

# SyncTrack4D: Cross-Video Motion Alignment and Video Synchronization with Multi-Video 4D Gaussian Splatting

## Supplementary Material

### A. Dataset and Evaluation Details

We evaluate our method on the CMU Panoptic Studio dataset [13] and the SyncNeRF Blender dataset [18], both of which provide multi-camera video sequences with sufficiently long temporal durations. For Panoptic Studio, we use the Dynamic3DGS split for ease of geometry acquisition. This split contains six human-centric scenes: *basketball*, *football*, *juggle*, *tennis*, *softball*, and *boxes*. Each scene includes 31 synchronized camera videos, and we use the first 150 frames for our experiments. The SyncNeRF Blender dataset consists of three synthetic scenes (*fox*, *blender*, *box*), each captured with 14 cameras and 270 frames. We use the Dynamic3DGS-provided depth maps for the Panoptic Studio dataset. Similarly, for the SyncNeRF Blender dataset, we obtain depth maps using the same Dynamic3DGS pipeline.

**Unsynchronization setup.** Since both datasets provide synchronized videos, we introduce custom temporal offsets to simulate unsynchronized conditions. We consider two evaluation settings. For measuring time differences, we report the mean absolute time-offset error relative to the reference camera, computed as the average of  $|\hat{\Delta}t_v - \Delta t_{\text{ref}}|$  over all query cameras  $v$  in the scene. We use the last camera as the reference in both datasets: camera ID 30 for Panoptic Studio and camera ID 13 for SyncNeRF Blender.

**(1) Two-view synchronization.** Used to evaluate our DTW-based coarse initialization (Table 1). We define three offset ranges:  $0 \leq |\Delta t| \leq 10$ ,  $10 \leq |\Delta t| \leq 30$ ,  $30 \leq |\Delta t| \leq 50$ . For each range, we randomly sample 180 video pairs with varying start frames and time offsets, ensuring a minimum temporal overlap of 90 frames.

**(2) Many-view synchronization and rendering.** Used for evaluating our full pipeline. For each scene, we randomly assign per-camera time offsets with a maximum pairwise difference of up to 50 frames, again requiring at least 90 overlapping frames between every camera pair.

### B. Implementation Details

For the per-video 4D track estimation module, we combine MoSca [23] with the feature map rasterizer of [50]. Each 4D Gaussian is augmented with a 1024-dimensional semantic feature attribute supervised by 2D DINOv3 feature maps. Since we assume multi-view consistent poses and depths are provided by sensors or external multi-view models, we skip the bundle adjustment stage of MoSca [23]. To reduce

memory overhead, we fix the number of static and dynamic Gaussians to 10,000 each, and we use a MoSca world-unit scaling factor of 0.05. We follow the default MoSca hyperparameters for this stage, except for the learning rates of scaffold nodes and rotations, which are set to 0.001 and 0.01, respectively. For feature map, we used L1 loss with learning rate 0.1

For the Gromov-Wasserstein method, we use the implementation in the *POT Python Optimal Transport* library [9, 10]. Specifically, we use the fused unbalanced Gromov-Wasserstein method, which can consider 4D tracks without matching correspondences. We use  $\alpha = 0.1$  for the linear terms for feature couplings, 0.1 for the marginal relaxation terms, and 0.01 for the entropic regularization terms. We use *dtw-python* for the implementation of the dynamic time warping (DTW) algorithms. We use DTW with open-begin and open-end boundary conditions. Also, we regularize the slope of correspondences to be near 0.5-2.0 by using the *rabinerJuangStepPattern*.

We implement multi-video motion-spline 4DGS on top of the MoSca backend [23]. While using the same loss configurations as MoSca [23], we disable the densification and pruning of 4DGS and nodes during multi-video 4DGS, since it leads to unbalanced pruning of 4DGS across video views. For the cubic Hermite spline, we follow the descriptions in [34]. We used 50 control points for all the tracks.

