Any4D: Unified Feed-Forward Metric 4D Reconstruction

any-4d.github.io

Jay Karhade Nikhil Keetha Yuchen Zhang Tanisha Gupta
Akash Sharma Sebastian Scherer Deva Ramanan

Carnegie Mellon University

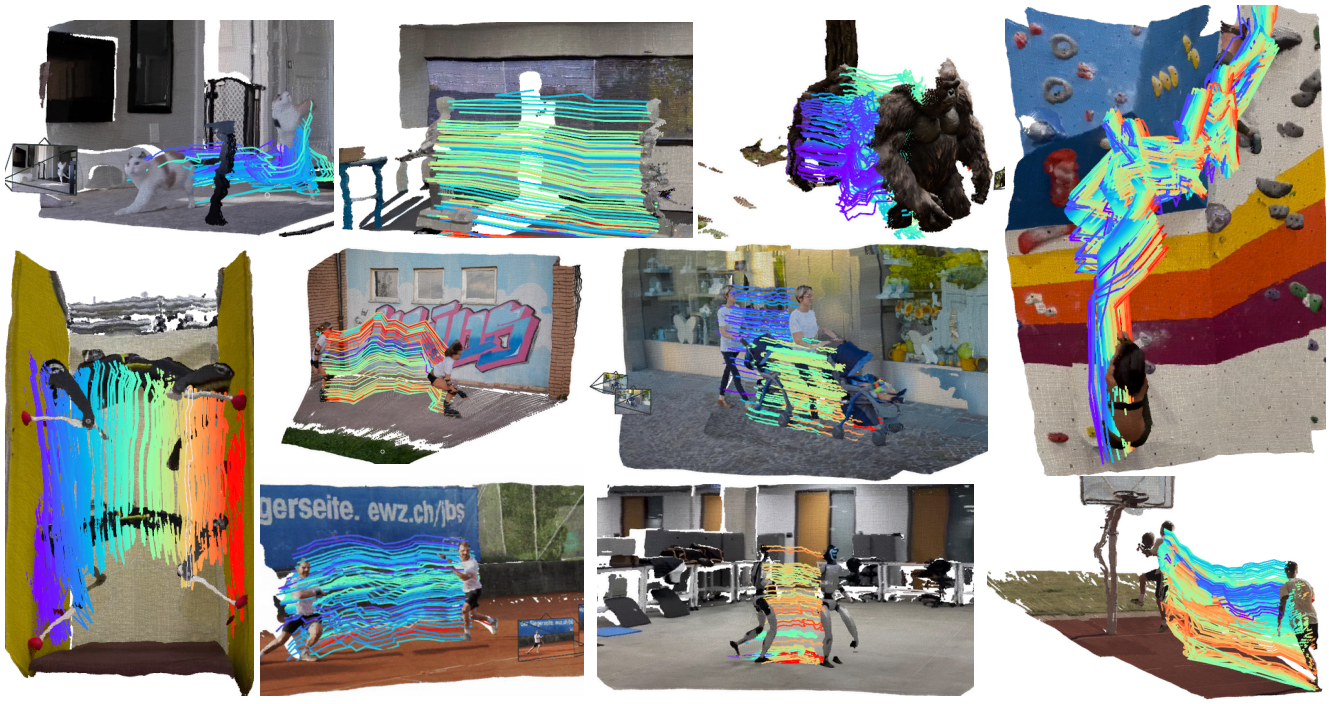

Figure 1. **Any4D is a flexible feed-forward model capable of producing dense metric 4D reconstructions using N frames as input.** Any4D is up to 15× faster and 3× better than prior state-of-the-art, where performance can be further boosted by using diverse sensors as input. Note that Any4D produces dense 3D tracking vectors, but here we visualize the sparse 3D motion tracks for simplicity.

Abstract

We present Any4D, a scalable multi-view transformer for metric-scale, dense feed-forward 4D reconstruction. Any4D directly generates per-pixel motion and geometry predictions for N frames, in contrast to prior work that typically focuses on either 2-view dense scene flow or sparse 3D point tracking. Moreover, unlike other recent methods for 4D reconstruction from monocular RGB videos, Any4D can process additional modalities and sensors such as RGB-D frames, IMU-based egomotion, and Radar Doppler measurements, when available. One of the key innovations that allows for such a flexible framework is a modular representation of a 4D scene; specifically, per-view 4D predictions are encoded using a variety of egocentric factors (depthmaps and camera intrinsics) represented in local camera coordinates, and allocentric factors (camera extrinsics and scene flow) represented in global world coordinates. We achieve superior performance across diverse setups - both in terms of accuracy (2 − 3× lower error) and compute efficiency (15× faster) - opening avenues for multiple downstream applications.

1. Introduction

Reconstructing the 4D (3D + t) world from sensor observations is a long-standing goal of computer vision. Such a technology can unlock a wide range of downstream tasks. In generative AI, 4D reconstruction can improve dynamic video synthesis [8, 41, 72, 84], video understanding [24, 96], and the creation of interactive dynamic assets such as VR avatars. In robotics, 4D scene reconstruction can significantly improve predictive control (MPC) for an agent navigating and manipulating in a physical world [44, 52].

Although there has been significant recent progress on 4D reconstruction [16, 27, 37, 41, 60, 78, 92], dynamic reconstruction of in the wild videos remains challenging for many reasons. First, 4D reconstruction is severely under-constrained, requiring simplifying assumptions such as rigid motion, smoothness priors, or a mostly-static world assumption. Second, there is a lack of large-scale 4D datasets. Unlike million-scale video [10] and 3D datasets [2, 3], *reliable high-quality* 4D reconstruction datasets are still limited to a few thousand scenes, primarily obtained via simulation [18, 95]. Third, because 4D reconstruction and tracking is such a challenging problem, progress has been largely achieved by treating dynamic attribute prediction as independent sub tasks (i.e., 3D tracking, video-consistent depth estimation, scene flow estimation, camera pose estimation in dynamic scenes). This focus on sub tasks has led to fragmented datasets and benchmarks that lack consistent 4D definitions and annotations. This is unsatisfying because all sub tasks observe the same underlying 4D world!

To create a universal system that can reliably work on in the wild videos, we seek to address the following desiderata: a) **efficiency**: much prior work often makes use of iterative optimization-based methods as a post-processing step that maybe too slow for real-time deployment. b) **multi-modality**: Many robotic platforms use additional sensors beyond cameras, but most prior work fails to exploit such diverse configurations. c) **metric scale outputs**: while existing 4D reconstruction methods produce outputs in a normalized coordinate frame, physical agents undeniably operate in the metric-scale physical world.

Taking a step in this direction, `Any4D` is a unified and scalable model with the following 3 core contributions:

- **Dense Metric-Scale 4D Reconstruction**: `Any4D` predicts the dense geometry and motion of the scene in metric coordinates, unlike existing methods that can reconstruct only up-to-scale or sparse tracks. We propose a factored 4D representation consisting of per-view *allocentric factors* (for scene flow and poses) and *egocentric factors* (for intrinsics and depth). This factored 4D representation allows us to train on diverse datasets with partial annotations, including metric-scale 3D reconstruction datasets without motion annotations, and non-metric datasets with motion annotations.
- **Flexible Multi-Modal Inputs**: When available, `Any4D` can further improve its 4D reconstruction by exploiting additional input modalities like depth from RGBD sensors, camera poses from IMUs or doppler velocity from RADARs compared to image-only 4D reconstruction.
- **Efficient Inference**: `Any4D` infers both geometry and motion from N video frames in a single feed-forward pass, bypassing existing work that only predict motion for 2 frame inputs or require computationally-expensive optimization, making `Any4D` up to $15\times$ faster than the next best performing method.

