Supplementary Material

A. Implementation Details

Training Loss. We train PIPs+ and PIPs++ using the weighted L_1 distance between the estimated trajectory and the ground truth trajectory across iterative updates as proposed by Harley et al. [2]. For any query point p_n, we compute the loss as follows:

$$\mathcal{L}_n = \sum_{k=1}^{K} \left(\gamma^{K-k} \frac{1}{T} \sum_{t=1}^{T} ||p_{t,n}^k - p_{t,n}^*||_1 \right) , \quad (1)$$

where $p_{t,n}^k$ denotes the estimated position at timestep t, from iteration k, and $p_{t,n}^*$ is the ground truth. We set $\gamma = 0.8$ in our experiment. The full training loss is obtained by averaging the per-point loss across all the N query points:

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}_n \, . \quad (2)$$

The loss is applied even when the target is occluded, which asks the model to estimate the track during visibility gaps. Note that PIPs [2] is also trained with a visibility classification loss, to predict whether a point is occluded, along with a score loss that supervises the similarity score map to peak at the correct locations to help the network converge faster. We find that those losses can be omitted without harming tracking performance, and therefore we only use the $\mathcal{L}$ loss presented in Eq. (2).

Test Time Trajectory Chaining. In order to track with PIPs for more than 8 frames, Harley et al. [2] link multiple 8-frame predictions. The linking strategy works per-point: after tracking for 8 frames and estimating visibility on each frame, the tracker is re-initialized on the last timestep whose visibility exceeds a threshold. While PIPs runs quickly within 8-frame clips, this chaining strategy leads to an overall FPS of 3.6 (at 720×1080 on an Nvidia V100 GPU). Although PIPs++ does not in principle require a linking strategy, our GPU memory constraints necessitate one. We use a very simple strategy: we predict 36 timesteps at a time, in sliding-window fashion, with *no overlap* between the windows. Since there is no visibility check, this is very fast, leading to an overall FPS of 55.2. The model can be run with larger or smaller windows, but in a small grid search we found that 36 works best, possibly because this was also the sequence length used at training time.

B. Additional Visualizations of PointOdyssey

Sample animated characters from the dataset are shown in Fig. 1. By retargeting motion data to these characters, we are able to generate a wide range of interacting sequences, as illustrated in Fig. 2.

Samples of our re-built motion capture environments are shown in Fig. 3.

Fig. 4 shows sample images pixel trajectories from the dataset.

We recommend watching the supplementary video for additional visualizations.

C. Additional Results

We show the performance of PIPs [2] and our PIPs++ method on real-world data in Fig. 5. While PIPs can track visible points effectively, it struggles with occlusions. Trajectories from PIPs++ are on average less sensitive to occlusions.

References

[1] 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 Datasets and Benchmarks*, 2022. 5