### C. Additional Results

Our 4D track results are best viewed in the supplementary videos. We also provide additional rendering results (Fig. 12) and per-scene synchronization performance (Fig. 11) on the Panoptic Studio dataset. From the synchronization results, we observe that the temporal offsets of most cameras successfully converge during our multi-video 4DGS optimization. As demonstrated in the per-camera temporal offset estimation results in the figure. We also report additional rendering results in Fig. 12, where our method demonstrates high-fidelity rendering quality along with the estimated 4D tracks.

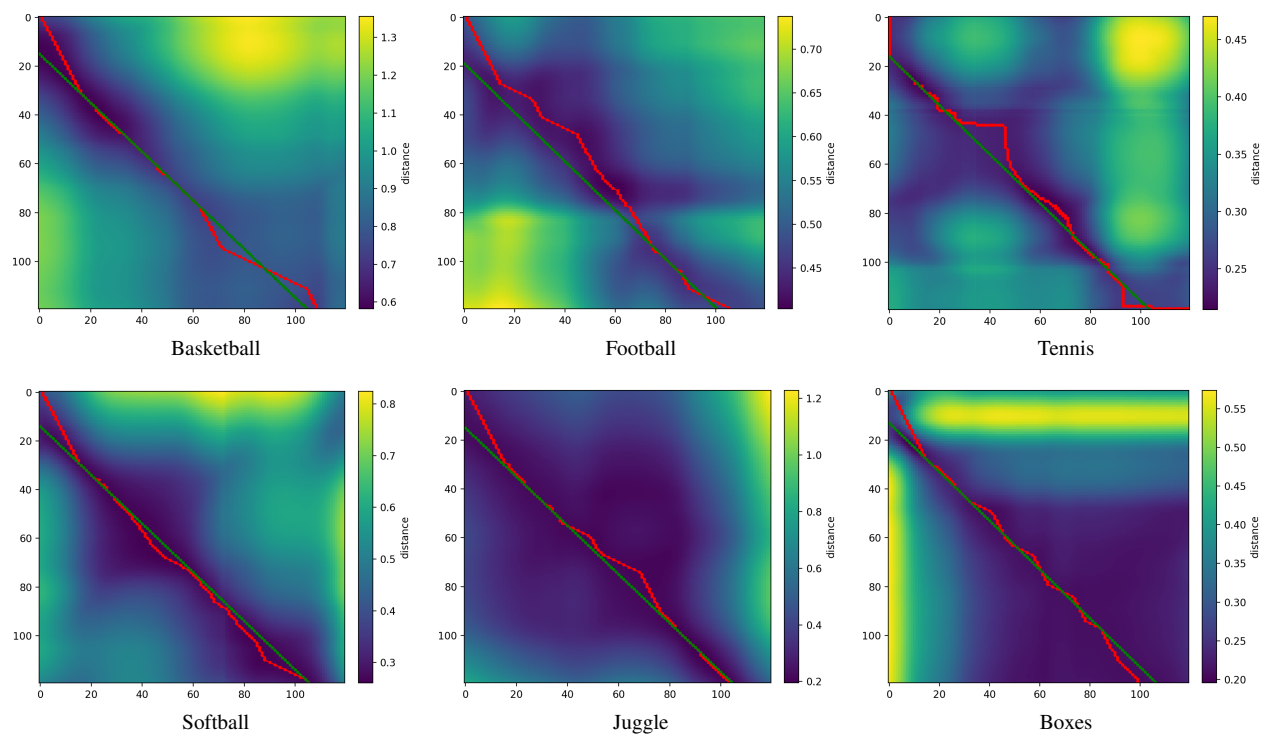


Figure 10. Example DTW cost maps for each scene. The distinctiveness of each map varies depending on the amount and type of motion observed in the scene. The red lines indicate the frame-to-frame correspondences estimated by DTW. We determine the temporal offset for each scene by selecting the most frequent offset among these correspondences (green line).

