LEAP-VO: Long-term Effective Any Point Tracking for Visual Odometry

Weirong Chen[1,2*] Le Chen[3] Rui Wang[4] Marc Pollefeys[4]
[1]TU Munich [2]Munich Center for Machine Learning [3]MPI for Intelligent Systems [4]Microsoft

Abstract

Visual odometry estimates the motion of a moving camera based on visual input. Existing methods, mostly focusing on two-view point tracking, often ignore the rich temporal context in the image sequence, thereby overlooking the global motion patterns and providing no assessment of the full trajectory reliability. These shortcomings hinder performance in scenarios with occlusion, dynamic objects, and low-texture areas. To address these challenges, we present the Long-term Effective Any Point Tracking (LEAP) module. LEAP innovatively combines visual, inter-track, and temporal cues with mindfully selected anchors for dynamic track estimation. Moreover, LEAP's temporal probabilistic formulation integrates distribution updates into a learnable iterative refinement module to reason about point-wise uncertainty. Based on these traits, we develop LEAP-VO, a robust visual odometry system adept at handling occlusions and dynamic scenes. Our mindful integration showcases a novel practice by employing long-term point tracking as the front-end. Extensive experiments demonstrate that the proposed pipeline significantly outperforms existing baselines across various visual odometry benchmarks.

1. Introduction

Visual odometry (VO) calculates the camera's position and movement from an image sequence. It is widely used in various fields including robotics [46], mixed reality [23], and autonomous driving [15]. The performance of VO highly depends on the accuracy of its point tracking front-end, which recovers relative camera motion between consecutive frames using projective geometry. It further requires the point correspondences to be static, as dynamic points can lead to inaccurate motion estimation and pose recovery.

Despite significant advances in feature tracking and optical flow estimation, most existing VO methods mainly focus on the two-view case, *i.e*., computing feature correspondences between a pair of images. However, these methods are limited as they rely solely on pairwise relative motion, thus neglecting the valuable temporal information in image sequences. Consequently, they often struggle with capturing dynamic motion patterns in tracks. Another challenge arises from the need to handle occlusions. In two-view scenarios, occlusion is not explicitly modeled, leading to misidentification of occluded points as incorrect matches. To address these challenges effectively, we propose to introduce a long-term point tracker, which can recover the trajectories of specific points across the image sequence given a sparse set of these query points. Its capability of leveraging temporal context and reliably tracking queries over multiple frames even under occlusion, makes it highly suitable for VO applications (refer to Figure 1).

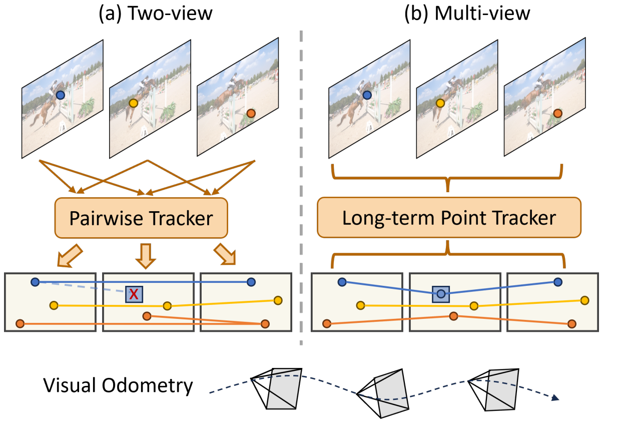


Figure 1. **Comparison between two-view approach and multi-view approach for visual odometry**. (a) In the two-view approach, correspondences are derived for every image pair and concatenated. Managing occlusion becomes challenging without the temporal context. (b) In the multi-view approach, long-term point trajectories can be obtained all at once, enabling the detection of occlusion and tracking points under partial occlusions.

Recently, Persistent Independent Particles (PIPs) [16] proposes a novel pipeline that brings the idea of iterative refinement into the long-term point tracking task, with the local correlation evidence to maintain feasible computation. While it is proficient in leveraging the temporal context present in the image sequence, it ignores the inter-track information in the image sequence and processes each query point independently. Consequently, PIPs struggles to cap-

[*] Work done at Microsoft Mixed Reality & AI Lab Zurich for a Master's degree.

ture the global motion patterns from the image sequence. Additionally, it lacks a reliability metric for its trajectory estimates, hindering its application in quality-sensitive tasks like video segmentation and visual odometry. To mitigate the aforementioned issues, we propose a pipeline that explores dynamic track estimation and temporal probabilistic formulation for long-term point tracking.

In this paper, we present Long-term Effective Point Tracking (LEAP), aiming to estimate the reliability of predicted correspondences and track moving object trajectories in dynamic scenes. To achieve this, we harness the global motion patterns—often overlooked in existing methods—through an anchor-based dynamic track estimation module, leveraging visual, temporal, and inter-track information to differentiate between the static and dynamic elements. We also incorporate a temporal probabilistic approach into our LEAP pipeline to handle uncertainties and refine the distribution of point correspondences iteratively, showcasing superior performance in complex real-world situations. More importantly, we pioneer the usage of continuous motion estimation from long-term point tracking to construct a robust visual odometry system. Our LEAP-VO distinguishes itself through its thoughtful integration of visual features, track distribution, and global motion patterns, significantly enhancing both the performance and robustness of visual odometry in dynamic environments.

2. Related Work

Long-term Point Tracking. In the realm of long-term point tracking or Tracking Any Point (TAP), various methods have been introduced recently. PIPs [16] reframes pixel tracking as a long-term motion estimation with local information and improves tracking accuracy by associating each pixel with a trajectory that spans multiple image frames. The subsequent works enhance performance by integrating spatial context features [2], tracking multiple queries concurrently [18], and extending temporal fields and templates [49]. TAP-Net [9] adopts a different method by locating candidate correspondences through a global cost volume, improved by TAPIR's trajectory refinement [10] and BootsTAP's semi-supervised training [11]. Other approaches explore self-supervision targets to learn visual correspondence via powerful feature representation [5, 17, 36]. Recently, OmniMotion [37] provides a pure test-time optimization method that infers the point trajectory and occlusion directly from videos with optical flow predictions. However, its slow inference speed makes it unsuitable for real-time applications such as visual odometry.

