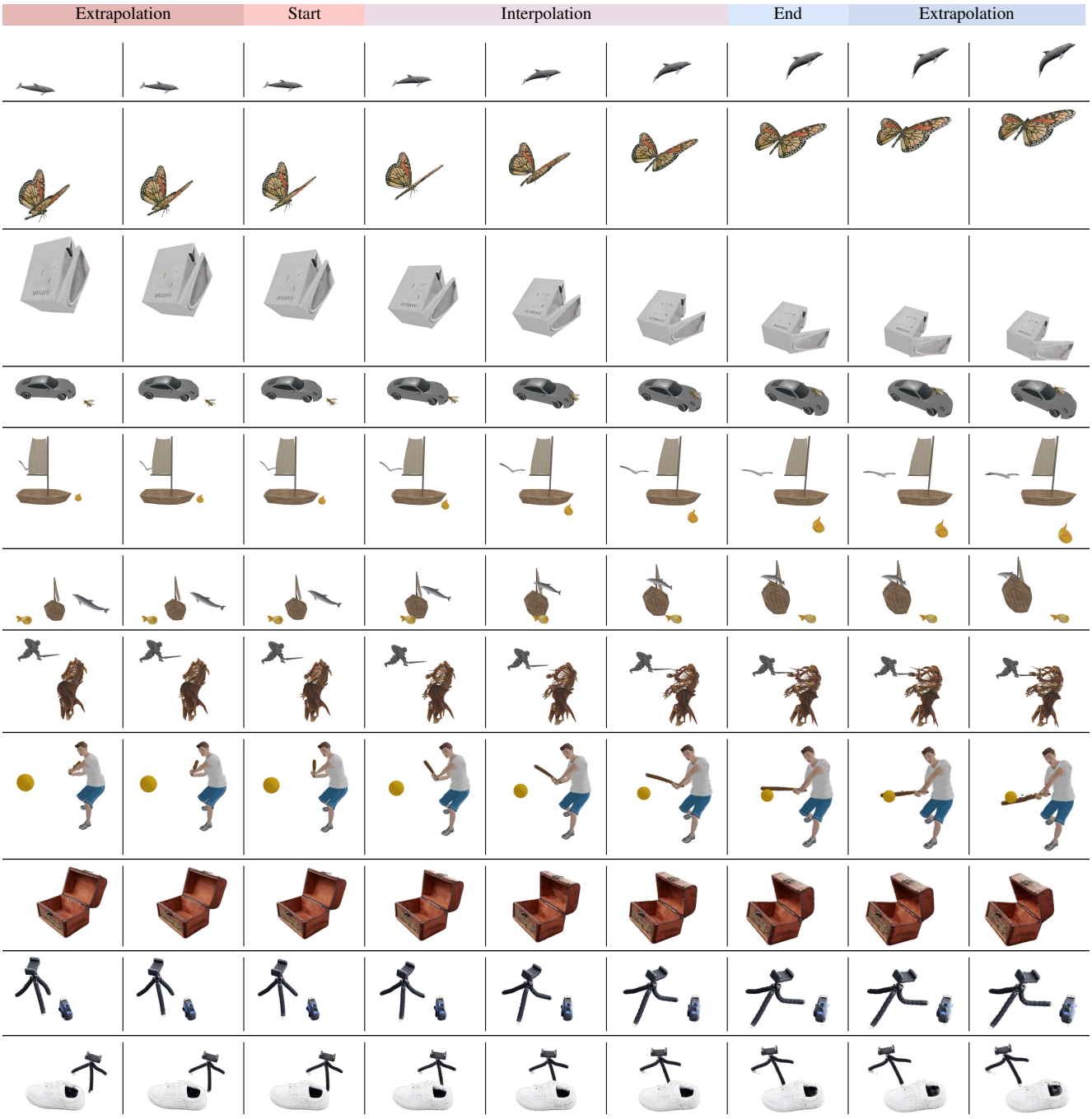

Figure 10. **Additional Interpolation and Extrapolation Results.** The figure presents interpolation and extrapolation novel-view synthesis results using our method on the global-motion dataset. From top to bottom, the scenes displayed are `Dolphin`, `Butterfly`, `Microwave`, `Car`, `Seagull`, `Boat`, `Knight`, `Ball`, `Box`, `tapeline`, and `Shoe`. The top nine scenes are synthetic, and the bottom three are real-world. The nine columns correspond to nine timesteps: $\{-0.20, -0.10, 0.00, 0.25, 0.50, 0.75, 1.00, 1.10.1.20\}$.

