L4P: Towards Unified Low-Level 4D Vision Perception

Abhishek Badki* Hang Su* Bowen Wen Orazio Gallo

NVIDIA

Abstract

The spatio-temporal relationship between the pixels of a video carries critical information for low-level 4D perception tasks. A single *model that reasons about it should be able to solve* several *such tasks well. Yet, most state-of-the-art methods rely on architectures specialized for the task at hand. We present* L4P*, a feedforward, general-purpose architecture that solves low-level 4D perception tasks in a unified framework. L4P leverages a pre-trained ViT-based video encoder and combines it with per-task heads that are lightweight and therefore do not require extensive training. Despite its general and feedforward formulation, our method is competitive with existing specialized methods on both dense tasks, such as depth or optical flow estimation, and sparse tasks, such as 2D/3D tracking. Moreover, it solves all tasks at once in a time comparable to that of single-task methods.*

1. Introduction

Large collections of videos are our most complete and compact source of priors about the world. Much like text did for large-language models, the corpus of videos we amassed over the years allowed video-language models (VLMs) to produce remarkable zero-shot results on high-level vision tasks such as video captioning, video question answering, and others. However, zero-shot, low-level 3D and 4D vision perception tasks, such as depth from video, tracking, optical flow, and others remain a challenge. Pre-trained diffusion models fine-tuned on target-domain data showed potential on dense vision perception tasks (*e.g.*, depth [23, 30], flow [48], *etc.*), but the fine-tuning makes them task-specific, and therefore limits their ability to leverage priors across multiple tasks at once. Sparse vision perception tasks, such as tracking, are even more challenging to tackle with a general foundation model, because their representation does not fit naturally into pixel-aligned data structures, and are typically addressed with optimization-based approaches [68] or specialized architectures [11, 29].

* indicates equal contribution.
Website: https://research.nvidia.com/labs/lpr/l4p

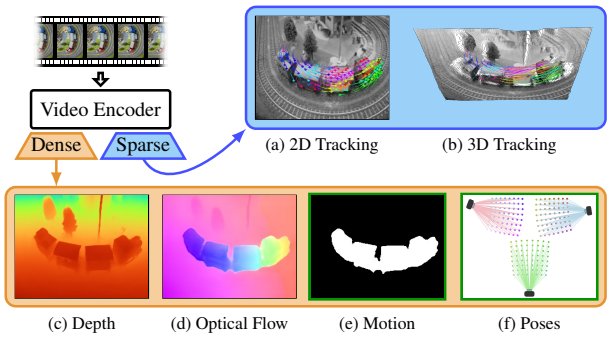

Figure 1. We propose L4P, a general-purpose architecture that solves several low-level 4D perception tasks. Building on a pre-trained video encoder and lightweight per-task heads, our unified model is competitive with existing methods specialized to solve individual tasks. L4P can easily be extended to additional tasks, (e) and (f), without compromising performance.

Can we leverage the priors learned from a large body of video data and solve *multiple* low-level 4D vision perception tasks, both *dense* and *sparse*, with a *unified* architecture with strong generalization abilities?

This is challenging: we need a shared backbone that is strong and versatile. On one hand, it must be general enough to allow for pre-training on auxiliary tasks. On the other hand, it must be able to support diverse tasks that require fundamentally different representations, such as dense 2D planes and 3D tracks. We tackle this challenge by combining a pre-trained video masked auto-encoder (VideoMAE) [63, 67] with per-task, lightweight heads (Figure 2). VideoMAEs offer a feedforward, online mechanism to tokenize videos within a small computational envelope. Indeed they have been successfully employed for a variety of mid- and high-level vision tasks [67], but their ability to capture the spatio-temporal relationship between pixels is underexplored in the context of low-level 4D perception. For dense tasks, we couple VideoMAE with heads based on the dense prediction transformer (DPT), which has been shown to perform well on depth estimation, segmentation, and others [46]. For sparse tasks, we focus on tracking. We posit that tracking is important for perception because understanding fine-grained, complex motions and the physical interaction between objects is critical to downstream applications, including 3D reconstruction [35, 65] and robotics

manipulation [2, 77, 78, 82]. We formulate the problem of tracking as that of predicting *2D* heatmaps for queried pixels with associated depth and visibility tokens. This formulation allows us to tackle sparse and dense tasks within a *unified* model. To implement it, we augment a general-purpose head with a novel memory mechanism to track points for arbitrarily long videos. We finetune our model for multiple tasks at once on a small collection of synthetic datasets. Yet, thanks to the training curriculum we propose, and to VideoMAE's priors, it shows strong generalization to real data.

Our formulation presents several desirable properties and advantages. First, the pre-trained VideoMAE model allows us to tap into priors learned from large datasets that are potentially different and more varied than those typically used for low-level 4D perception. It also affords us efficient computation: our system solves all tasks in roughly 300ms for a 16-frame video chunk ($\sim$19ms/frame), which is comparable to, or faster than, methods specialized for each task (see Table 1 in the Supplementary). Moreover, combining it with per-task heads allows us to train a relatively small number of parameters for new tasks, which we show by freezing the system and adding a head for motion-based segmentation and camera pose estimation (marked in green in Figures 1 and 2). Lastly, but perhaps most importantly, breaking the architecture into a general VideoMAE and per-task heads offers a mechanism to solve both dense and sparse tasks with a *unified* model (Figure 1). We emphasize that our main target is a general, multi-task framework, rather than the performance on any specific tasks. Nonetheless, our model performs competitively on the individual tasks, often on par with state-of-the-art methods. This is remarkable because the baselines we compare against are task-specific, carefully designed for, and specialized to excel at their respective tasks. Finally, VideoMAEs are already used as encoders for VLMs [37, 75], and we speculate that training them to reason about low-level 4D perception may impart those capabilities to the downstream VLMs they may be used with, though validating this is outside the scope of our paper. In summary:

- We show that priors from video representation learning can be leveraged for low-level 4D perception tasks with strong generalization capabilities.
- We present a feedforward video models that adopts a general-purpose architecture for multiple 4D perception tasks jointly—including both dense and sparse ones—and demonstrate its effectiveness on in-the-wild video inputs.
- We combine the general-purpose architecture with a novel memory mechanism to support long-range tracking to tackle the limited context of the video encoder.
- Our model demonstrates strong performance, solving multiple tasks jointly with results on par with state-of-the-art, within timeframes typical of single-task approaches.