Monocular Visual Odometry. Visual Odometry (VO) can be categorized into indirect (feature-based) and direct methods. Indirect methods, such as MonoSLAM [8] and ORB-SLAM [25], mostly rely on feature extraction and matching to estimate the camera pose. Direct methods, conversely, skip feature extraction and matching and utilize intensity to optimize camera pose and 3D scene point positions based on photometric errors [12, 13, 38], showing better performance on low-texture regions where keypoint tracking can easily fail. However, they are more vulnerable to illumination changes and issues with rolling shutter cameras. Advancements in deep learning have spawned end-to-end monocular VO methods in supervised [3, 28, 30, 33, 39, 40, 43, 44] and unsupervised [21, 27, 45, 51] settings. DeepVO [39] utilizes Recurrent Neural Networks to model sequential data, while SfMLearner [51] develops an unsupervised framework to learn the depth and camera motion from unlabeled monocular videos. TartanVO [40] proposes a VO model that generalizes to multiple datasets and real-world scenarios. DROID-SLAM [32] follows learning-based optimization from RAFT [31] and proposes an end-to-end system, integrating flow, confidence, and geometric optimization via iterative GRU updates. The followed-up work DPVO [33] further cuts computational complexity by replacing dense feature tracking with sparse patch tracking.

Dynamic Track Estimation. In addition to enhancing accuracy, numerous studies have focused on the detection of dynamic objects within videos. Methods based on segmentation, such as DynaSLAM [1] and RCVD [20], employ semantic cues from pre-trained models to exclude dynamic regions. However, these segmentation-based methods are constrained to predefined object categories and struggle to capture real-world motion (for example, differentiating between a stationary and a moving car). Alternatively, DytanVO [28] and ParticleSfM [48] adopt a trajectory-based motion detection approach. This method addresses dynamic scenes using estimated point track information, thereby generating reliable camera trajectories in complex motion scenarios. Nonetheless, they depend on dense optical flow estimation to capture global motion patterns. Furthermore, dynamic tracks can be detected using additional depth information for a 3D consistency check [7, 47], or by optimization based on photo consistency [35, 41, 42].

3. Method

Our goal is to tackle the aforementioned challenges in VO systems, particularly in handling dynamic environments, temporal occlusions, and low-texture regions. To this end, we introduce our LEAP module, which incorporates dynamic track estimation and temporal probabilistic formulation. Furthermore, we illustrate how this module serves as the cornerstone for constructing our advanced VO system.

3.1. Preliminary: Tracking Any Point

Tracking arbitrary pixels in an image sequence is traditionally akin to estimating optical flow, where each pixel is mapped to a corresponding location in a subsequent frame. However, two-view approaches often overlook the rich temporal context within the image sequence, leading to suboptimal outcomes. Recent advancements address this by revisiting the particle video concept and improving long-term point tracking through learning-based techniques [10, 16, 18]. These TAP methods can track points over multiple frames and detect occlusions, significantly boosting tracking robustness under challenging scenarios.

Formulation. Given a sequence of S consecutive RGB images $\mathbf{I} = [\mathbf{I}_1, ..., \mathbf{I}_S], \mathbf{I}_s \in \mathbb{R}^{3\times H\times W}$ as input, the goal of TAP is to track a set of query points across these frames. For a specific query point with the 2D pixel coordinate $\mathbf{x}_q \in \mathbb{R}^2$ in frame s_q, TAP predicts its trajectory across all S images as $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_S], \mathbf{x}_s \in \mathbb{R}^2$, given $\mathbf{x}_{s_q} = \mathbf{x}_q$. Additionally, TAP estimates the visibility of each point throughout the sequence as $\mathbf{V} = [v_1, ..., v_S]$, where $v_s \in \{0, 1\}$ is a binary label of point visibility indicating whether a point in frame s is visible or occluded. The TAP formulation for a single query can be described as

$$(\mathbf{X}, \mathbf{V}) = \text{TAP}(\mathbf{I}, \mathbf{x}_q, s_q). \tag{1}$$

PIPs [16], a notable learning-based TAP method, offers a promising solution featuring local cost volumes and iterative refinement. Initially, each image $\mathbf{I}_s \in \mathbf{I}$ is processed through a CNN feature extractor $\mathcal{F}$ to extract the feature map $\mathbf{Y_s} = \mathcal{F}(\mathbf{I}_s)$. Point features $\mathbf{f}_q$ are computed by bilinear sampling at the query positions $\mathbf{x_s}$ on $\mathbf{Y}_s$. These point features from each image are then concatenated to form a point feature tensor $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_S]$. PIPs approaches the TAP problem by iteratively updating the state variables $(\mathbf{X}, \mathbf{F})$, with the initial states obtained by duplicating the query positions and features. During the k-th iterative refinement, PIPs computes a multi-scale local cost volume $\mathbf{C}[\mathbf{X}^k]$ centered around the current estimated point position $\mathbf{X}^k$. Utilizing this local evidence, PIPs predicts updates for the state variables via

$$\begin{gathered}(\Delta\mathbf{X}, \Delta\mathbf{F}) = \text{Refiner}(\mathbf{F}^k, \text{pos}(\mathbf{X}^k - \mathbf{x_q}), \mathbf{C}^k[\mathbf{X}^k]), \\ \mathbf{X}^{k+1} \leftarrow \mathbf{X}^k + \Delta\mathbf{X}, \quad \mathbf{F}^{k+1} \leftarrow \mathbf{F}^k + \Delta\mathbf{F},\end{gathered} \tag{2}$$

where pos$(\cdot)$ refers to the positional embedding. The final point feature $\mathbf{F}^K$ is used to predict point visibility with a simple linear projection layer $\mathcal{G}_v$, yielding $\mathbf{V} = \mathcal{G}_v(\mathbf{F}^K)$. Recent advances extend PIPs by replacing the MLP-based refiner with an attention-based refiner and adding spatial feature aggregation to track multiple queries together [18].

3.2. Anchor-based Dynamic Track Estimation

The established TAP model enables us to extract long-term point trajectories and thereby recover camera motion from an image sequence. However, dynamic environments pose extra challenges for VO systems that rely on static 3D point correspondences linked to unchanging scene locations. Therefore, integrating dynamic track estimation into the VO system is crucial for real-world applications.