2. Related Work

Reconstruction of Dynamic Scenes: Reconstruction and camera pose estimation for static scenes has a rich history. It has been studied as Simultaneous Location and Mapping (SLAM) [13, 15, 31, 33, 50, 65] when visual observations occur in a temporal sequence, and as structure-from-motion (SFM) [1, 62, 64, 71] otherwise. Since traditional optimization-based reconstruction is at odds with dynamics reconstruction, many approaches relied on ad-hoc semantic and motion masks to discard dynamic regions of a scene [6, 21, 36, 57]. Subsequently, advances in data-driven monocular depth [14, 58, 59, 88] and optical flow [67, 94] estimation have not only enabled data-driven static reconstruction methods [68], but have also sparked research [34, 37, 40, 45, 63, 84] in dynamic scene reconstruction. Although methods such as MegaSaM [40] are promising, they rely on per-scene optimization, making them ill-suited for real-time use. More recently, following the success of end-to-end methods[26, 80], methods such as MonST3R [92] handle dynamic scenes by making independent per-frame predictions. However, they still require post-hoc optimization to establish explicit correspondences. To alleviate this, [32, 77, 83] also show the potential of feed-forward multi-view inference from a set of images. Following this line of work, `Any4D` is a feed-forward model that predicts camera poses, dense 3D motion (as scene flow) and geometry (as pointmaps), fully describing a dynamic scene captured by a set of N frames in its entirety.

Scene Flow: Scene flow was introduced in [75] as the problem of recovering the 3D motion vector field for every point on every surface observed in a scene. Any optical flow then is the perspective projection of scene flow onto the camera plane. Subsequently, it has been studied through a wide range of approaches, ranging from variational methods [5, 25, 55] to learning-based supervised methods [42, 81] and self-supervised methods [49, 56, 85]. Despite these advances, solutions to scene flow estimation have largely been tailored to specific downstream use cases, exploiting access to privileged information. In autonomous vehicles (AVs), scene flow approaches [11, 74] typically access sensor pose through inertial and proprioceptive sensors. Similarly, RAFT-3D [69] assumes access to depth. Recently, [41] proposed to build upon [77] for scene flow and view synthesis. However, in the spectrum of dynamism in a scene, we observe that all the above scene flow methods are limited to simplistic scenes like [7, 47, 48] with minimal dynamic motion. Our model is instead capable of directly predicting scene flow in the *allocentric* coordinate frame .

3D Tracking: While scene flow has been defined for short-range motion typically for a pair of image frames,

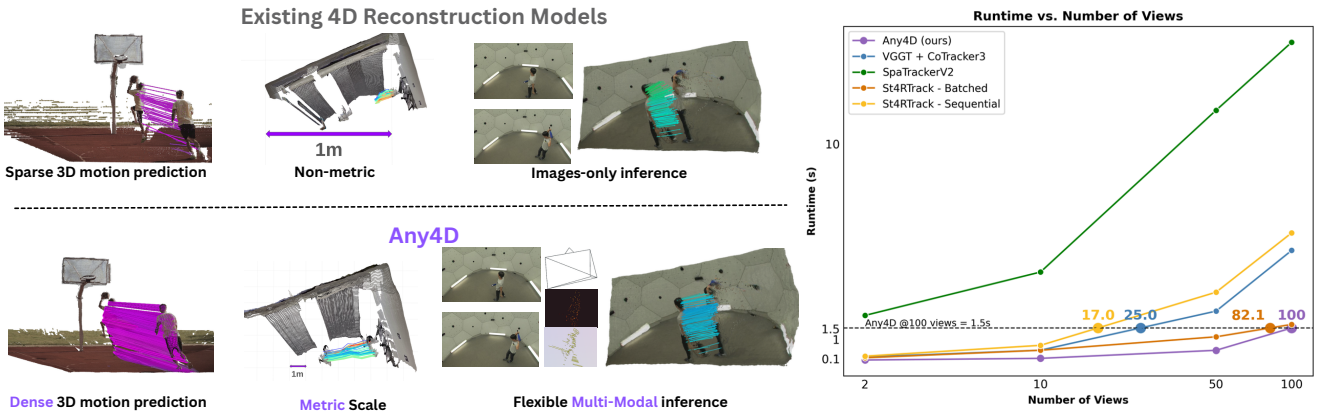

Figure 2. **Any4D's unified capabilities overcome major limitations of existing 4D reconstruction models [16, 87].**

point tracking [61] is the task of tracking a pixel trajectory over a long time horizon. Following methods such as [20, 29, 30] that show the success of 2D point-tracking, TAPVID-3D [35] introduced a benchmark to address the problem of 3D point tracking. Subsequently, [51, 86, 91] proposed methods for obtaining 3D point tracks and improving this benchmark. However, [51, 86] focus on ego-centric 3D point-tracking, unlike Any4D which regresses *allocentric* 3D point-tracks. [87] recently proposed a method for allocentric 3D point tracking, by jointly optimizing camera motion, 2D and 3D point tracks. However, it is important to note that these methods can only track sparse points and require knowledge of poses and depth, either from ground truth or from running off the shelf models, limiting real-time deployment. In contrast, Any4D natively supports dense 3D point tracking and can take flexible inputs, allowing adoption on a range of platforms.

Concurrent and Recent Work: We acknowledge concurrent works [16, 19, 27, 41, 43, 66, 93] that focus on predicting geometry and motion, with [27, 41] being limited to extremely small camera and scene motion. Any4D differs from all concurrent methods in 3 ways (see Fig. 2). **First**, all concurrent methods require multiple feedforward passes to infer the motion, whereas Any4D adopts a scalable architecture inspired by [32] and performs a single feedforward pass for all image frames at once. **Second**, these methods only accept image inputs, while Any4D which can exploit diverse multi-modal inputs. **Third**, unlike the concurrent works, we are the only method to produce metric scale 4D reconstructions. We believe that the open-source release of Any4D will set a strong foundation for the community.

3. Any4D

Any4D is a transformer that takes flexible multi-modal inputs and outputs a dense metric-scale 4D reconstruction in a single feed-forward pass. In addition to a set RGB images $\mathbf{I} \triangleq \{I_i\}_{i=1}^{N}$, Any4D can use auxiliary multi-modal sensor inputs which we denote as $\mathbf{O} \triangleq (O_i)_{i=1}^{N}$. Then, our model can be represented as a function that maps these inputs to a factored output representation as follows:

$$(\tilde{s}, \{\tilde{R}_i, \tilde{D}_i, \tilde{T}_i, \tilde{F}_i\}_{i=1}^{N}) = \text{Any4D}(\mathbf{I}, \mathbf{O}), \quad (1)$$

where the optional inputs $\mathbf{O}$ can contain information such as depth maps, camera intrinsics, camera poses from external systems or IMU and Doppler velocity from RADAR.

Model predictions are denoted with $\sim$ in order to differentiate them from ground-truth targets or auxiliary inputs. Predictions include a metric scaling factor $\tilde{s} \in \mathbb{R}$ for the entire scene, egocentric quantities predicted in the local camera coordinate frame, namely

- ray directions for each view, i.e., $\tilde{R}_i \in \mathbb{R}^{3\times H\times W}$
- scale-normalized depth along the rays for each view, i.e., $\tilde{D}_i \in \mathbb{R}^{1\times H\times W}$.

and *allocentric* quantities predicted in a consistent world coordinate frame, namely

- scale-normalized forward scene flow from the first view to all other views, i.e., $\tilde{F}_i \in \mathbb{R}^{3\times H\times W}$.
- camera pose of each view in the coordinate system of the first view, i.e., $\tilde{T}_i \triangleq [p_i, q_i] \in \mathbb{R}^7$ represented using a scale-normalized translation vector and quaternion.

