Efficiently Reconstructing Dynamic Scenes One 🔪 D4RT at a Time

Appendix

A. Model Overview & Training Details

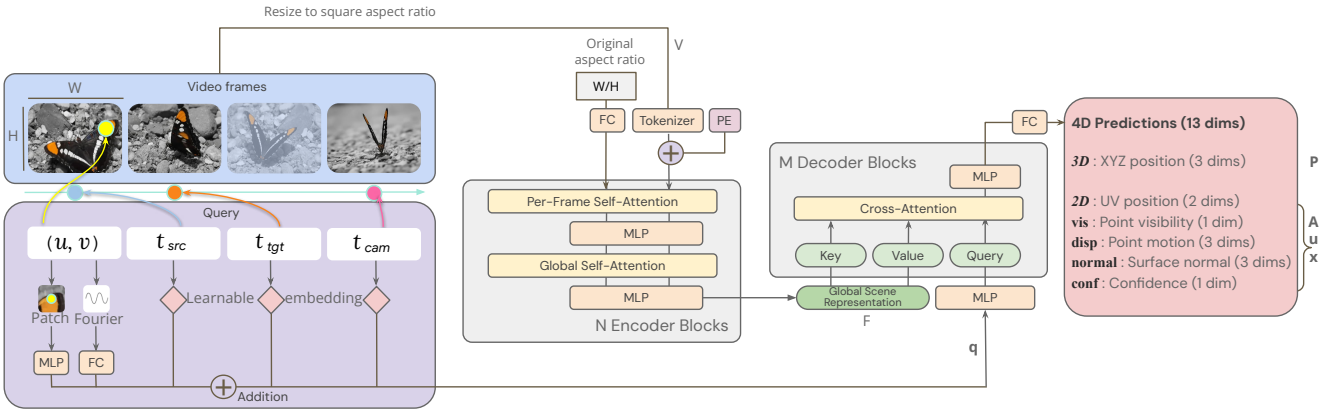


Figure 7. **Full D4RT model overview** – We provide a holistic overview of the model together with its inputs and outputs. FC corresponds to a fully connected layer, and PE for positional encoding. See Sec. 2 of the main paper for reference.

The model is trained end-to-end by minimizing a composite loss $\mathcal{L}$, which is a weighted sum of task-specific losses computed over a batch of N sampled queries:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\Big(c\lambda_{3D}\mathcal{L}_{3D} - \lambda_{\text{conf}}\log c + \lambda_{2D}\mathcal{L}_{2D} + \lambda_{\text{vis}}\mathcal{L}_{\text{vis}} + \lambda_{\text{disp}}\mathcal{L}_{\text{disp}} + \lambda_{\text{conf}}\mathcal{L}_{\text{conf}} + \lambda_{\text{normal}}\mathcal{L}_{\text{normal}}\Big)_i$$

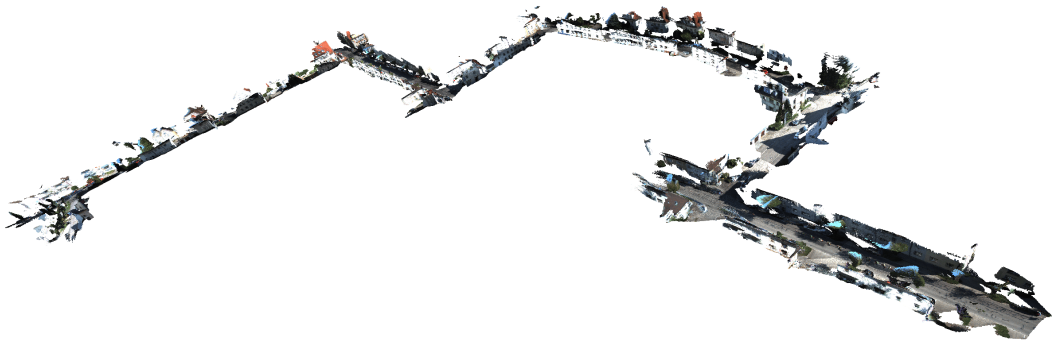
where c is the confidence score predicted by the model. We use the following loss weights: λ_{3D}=1.0, λ_{2D}=0.1, λ_{vis}=0.1, λ_{disp}=0.1, λ_{conf}=0.2, λ_{normal}=0.5, λ_{conf}=0.2. We train using the AdamW optimizer with a weight decay of 0.03. The learning rate is warmed up for 2,500 steps until it reaches a peak value of 10^{-4}. Subsequently, it follows a cosine annealing schedule, decaying to a final value of 10^{-6}. Gradients are clipped to a maximum L^2-norm of 10.