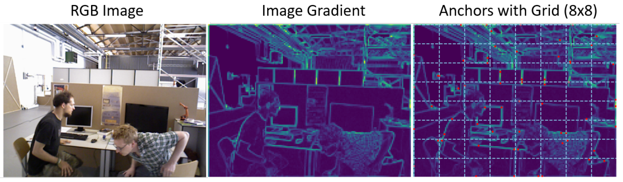


Figure 2. **Distributed image gradient-based sampling with $k = 8, N_a = 64$.** After computing the image gradient and pooling, we split the gradient map into 8×8 grids and select the point with the maximum gradient in each grid.

We propose an innovative method for dynamic track estimation that leverages visual, temporal, and inter-track information. While common techniques like monocular segmentation [20] excel at identifying moving objects using visual cues but can erroneously label static objects, like parked cars, as dynamic. Trajectory-based methods [48], utilizing optical flows from two or multiple frames, can differentiate static and moving objects apart but tend to ignore visual information and can be inefficient due to dense trajectories computation. Merging these two distinct methods is non-trivial; however, the formulation of the TAP pipeline, enhanced with inter-track attention, offers a viable solution.

Our method smartly integrates visual appearance, long-term motion, and inter-track cues, combining the strengths of segmentation and trajectory-based techniques. Utilizing TAP, we extract point features $\mathbf{F}$ and long-term trajectories $\mathbf{X}$ from the video input. These features capture visual information and the iterative refinement module exchanges information across tracks over time for accurate labeling. However, such a method depends on the query distribution and quantities. With only one query during inference time, it cannot utilize inter-track information. Likewise, if the queries focus on a single object, the system may overlook the motions of other objects due to similar motion patterns.

Anchor-based Inter-track Attention. To better capture global motion patterns, we introduce additional anchors alongside the given queries, denoted as $\mathcal{A}, |\mathcal{A}| = N_a$. Ideally, the anchor set should exhibit two characteristics: (1) the anchors should be easy to track, and (2) they should be well-distributed to encapsulate global motion patterns. Given the image $\mathbf{I}_{s_q}$ from which the queries $\mathbf{x}_q$ are extracted (we maintain the same notations for both single and multiple queries for simplicity), we first employ the Sobel Kernel to derive the image gradient $(\mathbf{G}_x, \mathbf{G}_y)$ for image $\mathbf{I}_{s_q}$. We then apply an average pooling layer on gradient maps for smoothing and downscaling feature maps. Subsequently, we divide the gradient map into $k \times k$ sub-regions and select top $\frac{N_a}{k^2}$ pixels with the highest image gradient magnitude in each local grid. The process is demonstrated in Figure 2.

To predict the dynamic label of a trajectory, we track queries with additional anchors and concatenate the original query features $\mathbf{X}$ and query trajectories $\mathbf{F}$ with anchor features $\mathbf{X}_A$ and anchor trajectories $\mathbf{F}_A$. We apply a shallow MLP layer $\mathcal{G}_d$ with average pooling over the temporal dimension. The dynamic track label $\mathbf{m}_d$ is estimated as

$$\mathbf{m}_d = \text{avgpool}(\mathcal{G}_d([\mathbf{X}^K; \mathbf{X}_A^K], [\mathbf{F}^K; \mathbf{F}_A^K])). \tag{3}$$

Since newly added anchors often lack ground truth information, we employ a semi-supervised training scheme that tracks queries and anchors together to leverage inter-track attention but only calculates losses based on query predictions. This approach prevents the network from developing bias towards specific query sampling methods while learning dynamic track labels effectively.

3.3. Temporal Probability Modeling

Supervised learning applied to large-scale datasets has yielded notable advancements in learning-based point tracking methods. Nonetheless, trajectory estimation errors remain a concern, especially under challenging conditions like occlusions, motion blur, or textureless regions. Recognizing the varying reliability of trajectories, integrating an uncertainty estimation mechanism into the correspondence estimation process is beneficial [6]. This is particularly crucial for quality-sensitive tasks such as visual odometry.

To evaluate the reliability of point correspondences, we incorporate a probabilistic formulation into the TAP pipeline. Given an image sequence $\mathbf{I}$ and a single query point $\mathbf{x}_q$, our objective is to compute the conditional probability density of point trajectory as $p(\mathbf{X}|\mathbf{I}, \mathbf{x}_q)$. Recognizing the strong correlation between points within the same track, treating each point in the track as independent becomes suboptimal. Therefore, we propose to model the distribution of point trajectories by applying two multivariate distributions, respectively for the 2D coordinates. Let $\mathbf{X} = [\mathbf{a}; \mathbf{b}]$, where $\mathbf{a} \in \mathbb{R}^S$ and $\mathbf{b} \in \mathbb{R}^S$ represent the X and Y coordinate of all points in $\mathbf{X}$, respectively. Assuming independence between these two coordinates, the joint probability distribution can be expressed as $p(\mathbf{X}|\mathbf{I}, \mathbf{x}_q) = p(\mathbf{a}|\mathbf{I}, \mathbf{x}_q) \cdot p(\mathbf{b}|\mathbf{I}, \mathbf{x}_q)$. We adopt the multivariate Cauchy distribution, which is associated with its heavy tails distribution and is more stable for optimization. The probability density function (PDF) of a single coordinate is given by

$$p(\mathbf{a}|\mathbf{I}, \mathbf{x}_q) = \frac{\Gamma(\frac{1+S}{2})}{\Gamma(\frac{1}{2})\pi^{\frac{S}{2}}|\mathbf{\Sigma}_a|^{\frac{1}{2}}[1+(\mathbf{a}-\boldsymbol{\mu}_a)^T\mathbf{\Sigma}_a^{-1}(\mathbf{a}-\boldsymbol{\mu}_a)]^{\frac{1+S}{2}}} \tag{4}$$

and a similar expression is applicable for $p(\mathbf{b}|\mathbf{I}, \mathbf{x}_q)$. Γ is the Gamma function. The parameters $(\boldsymbol{\mu}_a, \mathbf{\Sigma}_a, \boldsymbol{\mu}_b, \mathbf{\Sigma}_b)$ represent the location and scale matrices for the respective coordinates. During inference, the uncertainty associated with each point is quantified by the sum of diagonal scale estimates at position (s, s) as $\phi(\mathbf{x}_s) = \mathbf{\Sigma}_a[s, s] + \mathbf{\Sigma}_b[s, s]$.