Now, given these output factors from Any4D, one can recover the predicted metric-scale geometry $\tilde{\mathbf{G}}_i$ in the form of pointmaps [80] by composing the individual quantities as

$$\tilde{\mathbf{G}}_i = \tilde{s} \cdot \tilde{T}_i \cdot \tilde{R}_i \cdot \tilde{D}_i \quad \in \mathbb{R}^{3\times H\times W}. \quad (2)$$

Similarly, allocentric scene flow $\tilde{M}_i$ and pointmaps after motion $\tilde{\mathbf{G}'}_i$ can be recovered as

$$\tilde{M}_i = \tilde{s} \cdot \tilde{F}_i \quad \in \mathbb{R}^{3\times H\times W} \quad (3)$$

$$\tilde{\mathbf{G}'}_i = \tilde{\mathbf{G}}_i + \tilde{M}_i \quad \in \mathbb{R}^{3\times H\times W} \quad (4)$$

We show in Sec. 4, that this parameterization of motion and geometry is optimal for model performance compared to other parameterizations.

3.1. Architecture

Any4D largely follows a multi-view transformer architecture, similar to [32] (see Fig. 3). Conceptually, it can be

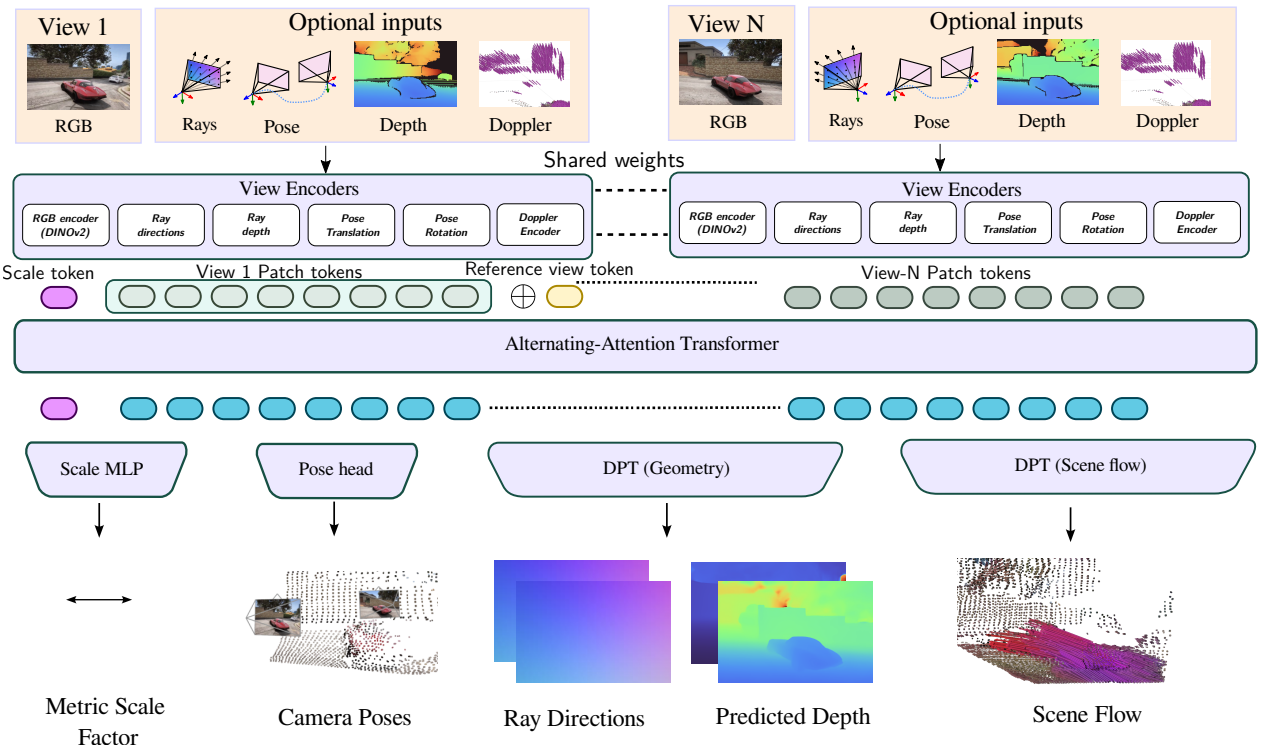

Figure 3. **Any4D predicts a factorized dense metric 4D reconstruction** represented as a global metric scale, per-view egocentric factors (depth maps and ray directions) and per-view allocentric factors (forward scene flow and camera poses) as explained in Sec. 3. Any4D is a N-view transformer, consisting of modality-specific encoders, followed by an alternating-attention transformer to produce contextual patch embeddings. The output tokens from the transformer are then decoded using individual decoders specific to each factor.

separated into three sections: a) modality specific input encoders, b) a multi-view transformer backbone that attends to the tokens from all views, and c) output representation heads which decode the tokens into the factorized output variables for each view.

Multi-Modal Input Encoders: RGB inputs $\mathbf{I}$ and auxiliary multi-modal sensor inputs $\mathbf{O}$ are mapped to view-specific patch tokens through multi-modal view encoders with shared weights for input views which map to a $\mathbb{R}^{1024\times H/14\times W/14}$ feature space. We follow the design choices in [32] for RGB, depth, camera poses and intrinsics encoders, and additionally, add a CNN encoder to encode doppler velocity. We summarize these below:

- **RGB Images**: DINOv2 [53] for encoding images, to extract the layer-normalized patch-level features from the final layer of DINOv2 ViT-Large, $F_{\mathrm{I}} \in \mathbb{R}^{1024\times H/14\times W/14}$.
- **Depth Images:** A shallow CNN encoder is used to encode depth images, where we normalize the input depth before passing it to the depth encoder. The normalization factor is computed independently for each local view.
- **Doppler Velocity:** Doppler velocity is also encoded using a CNN-based encoder. However, here the normalization factor for encoding the doppler velocity is computed from the first-view pointmap and shared globally.
- **Camera Intrinsics:** Camera intrinsics are encoded as rays, and also use a CNN that maps the 3-channel ray-directions into the same 1024-dimensional latent space.
- **Camera Poses:** Two 4-layer MLP encoders are used for camera rotation and translation that map normalized input poses to latent vectors, $f_{\text{rot}} \in \mathbb{R}^{1024}$ and $f_{\text{trans}} \in \mathbb{R}^{1024}$. The normalization factor for pose translation is computed globally across all views, and a positional encoding is used to indicate the reference view $p_{\text{ref}} \in \mathbb{R}^{1024}$.
- **Metric Scale Token**: For metric-scale data, the depth scale and pose scale obtained from normalizing depth and pose are first transformed to log-scale and then encoded using a 4-layer MLP, yielding two $\mathbb{R}^{1024}$ latent features.

All multimodal encodings thus obtained are aggregated via *summation* into a per-view embedding $F_{view} \in \mathbb{R}^{1024\times H/14\times W/14}$, which are flattened into tokens, along with an added learnable token to learn the metric-scale.

Transformer Backbone: We use an alternating-attention transformer [77] across the views, consisting of 12 blocks of 12 multi-head attention and MLPs. Each transformer block processes tokens with a latent dimension of 768 and contains MLPs with a ratio of 4, similar to the ViT-Base architecture. Furthermore, consistent with [32] we choose to not use 2-D rotary positional encoding (RoPE) for the inputs, and also employ Flash Attention [12] for efficiency.