Data augmentation. Extensive data augmentation techniques are applied to the video during training to improve model generalization. We apply temporally consistent color jittering by performing random brightness, saturation, contrast, and hue adjustments. We also apply random color drop with a probability of 0.2, and Gaussian blur augmentation with a probability of 0.4. For spatial augmentation, we use random crop augmentations with a scale ratio between 0.3 and 1.0 of the original size. After determining the crop size, a random aspect ratio is selected by sampling from a uniform distribution in the logarithmic domain; this ensures equal probability for wide and tall crops while respecting image boundaries. Additionally, during the random crop augmentation, the image is randomly zoomed in with a probability of 0.05. On the temporal dimension, frames are subsampled from the full video with a random stride.

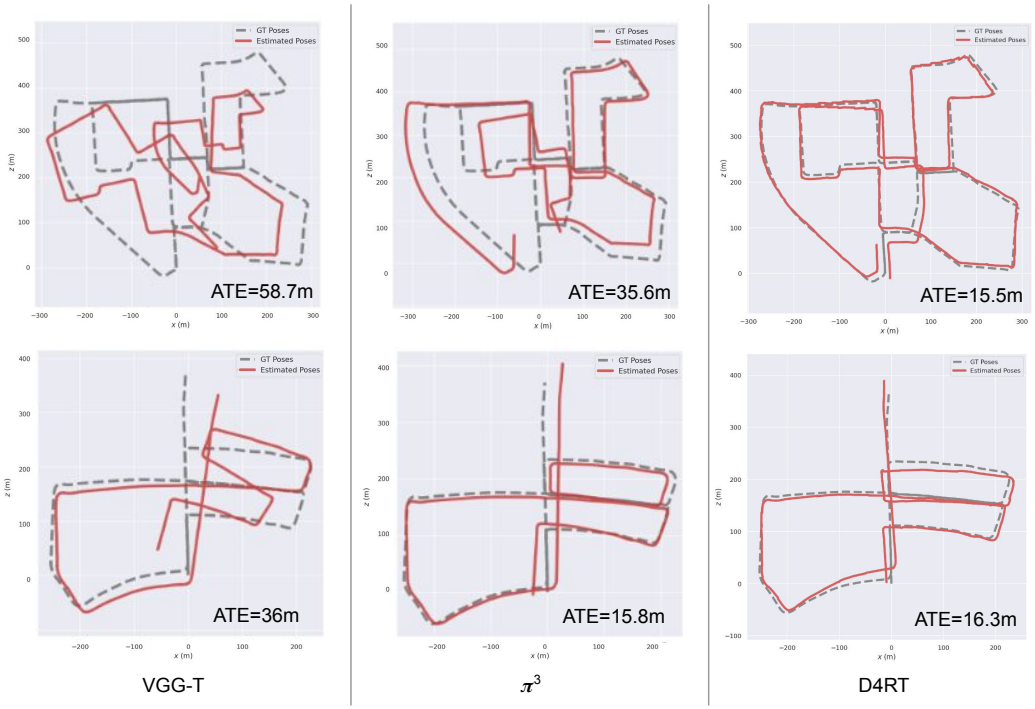
Training queries. Queries are sampled from available ground truth point trajectories. To focus the model on challenging regions, 30% of the queries (along dimensions u, v) are sampled near depth discontinuities or motion boundaries, which are pre-computed using a Sobel filter on depth maps. We find this biased sampling to be critical: removing the bias towards edges and dynamic objects significantly degrades performance (*e.g.*, Pose Average Error increases from 0.13 to 0.21, and 3D Tracking AJ drops from 0.225 to 0.168 on the Sintel dataset). Timesteps t_{src}, t_{tgt}, and t_{cam} are sampled uniformly at random, except that we enforce $t_{tgt} = t_{cam}$ with probability 0.4 to improve downstream performance.