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

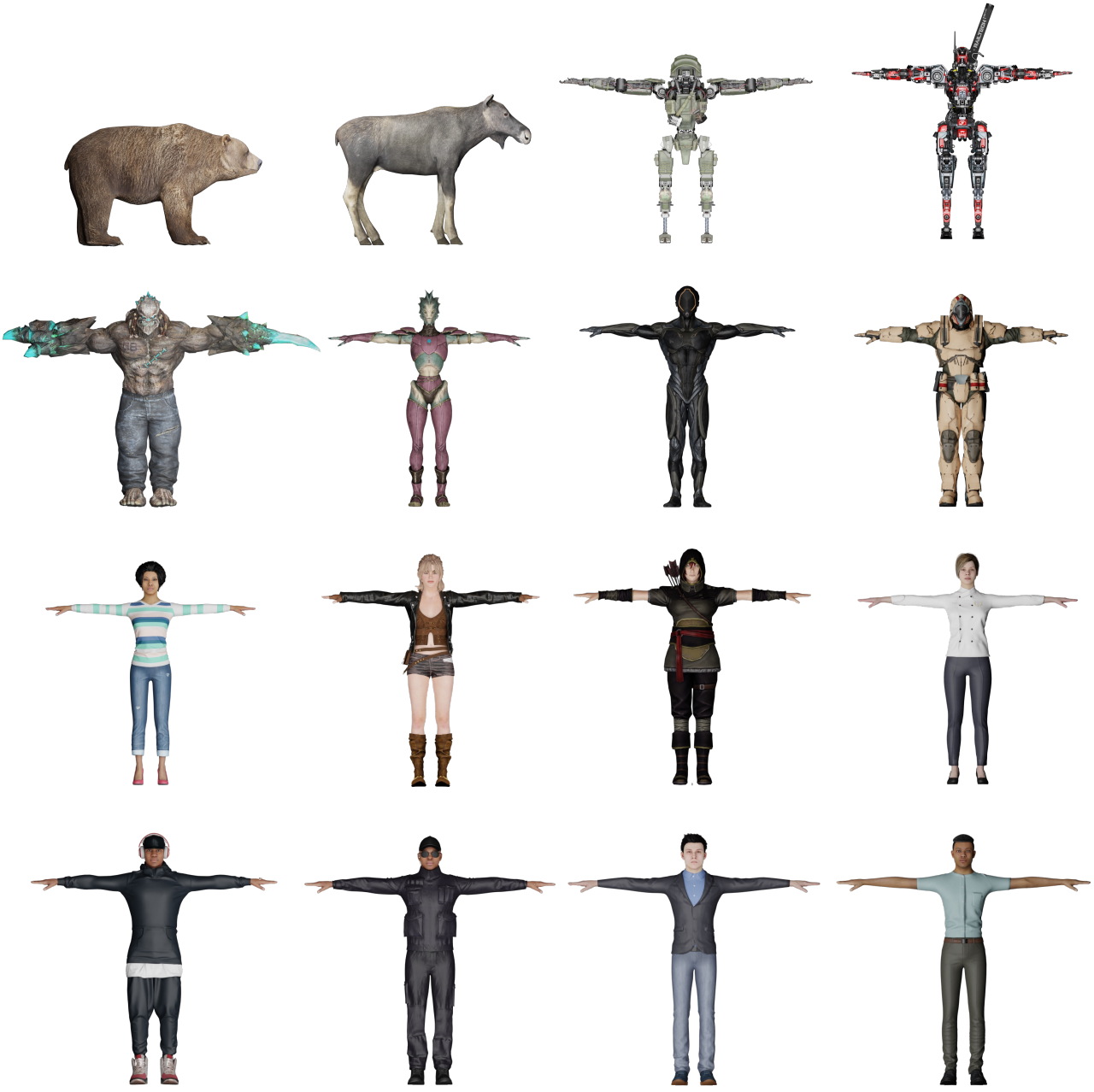

Figure 1: Sample characters from our dataset.

Figure 2: Sample motions from our dataset.