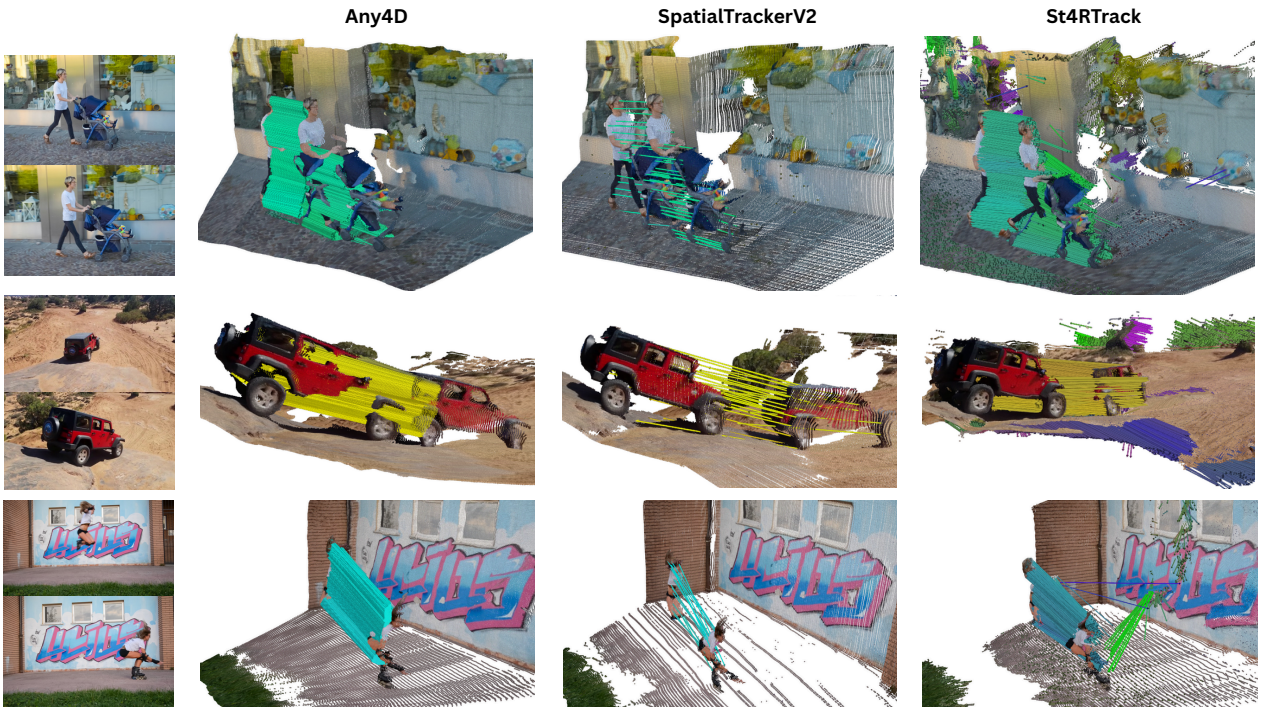

Figure 4. **Any4D provides dense and precise motion estimation, where on the other hand, state-of-the-art baselines either produce reliable but sparse motion (SpatialTrackerV2 [87]) or dense per-pixel motion that is not accurate (St4RTrack [16]).** For SpatialTrackerV2, we are only able to uniformly query a maximum of 2500 points with a H100 GPU using 80 gigabytes of GPU memory. Note that we don't use any pre-computed segmentation mask but purely threshold our scene flow output to get a binary motion mask. St4RTrack cannot produce good binary motion masks due to incorrect scene flow predictions on object boundaries and the background.

Output Representation Heads: We decode the multi-view tokens from the transformer backbone into a factored output representation as follows:

- **Geometry DPT Head**: We use a dense prediction transformer (DPT) [58] head to predict per-view ray directions $\tilde{R}_i$, up-to-scale ray depths $\tilde{D}_i$, and confidence masks.
- **Motion DPT Head**: A second DPT head is tasked to predict per-view forward *allocentric* scene flow $\tilde{F}_i$. The scene flow represents motion of points in the reference view-0 to all other views, and naturally enables long-range 3D tracking (see supplementary for details).
- **Pose Decoder**: The pose decoder is an average-pooling-based CNN decoder that predicts per-view, up-to-scale translations and quaternions $\tilde{T}_i \triangleq [p_i, q_i]$.
- **Metric Scale Decoder**: We use a lightweight MLP decoder to predict the log scale metric scaling factor, which is subsequently exponentiated.

3.2. Training Details

Datasets: Despite recent efforts [27], there is a lack of large-scale datasets that contain dynamic scene motion annotations. In fact, *reliable, high-quality* scene flow annotations are sparse and only available from simulation engines [18, 28]. We address this challenge in this work by a) finetuning large-scale pretrained geometry models and b) training with partial supervision. Owing to our factored representation, we are able to train on a mixture of both geometry-only and dynamic datasets, where they can be synthetic or real-world with varying sparsity of labels: BlendedMVS [89], MegaDepth [39], ScanNet++ [90], VKITTI2 [7], ParallelDomain4D [73], Waymo-DriveTrack [4], SAIL-VOS3D [23] PointOdyssey [95], Dynamic Replica [28] and Kubric [18] data generated by CoTracker3[29] and GCD[73]. Please refer to Supplementary for detailed information of all datasets used for training.

Training with Multi-Modal Conditioning: We preprocess the datasets and generate multi-modal inputs offline for faster training. Geometric inputs consisting of poses, depths and intrinsics are directly taken from the dataset annotations. To simulate doppler velocity, we take the radial component of egocentric scene flow between data pairs. During training, multi-modal conditioning is applied with a probability of 0.7, i.e., 70% of training iterations include multi-modal inputs alongside images. Additionally, we ensure that individual modalities (depth, rays, poses, and doppler) are independently removed with a probability of 0.5 to promote effective learning in flexible input configurations. Finally, we initialize our network with MapAnything weights [32]. For each training batch, we sample up to 4 views from

the datasets and train on 1 H100 node for 100 epochs.

Losses: Any4D is trained using a combination of geometric and motion losses based on the type of annotation available. Ray directions representing the camera intrinsics and quaternions are scale-agnostic, and therefore can be supervised via simple regression losses:

$$\mathcal{L}_{\text{rays}} \triangleq \sum_{i=1}^{N} \|R_i - \tilde{R}_i\| \tag{5}$$

$$\mathcal{L}_{\text{rotation}} \triangleq \sum_{i=1}^{N} \min(\|q_i - \tilde{q}_i\|, \| - q_i + \tilde{q}_i\|). \tag{6}$$

On the other hand, geometric quantities such as camera translations t_i, ray depths D_i and scene flow F_i are predicted in a scale-normalized coordinate frame. Following prior work [32, 38, 80], we use the ground-truth validity masks V_i and pointmaps X_i and compute the ground-truth scale as the average euclidean distance of valid points with respect to the world origin (given by the first view camera frame): $z = \|\{X_i[V_i]\}_i^N\| / \sum_i^N V_i$. To compute scale-invariant losses, we also compute a scale factor derived from our predictions $\tilde{z} = \|\{\tilde{X}_i[V_i]\}_i^N\| / \sum_i^N V_i$:

$$\mathcal{L}_{\text{trans}} \triangleq \sum_{i}^{N} \left\| \frac{t_i}{z_i} - \frac{\tilde{t}_i}{\tilde{z}_i} \right\|, \tag{7}$$

$$\mathcal{L}_{depth} \triangleq \sum_{i}^{N} \left\| f_{\log}\left(\frac{D_i}{z_i}\right) - f_{\log}\left(\frac{\tilde{D}_i}{\tilde{z}_i}\right) \right\| \tag{8}$$

where $f_{\log}(x) \triangleq (x/\|x\|)\log(1+\|x\|)$ converts quantities to log-space for numerical stability. A pointmap loss is also applied to the composed geometric predictions as follows:

$$\mathcal{L}_{pm} \triangleq \sum_{i}^{N} \left\| f_{\log}\left(\frac{X_i}{z_i}\right) - f_{\log}\left(\frac{\tilde{X}_i}{\tilde{z}_i}\right) \right\| \tag{9}$$