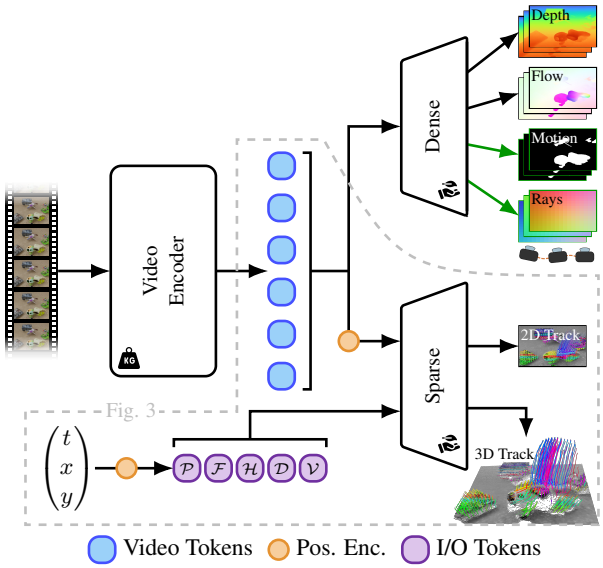

Figure 2. We use a pre-trained video encoder as tokenizer, and combine it with per-task lightweight heads. For sparse tasks we define additional query tokens: the point to track, $\mathcal{P}$, the corresponding feature token, $\mathcal{F}$, and output tokens (heatmap $\mathcal{H}$, depth $\mathcal{D}$, and visibility $\mathcal{V}$).

2. Related Works

Our method unites the strong generalization capabilities of pre-trained foundation models with lightweight task-specific heads. In this section, we review relevant literature for both the foundation models and the individual tasks.

2.1. Foundation models for vision perception

Self-supervised pre-training of large models on huge unlabeled data has shown great success. Among them, Vision Transformers (ViT) [32] pre-trained with masked autoencoding (MAE) [22] have become a common choice to fine-tune for many vision tasks, Segment Anything (SAM) [31] being a notable example. Several approaches have been proposed for spatio-temporal representation learning for videos: VideoMAE [63, 67], MAE-ST [15], V-JEPA [1], etc. We adopt VideoMAEv2 [67], which introduces a dual masking strategy that allows them to efficiently scale up the model to a billion parameters and to effectively leverage priors from large data. Their representation is effective for action recognition, and more recently is combined with language models [37, 75]. We show how to leverage them for low-level 4D perception tasks and achieve results competitive with specialized SOTA methods.

2.2. Dense Prediction Tasks

Depth estimation. Stereo depth estimation approaches [17, 18, 24, 49, 51, 86] and single image depth estimation approaches [14, 45, 46, 84, 85] have made significant

progress. We focus on estimating depth from videos of dynamic scenes, for which several solutions exist [33, 39, 58, 74]. Pioneered by DUSt3R [70], an emerging paradign estimates pointmaps instead of depthmaps. Several recent approaches [36, 66, 69, 88] follow this paradigm for joint depth and camera pose estimation. To tackle the data scarcity problem, the state-of-the-art approaches leverage priors from foundation models: DepthAnything [84, 85] uses DINOv2 [42], DUSt3R [70] uses cross-view completion pre-training [76], and Marigold [30] and DepthCrafter [23] use image and video diffusion priors respectively. We benefit from large-scale pre-training by using a video encoder pre-trained with MAE.

Optical flow estimation. Though straightforward when posed as a dense task, optical flow traditionally requires specialized architectures [56, 59, 71, 81]. Most related to ours are approaches that take multi-frame inputs [13, 26, 47, 54]. Similar to depth estimation, recent works adopt priors from large-scale pre-training, *e.g.* via diffusion priors [48] or cross-view completion pre-training [76].

Motion-based segmentation. While early learning-based approaches rely on combining appearance features with flow [2, 16, 25, 61, 62], more recent works [5, 83] extract geometric properties from flow before using it to train a classification network. However, they are affected by noisy flow and are limited by their two-frame formulation.

Camera pose estimation. Classical structure-from-motion solutions [50, 55] work robustly and accurately in constrained settings, while learning-based solutions [38, 60, 64, 66, 69, 70, 72, 88] address challenging scenarios like limited image observations, dynamic and texture-less regions, etc. We cast camera pose estimation as a dense task by representing cameras as dense bundles of rays [20, 89] using 6-D Plücker coordinates [44] and estimating them using our dense head.

We treat motion-based segmentation and camera pose estimation as additional tasks, by leveraging the priors from our video model trained on depth, flow and tracking tasks, and only training task-specific heads. Thus, we maintain main tasks performance while supporting additional tasks.

2.3. Sparse Prediction Tasks

Tracking Any Point (TAP) in a video has many applications [2, 8, 35, 65, 78]. Particle Video [21], PIPs [4] and TAP-Net [10] lay the initial foundation by adopting ideas from optical flow approaches. On the other hand, OmniMotion [68] optimizes a volumetric representation for each video to solve this task and is time-consuming. TAPIR [11] introduces the idea of coarse-to-fine track estimation, BootsTAPIR [12] further improves it by adopting a self-supervised learning, and CoTracker [29] proposes to jointly track multiple points to leverage spatial correlations. SpaTracker [79] is a feedforward method for 3D point tracking that lifts pixels to 3D using input depth maps and applies CoTracker's tracking formulation in 3D. Concurrent works, TAPIP3D [87] and SpatialTrackerV2 [80], adopt a similar approach but use posed RGBD inputs; notably, SpatialTrackerV2 jointly optimizes point tracks and initial camera poses via iterative bundle adjustment, which is inefficient for real-time settings. Seurat [9] shows that dense 2D tracks alone contain enough information to be lifted to 3D. In contrast to these prior works, which often rely on specialized architectures and additional inputs such as depth maps or 2D point tracks, we focus on efficiently unifying 2D and 3D point tracking with other tasks, paving the way for general purpose low-level 4D perception for real-time scenarios.

3. Method

We provide an overview of our approach in Figure 2. Our model uses a pre-trained ViT-based video encoder [67] (Section 3.1) to capture spatio-temporal features in an RGB video clip of length T. We use lightweight task-specific heads that decode the video features for low-level 3D/4D perception tasks. For pixel-wise dense tasks like depth, flow, motion-based segmentation, and camera rays estimation (for camera pose estimation), we propose an extension of the DPT architecture [46] that allows us to use them for videos, instead of just images, as in their existing formulation (Section 3.2). For the sparse task of tracking any pixel in a video, we take inspiration from the head architecture proposed in SAM [31] (Section 3.3). Given a pixel queried in any frame of the input video, we extend the head, also originally designed to work for images, to decode video tokens into a 2D trajectory, its depth with respect to the camera, and its visibility in each frame (Section 3.3.1). VideoMAE extracts video tokens from temporal windows of fixed length T, and cannot process longer sequences. We propose a memory mechanism to track points in arbitrarily long videos (Section 3.3.2).

