Back on Track: Bundle Adjustment for Dynamic Scene Reconstruction

Supplementary Material

A. Implementation Details

Model Architecture. We develop our 3D tracker based on CoTracker [18], incorporating additional modifications. The feature extractor is a CNN-based architecture, consisting of a 7×7 convolutional layer followed by several 3×3 residual blocks, which generate multi-scale feature maps. These feature maps are aggregated into a single feature map with additional convolutional layers, producing a resolution of $\frac{1}{4}$ of the input image. We adopt the depth map encoding strategy from SpatialTracker [51], omitting the tri-plane correlation for improved efficiency.

The 3D tracker $\mathcal{T}$ is a transformer-based architecture that alternates between spatial and temporal attention blocks, consisting of 6 layers. For motion decoupling, we employ a smaller 3-layer transformer $\mathcal{T}_{\text{dyn}}$ designed specifically to predict the static component. Each layer in both transformers includes a temporal and a spatial attention block, with each block comprising an attention layer followed by an MLP.

Refinement Formulation. The 3D tracker uses an iterative transformer-based refiner module $\mathcal{T}$ [18]. To provide the initial state as input to $\mathcal{T}$, we copy the 2D location of the query point across all frames $s \in (1, \ldots, S)$ and sample the point features as

$$X^{(0)}(s) = X, \qquad f^{(0)}(s) = F_t[\mathbf{x}], \tag{13}$$

where the superscript $^{(\cdot)}$ denotes the iteration index. Aggregating context over all timesteps in the window, the 3D tracker $\mathcal{T}$ iteratively updates the point features $f(s)$ and the 3D trajectories $X(s)$. Dropping timestep s to avoid clutter, the update in the k-th iteration is

$$(X^{(k+1)},\ f^{(k+1)}) = \mathcal{T}(f^{(k)}, \text{PE}(X^{(k)}), C^{(k)}), \tag{14}$$

where $\text{PE}(\cdot)$ represents the positional embedding of the point track, encoding its 3D location and timestep with periodic bases.

For the dynamic tracker, $\mathcal{T}_{\text{dyn}}$ predicts the object-induced motion X_{dyn} and dynamic point feature f_{dyn} as

$$(X^{(k+1)}_{\text{dyn}}, f^{(k+1)}_{\text{dyn}}) = \mathcal{T}_{\text{dyn}}(f^{(k)}_{\text{dyn}}, \text{PE}(X^{(k)}_{\text{static}}), C^{(k)}), \tag{15}$$

where $X^{(0)}_{\text{dyn}}$ is set to 0 and $f^{(0)}_{\text{dyn}}$ is set to $F_t[\mathbf{x}]$.

Local Context. The local correlation map $C^{(k)}$ is a function of the point track $X^{(k)}$, *i.e.*

$$C^{(k)}(s) = C[X^{(k)}(s)], \tag{16}$$

and serves to enhance the spatial context of each tracked point. To compute C, we follow [51] and first apply Fourier embedding to the depth map $\mathbf{D}$ to obtain depth features $\mathbf{D}^{\text{Fourier}}$. These are concatenated with the image features $\mathbf{F}$ to produce fused features $\mathbf{F}^{\text{hyb}}$. We then apply bilinear interpolation to generate multi-scale hybrid feature maps $\tilde{\mathbf{F}}^{\text{hyb}}$. For each point $X^{(k)}(s)$, we extract a local region $\tilde{\mathbf{F}}^{\text{hyb}}_s$ centered at $X^{(k)}(s)$, and compute the correlation as the inner product between the point feature $f^{(k)}(s)$ and the surrounding hybrid features [18]. The resulting correlation map $C^{(k)}$ captures both appearance and geometric cues in a unified representation.

Training. We train our model on the TAP-Vid-Kubric training set [9], which includes 11,000 sequences. Each sequence consists of 24 frames derived from the MOVi-F dataset. Following the data preprocessing steps from CoTracker [18], we additionally extract dynamic labels, 3D total trajectory, and static trajectory ground truth. The static trajectory ground truth is generated by back-projecting queries from their 3D positions in the source frame into the target frames using ground-truth camera poses. The original image resolution of each sequence is 512×512, and we crop it to 384×512 during training. For image augmentation, we apply random resizing, flipping, cropping, Gaussian blurring, and color jitter. For depth augmentation, we adopt the scale-shift augmentation and Gaussian blurring techniques described in [15]. The augmented ground-truth depth is consistently used during the training process.