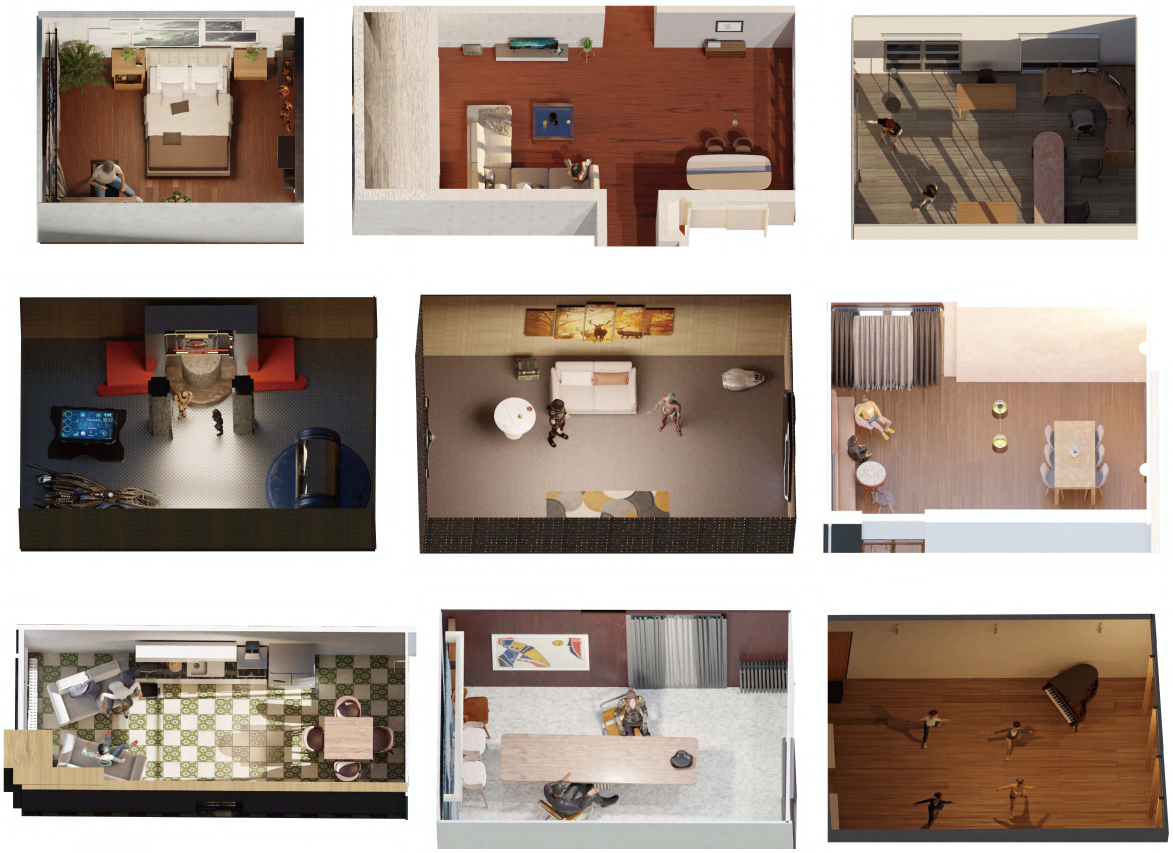

Figure 3: Examples of the rebuilt 3D scenes, with humans at a random timestep in their trajectories.

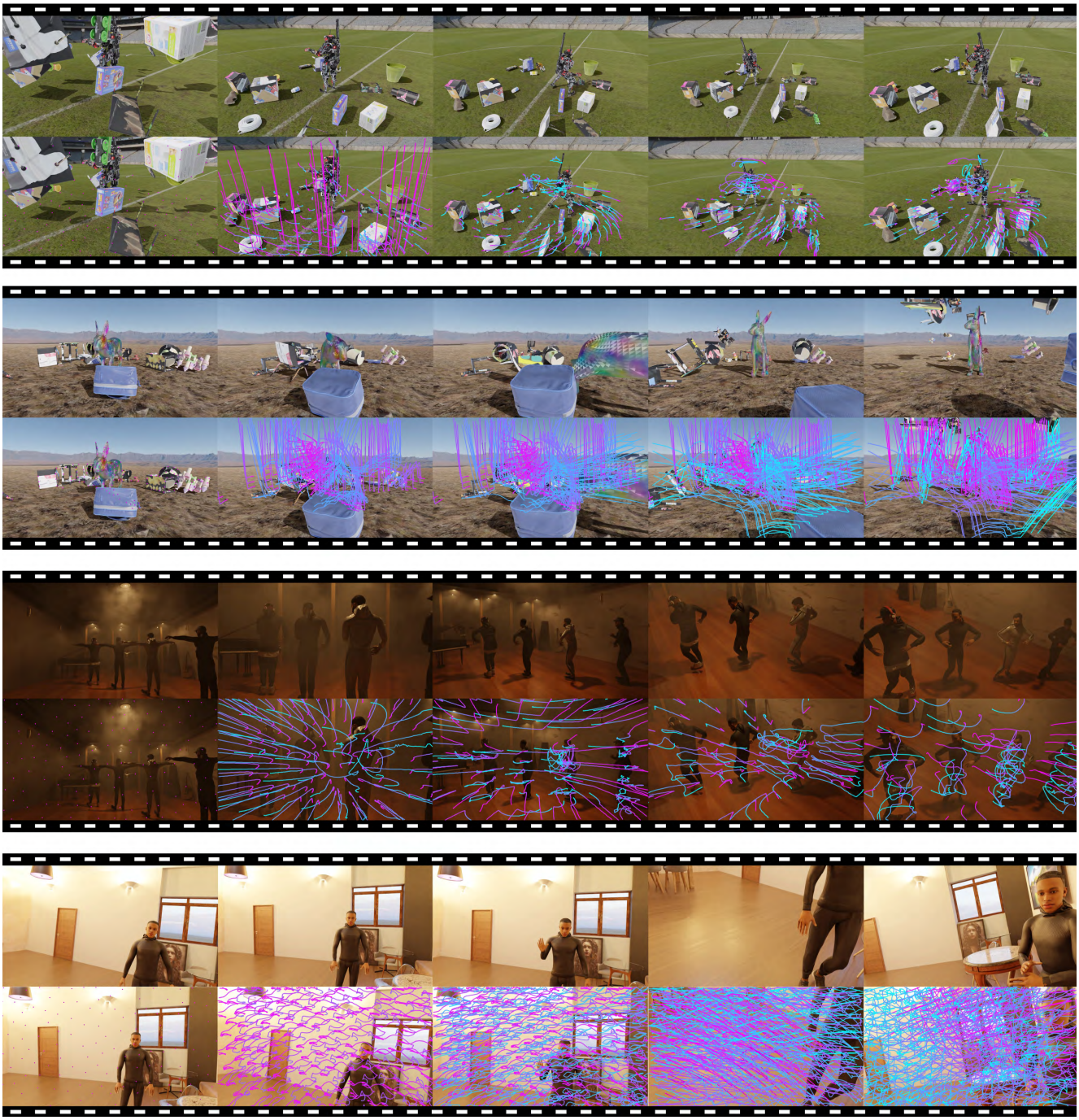

Figure 4: Sample RGB videos and point trajectories from PointOdyssey. The four rows show: (1) a robot walking in a stadium interacting with random objects; (2) a crystal rabbit in the desert interacting with random objects; (3) four human characters dancing in a room; (4) an ego-centric view of one character talking to another.