B. Generalization to Long Videos

We implement a long-sequence processing algorithm by partitioning videos into overlapping segments. We then align these segments by estimating Sim(3) transformations using the Umeyama algorithm [49], based on the top 85% of points with the highest confidence in the overlapping regions. The process is similar to the first stage proposed in VGGT-Long [8], we omit loop detection and optimization stage in [8] to directly evaluate the reconstruction model's raw precision. As shown in Fig. 8 on KITTI [16], our model yields the best ATE in the first example, significantly outperforming VGG-T and π^3 by a large margin. On the second example, we significantly outperform VGG-T and produce a result on par with π^3.



(a) Visualization of reconstruction of 1000 frames from KITTI sequence 00.



(b) Comparison of raw chunked prediction alignment results against VGG-T and π^3 baselines (no loop closure).

Figure 8. **Long sequence results on KITTI sequences** – We align and stitch predictions from overlapped chunks by Umeyama algorithm, without the loop detection and global optimization proposed in [8]. Our model obtains consistently better results than VGG-T, and significantly better results than π^3 on sequence "00" (top row).

C. High-Resolution Decoding with Subpixel Precision

A key advantage of D4RT's architecture is the decoupling of the global scene encoding from the point-wise decoding. As the query coordinates (u, v) are defined in a continuous normalized space $[0, 1]^2$, our decoder is able to probe the scene at arbitrary resolutions, independent of the resolution of the Global Scene Representation F.

We explore this capability in Tab. 10. To quantify the preservation of high-frequency details, we report the Pseudo Depth Boundary Error accuracy ($\epsilon^{\text{acc}}_{\text{PDBE}}$) proposed by Pham et al. [38], in addition to standard depth metrics. We compare four configurations using a ViT-g encoder fixed at a resolution of 256×256. The baseline output at the encoder's native resolution (**Config** ①) is naturally coarse. Incorporating the local appearance patch mechanism described in the main paper (**Config** ②) allows for the recovery of finer low-level structures, although pixelation artifacts persist. We further leverage the continuous nature of our query mechanism to predict at the *original* resolution. While naive dense querying (**Config** ③) recovers smoother edges, it still fails at recovering high frequencies.

In **Config** ④, we extract the local RGB patches from the source frames at their *original* resolution for decoding.

The significant drop in $\epsilon^{\text{acc}}_{\text{PDBE}}$ demonstrates how this allows D4RT to recover finer details. Importantly, the local patch is critical for recovering high-frequency boundary details but has less impact on smooth regions, which dominate global metrics like AbsRel. We show qualitative results in Fig. 9 where we observe that hair strands and object boundaries are resolved much more accurately, while smooth, texture-less regions remain robust and consistent regardless of the patch.

Config.	Encoder Resolution	RGB Patch	Output Resolution	RGB patch Resolution	Scale		Scale and Shift	
					AbsRel ↓	$\epsilon^{\text{acc}}_{\text{PDBE}}$ ↓	AbsRel ↓	$\epsilon^{\text{acc}}_{\text{PDBE}}$ ↓
①	256×256	✗	256×256	256×256	0.254	3.323	0.219	3.307
②	256×256	✓	256×256	256×256	0.218	2.254	0.179	2.243
③	256×256	✓	*Original*	256×256	0.217	2.266	0.178	2.258
④	256×256	✓	*Original*	*Original*	0.220	2.193	0.176	2.185

Table 10. **Quantitative impact of query density and patch fidelity** – Feeding RGB patches from the high-resolution video into the decoder (Config ④) yields significantly sharper edges in depth maps as measured by $\epsilon^{\text{acc}}_{\text{PDBE}}$.

