A. Overview

In this supplementary material, we provide additional details on implementation (Sec. B) and video data processing (Sec. C). We highly recommend viewing our *project page* for compelling demonstrations.

B. Additional Implementation Details

B.1. Details on Model Training

We train the single-view 2D motion diffusion model for 300,000 steps and the multi-view diffusion model for 120,000 steps. The Adam optimizer[10] is used with a learning rate of 1×10^{-4}, a batch size of 64, and no weight decay. All training is performed on a single NVIDIA L40S GPU. Model checkpoints are saved every 20,000 steps, and the final model is selected based on validation performance. During the training process of multi-view diffusion, we randomly mask out a subset of input keypoints to handle partial occlusions and potential tracking failures.

For the hybrid data-source training, we use a data ratio of 2:1 between 2D keypoints extracted from real Internet videos and local 2D projections reprojected from reconstructed 3D motions.

B.2. Details on Object Motion Optimization

We provide details on how we recover the 3D object motion from our sampled multi-view 2D results. We start with obtaining the 3D object keypoints $\mathbf{Q} \in \mathbb{R}^{T\times M\times 3}$ by minimizing the multi-view reprojection error. Using the reconstructed $\mathbf{Q}$ and the predefined canonical keypoints $\mathbf{P} \in \mathbb{R}^{M\times 3}$ on the object mesh, we then estimate the object pose $\mathcal{O}_t = \{\mathbf{r}_t, \mathbf{t}_t, s\}$, where $\mathbf{r}_t \in \mathbb{R}^6$ is the 6D rotation [53], $\mathbf{t}_t \in \mathbb{R}^3$ is the translation, and $s \in \mathbb{R}^+$ is a global scale factor. We denote the mapping from $\mathbf{r}_t$ to the rotation matrix as $\mathbf{R}_t = \mathrm{Rot}(\mathbf{r}_t) \in \mathrm{SO}(3)$.

We first initialize each frame independently by aligning the canonical keypoints $\mathbf{P}$ with the observed 3D keypoints $\mathbf{Q}_t$ through a rigid transformation and scale estimation as:

$$\hat{\mathbf{r}}_t, \hat{\mathbf{t}}_t = \arg\min_{\mathbf{r}_t,\mathbf{t}_t} \sum_{i=1}^{M} \left\| \hat{s}\,\mathrm{Rot}(\mathbf{r}_t)\mathbf{p}_i + \mathbf{t}_t - \mathbf{q}_{i,t} \right\|_2, \quad \text{(S1)}$$

where the global scale is computed from the distance ratio of a reference keypoint pair in the observed and canonical spaces:

$$\hat{s} = \frac{\| \mathbf{q}_{1,t} - \mathbf{q}_{2,t} \|_2}{\| \mathbf{p}_1 - \mathbf{p}_2 \|_2}. \quad \text{(S2)}$$

This initialization yields per-frame pose estimates without temporal coupling. After initialization, we further optimize the object pose sequence with the object fitting loss:

$$\mathcal{L}_{\text{fit}}^{\text{obj}} = \frac{1}{TM} \sum_{t=1}^{T} \sum_{i=1}^{M} m_i \left\| s\,\mathrm{Rot}(\mathbf{r}_t)\mathbf{p}_i + \mathbf{t}_t - \mathbf{q}_{i,t} \right\|_2, \quad \text{(S3)}$$

where $m_i \in \{0, 1\}$ is a point-wise visibility mask shared across all frames.

Additionally, we apply a regularization loss on consecutive rotations to ensure temporal smoothness:

$$\mathcal{L}_{\text{smooth}}^{\text{obj}} = \frac{1}{T-1} \sum_{t=1}^{T-1} \left\| \mathbf{r}_t - \mathbf{r}_{t+1} \right\|_2. \quad \text{(S4)}$$

The optimization loss for object motion is defined as:

$$\mathcal{L}_{\text{obj}} = \mathcal{L}_{\text{fit}}^{\text{obj}} + \lambda_{\text{smooth}}^{\text{obj}} \mathcal{L}_{\text{smooth}}^{\text{obj}}. \quad \text{(S5)}$$

The Adam optimizer [10] is employed for all optimization stages. To reconstruct the 3D keypoint sequence from the sampled multi-view 2D motions, we run 500 optimization iterations with a learning rate of 0.01. During the object pose fitting stage, we perform 2,000 iterations with a learning rate of 0.05.