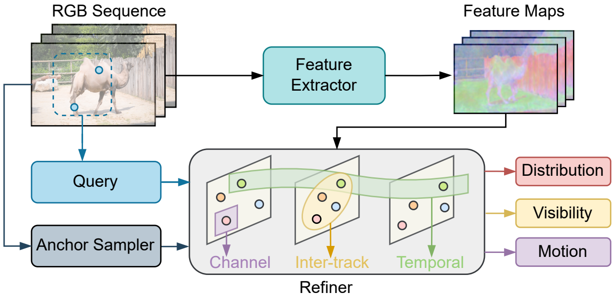


Figure 3. **LEAP Front-end**. Once image feature maps are obtained, selected anchors aid in tracking. The queries and anchors are processed by a refiner to iteratively update states. The model outputs trajectory distribution, visibility, and dynamic track label.

Kernel-based Estimation. Rather than just predicting the positions of points within a trajectory, our model focuses on recovering the distribution of point trajectories by estimating its parameters. The location parameter predictions can be associated with the previous point trajectory as $[\boldsymbol{\mu}_a; \boldsymbol{\mu}_b] = \mathbf{X}$. However, directly deriving a symmetric and positive definite scale matrix from model outputs is challenging. To address this, we employ a kernel-based approach with the linear kernel $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T\mathbf{y}$. During each iteration, two scale matrices $(\mathbf{\Sigma}_a, \mathbf{\Sigma}_b)$ are constructed by first applying two linear projection layers to the point features, represented as $\mathbf{F}_a^k = \mathcal{G}_a(\mathbf{F}^k)$ and $\mathbf{F}_b^k = \mathcal{G}_b(\mathbf{F}^k)$. We then compute the scale matrices as $\mathbf{\Sigma}_a = K(\mathbf{F}_a, \mathbf{F}_a) + \sigma\mathbf{I}$, and $\mathbf{\Sigma}_b = K(\mathbf{F}_b, \mathbf{F}_b) + \sigma\mathbf{I}$, where σ is a small positive value and $\mathbf{I}$ is the identity matrix. The model parameters are refined through Maximum Likelihood Estimation (MLE) using a negative log-likelihood (NLL) loss function.

Loss Functions. We supervise the point trajectory mainly with the NLL loss, which is based on the predicted distribution parameters and the ground-truth point trajectory. The main point trajectory loss is given as

$$\mathcal{L}_{main} = \sum_{k}^{K} \gamma^{K-k} \mathcal{L}_{NLL}(\mathbf{X}^k, \mathbf{X}^*, \mathbf{\Sigma}_a^k, \mathbf{\Sigma}_b^k), \tag{5}$$

where K is the number of iteration and $\gamma = 0.8$.

For the visibility and dynamic track label supervision, we use the cross entropy loss with the estimates $\mathbf{V}, \mathbf{m}_d$ and ground truth $\mathbf{V}^*, \mathbf{m}_d^*$ as

$$\mathcal{L}_{vis} = (1 - \mathbf{V}^*) \log(1 - \mathbf{V}) + \mathbf{V}^* \log \mathbf{V}. \tag{6}$$

$$\mathcal{L}_{dyn} = (1 - \mathbf{m}_d^*) \log(1 - \mathbf{m}_d) + \mathbf{m}_d^* \log \mathbf{m}_d. \tag{7}$$

The total loss is a weighted sum of three losses:

$$\mathcal{L}_{total} = w_1\mathcal{L}_{main} + w_2\mathcal{L}_{vis} + w_3\mathcal{L}_{dyn}. \tag{8}$$

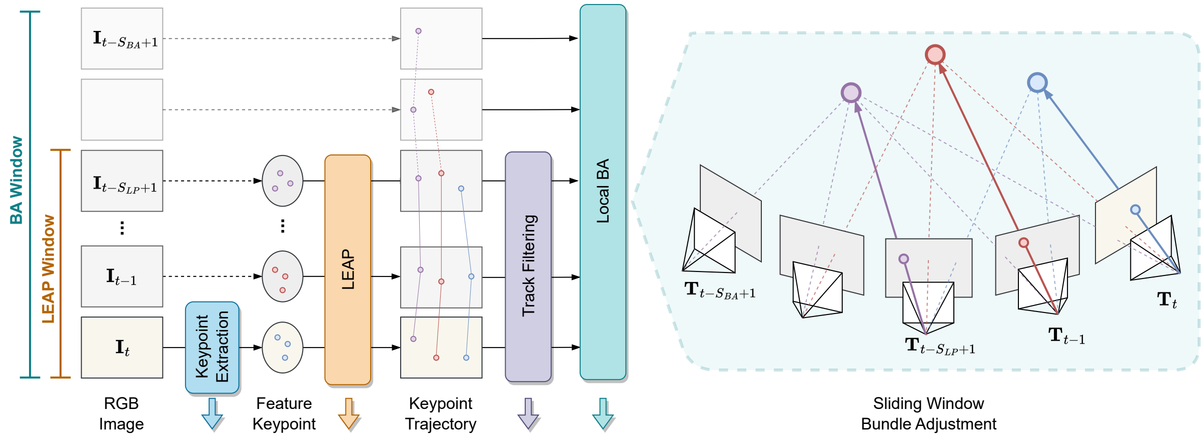


Figure 4. **LEAP-VO pipeline.** Given a new image $\mathbf{I}_t$ received at time t, the keypoint extractor extracts new keypoints from $\mathbf{I}_t$. Then, all the keypoints from the latest S_{LP}-frame $\mathbf{I}_{t-S_{LP}+1:t}$ are tracked across all other frames within the current LEAP window, followed by a track filtering step to remove outliers. Finally, the local BA module is used on the current BA window to update the camera poses and 3D positions of the extracted keypoints. The colored arrows denote the moving direction of each module when a new image is received.

3.4. LEAP-VO

Leveraging the anchor-based dynamic track estimations and temporal probabilistic formulation, we develop our Long-term Effective Point Tracking (LEAP) pipeline, boosting the accuracy of point tracking and simultaneously capturing global motion patterns and uncertainties of the tracks (refer to Figure 3). In this section, we demonstrate how we can build visual odometry for dynamic scenes based on the long-term tracking capabilities of LEAP. Our LEAP-VO pipeline is illustrated in Figure 4.