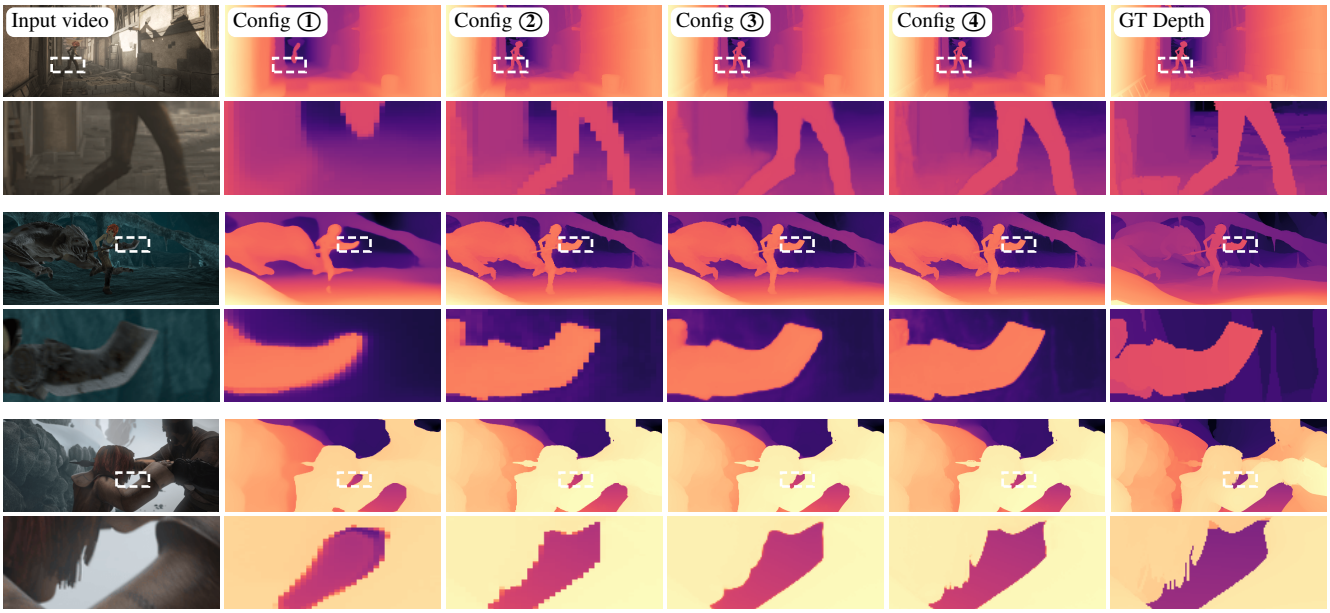


Figure 9. **Visualizing sub-pixel detail recovery** – We propose a visual comparison of the different high-res configurations. Config ④ achieves the highest fidelity, it preserves sharp edges and recovers fine details—such as the hair in the bottom row—without increasing the computational cost or memory requirements of the overall model.

D. Further Ablations

Pretrained encoder. In Tab. 11, we show our model performance when the video encoder is initialized with random weights compared to using pre-trained VideoMAE [52] weights. We observe significant improvements across the board.

Model weight initialization	Video Depth Estimation		Camera Pose Estimation		
	AbsRel (S) ↓	AbsRel (SS) ↓	ATE ↓	RPE-T ↓	RPE-R ↓
None	0.738	0.520	0.334	0.139	1.126
VideoMAE [52]	0.302	0.257	0.091	0.028	0.245

Table 11. **Model initialization** – Initializing the model with VideoMAE [52] weights leads to significant improvements.

Local RGB patch size. We perform an ablation study on the size of the local RGB patch. As shown in Fig. 10, our results indicate that the patch size 9×9 yields the best overall performance across camera pose and depth estimation tasks.

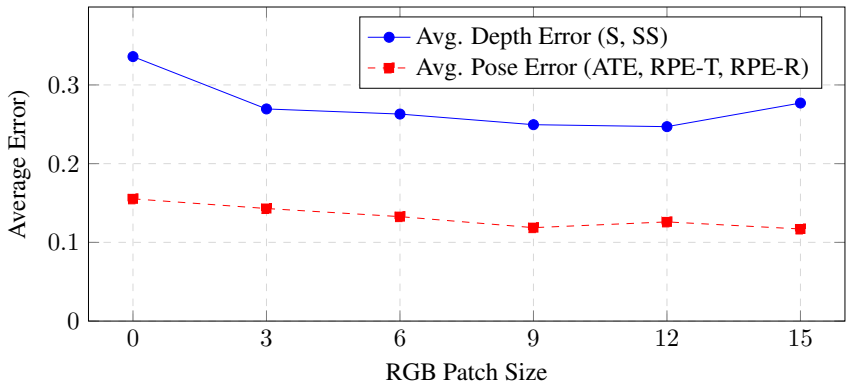


Figure 10. **Ablation study on RGB patch size** – The plot shows the average error for depth and pose estimation on Sintel. A patch size between 9 and 12 yields the best performance for both tasks.