Similarly, scene flow is also supervised in a scale-invariant manner. We find that scene flow loss is dominated by static points since most of the scene is static. Therefore, we find it is crucial to calculate a static-dynamic motion mask M from the ground truth scene flow, and upweight the scene flow loss in the dynamic regions by 10x more compared to static regions:

$$\mathcal{L}_{sf} \triangleq \sum_{i}^{N} M \cdot \left\| f_{\log}\left(\frac{F_i}{z_i}\right) - f_{\log}\left(\frac{\tilde{F}_i}{\tilde{z}_i}\right) \right\| \tag{10}$$

Finally, the predicted metric scale factor $\tilde{s}$ is also supervised in the log space as follows: $\mathcal{L}_{\text{scale}} \triangleq \|f_{\log}(z) - f_{\log}(\tilde{s} \cdot \text{sg}(\tilde{z})\|$, where sg denotes the stop-gradient operation and prevents the scale supervision from affecting other predicted quantities. The final loss is expressed as:

$$\mathcal{L} = \mathcal{L}_{\text{trans}} + \mathcal{L}_{\text{rot}} + \mathcal{L}_{\text{rays}} + \mathcal{L}_{\text{depth}} + \mathcal{L}_{\text{pm}} + \mathcal{L}_{\text{sf}} + \mathcal{L}_{\text{mask}} \tag{11}$$

4. Results & Analysis

We evaluate Any4D on diverse benchmarking setups specifically designed for allocentric 4D reconstruction, and compare against state-of-the-art (SOTA) methods.

3D Tracking: There is a lack of standard and unified benchmarks for evaluating 4D reconstruction in the existing literature. To create allocentric 3D tracking benchmarks, we follow [16] and repurpose existing 3D tracking benchmark, particularly TAPVID-3D [35]. However, TAPVID-3D have their own limitations: the Aria Digital Twin (ADT) sequences are largely static, Parallel-Studio sequences contain fixed-camera viewpoints, while the DriveTrack sequences are extremely sparse. Hence, we choose to drop ADT, and keep Parallel-Studio and Drive-Track benchmarking sequences. We also add unseen held-out sequences from Dynamic Replica [28] and a zero-shot dataset LSFOdyssey [76], both of which contain camera motion along with 3D tracking labels. The final benchmark contains $\sim$ 170 sequences across 4 datasets of up to 64 frames in length. We evaluate Any4D against SOTA 3D trackers SpatialTrackerV2 [87] and St4RTrack [16]. We also compose 3D reconstruction models [32, 38, 77, 92] with 2D tracks from CoTracker3 [29] for comparison.

We use standard benchmarking protocols [16, 35, 69, 87] to evaluate the quality of our 4D reconstruction. Following [16], we first perform median-scaling to align to metric space. We report average percent of points within delta for 3D points after motion (APD) and inlier percentage τ for scene flow. We also report End Point Error (EPE) for 3D points after motion (dynamic points) and 3D scene flow vectors. APD and τ are defined as:

$$\text{APD} = \sum_{i,t} \mathbb{1} \cdot \left(\left\| P_{i,t} - \tilde{P}_{i,t} \right\| < \delta_{3\text{D}} \right) \tag{12}$$

$$\tau = \sum_{i,t} \mathbb{1} \cdot \left(\left\| F_{i,t} - \tilde{F}_{i,t} \right\| < 0.1\text{m} \right) \tag{13}$$

where $\tilde{P}_i$ represents the predicted 3D point after motion and $\tilde{F}_i$ is the corresponding scene flow vector at time t. For APD, we use thresholds $\delta_{3D} \in \{0.1, 0.3, 0.5, 1.0\}$ m. As evident in Tab. 1, Any4D shows state-of-the-art performance across all datasets. Furthermore, it is $15\times$ faster than the closest performing method, SpatialTrackerV2. This is further reinforced qualitatively in Fig. 4.

Dense Scene Flow: We construct allocentric scene flow benchmarks by repurposing 2 egocentric scene flow benchmarking datasets: VKITTI-2 [7] and Kubric-4D [72]. While scene flow in VKITTI-2 is limited to small consecutive frame motion, we can simulate scene flow across 60 frames and 16 camera viewpoints from Kubric4D (GCD). Hence we create 2 variants for Kubric4D (GCD): (a) scene flow from static camera movement and (b) scene flow from wide-baseline dynamic camera movement. Importantly, all pairs from both datasets are from held-out scenes to ensure there is no data leak from the training datasets. We evaluate Any4D against St4RTrack which can predict dense scene flow, and 3D reconstruction method outputs composed with optical flow from SEA-RAFT [82], to calculate *covisible*

Table 1. **Any4D showcases state-of-the-art sparse 3D point tracking, while providing dense motion predictions and being an order of magnitude faster than the closest performing baseline.** We report end-point error (EPE), average points within delta (APD) and inlier ratio at 0.1m (τ) for dynamic points in the benchmark. The runtime is computed on a H100 using 50 frames as input. Best results are **bold**.

		Drive Track [4]				Dynamic Replica [28]				LSFOdyssey [76]				PStudio [35]			
		Dynamic Points		Scene Flow		Dynamic Points		Scene Flow		Dynamic Points		Scene Flow		Dynamic Points		Scene Flow	
Method	**Runtime (s)**	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑
MonST3R + CoTracker3	146.40	16.81	0.44	21.87	0.06	0.81	43.34	0.18	25.99	0.61	50.96	0.41	43.64	0.51	51.87	0.52	21.06
MASt3R + CoTracker3	13.82	17.16	1.22	20.01	0.20	0.40	57.72	0.23	53.98	0.83	45.95	0.62	41.10	0.43	54.11	0.43	14.69
VGGT + CoTracker3	2.31	8.30	4.80	11.69	0.77	0.26	69.12	0.06	89.37	0.47	59.21	0.22	74.11	0.26	69.34	0.17	45.77
MapAnything + CoTracker3	0.73	9.42	2.45	12.88	0.43	0.25	70.51	0.06	**89.59**	0.63	35.51	0.51	58.00	0.63	50.85	0.35	**58.01**
St4RTrack	1.12	11.82	1.03	14.63	0.10	0.17	80.87	0.07	77.90	0.56	48.11	0.25	38.31	0.41	53.12	0.21	28.46
SpatialTrackerV2	11.56	5.45	4.48	10.63	0.10	0.69	62.34	0.06	83.66	0.34	68.37	**0.09**	**78.75**	**0.21**	**74.46**	**0.14**	50.70
Any4D	**0.50**	**3.89**	**7.81**	**3.14**	**1.83**	**0.07**	**93.44**	**0.05**	86.99	**0.27**	**71.70**	0.10	71.41	0.27	67.43	0.19	33.57

Table 2. **Any4D achieves state-of-the-art dense scene flow estimation performance.** We report end-point error (EPE), average points within delta (APD) and inlier ratio at 0.1m (τ) for dynamic points and scene flow across three datasets, where best results are **bold**.