B.3. Details on Predefined Camera Trajectory

To enrich camera trajectory diversity, we predefine six camera movement modes: zoom in, zoom out, move left, move right, rotate clockwise, and rotate counterclockwise. Each mode generates a sequence of camera transformations simulating the corresponding motion pattern. For each sequence, we randomly determine whether the camera keeps moving in one direction throughout the sequence or moves back to its original position after reaching the maximum displacement. In addition, with a certain probability, the camera tracks the pelvic joint to keep the human roughly centered in the view.

During training, the camera configuration is dynamically determined at each step. For each 3D motion sequence, we randomly choose to use either a camera trajectory extracted from real Internet videos (70%) or one of the predefined camera trajectories (30%) to reproject it into 2D for training.

B.4. Details on Experimental Setup

We provide details on how we synthesize additional training and evaluation data by projecting the 3D ground-truth motions using simulated moving cameras on the AIST++ [19] and BEHAVE [1] datasets. Following our hybrid data-source training, we reproject the 3D data using camera viewpoints sampled from a large camera motion base estimated from our gymnastic and martial arts videos, and we also incorporate a small set of predefined camera trajectories (Sec. B.3) with a ratio of 7:3 between the two sources.

We ensure that camera motions used for training do not overlap with those in the testing set. The estimated camera motion base is inherently divided into separate training and testing subsets to enforce this separation. For the predefined cameras, the range of motion, the decision of whether to return to the origin, and whether to track the human pelvic joint are all randomized, ensuring that no camera trajectory in the testing phase exactly matches any trajectory seen during training.

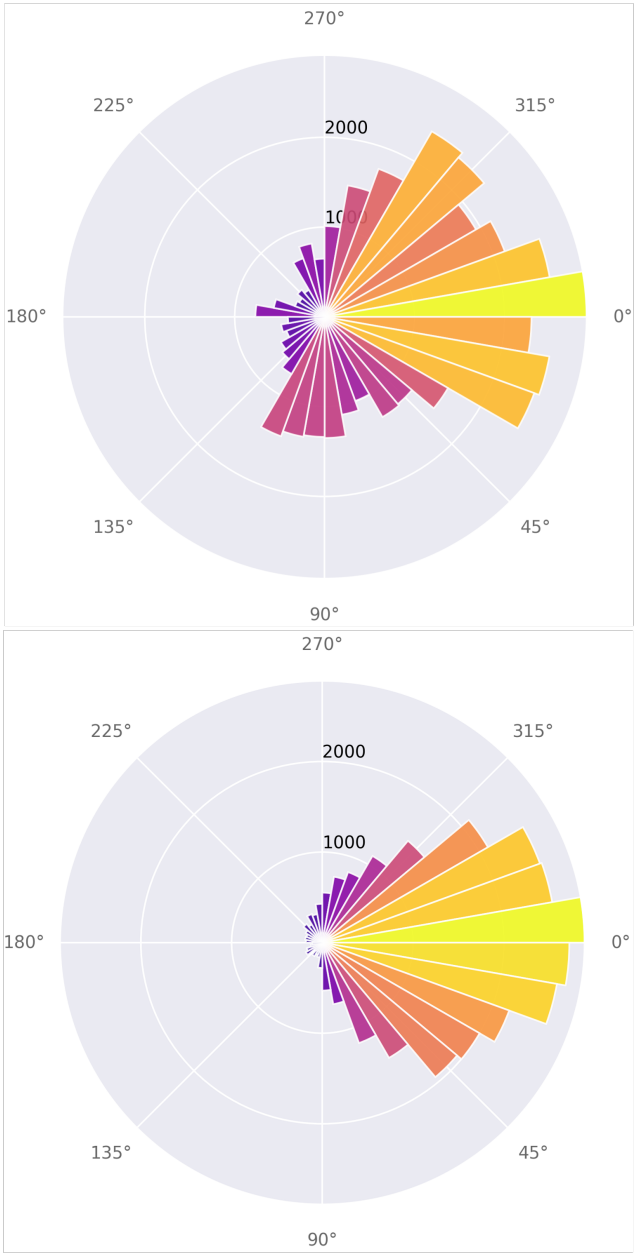

Figure S1. **Facing direction distributions of estimated humans** in the gymnastics (upper) and martial arts (lower) videos under the camera coordinate system. The angular axis indicates the facing direction and the radial axis represents number of frames.