Training data. We evaluate D4RT when trained exclusively on publicly available datasets. As shown in Tab. 12, this ablation confirms that the core architectural design of D4RT – rather than simply scaling proprietary data – is the primary driver of its strong performance.

Model	Training Datasets	ATE ↓		RPE-T ↓		RPE-R ↓		Point Cloud L1 ↓		SSI AbsRel ↓			3D Tracking AJ ↑		
		Sintel	Scannet	Sintel	Scannet	Sintel	Scannet	Sintel	Scannet	Sintel	Scannet	KITTI	DriveTrack	ADT	PStudio
VGGT [51]	16 public + internal	0.168	0.16	0.056	0.012	0.428	0.316	1.582	0.063	0.247	0.094	0.067	-	-	-
STv2 [59]	26 public + internal	0.126	0.018	0.053	0.012	1.052	0.324	2.609	0.036	0.175	0.175	0.025	0.064	0.26	0.097
π^3 [57]	22 public + internal	0.086	0.015	0.039	0.010	0.248	0.291	1.375	0.030	0.163	0.019	0.053	-	-	-
CoTracker [25]+VGGT	16 public + internal	-	-	-	-	-	-	-	-	-	-	-	0.129	0.132	0.045
D4RT	14 public	0.077	0.014	0.026	0.010	0.204	0.307	1.098	0.030	0.198	0.020	0.053	0.306	0.264	0.284
	14 public + internal	0.080	0.014	0.028	0.010	0.177	0.307	0.909	0.028	0.150	0.019	0.051	0.309	0.310	0.329

Table 12. **D4RT with public-only training data.** When trained exclusively on public data, D4RT still outperforms competing methods across pose estimation, point cloud reconstruction, and 3D tracking. This is achieved despite being trained on significantly less data and without hyperparameter tuning for this restricted regime. We note that competing models benefit from large-scale proprietary data (*e.g.*, internal Objaverse in VGGT, OmniWorld in π^3) or were initialized from checkpoints derived from internal data (*e.g.*, STv2 and π^3).

E. Further Qualitative Results

Complementing the qualitative visualizations in the main text, we provide additional examples of our reconstruction results in Fig. 11, along with further baseline comparisons in Fig. 12 and Fig. 13.