Bundle Adjustment. We build our bundle adjustment framework based on DPVO [44] and extend it to support RGB-D bundle adjustment. To enhance the robustness of pose estimation, we incorporate weight filtering during the pose update computation. We set the visibility threshold $\delta_v = 0.9$ and the dynamic label threshold $\delta_m = 0.9$ to ensure that camera pose updates rely exclusively on reliable point trajectories. This approach is adopted because recovering the camera pose primarily depends on a few accurate correspondences. In contrast, for depth updates, we consider all point trajectories to fully leverage the static components estimated through motion decoupling.

B. Additional Ablation Experiments

Comparison of single and dual network architectures. In Fig. 8, we further compare *(1)* a single-network tracker predicting only the static motion (Static*) and *(2)* our default tracker predicting the total and dynamic components. Static* struggles to capture the camera motion for dynamic points and produces motion label outliers, degrading pose estimates. This is likely because Static* predicts similar camera-induced motion for all points, making dynamic tracks hard

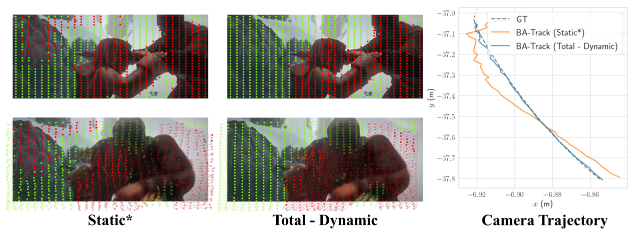

Figure 8. **Comparison of different trackers.** Red: estimated dynamic point tracks. Green: estimated static point tracks. The resulting camera trajectories from different trackers are shown on the right. Our dual networks with motion decoupling yield more accurate camera poses.

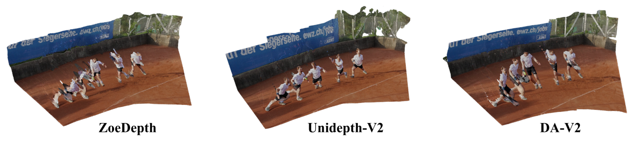

Figure 9. **Qualitative comparison across different depth priors.** Our method is robust to different types of depth priors.

to distinguish.

Robustness with different depth priors. To evaluate the robustness of our method to different depth prior models, we use two additional depth backbones: UniDepth-V2 [32] and Depth-Anything-V2 (DA-V2) [33], with per-video metric alignment based on the UniDepth-V2 scale. We compare the dynamic reconstruction results of our method using ZoeDepth, UniDepth-V2, and DA-V2. As shown in Fig. 9, our method remains robust across different depth priors for both camera pose estimation and reconstruction.

Failure modes. Our motion-decoupled tracker is robust to short-term occlusions and moderate motion but may fail under prolonged occlusions or complex, unseen motions. While BA-Track generalizes well to diverse rigid motion and performs strongly on most crowded scenes in Shibuya and DAVIS, its accuracy can degrade under severe occlusion or when tracking many small objects. Additionally, the method struggles with highly deformable or non-rigid objects, which are underrepresented in the TAP-Vid-Kubric [9] training set. Expanding training to larger and more diverse dynamic datasets could address these limitations.

C. Additional Qualitative Results

Motion Decoupling. We present additional results for motion decoupling on various video samples from the DAVIS [21] dataset, as shown in Fig. 10. The examples cover different motion scenarios, including single-object motion, multi-object motion, and occlusions. Our motion decoupling point tracker effectively distinguishes dynamic trajectories from static trajectories, providing robust estimates of the static components (illustrated in the last column). Treating the dynamic parts as static enables us to incorporate them into the geometry recovery process through bundle adjustment.

D. Limitations and Discussion

Joint Refinement with Camera Parameters. By design, BA-Track assumes the camera intrinsic parameters are provided either as ground truth or as estimates from off-the-shelf methods. These parameters are essential for bundle adjustment, which recovers the camera pose and the sparse point depths. However, calibration errors or estimation noise can introduce inaccuracies in the intrinsic parameters and lead to unreliable reprojection loss. One way to address this limitation is to extend our framework to jointly optimize camera intrinsics along with pose and point depths. Starting from an initial estimate, the intrinsics can be iteratively refined during optimization [37]. Future work could integrate this refinement with better initialization to improve convergence and reduce computational cost.