3.1. Video Masked Auto-Encoders

Motivated by its scalable and powerful pre-training and architecture, we use the ViT-based video encoder from VideoMAEv2 [67]. The encoder works with videos of size $T \times H \times W$ and uses a spatio-temporal patch size of $t \times h \times w$. It uses cube embedding [63] to transform an input video into a sequence of tokens, which are then processed by the spatio-temporal attention blocks to generate video tokens $\mathcal{S} \in \mathbb{R}^{P \times C}$, where P is the number of tokens and C is the embedding dimension. We run the video encoder only once per video clip, and then apply the lightweight heads to decode the tokens to the desired outputs. For the point tracking task, we can independently prompt these tokens to track many points in parallel.

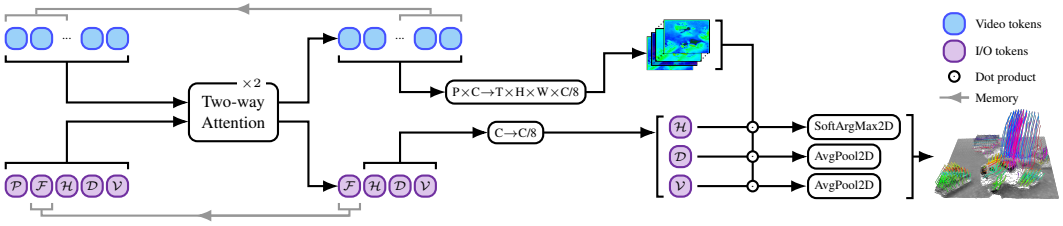

Figure 3. Sparse head. The video and I/O tokens (query $\mathcal{P}$oint, query point $\mathcal{F}$eature, and query point $\mathcal{H}$eat map, $\mathcal{D}$epth, and $\mathcal{V}$isibility) are processed by a two-way attention layer. The reshaped and resized per-frame feature maps and processed output tokens are combined via a dot product. We also introduce a memory mechanism to combine video and the $\mathcal{F}$eature tokens from time t and $t+1$.

3.2. Dense Prediction Heads

Dense prediction tasks produce outputs with spatial dimensions aligned with their inputs, typically at the same resolution $H \times W$. A wide array of common computer vision problems can be formulated as dense prediction tasks. In this work, we explore depth estimation, optical flow estimation, motion-based segmentation, and camera pose estimation as examples.

We adopt the DPT [46] architecture for dense prediction due to its proven performance and efficiency on single-image depth estimation. DPT progressively assembles and combines tokens from various layers, thus capturing both local and global spatial structures. We adapt DPT to work with the 3D tokens from VideoMAE by mapping the video tokens to 3D feature maps and enabling temporal reasoning by introducing 3D convolutions inside the DPT head. We find that this modification is enough to bring in temporal consistency with minimal computation overhead. For most of our dense tasks, the DPT heads differ only in the final layer, which outputs one channel for depth and motion-based segmentation, and two channels for optical flow. We cast camera pose estimation as a dense task by estimating a bundle of rays for each camera, and recover camera parameters from the estimated rays [89] (see Supplementary for more details).

For videos longer than T frames, we run inference with stride $T/2$ and enforce consistency in the overlapping frames between temporal windows. For depth, we align the predictions with an affine transformation. This strategy has no effect on the individual windows for relative depth, but can greatly improve long-term temporal consistency. For optical flow and motion-based segmentation, we simply overwrite the predictions in the overlap between windows. For pose estimation, we align camera trajectories across overlapping windows using 3D similarity transformations between their uplifted point clouds.

3.3. Sparse Prediction Heads

Given a video and pixel prompt, (t_i, x_i, y_i), we want to estimate the corresponding 3D trajectory, $\mathcal{T}_i = \{\hat{x}_i(t), \hat{y}_i(t), \hat{d}_i(t), \hat{v}_i(t)\}_{t=0}^{S-1}$, where at time t, $(\hat{x}_i(t), \hat{y}_i(t))$ denotes the 2D track location, $\hat{d}_i(t)$ is the track depth with respect to camera, and $\hat{v}_i(t)$ is the track visibility indicating if a track is visible or occluded at time t. This is a challenging task since it requires tracking the pixel in 2D when visible, tracking it through occlusions, and reasoning about its depth along the track. Moreover, our video encoder has limited temporal context, since it can only process videos with fixed temporal window of T frames, and we want to enable tracking for arbitrarily long videos with $S > T$ frames. This makes adapting a general-purpose head particularly challenging. To tackle this, we introduce an online approach. We design a head that allows us to estimate the 3D track for an input pixel prompt within the temporal context (T frames) of the video encoder. For online estimation beyond T frames, we propose a memory mechanism for our head and a recipe to train it efficiently (see Figure 3).

3.3.1 Tracking within the Temporal Context

Posing the sparse tracking task within our unified framework requires special care. Instead of directly estimating point-track positions, we propose to represent tracks using dense probability heatmaps. Casting tracking as a problem of estimating pixel-aligned 2D maps affords us a consistent representation between sparse and dense tasks, which is critical for using a shared backbone. To achieve this, we build on the prompt-encoding and mask-decoding mechanisms from SAM [31]. We encode the input pixel prompt using 3D positional encoding and a learnable embedding to generate input point token $\mathcal{P}$ with embedding dimension of C. Similarly, we define output tokens with learnable embeddings to estimate different components of a 3D track: a heatmap token ($\mathcal{H}$) to estimate the 2D pixel position of the track across the video, a depth token ($\mathcal{D}$), and a visibility token ($\mathcal{V}$). Input and output tokens interact with the video tokens, $\mathcal{S}$, also encoded with 3D positional encoding, using a two-way attention mechanism to decode the video features. These video features are then reshaped and upsampled, and a final inner product with the processed output tokens gives us output masks of size $T \times H \times W$. For 2D track estimation, we interpret this output mask as a probability heatmap that

encodes the 2D track position and apply a 2D soft-argmax operation to estimate the 2D track position $(\hat{x}_i(t), \hat{y}_i(t))$ at each frame t. For depth and visibility, we simply apply a 2D average pooling operation, followed by exponential and sigmoid operations respectively to estimate the track depth $\hat{d}_i(t)$ and the visibility $\hat{v}_i(t)$ at each frame t. This design also allows us to query points anywhere in the video and track them in parallel. We use two lightweight, two-way attention layers [31]. We use 3D convolutions to enable temporal reasoning.