Keypoint Extraction and Tracking. In our VO system, the LEAP model serves as the front-end. When a new image frame $\mathbf{I}_t$ is captured at time t, we sample N new keypoints from this image to initiate point tracking. We consistently employ the distributed image gradient-based sampling method (refer to Section 3.2) for keypoint selection due to its favorable properties in feature distinctiveness and distribution. This approach also obviates the need for additional anchors during tracking. The LEAP model then tracks these keypoints across the last S_{LP} frames, performing bidirectional tracking (both forward and backward in time). Notably, S_{LP} can exceed the model window S, thus enabling more expansive tracking capabilities through sequential prediction [16, 18].

Track Filtering. Theoretically, precise camera pose estimation between two images can be attained with just a few accurate correspondences under certain conditions. However, the quality of correspondences obtained from feature matching or optical flow can vary. Traditional VO pipelines often use RANSAC [14] to filter out incorrect matches, but their efficiency decreases with more keypoints and views. Our approach, in contrast, bypasses the sampling-based method and uses trajectory quality assessments from LEAP for more efficient and informative track filtering.

(i) Visibility and Dynamic Track Label: The LEAP model has the capability to estimate not only the 2D point positions within a trajectory but also the visibility of each point and the dynamic track label of the entire trajectory. Thresholds γ_v for visibility and γ_d for dynamic track labels are set to ensure that only visible and static points are utilized in the bundle adjustment process.

(ii) Track Uncertainty: In Section 3.3, we introduce probabilistic modeling to LEAP, allowing us to gauge confidence via distribution measurements. We select only high-confidence points, which correspond to low uncertainty measurements. Denoting our model's uncertainty estimation as $\mathbf{\Phi}$, we retain points that exhibit high confidence using the criterion $\mathbf{\Phi} \leq Q(\gamma_u)$, where γ_u is the uncertainty quantile and $Q : [0, 1] \rightarrow \mathbb{R}$ is the quantile function.

(iii) Track Length: After filtering, we evaluate and remove tracks with insufficient observations, as bundle adjustment is more reliable with increased observations and wider baselines. Tracks with fewer than γ_{track} valid points are excluded from optimization cost computation due to their potential unreliability.

Sliding-window Optimization. With the point validation mask established, we proceed to define the optimization cost function for our sliding window bundle adjustment. The window size for local bundle adjustment, denoted as S_{BA}, may differ from the point tracking window size S_{LP}. Our geometric bundle adjustment aims to optimize camera poses $\mathbf{T}$ and 3D scene point positions $\mathbf{Q}$ by aligning the induced point trajectory from the projective relationship, with the estimated point trajectory from LEAP. We parameterize a 3D scene point $\mathbf{Q}_i$ by its 2D keypoint location $\mathbf{x}_i$ in image $\mathbf{I}_i$ and depth d_i. Let LEAP$_{i\rightarrow j}$ denote the mapping of tracking keypoint from $\mathbf{I}_i$ to $\mathbf{I}_j$ using LEAP model. The reprojection cost is formulated as:

$$\sum_{i} \sum_{j \in |i-j| \leq S_{BA}} \sum_{n} w_{i \to j,n} \| \mathcal{P}(\mathbf{T}_i, \mathbf{T}_j, \mathbf{K}, d_{i,n}) - \text{LEAP}_{i \to j}(\mathbf{x}_{i,n}) \|_\rho, \quad (9)$$

where $\|\cdot\|_\rho$ is the distance metric, and the weight $w_{i\to j,n}$ is derived from the track filtering results. We use the Gauss-Newton method [34] to optimize Eq. (9) for K_{BA} iterations. At each iteration, we compute the camera poses update $\Delta\xi^{(k)} \in \mathfrak{se}(3)$ (lie-algebra corresponding to $\mathbf{T}$), and the depth update $\Delta\mathbf{D}^{(k)}$ for each point. The optimization can be solved efficiently with the Schur decomposition. We use the robust Huber loss function for the distance metric.

4. Experiment

4.1. Datasets and Metrics

Replica. Replica [29] provides synthetic indoor environments for simulating the image-capturing process with a moving camera. We use the camera trajectories provided by [50] for the localization task, resulting in 16 camera trajectories derived from 8 static scenes. Each sequence comprises 900 frames with rendered RGB images. We use "Sequence 1" for the visual odometry evaluation.

MPI Sintel. The MPI Sintel [4] dataset contains dynamic image sequences from open-source 3D animated short films. These sequences consist of large and complex object motion, along with other imaging effects such as motion blur and defocus blur. Following prior works [20, 48], we assess dynamic VO performance on the MPI Sintel dataset, with each sequence containing 20 to 50 image frames.

TartanAir Shibuya. The TartanAir Shibuya dataset [26] features dynamic scenes from two scenarios: "Standing Humans" and "Road Crossing". The majority of the humans remain stationary in "Standing Humans" sequences, while "Road Crossing" sequences provide a more challenging setting where multiple humans are moving in different directions. Each sequence in the dataset includes 100 frames with more than 30 tracked moving humans.

Metrics. For visual odometry, we compare the Absolute Translation Error (ATE), Relative Translation Error (RPE trans), and Relative Rotation Error (RPE rot) following [33, 47, 48]. ATE measures how much the estimated trajectory deviated from the ground truth trajectory, which is calculated as the root mean square error (RMSE) overall all pose transformations after scaling and alignment. "RPE trans" computes translation errors made over a certain distance (meter), and "RPE rot" computes the estimated rotation over a certain distance (degree). Both "RPE trans" and "RPE rot" are calculated on all poses and then averaged.

Method	Replica		
	ATE (m)	RPE trans (m)	RPE rot (deg)
ORB-SLAM2 [24]	0.086	0.030	0.650
DynaSLAM [1]	0.039	0.017	0.366
DROID-SLAM [32]	0.267	0.036	2.631
TartanVO [40]	0.406	0.036	2.063
DytanVO [28]	0.289	0.035	2.146
DPVO [33]	0.257	0.036	2.635
LEAP-VO (Ours)	**0.204**	**0.030**	**1.992**

Table 1. **Camera tracking results on Replica [29].** The statistics demonstrate that our method achieves better results than other competitive baselines.