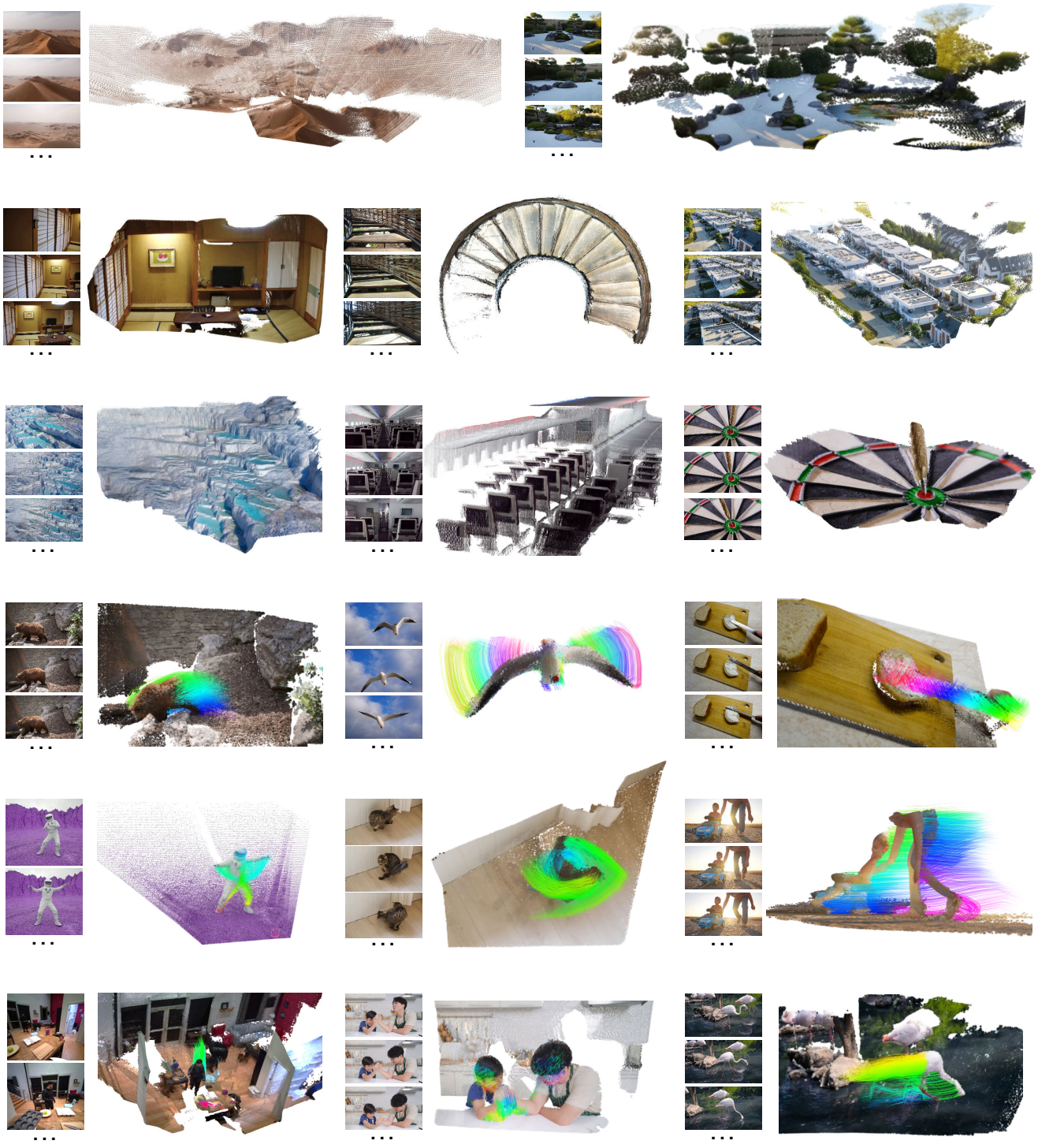


Figure 11. **Additional visualizations of D4RT** – D4RT produces accurate reconstructions for both static environments (top three rows) and dynamic sequences (bottom three rows).