3.3.2 A Memory Mechanism for Long Videos

To track beyond a single window of length T frames, we propose an online strategy. A naïve approach would be to chain tracks across windows. Given two consecutive and overlapping windows, and a 2D track estimated in the first one, we can use a point on the track in the temporal overlap between the two windows as the query for tracking in the second one. To pick a good point to chain the tracks, we can select the one with highest visibility score. However, this solution is brittle for two reasons. First, a tracked point may not be visible in the overlap between the windows. To tackle this problem, inspired by Karaev *et al*. [29], we introduce a track-feature token $\mathcal{F}$ that is passed to subsequent windows as an additional prompt (see Figure 3). However, unlike Karaev *et al*., we do not initialize it explicitly with the local appearance around the query point, so the two-way attention head is free to capture the most useful information to track through occlusions. Second, the naïve solution described above does not allow the system to reason across temporal windows, which makes it prone to drifting or to losing tracks. The track-feature tokens help, but to provide even more cross-window information, we pass the video tokens decoded by the two-way attention stage of the current window to the next, as shown in Figure 3. We achieve this by projecting the decoded video tokens in the overlapping region via a linear layer, and by adding them to the corresponding incoming video tokens to the two-way attention stage in the next window. Our memory strategy based on these two mechanisms is critical to allow proper reasoning across temporal windows as shown by the comparison in ablation study in Section 5.4.

Online training. Training the memory mechanism requires unrolled-window training [29], in which we compute the video features for all the overlapping windows in a video of length S, and then compute the tracks for the entire video in an online fashion. However, training such an approach end-to-end is prohibitive due to memory constraints. To alleviate this, we adapt a multi-stage training strategy. First, we train only for a single window but train all the parameters of our network. In the next stage, we freeze all but the last few layers of our video encoder, and fine-tune it along with the tracking head for unrolled window training. In Table 5, we show this approach improves the performance over the naïve chaining approach.

4. Implementation

The training curriculum is critical to leverage the VideoMAE priors and fine-tune our system for all tasks at once.

Training datasets. We use the video encoder from Wang *et al*. [67], which is pre-trained on 1.35M video clips for masked auto-encoding. To fine-tune our model, we use four synthetic datasets covering various types of annotations, and rely on the priors from the pre-trained video encoder for generalization. We use Kubric [19], a synthetic dataset in which multiple objects interact, annotated with ground truth (GT) depth, flow, and 2D/3D point tracking. To include videos with long 3D trajectories, we add PointOdyssey [90] and DynamicReplica [28]. Both are synthetic datasets with animated characters in mostly indoor scenes. Both have depth GT and DynamicReplica also offers optical flow GT. To further increase scene diversity, we also include TartanAir [73], which provides GT for flow and depth.

Architectures. Our video encoder processes $16 \times 224 \times 224$ clips. It uses a patch size of $2 \times 14 \times 14$, which results in $P = 2048$ video tokens, and an embedding dimension of $C = 1408$. It has 40 encoder blocks, and we use the output from blocks 14, 21, 28, 36 for DPT heads for dense tasks, while the sparse heads use features only from the last block (block 39). Feeding the sparse and dense heads with tokens from different blocks is critical for maintaining the performance on dense tasks while we fine-tune our model to train the memory mechanism for the tracking tasks, as we discuss below. For a $16 \times 224 \times 224$ video clip, our method generates the outputs for all our tasks, including the additional tasks, in $\sim$300ms on an NVIDIA A6000 GPU. This corresponds to $\sim$19ms for a single frame, which is competitive with single-task approaches (see Supplementary for detailed comparisons). However, our method's latency may prevent its use for applications that require strict real-time performance.

Training. Perhaps unsurprisingly, training our system to maximize the performance on all tasks at once requires particular care. We initialize the video encoder using a pre-trained VideoMAE [67] and fine-tune the complete system in three stages. All the stages use a batch size of 8 and are trained on a single 8-GPU (NVIDIA A100) node for 100k iterations. In all the stages, we construct a batch of multiple tracks per video for the tracking task. In the first stage, we train end-to-end for depth, flow and point tracking tasks on a single window of $T = 16$ frames and train only on Kubric. In the second stage of training, we add the remaining three datasets and further fine-tune the model on a single window for all the tasks. Since not all the datasets offer GT for all tasks, we form our batches by sampling two videos from

	Sintel (~50 frames)		ScanNet (90 frames)		KITTI (~110 frames)		Bonn (110 frames)		NYUv2 (1 frame)	
	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑
Marigold [30]	0.532	0.515	0.166	0.769	0.149	0.796	0.091	0.931	0.070	0.946
DA [84]	0.325	0.564	0.130	0.838	0.142	0.803	0.078	0.939	**0.042**	**0.981**
DA-V2 [85]	0.367	0.554	0.135	0.822	0.140	0.804	0.106	0.921	0.043	0.978
NVDS [74]	0.408	0.483	0.187	0.677	0.253	0.588	0.167	0.766	0.151	0.780
ChronoDepth [53]	0.587	0.486	0.159	0.783	0.167	0.759	0.100	0.911	0.073	0.941
MonST3R [88]	0.335	0.585	-	-	0.104	0.895	0.063	0.964	0.091	0.888
DepthCrafter [23]	0.270	0.697	0.123	0.856	0.104	0.896	0.071	0.972	0.072	0.948
L4P	**0.219**	**0.700**	**0.071**	**0.953**	**0.084**	**0.935**	**0.056**	**0.973**	0.084	0.916

Table 1. **Depth estimation results.** We compare our full model with single-image (Row 1-3) and video depth estimation baselines (Row 4-7). On video datasets (all except NYUv2), our model consistently performs better than DepthCrafter, the closest competition, and by a large margin on ScanNet and KITTI. Frames indicate average video length for each dataset.

each of the four datasets. This forces each batch to provide training signal for all task, and thus prevents training instability. In the third stage, we further fine-tune our model for the tracking tasks using unrolled-window training and our memory mechanism for online tracking. We train on videos of length $S = 40$ by using 4 overlapping windows of size 16 frames and a stride of 8. Due to memory constraints, in the third stage, we freeze all the parameters, except the last three layers (37-39) of the video encoder and the sparse task head. This allows us to maintain the performance on depth and flow, while training the memory mechanism to improve the tracking tasks. Training takes around 4 days for all the stages combined.