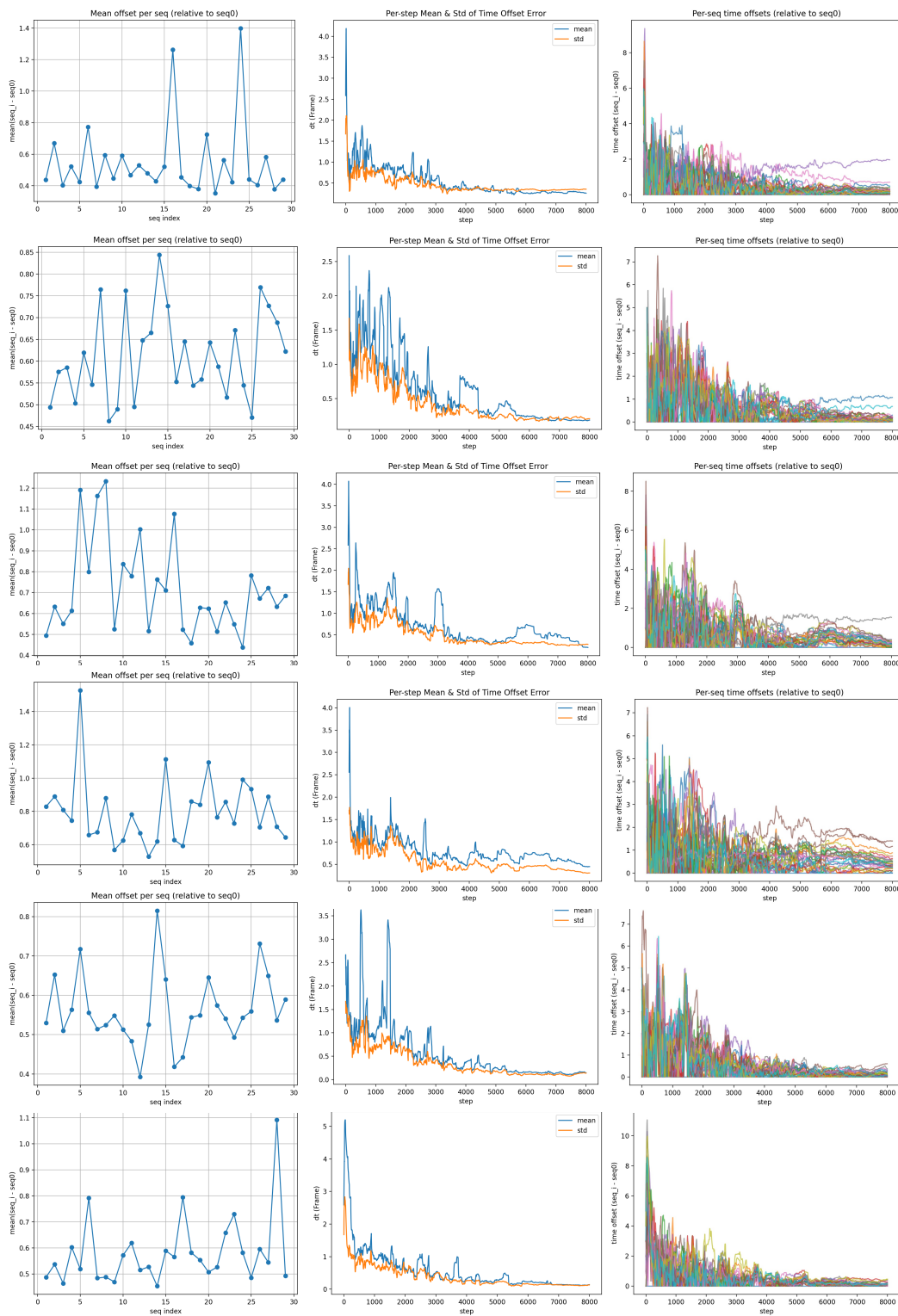


Figure 11. Additional DTW results on Panoptic Studio dataset. From the top: basketball, tennis, boxes, juggle, softball, football.