Method	MPI Sintel		
	ATE (m)	RPE trans (m)	RPE rot (deg)
ORB-SLAM2 [24]	X	X	X
DynaSLAM [1]	X	X	X
DROID-SLAM [32]	0.175	0.084	1.912
TartanVO [40]	0.238	0.093	1.305
DytanVO [28]	0.131	0.097	1.538
DPVO [33]	0.076	0.078	1.722
LEAP-VO (Ours)	**0.037**	**0.055**	**1.263**

Table 2. **Camera tracking results on MPI Sintel [4].** The statistics show that our method outperforms other competitive baselines.'X' denotes the tracking failure case.

4.2. Implementation details

We employ CoTracker [18], a recent PIPs variant, as our base TAP model. We train our model on the TAP-Vid-Kubric [9] training set, following [18]. We load the pre-trained CoTracker weights for the image feature extractor and train LEAP for 100,000 steps using 4 NVIDIA A100 GPUs in parallel. During training, we use $K = 4$ steps for iterative refinements, $N = 256$ for the number of queries, $S = 8$ for the model window, and $S_{LP} = 12$ for the tracking window. For inter-track anchor-based attention, the number of anchors is 64. The loss weights are set to $w_1 = 1.0, w_2 = 0.5, w_3 = 0.5$. In LEAP-VO, we perform 4 steps for each bundle adjustment with the Huber Loss for distance metric. The bundle adjustment window size S_{BA} is set to 15. Track filtering parameters are set at $\gamma_v = 0.9, \gamma_d = 0.9, \gamma_u = 0.8, \gamma_{track} = 3$. Please refer to our supplementary materials for more details.

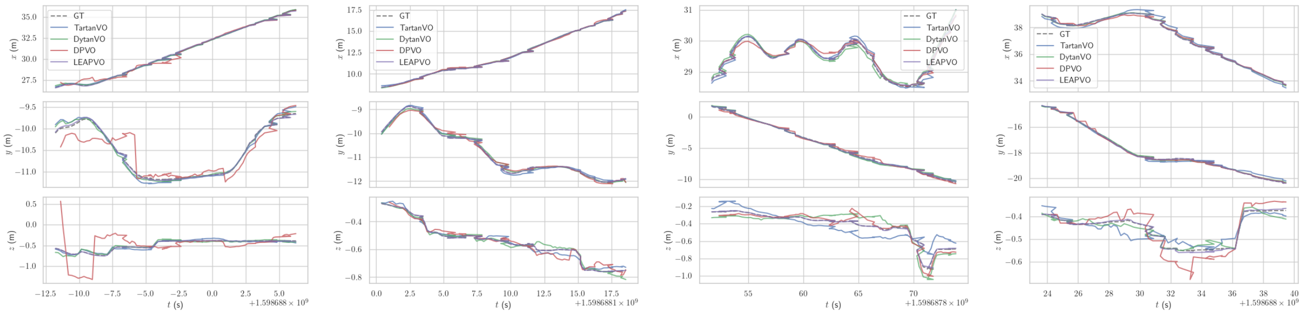
4.3. Visual Odometry Results

We compare our methods against the state-of-the-art SLAM and VO approaches, including ORB-SLAM2 [24], DynaSLAM [1], DROID-SLAM [32], TartanVO [40], DytanVO [28], and DPVO [33]. Evaluations are conducted on the static Replica dataset [29] as well as on the dynamic datasets MPI-Sintel [4] and TartanAir-Shibuya [26].

Replica. We compare our method with other baselines on the Replica dataset, which contains eight static scenes

Method	StandingHuman		RoadCrossing (Easy)			RoadCrossing (Hard)		Average
	01	02	03	04	05	06	07	
ORB-SLAM2 [24] w/ mask	0.0788	0.0060	0.0657	0.0196	0.0148	1.0984	0.8476	0.3044
DynaSLAM [1]	X	0.8836	0.3907	0.4196	0.4925	0.6446	0.6539	(0.5808)
AirDOS [26] w/ mask	0.0606	0.0193	0.0951	0.0331	0.0206	0.2230	0.5625	0.1449
DROID-SLAM [32]	0.0051	0.0073	0.0103	0.0120	0.2278	0.0253	0.5788	0.1238
TartanVO [40]	0.0600	0.1605	0.2762	0.1814	0.2174	0.3228	0.5009	0.2456
DytanVO [28]	0.0327	0.1017	0.0608	0.0516	0.0755	**0.0365**	0.0660	0.0607
DPVO [39]	0.0539	0.1603	0.1788	0.2858	0.1196	0.1119	0.1498	0.1514
LEAP-VO (Ours)	**0.0081**	**0.0185**	**0.0399**	**0.0317**	**0.0113**	0.0689	**0.0246**	**0.0290**

Table 3. **ATE (m) results on TartanAir-Shibuya Sequences [26].** $(\cdot)$ denotes averaging on valid scenes only. We can observe that our method shows better results for ATE on most sequences, outperforming all SLAM and VO baselines on average ATE.



(a) RoadCrossing 04 (b) RoadCrossing 05 (c) RoadCrossing 06 (d) RoadCrossing 07

Figure 5. **Qualitative results of camera trajectory estimation on TartanAir-Shibuya [26].** The visualizations show that our method provides more robust and accurate camera pose trajectories, especially in hard cases (RoadCrossing 06 and RoadCrossing 07).

without any moving objects. Challenges arise in two test sequences that contain prolonged occlusions, leading to the potential camera tracking loss. As illustrated in Table 1, our approach outperforms other VO methods on all three metrics, showing effectiveness under static environments and the capability to handle occlusions. Further insights are available in Section 4.5. For classical methods, ORB-SLAM2 [24] and DynaSLAM [1], we observe tracking loss and resuming via re-localization, which reduces long-term drift and shows high accuracy for camera pose tracking.

MPI-Sintel. We also evaluate our VO camera trajectory accuracy on 14 dynamic scenes from the MPI Sintel dataset, following the setting of [48]. As shown in Table 2, our method substantially surpasses other VO baselines in dynamic scenes, thanks to our temporal uncertainty estimation and dynamic track detection. Traditional feature-based methods like ORB-SLAM2 [24] and DynaSLAM [1] sometimes fail to track camera movements in highly dynamic sequences, primarily due to outliers from dynamic object correspondences and segmentation-based only dynamic track estimation. In contrast, our approach, which considers visual, temporal, and inter-track information, demonstrates notable improvements in dynamic scene tracking accuracy.