Losses. We use the SILog [14] loss for depth and L1 loss for optical flow. For tracking, we use L1 loss for 2D track positions, scale-invariant loss for track depth (similar to dense depth), and binary cross entropy loss for track visibility. Like with the choices of tasks heads, we pick the most widely used losses for each of our tasks. However, since we train for multiple tasks at once, weighting the losses appropriately is critical. We find the loss weights empirically by bringing the losses in the same order of magnitude and then doing a small hyperparameter search around those weights.

Please refer to the Supplementary for additional important implementation details.

5. Experiments

For each task included in our main model, we compare against a few recent SOTA methods on standard datasets. Our focus is a general, multi-task framework rather than achieving SOTA on any one task. Overall, we demonstrate competitive performance across all tasks, with computation times that are comparable to or faster than those of single-task approaches. We also show that L4P can be extended to new tasks without degrading existing performance.

5.1. Video Depth Estimation

We follow DepthCrafter [23] and evaluate video depth estimation on a collection of five datasets. We do not use any of the datasets for training our models or the baselines to better understand their generalization abilities. When estimating depth for videos longer than $T = 16$ frames, we follow the online inference and alignment strategies explained in Section 3.2. There is an inherent scale-ambiguity in the estimated depthmaps. We follow the common practice of aligning linearly the estimation with the GT before calculating evaluation metrics. The alignment is done for all the frames at once, and is carried out in *disparity* space via least-square fitting. For comparison on single image datasets, we repeat the single frame 16 times to compute our estimations. We report two metrics: AbsRel (mean($|\hat{\mathbf{d}} - \mathbf{d}|/\mathbf{d}$)) and δ_1 (ratio of pixels satisfying $\max(\mathbf{d}/\hat{\mathbf{d}}, \hat{\mathbf{d}}/\mathbf{d}) < 1.25$), where $\mathbf{d}$ represents GT, and $\hat{\mathbf{d}}$ is depth estimation after alignment. We upsample our estimations from 224×224 to each dataset's original resolution for evaluation. We consider video approaches including NVDS [74], ChronoDepth [53], DepthCrafter [23], MonST3R [88], as well as single-image ones, including Marigold [30] and DepthAnything [84, 85].

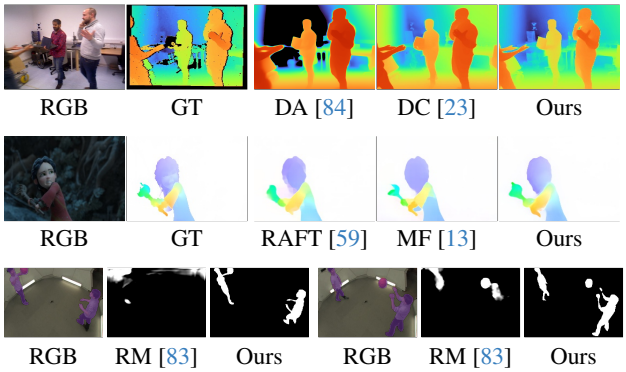

Figure 4. Comparisons with SOTA methods. We run inference on videos but show one frame here. The purple masks show GT motion segmentation.

Our results show strong performance against both single-image and video depth approaches on the four video datasets (Table 1). Since L4P is a video approach, applying it on single images from NYUv2 does not provide the necessary temporal context for it to perform well. DepthCrafter and MonST3R similarly suffer on NYUv2. Figure 4 (first row) shows a qualitative comparison. L4P produces a level of details comparable to that of diffusion models such as DepthCrafter, while generally capturing more accurate relative scales. It is also worth noting that our approach performs competitively on KITTI, despite the fine-tuning not including driving scenarios.

5.2. Multi-Frame Optical Flow Estimation

We evaluate on the Spring [41], Sintel [6], and Virtual KITTI [7] datasets. These video datasets are not used to train ours or other approaches, allowing us to evaluate generalization ability. For each dataset, we use all the videos and all the non-overlapping 16-frame clips from each video. The input frames are resized to 224×224 for all evaluation. We report Endpoint Error (EPE) and the ratio of EPE < 1.

	Spring		Sintel		Virtual KITTI	
	$EPE \downarrow$	$EPE < 1 \uparrow$	$EPE \downarrow$	$EPE < 1 \uparrow$	$EPE \downarrow$	$EPE < 1 \uparrow$
RAFT* [59]	0.13	98.4	1.31	85.1	1.13	77.5
MemFlow [13]	0.11	**98.7**	**1.03**	**87.2**	0.72	85.4
L4P	**0.09**	**98.7**	1.06	84.6	**0.63**	**86.6**

Table 2. **Optical flow estimation results.** We compare with SOTA flow methods: RAFT (two-frame, marked with *) and MemFlow (multi-frame). We evaluate cross-dataset generalization at 224x224 resolution. Our approach is competitive at this resolution, despite using generic architecture.

	Aria	DriveTrack	PStudio	**Overall**		
	2D-AJ ↑	2D-AJ ↑	2D-AJ ↑	2D-AJ ↑	APD ↑	OA ↑
TAPIR [11]	48.6	57.2	48.7	53.2	67.4	80.5
BootsTAPIR [12]	54.7	**62.9**	**52.4**	**59.1**	**74.7**	85.6
CoTracker [29]	54.2	59.8	51.0	57.2	74.2	84.5
L4P (2D Only)	**55.5**	53.0	48.7	52.4	68.2	**88.7**
L4P	52.4	50.0	48.1	50.2	66.1	**88.5**

Table 3. **Evaluation of 2D tracking on TAPVid-3D (*full_eval* split).** 2D GT trajectories are obtained by projecting 3D GT trajectories onto 2D. Though behind 2D SOTA approaches, L4P is competitive once fine-tuned only for 2D tracking ("2D Only").