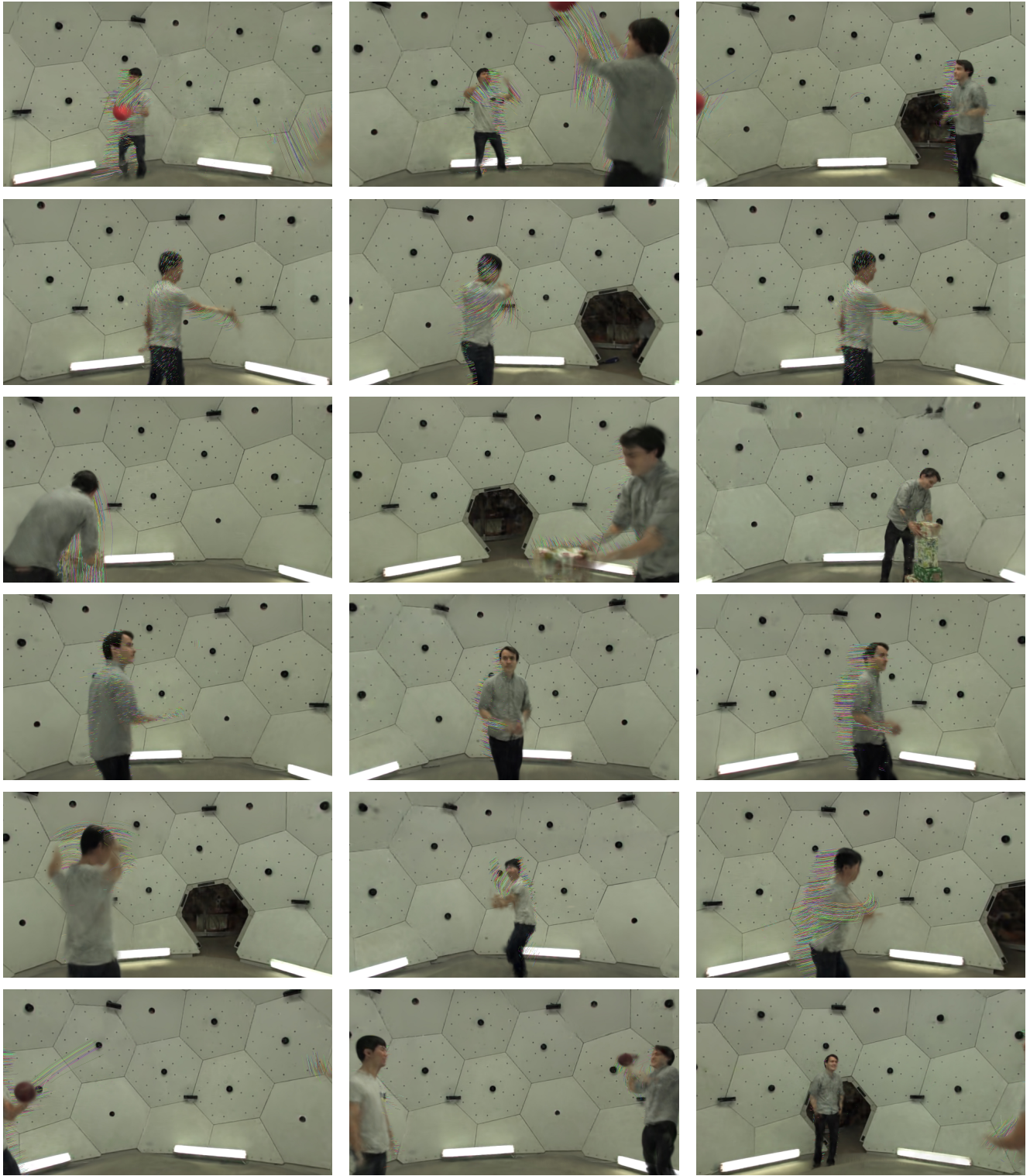


Figure 12. Additional rendering results on Panoptic Studio dataset. From the top, basketball, tennis, boxes, juggle, softball, football scenes.