	Kubric-4D Dynamic Camera				Kubric-4D Static Camera				VKITTI-2			
	Dynamic Points		Scene Flow		Dynamic Points		Scene Flow		Dynamic Points		Scene Flow	
Method	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑
MonST3R + SEA-RAFT	5.23	2.20	3.73	14.69	2.26	6.80	1.16	61.79	12.31	0.44	1.21	12.93
MASt3R + SEA-RAFT	6.35	1.92	1.45	13.95	2.85	7.58	1.26	53.62	12.25	2.50	13.05	10.20
VGGT + SEA-RAFT	11.80	3.60	11.76	14.53	1.92	15.01	0.78	86.54	6.57	2.61	0.70	37.63
MapAnything + SEA-RAFT	17.65	2.67	17.70	9.16	2.82	**19.99**	1.75	73.33	8.46	2.42	1.32	13.78
St4RTrack	2.44	5.79	1.70	11.83	2.61	6.53	0.72	20.51	14.71	0.00	0.97	3.37
Any4D	**1.13**	**18.14**	**0.17**	**83.38**	**1.23**	19.53	**0.10**	**87.51**	**4.97**	**11.70**	**0.04**	**93.08**

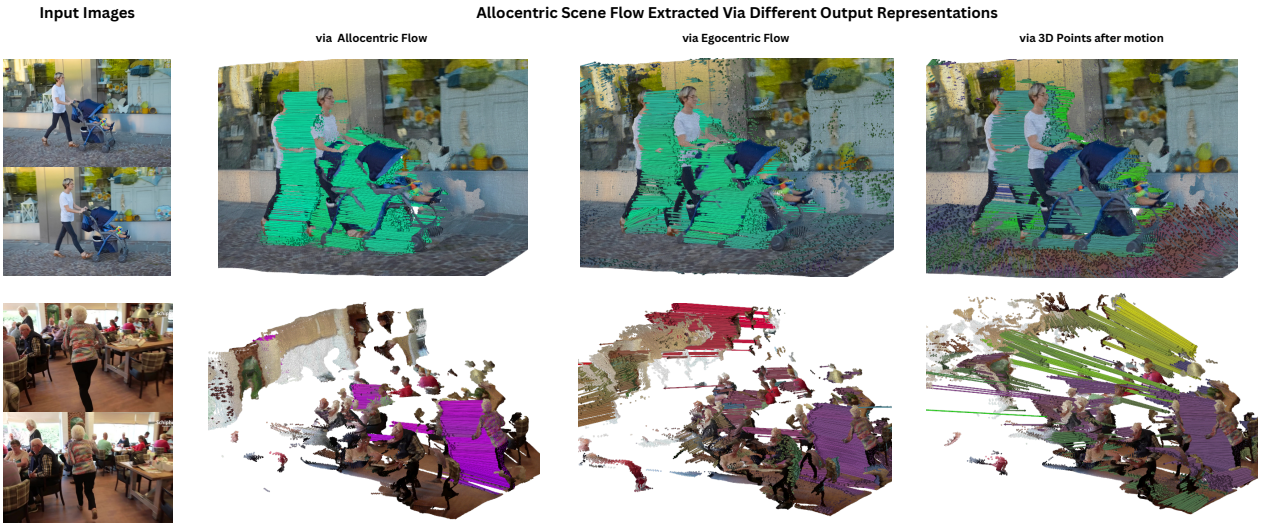

Figure 5. **Scene motion parametrized as *allocentric* scene flow provides the cleanest 4D reconstructions.** We find that other parameterizations such as 3D points after motion (proposed in St4RTrack [16]) provide extreme noise on object boundaries and background.

scene flow. We are unable to run SpatialTrackerV2 or CoTracker3 as they do not support per-pixel point queries and run out-of-memory(OOM). From Tab. 2, we see that Any4D outperforms baselines by $2-3\times$ on average on APD, and by even more on scene-flow metrics.

Video Depth: We also evaluate Any4D on standard video depth benchmarks [17, 46, 54] in Tab. 3, against specialized video depth baselines [9, 22], feed-forward + iterative optimization baselines [40, 80, 87, 92], and single-step feed-forward baselines [32, 77, 79]. Any4D shows state of the art video depth estimation over other single-shot feed-forward inference baselines while being competitive with optimization based and task-specific methods.

Support for Multi-Modal Inputs: Since Any4D can utilize flexible inputs for inference to enhance performance, we study improvements to scene flow on the dense Kubric-4D static benchmark and 3D tracking on LSFOdyssey benchmark by incorporating different input modalities. From Tab. 4, we observe that adding geometry significantly improves APD and EPE for 3D points. Adding

Table 3. **Any4D shows state-of-the-art video depth estimation over other single-step feed-forward baselines.** It is also competitive to iterative/optimization-based methods or ones trained specifically for this task. We report the absolute relative error (rel) and the inlier ratio at 1.25% ($\delta_{1.25}$), where the best is **bold**.

Method	Average		Bonn		KITTI		Sintel	
	rel ↓	$\delta_{1.25}$ ↑	rel ↓	$\delta_{1.25}$ ↑	rel ↓	$\delta_{1.25}$ ↑	rel ↓	$\delta_{1.25}$ ↑
a) Video Depth:								
DepthCrafter	**0.15**	85.23	0.07	97.90	0.11	88.50	**0.27**	**69.30**
VDA	0.17	**86.90**	**0.05**	**98.20**	**0.08**	**95.10**	0.37	67.40
b) Feed-Forward + Iterative Optimization:								
DUSt3R	0.26	75.83	0.17	83.50	0.12	84.90	0.48	59.10
MonST3R	0.16	82.73	0.06	95.40	0.08	93.40	0.34	59.40
MegaSAM	0.10	87.97	0.04	97.70	0.07	91.60	**0.18**	**74.60**
SpatialTrackerV2	**0.09**	**88.80**	**0.03**	**98.80**	**0.05**	**97.30**	0.20	70.30
c) Single-Step Feed-Forward:								
CUT3R	0.21	80.30	**0.07**	95.00	0.10	89.90	0.47	56.00
VGGT	**0.13**	85.85	**0.07**	**97.27**	**0.09**	**94.37**	**0.24**	65.90
MapAnything	0.14	84.97	0.09	94.77	**0.09**	94.26	0.25	65.87
Any4D	**0.13**	**86.28**	**0.07**	**97.27**	**0.09**	93.97	**0.24**	**67.59**

Table 4. **Auxiliary inputs improve the 4D motion estimation performance of Any4D.** We compare different inputs on both dense scene flow (Kubric) and sparse 3D point tracking (LSFOdyssey) benchmarks using end-point error (EPE), average points within delta (APD) and inlier ratio at 0.1m (τ), where best is **bold**. "Geometry" indicates use of depth, intrinsics and poses.

Any4D Inputs	Kubric-4D Static Camera				LSFOdyssey			
	Dynamic Points		Scene Flow		Dynamic Points		Scene Flow	
	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑
Images Only	1.17	21.33	0.11	86.25	0.28	71.47	0.12	68.03
Images + Geometry	**0.23**	80.18	**0.09**	86.26	**0.19**	80.80	0.12	68.71
Images + Doppler	1.17	21.70	0.12	86.90	0.29	71.26	**0.11**	70.32
Images + Geometry + Doppler	**0.23**	**81.72**	**0.09**	**87.27**	**0.19**	**81.10**	**0.11**	**71.37**

doppler further improves scene-flow, with the best performance achieved when all modalities are provided.

Choice of Motion Representation: While allocentric motion F_{allo} is arguably the useful quantity for downstream applications, it is possible to represent the predicted scene flow output in 4 ways:

- **Allocentric Scene Flow:** Directly predicting $\tilde{F}_{\text{allo}}$.
- **Egocentric Scene Flow:** Predicting egocentric scene flow $\tilde{F}_{\text{ego}}$, and using estimated geometry to recover allocentric motion as:
$$\tilde{F}_{\text{allo}} = T_{t\to t+1}(P_0^v + \tilde{F}_{\text{ego}}) - p$$
- **3D Points After Motion:** Predicting view-aligned pointmaps at time 0 and t - P_0^v and P_t^v, and recovering the allocentric motion:
$$\tilde{F}_{\text{allo}} = P_t^v - P_0^v$$
- **Backprojected 2D Flow:** Unprojecting optical flow to obtain covisible scene flow between pointmaps.

We systematically investigate these choices in Tab. 5 and Fig. 5. We find that directly predicting allocentric motion leads to optimal performance not only on *scene flow*, but surprisingly, also on *dynamic pointmaps after motion*, compared to directly predicting points after motion as adopted otherwise in [16]. This is because allocentric flow targets are easier to learn since they are sparse (mostly 0) for mostly-static scenes. In contrast, 3D points after motion and ego scene flow produce non-zero targets, even for mostly-static scenes when observed from moving cameras.