C. Additional Details on Data Processing

C.1. Details on Internet Video Processing

We provide details on how we process the manually collected Internet videos. We first filter out low-quality videos, such as those containing advertisements or long idle segments. The remaining videos are then divided into 10-second clips, resulting in roughly 4,000 gymnastics clips and 5,000 martial arts clips. In subsequent processing, we further filter out these clips to handle failures in human tracking and 2D pose estimation. Segments with low pose confidence score or severe 2D jittering are discarded. After the filtering process, we obtain around 1,600 gymnastics clips and 3,000 martial arts clips that are used for training and evaluation. The processed data are then split into training and testing sets with a 9:1 ratio.

As shown in Fig. S1, we plot the facing directions of the estimated humans in the gymnastics and martial arts videos in the camera coordinate system using polar plots, where the angular axis represents the facing direction and the radial axis corresponds to the number of frames. The results show that the facing directions in both datasets are mostly concentrated within the same semicircle, which supports our observation that the viewpoint coverage of online videos is severely limited.

C.2. Details on HOI Video Processing

Processing Videos from the BEHAVE Dataset. The BEHAVE dataset [1] involves complex object motions, particularly frequent rotations, which makes off-the-shelf tracking methods [28] unreliable. To obtain more reliable 2D keypoints, we avoid tracking in the image plane and instead recover the per-frame object pose by aligning the object mesh to the observed object mask using a point sampling, projection-based method.

Concretely, for each object, we start from its template mesh and pre-select a small set of mesh vertices as semantic keypoints. In each frame, we first obtain an object segmentation mask using Grounded-SAM [32], and randomly sample 5,000 2D points $\mathcal{Q} = \{\mathbf{q}_j\}$ from the foreground pixels. We also randomly sample 5,000 3D surface points $\mathcal{X} = \{\mathbf{x}_i\}$ from the object mesh. For projection, we fix a pinhole camera model with focal length $f_x = f_y = 1000$ and principal point $(c_x, c_y) = (\frac{W}{2}, \frac{H}{2})$ determined by the image width W and height H. With these intrinsics and an unknown $\mathrm{SE}(3)$ pose $(\mathbf{R}, \mathbf{t})$ of the object in the camera coordinate system, each 3D point is projected to the image as

$$\mathbf{p}_i = \Pi(\mathbf{R}\mathbf{x}_i + \mathbf{t}), \tag{S6}$$

where $\Pi(\cdot)$ denotes the pinhole projection. Let $\mathcal{P} = \{\mathbf{p}_i\}$ be the set of projected points. We estimate $(\mathbf{R}, \mathbf{t})$ by minimizing a symmetric 2D Chamfer distance between $\mathcal{P}$ and $\mathcal{Q}$:

$$\mathcal{L}_{\text{chamfer}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p}_i \in \mathcal{P}} \min_{\mathbf{q}_j \in \mathcal{Q}} \|\mathbf{p}_i - \mathbf{q}_j\|_2^2 + \frac{1}{|\mathcal{Q}|} \sum_{\mathbf{q}_j \in \mathcal{Q}} \min_{\mathbf{p}_i \in \mathcal{P}} \|\mathbf{q}_j - \mathbf{p}_i\|_2^2. \tag{S7}$$

In implementation, we parameterize $\mathbf{R}$ using a continuous 6D rotation representation [53] and initialize $\mathbf{t}$ heuristically

from the 2D mask bounding box and the object's 3D extent, which stabilizes optimization under large rotations. For the first frame of each sequence, we perform 200 random restarts of $\mathbf{R}$ and retain the solution with the lowest Chamfer loss. For each subsequent frame, we use the optimized pose from the previous frame as initialization, allowing the optimizer to refine the pose smoothly over time. After convergence, we apply the estimated $(\mathbf{R}, \mathbf{t})$ to the predefined semantic keypoints and project them into the image, producing temporally consistent 2D object keypoints on BEHAVE RGB videos. The Adam optimizer [10] is adopted in this optimization process.

Processing Captured Real-World Videos. For the videos we captured, we manually select a small set of visible object keypoints in the first RGB frame and track them across the sequence using DELTA [28]. The reconstructed human-object interactions and tracked object keypoints are visualized on our project webpage.