	Aria			DriveTrack			PStudio			**Overall**		
	3D-AJ ↑	APD ↑	OA ↑	3D-AJ ↑	APD ↑	OA ↑	3D-AJ ↑	APD ↑	OA ↑	3D-AJ ↑	APD ↑	OA ↑
Static Baseline	4.9	10.2	55.4	3.9	6.5	80.8	5.9	11.5	75.8	4.9	9.4	70.7
TAPIR + CM	7.1	11.9	72.6	8.9	14.7	80.4	6.1	10.7	75.2	7.4	12.4	76.1
CoTracker + CM	8.0	12.3	78.6	11.7	**19.1**	81.7	8.1	13.5	77.2	9.3	15.0	79.1
BootsTAPIR + CM	9.1	14.5	78.6	**11.8**	18.6	83.8	6.9	11.6	81.8	9.3	14.9	81.4
TAPIR + ZD	9.0	14.3	79.7	5.2	8.8	81.6	10.7	18.2	78.7	8.3	13.8	80.0
CoTracker + ZD	10.0	15.9	87.8	5.0	9.1	82.6	11.2	19.4	80.0	8.7	14.8	83.4
BootsTAPIR + ZD	9.9	16.3	86.5	5.4	9.2	85.3	11.3	19.0	82.7	8.8	14.8	84.8
TAPIR-3D	2.5	4.8	86.0	3.2	5.9	83.3	3.6	7.0	78.9	3.1	5.9	82.8
SpatialTracker	9.9	16.1	89.0	6.2	11.1	83.7	10.9	19.2	78.6	9.0	15.5	83.7
L4P	**11.1**	**17.7**	**90.3**	6.5	11.3	**87.8**	**18.4**	**28.1**	**87.5**	**12.0**	**19.0**	**88.5**

Table 4. **Evaluation of 3D tracking on TAPVid-3D (*full_eval* split).** The top approaches combine 2D point tracking approaches with COLMAP (CM) [50], while the bottom ones, including ours are feedforward. "ZD" refers to ZoeDepth. Our approach outperforms previous approaches on average across all the metrics.

We consider two baselines for comparison. RAFT [59], a competitive and widely used two-frame approach, creates dense pairwise pixel features and uses recurrent updates to estimate optical flow. MemFlow [13], a recently published work, ranks among the top methods on the Spring benchmark. It is a multi-frame approach that relies on a memory mechanism to leverage temporal context. Quantitatively, L4P compares favorably to both RAFT and MemFlow (Table 2). Our model captures both small and large motions and presents more precise motion boundaries (Figure 4, second row). In addition, multi-frame approaches like MemFlow and ours generally have an edge in temporal stability (see Supplementary). Unlike many specialized approaches, our model currently only operates on low-resolution videos and further work is needed to enable efficient high-res estimation. We discuss this in the Limitations (Section 6).

5.3. Sparse 2D/3D Track Estimation

We evaluate on TAPVid-3D [34], a benchmark containing long-range 3D point trajectories over three datasets: Aria [43], DriveTrack [57], and PStudio [27]. It introduced several baselines by combining SOTA 2D point tracking approaches with depth solutions (eg. ZoeDepth [3], COLMAP [50, 52]). We mainly compare against CoTracker [29], TAPIR [11], and BootsTAPIR [12] for 2D tracking, and SpaTracker [79] for 3D tracking.

The benchmark evaluates both 3D and 2D tracking approaches, and uses metrics that measure the ability to predict point visibility using an occlusion accuracy metric (OA), the accuracy of predicted trajectories in the visible regions (APD), and joint occlusion and geometric estimation (AJ). To resolve the scale ambiguity in depth estimation, the benchmark uses global median scaling by computing the median of the depth ratios between the estimated and ground-truth 3D tracks over all the points and frames in a video. We use the *full_eval* split evaluation numbers provided in the TAPVid-3D benchmark for the comparison.

On 3D tracking, we outperform SpaTracker (Table 4), demonstrating the effectiveness of our general architecture compared to a specialized one. Approaches that combine 2D track estimation with COLMAP perform better on the DriveTrack [57] dataset. A reason could be that many query points are on static vehicles, for which COLMAP gives accurate depth. Such COLMAP-based baselines, however, perform poorly on Aria [43] and PStudio [27], which are dynamic. See Figure 5 for comparison with SpaTracker.

On 2D tracking, we are slightly behind the SOTA 2D tracking approaches (Table 3). Our approach becomes more competitive when we fine-tune our model only for the 2D tracking task. We believe our reduced performance on 2D tracking comes from working at lower image resolution, 224×224 for us as compared to 384×512 for CoTracker, and 256×256 for others, and a lack of task-specific tricks, like tracking multiple points together (CoTracker), or assuming access to all frames in the video and performing a global track-refinement (TAPIR and BootsTAPIR), both of which could also benefit our tracking head.

5.4. Ablations

To understand the contribution of different components of our approach, we perform an ablation study for depth, flow and 3D point tracking, as shown in Table 5. For each of these tasks, we report average numbers over the datasets not used in our training: for depth we use datasets in Table 1, for optical flow we use the Spring dataset, and for tracking we use the *minival* split from the TAPVid-3D [34] benchmark. Our main contribution is a method that can leverage the priors of a pre-trained VideoMAE for dense and sparse low-level perception tasks. To show the usefulness of our end-to-end fine-tuning strategy, we compare against a pre-trained and frozen VideoMAE, where we only fine-tune the task-specific heads. Table 5 shows that our fine-tuned VideoMAE (row 4) produces better results than the pre-trained and frozen VideoMAE across all tasks (row 2). A version trained end-to-end from scratch results in worse

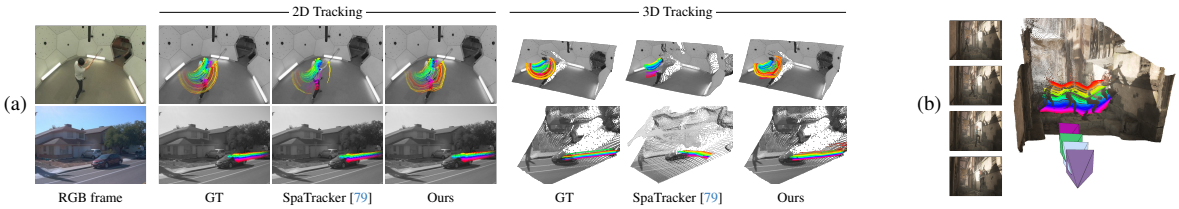

Figure 5. Comparisons on 2D/3D tracking (a). L4P can estimate camera poses, depth, and tracking in the same reference system (b).