Table 5. **Allocentric scene flow is the optimal output representation for 4D motion.** We compare different representation types on dense scene flow (Kubric) and sparse 3D point tracking (LSFOdyssey) using end-point error (EPE), average points within delta (APD) and inlier ratio at 0.1m (τ). Best results are **bold**.

Representation Type	Kubric-4D Static Camera				LSFOdyssey			
	Dynamic Points		Scene Flow		Dynamic Points		Scene Flow	
	EPE ↓	APD ↑	EPE ↓	τ ↑	EPE ↓	APD ↑	EPE ↓	τ ↑
Backprojected 2D Flow	2.14	19.44	1.16	75.69	0.49	57.21	0.27	70.11
3D Points After Motion	1.24	17.33	0.58	21.84	**0.24**	69.30	0.38	21.87
Egocentric Scene Flow	1.26	19.43	0.12	85.37	**0.24**	71.80	0.14	65.13
Allocentric Scene Flow	**1.23**	**19.53**	**0.10**	**87.51**	**0.24**	**73.95**	**0.10**	**71.46**

Limitations: Although Any4D takes a step forward towards achieving 4D reconstruction models, we identify important limitations. Firstly, we always calculate scene-flow from the reference (first) view to all other frames in the sequence, necessitating that the object of interest should be present at the start of the video. One possible way to alleviate this is by training Any4D in a permutation invariant manner as in [83]. Secondly, we assume perfectly simulated multi-modal input and do not account for sensor noise - which is hardly true for real-world deployment. Finally, as with all data-driven architectures, generalization is a function of the diversity and size of the training set. We believe that Any4D's performance on highly dynamic scenes and wide baselines (or low frame-rate videos) can be improved with the availability of richer dynamic 3D datasets[70].

5. Conclusion

We presented Any4D, a unified model that enables dense 4D reconstruction of dynamic scenes from both monocular and multi-modal setups. In Any4D, we chose a *factorized* output representation of 4D scenes, which allowed the use of diverse data for training at scale with partial supervision for auxiliary sub-tasks, in addition to the target task of dense scene flow estimation. Any4D is flexible, and supports optional *multi-modal inputs*. Importantly, we showed through our experiments that our joint training scheme produces generalizable view embeddings that improve performance whenever inputs such as depth and egocentric radial velocity (doppler) may be available to support the output prediction quantities. Finally, due to the feed-forward nature of Any4D, we saw that one can obtain dynamic scene estimates an order of magnitude faster than existing methods such as SpatialTrackerV2 [86], by exploiting N-view inference. We believe Any4D will ultimately enable real-time 4D scene reconstruction for applications such as Generative AI, AR/VR and Robotics, and serve as a foundational 4D reconstruction model.

References

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *2009 IEEE 12th International Conference on Computer Vision*, pages 72–79, 2009. 2

[2] Manuel López Antequera, Pau Gargallo, Markus Hofinger, Samuel Rota Bulo, Yubin Kuang, and Peter Kontschieder. Mapillary planet-scale depth dataset. In *European Conference on Computer Vision*, pages 589–604. Springer, 2020. 2

[3] Armen Avetisyan, Christopher Xie, Henry Howard-Jenkins, Tsun-Yi Yang, Samir Aroudj, Suvam Patra, Fuyang Zhang, Duncan Frost, Luke Holland, Campbell Orme, et al. Scenescript: Reconstructing scenes with an autoregressive structured language model. In *European Conference on Computer Vision*, pages 247–263. Springer, 2024. 2

[4] Arjun Balasingam, Joseph Chandler, Chenning Li, Zhoutong Zhang, and Hari Balakrishnan. Drivetrack: A benchmark for long-range point tracking in real-world videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22488–22497, 2024. 5, 7

[5] Tali Basha, Yael Moses, and Nahum Kiryati. Multi-view scene flow estimation: A view centered variational approach. *International journal of computer vision*, 101:6–21, 2013. 2

[6] Berta Bescos, José M Fácil, Javier Civera, and José Neira. Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE robotics and automation letters*, 3(4):4076–4083, 2018. 2

[7] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020. 2, 5, 6

[8] Kaihua Chen, Tarasha Khurana, and Deva Ramanan. Reconstruct, inpaint, finetune: Dynamic novel-view synthesis from monocular videos. *arXiv preprint arXiv:2507.12646*, 2025. 1

[9] Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22831–22840, 2025. 7

[10] 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. 2

[11] Nathaniel Chodosh, Deva Ramanan, and Simon Lucey. Re-evaluating lidar scene flow. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 6005–6015, 2024. 2

[12] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023. 4

[13] Frank Dellaert, Michael Kaess, et al. Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2): 1–139, 2017. 2

[14] Bardienus P Duisterhof, Jan Oberst, Bowen Wen, Stan Birchfield, Deva Ramanan, and Jeffrey Ichnowski. Rayst3r: Predicting novel depth maps for zero-shot object completion. *Advances in Neural Information Processing Systems*, 2025. 2

[15] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014. 2

[16] Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. *arXiv preprint arXiv:2504.13152*, 2025. 2, 3, 5, 6, 7, 8

[17] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013. 7

[18] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: a scalable dataset generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 5

[19] Jisang Han, Honggyu An, Jaewoo Jung, Takuya Narihira, Junyoung Seo, Kazumi Fukuda, Chaehyun Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Dˆ 2ust3r: Enhancing 3d reconstruction with 4d pointmaps for dynamic scenes. *arXiv preprint arXiv:2504.06264*, 2025. 3

[20] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022. 3

[21] Mina Henein, Jun Zhang, Robert Mahony, and Viorela Ila. Dynamic slam: The need for speed. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2123–2129. IEEE, 2020. 2

[22] Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. Depthcrafter: Generating consistent long depth sequences for open-world videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2005–2015, 2025. 7

[23] Yuan-Ting Hu, Jiahong Wang, Raymond A Yeh, and Alexander G Schwing. Sail-vos 3d: A synthetic dataset and baselines for object detection and 3d mesh reconstruction from video data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1418–1428, 2021. 5

[24] Junsheng Huang, Shengyu Hao, Bocheng Hu, and Gaoang Wang. Understanding dynamic scenes in ego centric 4d point clouds. *arXiv preprint arXiv:2508.07251*, 2025. 1

[25] Frédéric Huguet and Frédéric Devernay. A variational method for scene flow estimation from stereo sequences. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–7. IEEE, 2007. 2

[26] Wonbong Jang, Philippe Weinzaepfel, Vincent Leroy, Lourdes Agapito, and Jerome Revaud. Pow3r: Empowering unconstrained 3d reconstruction with camera and scene priors. *arXiv preprint arXiv:2503.17316*, 2025. 2

[27] Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. *arXiv preprint*, 2024. 2, 3, 5

[28] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13229–13239, 2023. 5, 6, 7

[29] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *arxiv*, 2024. 3, 5, 6

[30] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *Proc. ECCV*, 2024. 3

[31] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21357–21366, 2024. 2

[32] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, et al. MapAnything: Universal feed-forward metric 3d reconstruction. In *2026 International Conference on 3D Vision (3DV)*. IEEE, 2026. 2, 3, 4, 5, 6, 7

[33] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007. 2

[34] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 2

[35] Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, João Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. Tapvid-3d: A benchmark for tracking any point in 3d, 2024. 3, 6, 7

[36] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Monocular dense 3d reconstruction of a complex dynamic scene from two perspective frames. In *Proceedings of the IEEE international conference on computer vision*, pages 4649–4657, 2017. 2

[37] Jiahui Lei, Yijia Weng, Adam W Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6165–6177, 2025. 2

[38] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, pages 71–91. Springer, 2024. 6

[39] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2041–2050, 2018. 5

[40] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. Megasam: Accurate, fast, and robust structure and motion from casual dynamic videos. *arXiv preprint arXiv:2412.04463*, 2024. 2, 7