TartanAir-Shibuya. Following DytanVO [28], we further evaluate the VO performance using the dynamic TartanAir-Shibuya dataset. We assess the Average Trajectory Error (ATE) across seven diverse scenes, ranging from simple to challenging. As depicted in Table 3, our method consistently outperforms all other SLAM and VO baselines in the majority of these scenes. Moreover, trajectory visualizations in Figure 5 also highlight the robust camera tracking capabilities of our method in dynamic environments.

4.4. Dynamic Track Estimation Results

In our LEAP formulation, we have introduced the anchor-based dynamic track estimation module to distinguish between static and dynamic point trajectories, which provides valuable information for handling dynamic scenes in visual odometry. Our method leverages both point feature information and inter-track relations to predict the dynamic track label for each trajectory. To demonstrate the performance of the proposed dynamic track estimation module, we take the LEAP model and evaluate it on novel image sequences from diverse datasets. For a comprehensive demonstration, we uniformly sample 16×16 queries among the images and track them for $T = 8$ frames. The results, shown in Figure 6, illustrate that LEAP can effectively handle various scenarios, including single-object motion, complex and rapid motion, and multi-object motion with occlusion.

4.5. Ablation Study

Track Filtering. To highlight the efficacy of our track filtering module in LEAP-VO, we conduct an ablation study on different masks used in track filtering. We study three types of filtering criteria: visibility, dynamic

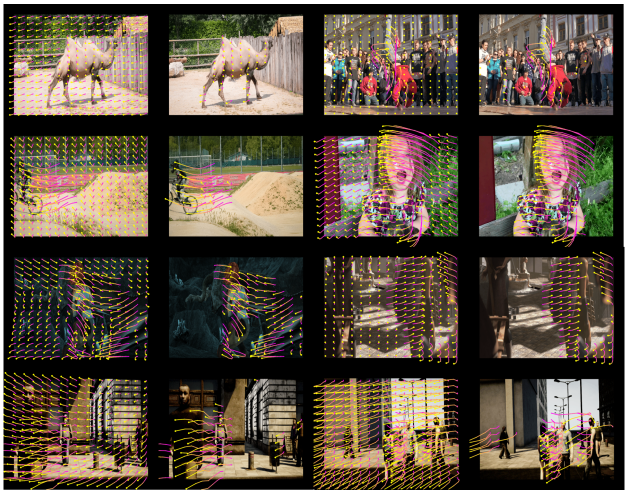


Figure 6. **Visualization of dynamic track estimation on DAVIS [19], MPI-Sintel [4], and TartanAir-Shibuya [26].** Odd columns: all point trajectories. Even columns: estimated dynamic point trajectories.

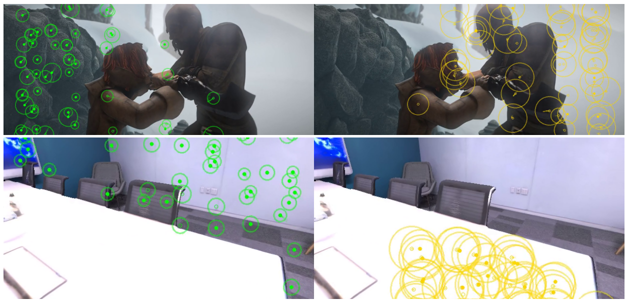


Figure 7. **Visualization of point-wise uncertainty measurements.** Keypoints with the lowest 20% (left) and highest 20% (right) uncertainty are shown in green and yellow, respectively.

track, and confidence filtering. Table 4 shows the impact of applying various filtering types on the dynamic MPI Sintel benchmark. Without dynamic filtering or confidence filtering, the ATE increases by 46% and 130%, respectively. Combining all three types of filtering methods altogether, we achieve the best results in all three metrics.

Method	Track Filtering			MPI Sintel		
	Vis.	Dyn.	Conf.	ATE	RPE trans	RPE rot
LEAP-VO	✓	-	-	0.118	0.078	1.340
	✓	✓	-	0.085	0.062	1.281
	✓	-	✓	0.054	0.058	1.274
	✓	✓	✓	**0.037**	**0.055**	**1.263**

Table 4. **Effect of the proposed track filtering method on MPI-Sintel [4].** Tracking using all criteria yields the best results.

Uncertainty Estimation. As illustrated in Figure 7, we demonstrate the LEAP's capability to provide meaningful per-point uncertainty. It models the uncertainty as distribution parameters for both the entire track and individual points. Two key observations can be found from Figure 7. Firstly, our model reliably identifies high-uncertainty points, typically located in low-texture areas or in regions with repetitive patterns. Secondly, although not explicitly supervised, the model tends to assign higher uncertainty to dynamic objects, which are inherently more challenging to track. These findings highlight the efficacy of LEAP's probabilistic formulation in handling uncertainty.

Keypoint Extraction. Table 5 showcases the effectiveness of various keypoint extraction strategies for VO performance. Despite its distinctive features, SIFT extraction provides inferior performance due to its unpredictable keypoint distribution. In contrast, the random sampling tends to generate well-separated keypoints, leading to a performance similar to SIFT's. Our distributed image gradient-based sampling shows the best performance, underscoring the significance of feature distribution and distinctiveness.

Policy:	Uniform	Random	SIFT [22]	Ours
ATE (m):	0.1511	0.0366	0.0355	**0.0290**

Table 5. **Comparison of keypoint extraction policies** ($N_a = 64$) **on TartanAir-Shibuya [26].** Our distributed image-gradient sampling method achieves better results compared to other policies.

Baseline comparison. We evaluate the pretrained CoTracker [18] for the end VO performance. Table 6 shows that LEAP front-end outperforms CoTracker on all datasets, demonstrating the effectiveness of the proposed dynamic track estimation and temporal probability modeling.

Front-end	Replica			MPI Sintel			Shibuya
	ATE	RPE trans	RPE rot	ATE	RPE trans	RPE rot	ATE
CoTracker	0.301	0.031	2.043	0.126	0.071	1.366	0.160
LEAP	**0.204**	**0.030**	**1.992**	**0.037**	**0.055**	**1.263**	**0.029**

Table 6. **Baseline comparison with CoTracker front-end [18].** Our LEAP front-end achieves better results on all datasets.