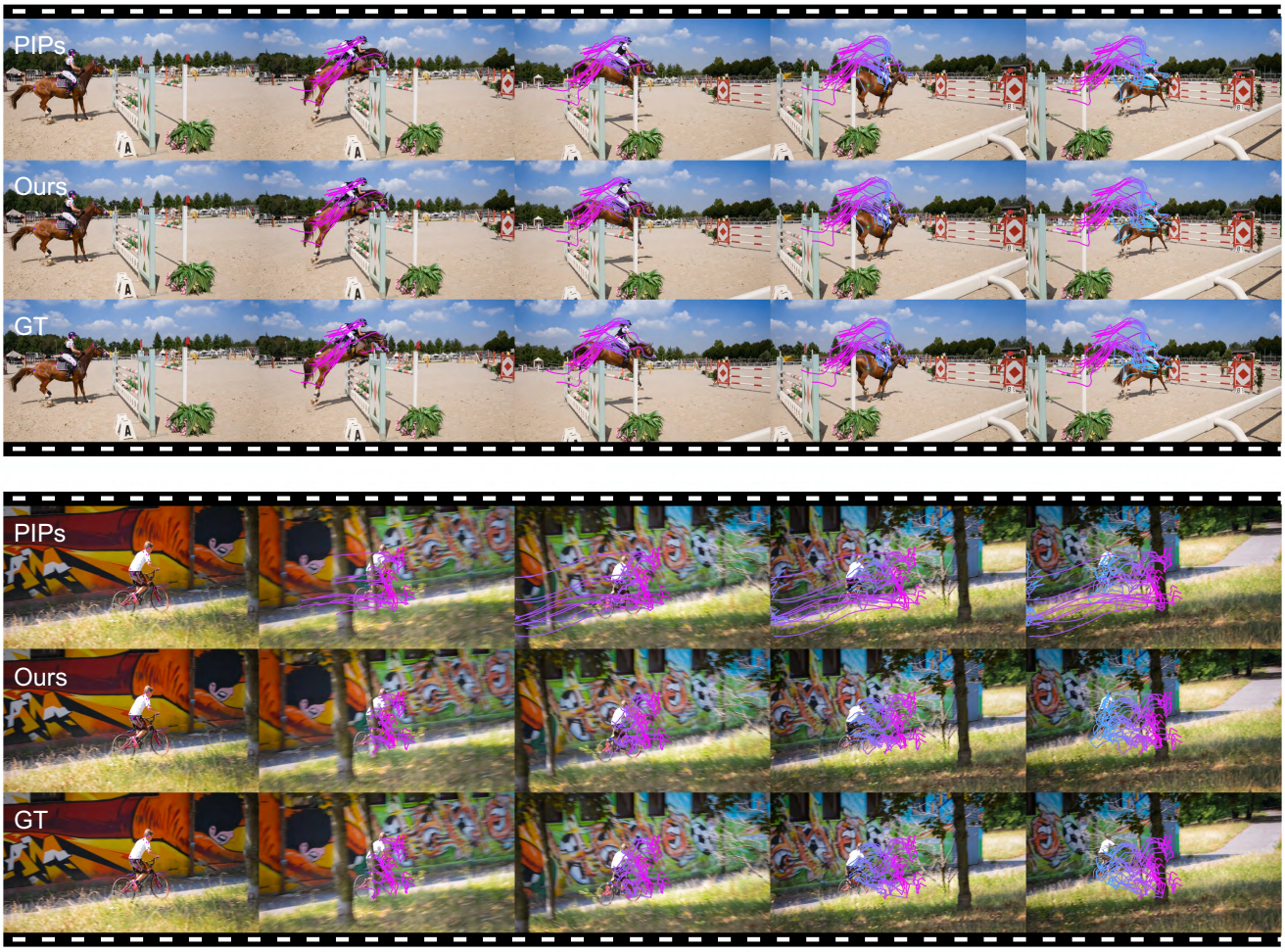

Figure 5: Qualitative results on Tap-Vid-DAVIS [1]. While PIPs [2] can track well during high visibility (as shown in the top horse riding sequence), it becomes unreliable when occlusions occur (as shown in the bike riding sequence). PIPs++ can withstand such occlusions more frequently than PIPs, likely thanks to its wider temporal receptive field.