	Depth	Optical flow	3D Track
	AbsRel↓ / δ_1 ↑	EPE↓ / EPE< 1 ↑	2D-AJ↑ / 3D-AJ↑
L4P (from scratch, w/o mem)	0.274 / 0.560	0.319 / 95.0	15.9 / 1.2
L4P (VideoMAE frozen, w/o mem)	0.140 / 0.830	0.114 / 98.3	33.3 / 4.2
L4P (task-specific, w/o mem)	*0.108 / 0.894*	**0.084 / 98.8**	39.4 / 8.4
L4P (w/o mem)	**0.103 / 0.895**	*0.092 / 98.7*	*40.4 / 8.6*
L4P	**0.103 / 0.895**	*0.092 / 98.7*	**49.0 / 10.9**

Table 5. **Ablation study.** See Section 5.4 for an analysis.

performance (row 1), which shows that our system successfully builds on the priors of the pre-trained VideoMAE. Training our system for all tasks jointly (row 4) performs roughly on par as training for each task individually (row 3), which shows effectiveness of our training strategy to leverage VideoMAE priors for multiple tasks at once. Finally, by adding the proposed memory mechanism for tracking and using our multi-stage training process for online tracking, we obtain improvements for point tracking, while maintaining the performance on other tasks (row 5 vs. 4).

5.5. Additional Tasks

L4P can be extended to new tasks, *without* degrading the performance of the original tasks. We do this by freezing our video encoder trained for the original tasks, and training a task-specific head. Here we provide two examples.

Motion-based segmentation. We train on the Kubric [19], and evaluate on the Virtual KITTI (VKITTI) [7] and Spring [41]. We compare against RigidMask (RM) [83], a SOTA two-frame approach. It is trained on the SceneFlowDatasets [40]; however, they also train a version for driving scenarios (RM-Drive). To evaluate, we report foreground IoU (higher is better). On both datasets, our video-based approach performs better. Note that while fine-tuning on driving scenes helps RigidMask (RM-Drive) on VKITTI, it significantly hurts performance on Spring, highlighting the benefit of our model's generalization ability. As shown in Figure 4 (third row), our approach performs better and can detect small motions. See more comparisons in the Supplementary.

	VKITTI	Spring
RM	32.4	17.0
RM-Drive	36.5	8.4
L4P	**56.0**	**21.5**

Camera pose estimation. We follow [89] and represent a camera as a bundle of rays. More specifically, each camera is represented by the 16×16 camera rays towards the image patch centers, and each ray by its 6-D Plücker coordinates. Our DPT dense head can be easily adapted to output a 16×16 grid of rays. From the estimated ray bundles, we can then recover camera poses by solving least-square optimizations for camera centers, rotation matrices and even camera intrinsics [89]. For evaluation, we follow [88] and show results on both dynamic (Sintel and TUM-dynamics) and static scenes (ScanNet). We provide ATE (lower is better) and focus on feedforward approaches using general architectures. We show two versions: one by only fine-tuning the camera head (head FT), and one by fine-tuning the entire model end-to-end (e2e FT) for all tasks, including camera pose estimation. In either settings, our approach performs favorably with recent approaches like DUSt3R [70], Spann3R [64], CUT3R [69]. Once we have the camera pose estimation, we can visualize the camera trajectory, depth and 3D tracks in the same reference system (Figure 5(b)). Additional metrics and methods are in the Supplementary.

	Sintel	TUM-dyn.	ScanNet
DUSt3R [CVPR24]	0.290	0.140	0.246
Spann3R [3DV25]	0.329	0.056	0.096
CUT3R [CVPR25]	0.213	**0.046**	0.099
L4P (head FT)	0.162	0.060	0.100
L4P (e2e FT)	**0.132**	0.055	**0.080**

6. Limitations

Our main limitation is the low input resolution (224×224) inherited from VideoMAE. We emphasize this is not a fundamental limitation, and could be addressed through high-resolution fine-tuning of VideoMAE (following DINOv2 [42]) or by incorporating convex upsampling layers (common in opticalflow approaches). A second limitation is weaker performance on single-image depth estimation (Table 1, NYUv2), since L4P is trained only on videos. Incorporating image-only training into pre-training of VideoMAE and fine-tuning of L4P could help close this gap.

7. Conclusions

We present a unified framework to solve multiple low-level 4D vision perception tasks, both dense and sparse. We achieve this by adopting a strong pre-trained video masked auto-encoder and design lightweight task heads to harness its representation power. Our simple yet versatile designs for task heads allow for effortless and generalizable adaptation to multiple 4D vision perception tasks.

8. Acknowledgements

We thank Jan Kautz for discussions and for reviewing an early draft, Zhiding Yu and Hongxu Yin for initial discussions on video models, and Yiqing Liang for help with data.

References

[1] Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *ArXiv Preprint*, 2025. 2

[2] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision (ECCV)*, 2024. 2, 3

[3] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. ZoeDepth: Zero-shot transfer by combining relative and metric depth. *ArXiv Preprint*, 2023. 7

[4] Weikang BIAN, Zhaoyang Huang, Xiaoyu Shi, Yitong Dong, Yijin Li, and Hongsheng Li. Context-PIPs: Persistent independent particles demands context features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[5] Pia Bideau, Rakesh R. Menon, and Erik Learned-Miller. MoA-Net: Self-supervised motion segmentation. In *ECCV 2018 Workshops*, 2019. 3

[6] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12*, pages 611–625. Springer, 2012. 6

[7] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. *ArXiv Preprint*, 2020. 6, 8

[8] Yangming Cheng, Liulei Li, Yuanyou Xu, Xiaodi Li, Zongxin Yang, Wenguan Wang, and Yi Yang. Segment and track anything. *ArXiv Preprint*, 2023. 3

[9] Seokju Cho, Jiahui Huang, Seungryong Kim, and Joon-Young Lee. Seurat: From Moving Points to Depth. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3

[10] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. TAP-Vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[11] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 1, 3, 7

[12] Carl Doersch, Yi Yang, Dilara Gokay, Pauline Luc, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ross Goroshin, João Carreira, and Andrew Zisserman. BootsTAP: Bootstrapped training for tracking-any-point. *ArXiv Preprint*, 2024. 3, 7

[13] Qiaole Dong and Yanwei Fu. MemFlow: Optical flow estimation and prediction with memory. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3, 7

[14] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014. 2, 6

[15] Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[16] Katerina Fragkiadaki, Pablo Arbelaez, Panna Felsen, and Jitendra Malik. Learning to segment moving objects in videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[17] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 2

[18] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE international conference on computer vision*, pages 873–881, 2015. 2

[19] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5, 8

[20] M.D. Grossberg and S.K. Nayar. A general imaging model and a method for finding its parameters. In *IEEE International Conference on Computer Vision (ICCV)*, 2001. 3

[21] Adam W. Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision (ECCV)*, 2022. 3