5. Conclusion

We introduce Long-term Effective Any Point Tracking (LEAP), designed to tackle the limitations of previous point-tracking methods. By leveraging visual, inter-track, and temporal cues within an anchor-based dynamic track estimation module, LEAP captures global motion patterns and effectively discriminates between static and dynamic elements. Moreover, our approach incorporates a temporal probabilistic formulation into the point-tracking pipeline, emulating recursive Bayesian filtering through a learning-based refinement module. This enables our model to accurately assess the reliability of point trajectory estimates. Using LEAP as the front-end, we then develop the LEAP-VO system, featuring novel integration of visual information, global motion patterns, and track distribution for handling dynamic scenes. Additionally, our takes for LEAP-VO can be integrated with other TAP-based front-ends, offering the potential to markedly improve camera-tracking accuracy and robustness.

References

[1] 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, 6, 7

[2] Weikang Bian, Zhaoyang Huang, Xiaoyu Shi, Yitong Dong, Yijin Li, and Hongsheng Li. Context-tap: Tracking any point demands spatial context features. *arXiv preprint arXiv:2306.02000*, 2023. 2

[3] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *CVPR*, pages 2560–2568, 2018. 2

[4] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625. Springer-Verlag, 2012. 6, 8

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *CVPR*, pages 9650–9660, 2021. 2

[6] Weirong Chen, Suryansh Kumar, and Fisher Yu. Uncertainty-driven dense two-view structure from motion. *IEEE Robotics and Automation Letters*, 8(3):1763–1770, 2023. 4

[7] Yuhua Chen, Cordelia Schmid, and Cristian Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *ICCV*, pages 7063–7072, 2019. 2

[8] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE TPAMI*, 29(6):1052–1067, 2007. 2

[9] 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. *NeurIPS*, 35:13610–13626, 2022. 2, 6

[10] 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. *ICCV*, 2023. 2, 3

[11] 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 arXiv:2402.00847*, 2024. 2

[12] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, pages 834–849. Springer, 2014. 2

[13] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE TPAMI*, 40(3):611–625, 2017. 2

[14] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 5

[15] 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. 1

[16] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *ECCV*, pages 59–75. Springer, 2022. 1, 2, 3, 5

[17] Allan Jabri, Andrew Owens, and Alexei Efros. Space-time correspondence as a contrastive random walk. *NeurIPS*, 33: 19545–19560, 2020. 2

[18] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 2, 3, 5, 6, 8

[19] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *ACCV*, 2018. 8

[20] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *CVPR*, pages 1611–1621, 2021. 2, 3, 6

[21] Shunkai Li, Xin Wang, Yingdian Cao, Fei Xue, Zike Yan, and Hongbin Zha. Self-supervised deep visual odometry with online adaptation. In *CVPR*, pages 6339–6348, 2020. 2

[22] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157. Ieee, 1999. 8

[23] Annette Mossel and Manuel Kroeter. Streaming and exploration of dynamically changing dense 3d reconstructions in immersive virtual reality. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*, pages 43–48. IEEE, 2016. 1

[24] Raúl 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. 6, 7

[25] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 2

[26] Yuheng Qiu, Chen Wang, Wenshan Wang, Mina Henein, and Sebastian Scherer. Airdos: Dynamic slam benefits from articulated objects. In *ICRA*, pages 8047–8053. IEEE, 2022. 6, 7, 8

[27] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *CVPR*, pages 12240–12249, 2019. 2

[28] Shihao Shen, Yilin Cai, Wenshan Wang, and Sebastian Scherer. Dytanvo: Joint refinement of visual odometry and motion segmentation in dynamic environments. In *ICRA*, pages 4048–4055. IEEE, 2023. 2, 6, 7

[29] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 6

[30] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. In *ICLR*, 2019. 2

[31] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419. Springer, 2020. 2

[32] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *NeurIPS*, 34: 16558–16569, 2021. 2, 6, 7

[33] Zachary Teed, Lahav Lipson, and Jia Deng. Deep patch visual odometry. *NeurIPS*, 36, 2024. 2, 6

[34] 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. 6

[35] Vadim Tschernezki, Diane Larlus, and Andrea Vedaldi. Neuraldiff: Segmenting 3d objects that move in egocentric videos. In *3DV*, pages 910–919. IEEE, 2021. 2

[36] Narek Tumanyan, Assaf Singer, Shai Bagon, and Tali Dekel. Dino-tracker: Taming dino for self-supervised point tracking in a single video. *arXiv preprint arXiv:2403.14548*, 2024. 2

[37] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once. *arXiv preprint arXiv:2306.05422*, 2023. 2

[38] Rui Wang, Martin Schworer, and Daniel Cremers. Stereo dso: Large-scale direct sparse visual odometry with stereo cameras. In *ICCV*, pages 3903–3911, 2017. 2

[39] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *ICRA*, pages 2043–2050. IEEE, 2017. 2, 7

[40] Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. Tartanvo: A generalizable learning-based vo. In *CoRL*, pages 1761–1772. PMLR, 2021. 2, 6, 7

[41] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. Dˆ 2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *NeurIPS*, 35:32653–32666, 2022. 2

[42] Jiawei Yang, Boris Ivanovic, Or Litany, Xinshuo Weng, Seung Wook Kim, Boyi Li, Tong Che, Danfei Xu, Sanja Fidler, Marco Pavone, et al. Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. *arXiv preprint arXiv:2311.02077*, 2023. 2

[43] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In *CVPR*, pages 1281–1292, 2020. 2

[44] Weicai Ye, Xinyue Lan, Shuo Chen, Yuhang Ming, Xingyuan Yu, Hujun Bao, Zhaopeng Cui, and Guofeng Zhang. Pvo: Panoptic visual odometry. In *CVPR*, pages 9579–9589, 2023. 2

[45] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, pages 1983–1992, 2018. 2

[46] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4):289–311, 2015. 1

[47] Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T Freeman. Structure and motion from casual videos. In *ECCV*, pages 20–37. Springer, 2022. 2, 6

[48] Wang Zhao, Shaohui Liu, Hengkai Guo, Wenping Wang, and Yong-Jin Liu. Particlesfm: Exploiting dense point trajectories for localizing moving cameras in the wild. In *ECCV*, pages 523–542. Springer, 2022. 2, 3, 6, 7

[49] 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*, pages 19855–19865, 2023. 2

[50] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *CVPR*, pages 15838–15847, 2021. 6

[51] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, pages 1851–1858, 2017. 2