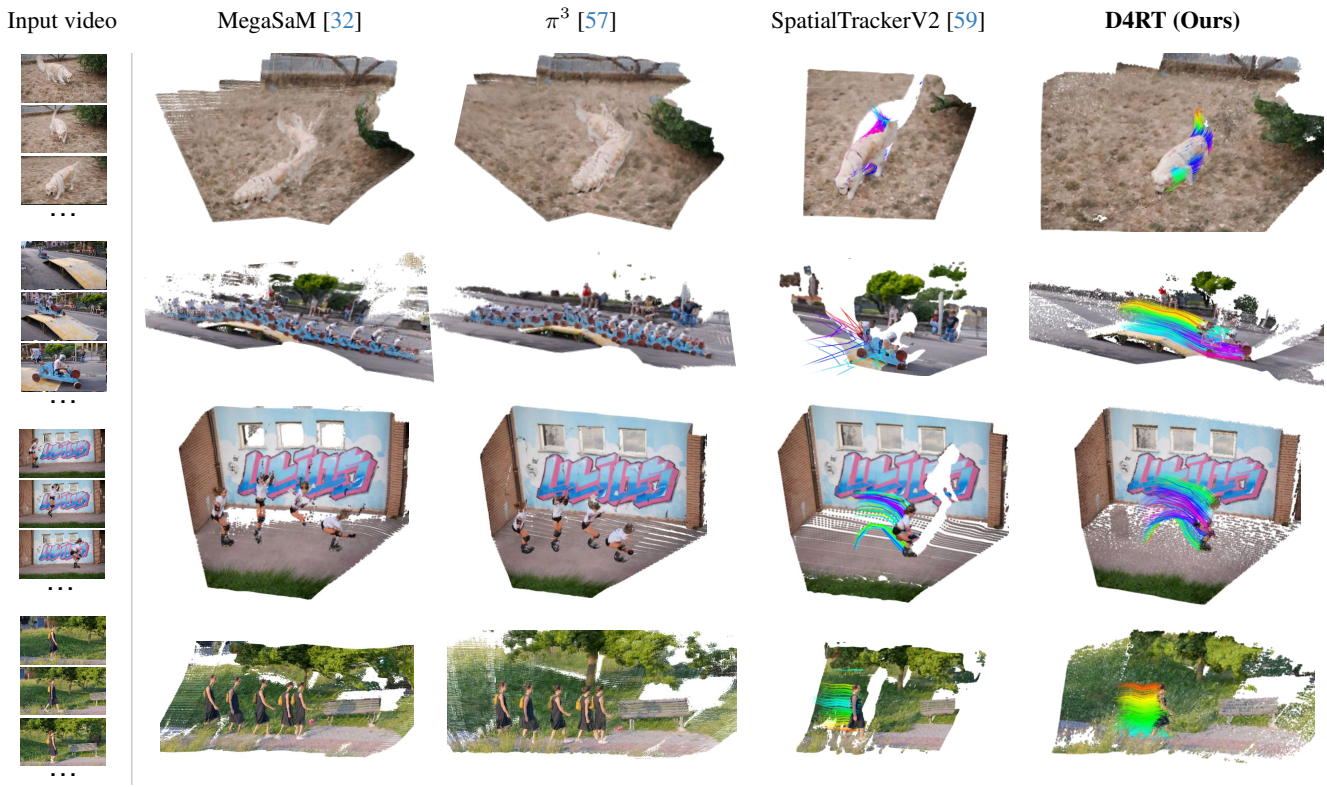


Figure 12. **Additional reconstruction results across methods** – Pure reconstruction methods (MegaSaM and π^3) are visualized as accumulated point clouds. For SpatialTrackerV2, we visualize sparse tracks on a representative frame. In contrast, D4RT reconstructs a complete 4D scene representation, tracking *all* pixels across the entire video.

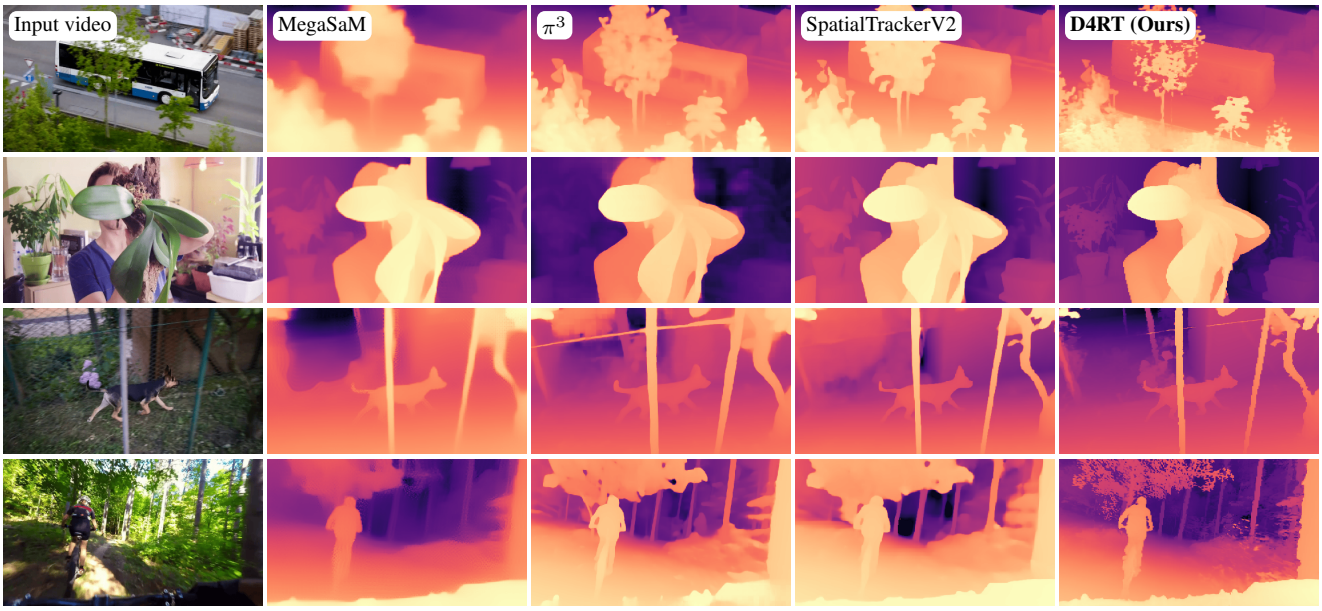


Figure 13. **Qualitative depth comparison across methods** – D4RT is able to perform dense depth estimation with finer details than current state-of-the-art methods, preserving geometric accuracy even in scenarios with large motion blur.