[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 2

[23] 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. *ArXiv Preprint*, 2024. 1, 3, 6

[24] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2821–2830, 2018. 2

[25] Suyog Jain, Bo Xiong, and Kristen Grauman. FusionSeg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. *ArXiv Preprint*, 2017. 3

[26] Joel Janai, Fatma Guney, Anurag Ranjan, Michael Black, and Andreas Geiger. Unsupervised learning of multi-frame

optical flow with occlusions. In *Proceedings of the European conference on computer vision (ECCV)*, pages 690–706, 2018. 3

[27] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic Studio: A massively multiview system for social motion capture. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 7

[28] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. DynamicStereo: Consistent dynamic depth from stereo videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 5

[29] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. In *European Conference on Computer Vision (ECCV)*, 2024. 1, 3, 5, 7

[30] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9492–9502, 2024. 1, 3, 6

[31] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, et al. Segment anything. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 2, 3, 4, 5

[32] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[33] 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. 3

[34] 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. *ArXiv Preprint*, 2024. 7

[35] Jiahui Lei, Yijia Weng, Adam Harley, Leonidas Guibas, and Kostas Daniilidis. MoSca: Dynamic gaussian fusion from casual videos via 4D motion scaffolds. *ArXiv Preprint*, 2024. 1, 3

[36] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024. 3

[37] Kunchang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. VideoChat: Chat-centric video understanding. *ArXiv Preprint*, 2023. 2

[38] 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. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3

[39] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (ToG)*, 39(4):71–1, 2020. 3

[40] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 8

[41] 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. 6, 8

[42] 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. 3, 8

[43] Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Yuheng Carl Ren. Aria digital twin: A new benchmark dataset for egocentric 3D machine perception. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 7

[44] Julius Plücker. *Analytisch-Geometrische Entwicklungen*. GD Baedeker, 2025. 3

[45] 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 (TPAMI)*, 2020. 2

[46] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 3, 4

[47] Zhile Ren, Orazio Gallo, Deqing Sun, Ming-Hsuan Yang, Erik B Sudderth, and Jan Kautz. A fusion approach for multi-frame optical flow estimation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2077–2086. IEEE, 2019. 3

[48] Saurabh Saxena, Charles Herrmann, Junhwa Hur, Abhishek Kar, Mohammad Norouzi, Deqing Sun, and David J Fleet. The surprising effectiveness of diffusion models for optical flow and monocular depth estimation. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 3

[49] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. Simplerecon: 3d reconstruction without 3d convolutions. In *European Conference on Computer Vision*, pages 1–19. Springer, 2022. 2

[50] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3, 7

[51] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for

unstructured multi-view stereo. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 501–518. Springer, 2016. 2

[52] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 7

[53] Jiahao Shao, Yuanbo Yang, Hongyu Zhou, Youmin Zhang, Yujun Shen, Matteo Poggi, and Yiyi Liao. Learning temporally consistent video depth from video diffusion priors. *arXiv preprint arXiv:2406.01493*, 2024. 6

[54] Xiaoyu Shi, Zhaoyang Huang, Weikang Bian, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. VideoFlow: Exploiting temporal cues for multi-frame optical flow estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12469–12480, 2023. 3

[55] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Transactions on Graphics (SIGGRAPH)*, 2006. 3

[56] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 3

[57] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurélien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7

[58] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 3

[59] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020. 3, 7

[60] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *neurips*, 34: 16558–16569, 2021. 3

[61] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning motion patterns in videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[62] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning video object segmentation with visual memory. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 3

[63] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 2, 3

[64] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *ArXiv Preprint*, 2024. 3, 8

[65] Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. VGGSfM: Visual geometry grounded deep structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 3

[66] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3

[67] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. VideoMAE V2: Scaling video masked autoencoders with dual masking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 3, 5

[68] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 1, 3

[69] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. *ArXiv Preprint*, 2025. 3, 8

[70] 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*, pages 20697–20709, 2024. 3, 8

[71] Shizun Wang, Xingyi Yang, Qiuhong Shen, Zhenxiang Jiang, and Xinchao Wang. Gflow: Recovering 4d world from monocular video. *arXiv preprint arXiv:2405.18426*, 2024. 3

[72] Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. TartanVO: A generalizable learning-based vo. *Conference on Robot Learning (CoRL)*, 2020. 3

[73] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. TartanAir: A dataset to push the limits of visual slam. In *International Conference on Intelligent Robots and Systems (IROS)*, 2020. 5

[74] Yiran Wang, Min Shi, Jiaqi Li, Zihao Huang, Zhiguo Cao, Jianming Zhang, Ke Xian, and Guosheng Lin. Neural video depth stabilizer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9466–9476, 2023. 3, 6

[75] Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Chenting Wang, Guo Chen, Baoqi Pei, Rongkun Zheng, Jilan Xu, Zun Wang, et al. Internvideo2: Scaling video foundation models for multimodal video understanding. *ArXiv Preprint*, 2024. 2

[76] Philippe Weinzaepfel, Thomas Lucas, Vincent Leroy, Yohann Cabon, Vaibhav Arora, Romain Brégier, Gabriela Csurka, Leonid Antsfeld, Boris Chidlovskii, and Jérôme Revaud. Croco v2: Improved cross-view completion pre-training for stereo matching and optical flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17969–17980, 2023. 3

[77] Bowen Wen, Wenzhao Lian, Kostas Bekris, and Stefan

Schaal. You only demonstrate once: Category-level manipulation from single visual demonstration. *rss*, 2022. 2

[78] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *ArXiv Preprint*, 2024. 2, 3

[79] Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. SpatialTracker: Tracking any 2D pixels in 3D space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3, 7

[80] 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 *IEEE International Conference on Computer Vision (ICCV)*, 2025. 3

[81] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. Unifying flow, stereo and depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3

[82] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *corl*, 2024. 2

[83] Gengshan Yang and Deva Ramanan. Learning to segment rigid motions from two frames. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3, 8

[84] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything: Unleashing the power of large-scale unlabeled data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 3, 6

[85] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything V2. *arXiv preprint arXiv:2406.09414*, 2024. 2, 3, 6

[86] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 2

[87] 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

[88] 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*, 2024. 3, 6, 8

[89] Jason Y Zhang, Amy Lin, Moneish Kumar, Tzu-Hsuan Yang, Deva Ramanan, and Shubham Tulsiani. Cameras as rays: Pose estimation via ray diffusion. In *International Conference on Learning Representations (ICLR)*, 2024. 3, 4, 8

[90] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 5