Depth Refinement. To enable track-guided depth refinement, we introduce a deformable scale map applied to the original dense depth map. This scale map facilitates refinement by aggregating information from sparse point tracks in a smooth and coherent manner, bridging the gap between sparse tracks and dense depth mapping. While effective, the scale map has limitations in handling complex error patterns present in monocular depth estimates. Future work could explore refinement models based on dense vector fields, which may improve continuity and smoothness. Additionally, representing the depth map with neural networks and refining it by backpropagating gradients to update the network weights presents another promising avenue for exploration [54].

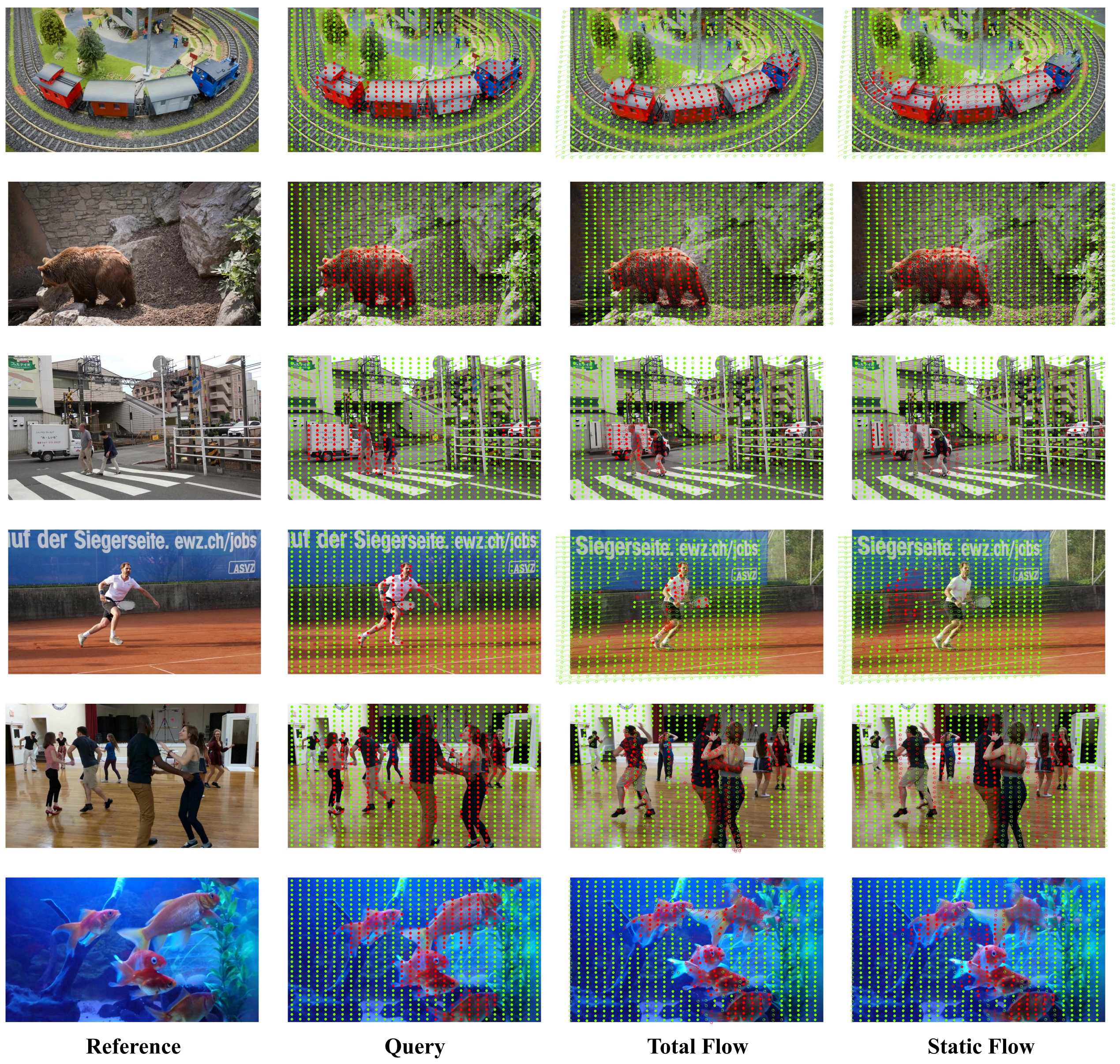

Figure 10. **Visualization of motion decoupling on the DAVIS dataset [21].** The reference frame corresponds to the first frame of the video, from which the queries are extracted. The total flow represents the combined motion of points, while the static flow refers to the motion attributed solely to its static components. The estimated static and dynamic trajectories are depicted in green and red, respectively. We observe that the red points in the static flow largely remain in their original reference frame, signifying successful motion decoupling.