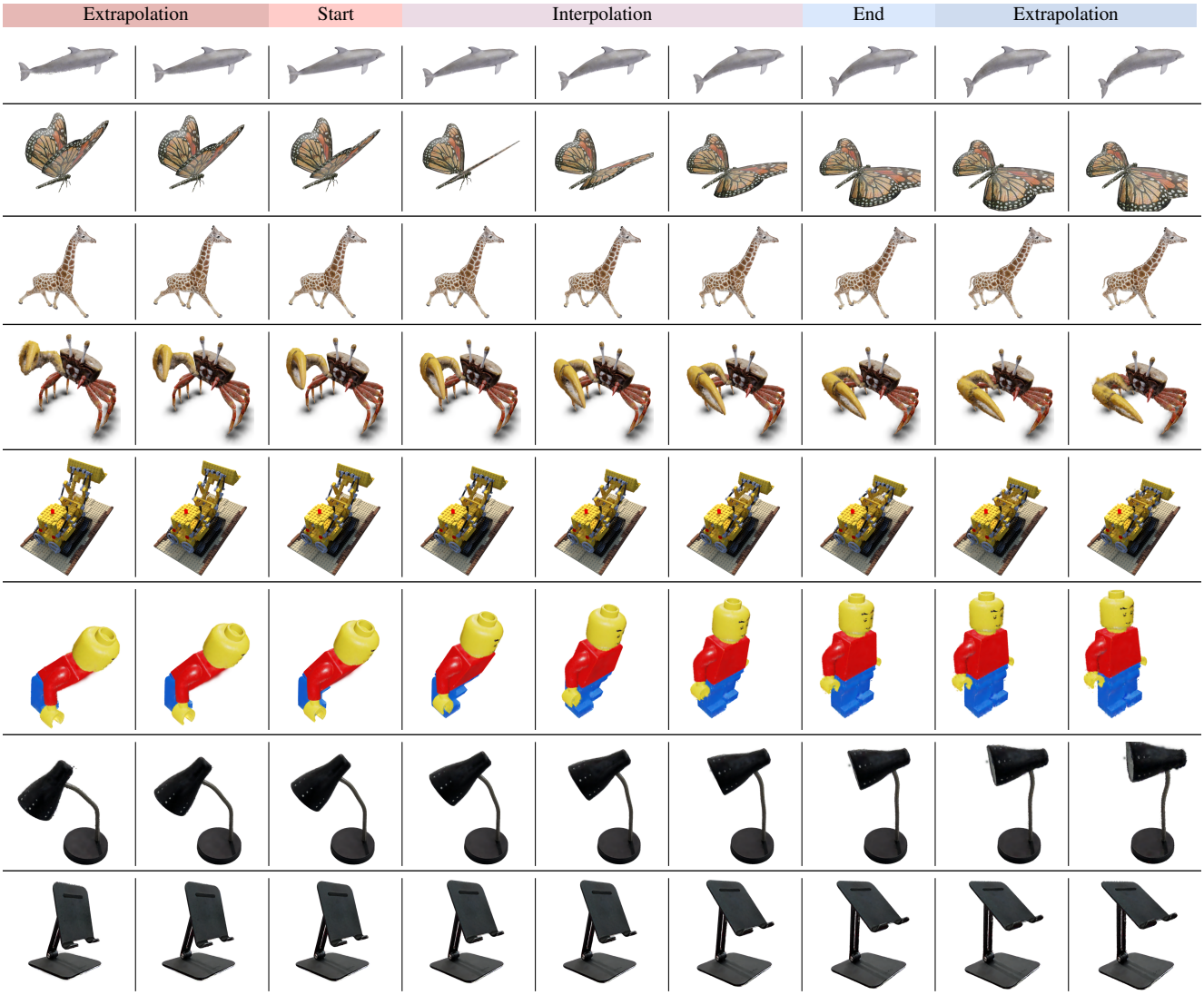

Figure 11. **Additional Interpolation and Extrapolation Results.** The figure shows interpolation and extrapolation novel-view synthesis results using our method on the PAPR in Motion dataset [20]. From top to bottom, the scenes displayed are `Dolphin`, `Butterfly`, `Giraffe`, `Crab`, `Lego Bulldozer`, `Lego Man`, `Lamp`, and `Stand`.