[41] Chenguo Lin, Yuchen Lin, Panwang Pan, Yifan Yu, Honglei Yan, Katerina Fragkiadaki, and Yadong Mu. Movies: Motion-aware 4d dynamic view synthesis in one second. *arXiv preprint arXiv:2507.10065*, 2025. 1, 2, 3

[42] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 529–537, 2019. 2

[43] Xinhang Liu, Yuxi Xiao, Donny Y Chen, Jiashi Feng, Yu-Wing Tai, Chi-Keung Tang, and Bingyi Kang. Trace anything: Representing any video in 4d via trajectory fields. *arXiv preprint arXiv:2510.13802*, 2025. 3

[44] Zeyi Liu, Shuang Li, Eric Cousineau, Siyuan Feng, Benjamin Burchfiel, and Shuran Song. Geometry-aware 4d video generation for robot manipulation. *arXiv preprint arXiv:2507.01099*, 2025. 1

[45] Hidenobu Matsuki, Gwangbin Bae, and Andrew J Davison. 4dtam: Non-rigid tracking and mapping via dynamic surface gaussians. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26921–26932, 2025. 2

[46] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 7

[47] 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. 2

[48] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[49] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11177–11185, 2020. 2

[50] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 2

[51] Tuan Duc Ngo, Peiye Zhuang, Chuang Gan, Evangelos Kalogerakis, Sergey Tulyakov, Hsin-Ying Lee, and Chaoyang Wang. Delta: Dense efficient long-range 3d tracking for any video. *arXiv preprint arXiv:2410.24211*, 2024. 3

[52] Dantong Niu, Yuvan Sharma, Haoru Xue, Giscard Biamby, Junyi Zhang, Ziteng Ji, Trevor Darrell, and Roei Herzig. Pretraining auto-regressive robotic models with 4d representations. *arXiv preprint arXiv:2502.13142*, 2025. 1

[53] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 4

[54] Emanuele Palazzolo, Jens Behley, Philipp Lottes, Philippe Giguere, and Cyrill Stachniss. Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7855–7862. IEEE, 2019. 7

[55] Jean-Philippe Pons, Renaud Keriven, and Olivier Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72:179–193, 2007. 2

[56] Gilles Puy, Alexandre Boulch, and Renaud Marlet. FLOT: Scene Flow on Point Clouds Guided by Optimal Transport. In *European Conference on Computer Vision*, 2020. 2

[57] Yuheng Qiu, Chen Wang, Wenshan Wang, Mina Henein, and Sebastian Scherer. Airdos: Dynamic slam benefits from articulated objects. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8047–8053. IEEE, 2022. 2

[58] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 2, 5

[59] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3), 2022. 2

[60] Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, and Huan Ling. L4gm: Large 4d gaussian reconstruction model. In *Advances in Neural Information Processing Systems*, 2024. 2

[61] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2195–2202, 2006. 3

[62] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 2

[63] Jenny Seidenschwarz, Qunjie Zhou, Bardienus P Duisterhof, Deva Ramanan, and Laura Leal-Taixé. Dynomo: Online point tracking by dynamic online monocular gaussian reconstruction. In *2025 International Conference on 3D Vision (3DV)*, pages 1012–1021. IEEE, 2025. 2

[64] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, pages 519–528. IEEE, 2006. 2

[65] Akash Sharma, Wei Dong, and Michael Kaess. Compositional and scalable object slam. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11626–11632. IEEE, 2021. 2

[66] Edgar Sucar, Zihang Lai, Eldar Insafutdinov, and Andrea Vedaldi. Dynamic point maps: A versatile representation for dynamic 3d reconstruction. *arXiv preprint arXiv:2503.16318*, 2025. 3

[67] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 2

[68] 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. 2

[69] Zachary Teed and Jia Deng. Raft-3d: Scene flow using rigid-motion embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 6

[70] Joachim Tesch, Giorgio Becherini, Prerana Achar, Anastasios Yiannakidis, Muhammed Kocabas, Priyanka Patel, and Michael J. Black. BEDLAM2.0: Synthetic humans and cameras in motion. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. 8

[71] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999. 2

[72] Basile Van Hoorick, Rundi Wu, Ege Ozguroglu, Kyle Sargent, Ruoshi Liu, Pavel Tokmakov, Achal Dave, Changxi Zheng, and Carl Vondrick. Generative camera dolly: Extreme monocular dynamic novel view synthesis. In *European Conference on Computer Vision*, pages 313–331. Springer, 2024. 1, 6

[73] Basile Van Hoorick, Rundi Wu, Ege Ozguroglu, Kyle Sargent, Ruoshi Liu, Pavel Tokmakov, Achal Dave, Changxi Zheng, and Carl Vondrick. Generative camera dolly: Extreme monocular dynamic novel view synthesis. In *European Conference on Computer Vision (ECCV)*, 2024. 5

[74] Kyle Vedder, Neehar Peri, Ishan Khatri, Siyi Li, Eric Eaton, Mehmet Kocamaz, Yue Wang, Zhiding Yu, Deva Ramanan, and Joachim Pehserl. Neural eulerian scene flow fields. *arXiv preprint arXiv:2410.02031*, 2024. 2

[75] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 722–729. IEEE, 1999. 2

[76] Bo Wang, Jian Li, Yang Yu, Li Liu, Zhenping Sun, and Dewen Hu. Scenetracker: Long-term scene flow estimation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 6, 7

[77] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 2, 4, 6, 7

[78] Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. In *International Conference on Computer Vision (ICCV)*, 2025. 2

[79] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10510–10522, 2025. 7

[80] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20697–20709, 2024. 2, 3, 6, 7

[81] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 2

[82] Yihan Wang, Lahav Lipson, and Jia Deng. Sea-raft: Simple, efficient, accurate raft for optical flow. In *European Conference on Computer Vision*, pages 36–54. Springer, 2024. 6

[83] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Scalable permutation-equivariant visual geometry learning. arXiv preprint arXiv:2507.13347, 2025. 2, 8

[84] Rundi Wu, Ruiqi Gao, Ben Poole, Alex Trevithick, Changxi Zheng, Jonathan T Barron, and Aleksander Holynski. Cat4d: Create anything in 4d with multi-view video diffusion models. *arXiv preprint arXiv:2411.18613*, 2024. 1, 2

[85] Wenxuan Wu, Zhi Yuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. Pointpwc-net: Cost volume on point clouds for (self-) supervised scene flow estimation. In *European Conference on Computer Vision*, pages 88–107. Springer, 2020. 2

[86] Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20406–20417, 2024. 3, 8

[87] Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Iurii Makarov, Bingyi Kang, Xin Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. Spatialtrackerv2: 3d point tracking made easy. In *ICCV*, 2025. 3, 5, 6, 7

[88] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10371–10381, 2024. 2

[89] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1790–1799, 2020. 5

[90] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 5

[91] Bowei Zhang, Lei Ke, Adam W Harley, and Katerina Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *arXiv preprint arXiv:2504.14717*, 2025. 3

[92] 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. 2, 6, 7

[93] Songyan Zhang, Yongtao Ge, Jinyuan Tian, Guangkai Xu, Hao Chen, Chen Lv, and Chunhua Shen. Pomato: Marrying pointmap matching with temporal motion for dynamic 3d reconstruction. *arXiv preprint arXiv:2504.05692*, 2025. 3

[94] Yuchen Zhang, Nikhil Keetha, Chenwei Lyu, Bhuvan Jhamb, Yutian Chen, Yuheng Qiu, Jay Karhade, Shreyas Jha, Yaoyu Hu, Deva Ramanan, et al. UFM: A simple path towards unified dense correspondence with flow. *Advances in Neural Information Processing Systems*, 2025. 2

[95] 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. 2, 5

[96] Hanyu Zhou and Gim Hee Lee. Llava-4d: Embedding spatiotemporal prompt into lmms for 4d scene understanding. *arXiv preprint arXiv:2505.12253*, 2025. 1