Egocentric Visibility-Aware Human Pose Estimation

Supplementary Material

7. Dataset Acquisition

Capture Setup. Our capture rig consists of a Pico4 Ultra VR-MR HMD device, 16 industrial cameras with image resolution of 2488×2048 and 10 OptiTrack cameras. All recordings are conducted within a cylindrical capture volume of 4m in diameter. We temporally synchronize the egocentric cameras and multi-view recordings via the OptiTrack system, and spatially align them into a unified coordinate frame. Ground-truth poses are then obtained by multi-view triangulation followed by SMPL fitting.

Temporal Synchronization. The industrial cameras and OptiTrack system are hardware-synchronized using a dedicated sync hub device, ensuring that all external cameras and motion capture hardware share a common trigger signal. Meanwhile, the Pico4 Ultra VR-MR HMD is synchronized offline. Specifically, an optical rigid body (ORBs) marker is mounted to the VR device, and we align its motion with the OptiTrack data using a motion-correlation algorithm [34]. Finally, both egocentric camera frames and external recordings are mapped into the same timeline via OptiTrack. Since the egocentric images are captured at 30Hz, a maximum temporal misalignment of approximately 16.5ms may persist. To mitigate this problem, the final SMPL parameters are post-processed using interpolation to better align with the egocentric images.

Spatial Alignment. We attach optical rigid body markers to the HMD device in order to capture HMD's 6-DoF pose $\boldsymbol{T}_{head}^{opti}$ in the OptiTrack coordinate system. Formally,

$$\boldsymbol{T}_{head}^{opti} = \boldsymbol{T}_{orb}^{opti} \cdot \boldsymbol{T}_{head}^{orb},$$

where $\boldsymbol{T}_{orb}^{opti}$ is the pose of the ORB in OptiTrack, and $\boldsymbol{T}_{head}^{orb}$ is the rigid transform from HMD to ORB which can be pre-calibrated. We apply the same method to compute the pose of each industrial camera. Thus we obtain all necessary transform matrices among the HMD, cameras, and the OptiTrack coordinate systems.

3D Keypoints Estimation. We first employ the HRNet [37] to extract 2D keypoints in the Body25 [8] format for all industrial camera views. Then, the multi-view 2D detections are used to generate 3D keypoints via an iterative triangulation procedure. Note that the low-confidence detections and outliers with large re-projection errors are filtered out in the iterative triangulation process. To further improve temporal smoothness and physical plausibility, the resulting 3D keypoints $\boldsymbol{p}^{tri} \in \mathbb{R}^{75}$ are optimized with a smoothness regularization and a bone-length constraint, yielding stable and anatomically plausible 3D trajectories.

SMPL Fitting. We perform multi-stage optimization to fit SMPL parameters $\theta \in \mathbb{R}^{75}$ and $\beta \in \mathbb{R}^{10}$ to the triangulated keypoints $\boldsymbol{p}^{tri}$. In the first stage, we optimize the body shape parameters $\beta \in \mathbb{R}^{10}$ by minimizing:

$$\mathcal{L}_{shape}(\beta) = \sum_{(i,j)\in\mathcal{B}} \| \|\boldsymbol{v}_i - \boldsymbol{v}_j\| - l_{ij}\|^2 + \|\beta\|^2$$

where $\mathcal{B}$ is a set of bone edges, $\boldsymbol{v}_i, \boldsymbol{v}_j$ are the SMPL joint positions in rest pose, and l_{ij} is the target bone length from the triangulated keypoints. The second term serves as a regularization to penalize large shape values.

Next, we optimize the global rotation R and translation T, and finally the local pose parameters by minimizing the following loss function:

$$\begin{aligned}\mathcal{L}_{pose}(\beta,\theta) = \lambda_{fk}\mathcal{L}_{fk} + \lambda_{smooth}\mathcal{L}_{smooth} \\ + \lambda_{reg}\mathcal{L}_{reg} + \lambda_{init}\mathcal{L}_{init}\end{aligned} \tag{9}$$

where λ_{fk}, λ_{smooth}, λ_{reg}, and λ_{init} are the weights for the respective loss terms. $\mathcal{L}_{fk}$ calculates the mean square errors between the triangulated keypoints and the SMPL forward-kinematics (FK) keypoints. Note that we use the regression matrix of the Body25 format in this FK process to adapt to the triangulated keypoints $\boldsymbol{p}^{tri}$. $\mathcal{L}_{smooth}$ is a temporal smoothness term, $\mathcal{L}_{reg}$ is a regularization term which penalizes large pose parameters. Since optimization is performed in batches of 10 frames, the first frame of each batch is initialized from the last frame of the previous one, and an additional loss $\mathcal{L}_{init}$ is applied to prevent the solution from drifting too far. In Fig. 4, we visualize some representative fitted SMPL meshes that are re-projected onto external industrial cameras for better illustration.

After obtaining SMPL parameters $\theta \in \mathbb{R}^{75}$ and $\beta \in \mathbb{R}^{10}$ as described above, we can generate the 3D keypoints in the SMPL format via a standard FK process. And these 3D keypoints are used for model training and evaluation.

Visibility Labeling. Recall that we have collected 3.0M frames of stereo egocentric image pairs in our Eva-3M dataset. To improve the keypoint visibility labeling efficiency, we discard the first 600 calibration frames of each sequence and annotate keypoint visibility every 10 frames thereafter. Specifically, we project SMPL keypoints onto the stereo egocentric images and require professional annotators to manually annotate the visibility of each keypoint on each image. In total, we annotate 435K frames with keypoint visibility labels. In Fig. 5, we presents a few representative samples of the keypoint visibility labels in our Eva-3M dataset. It demonstrates that the keypoint invisibility issue is evident in the egocentric settings due to self-occlusions and out-of-FoV.

Figure 4. Visualization of some representative fitted SMPL meshes.

Dataset	Methods	Backbone	MPJPE↓	PA-MPJPE↓	Jitter↓
EMHI: P1	UnrealEgo [2]	ResNet18	289.7	171.0	60.4
	EgoPoseFormer [48]	ResNet18	180.5	112.5	70.3
	FRAME [7]	ResNet50	135.6	126.7	12.9
	EvaPose-ResNet50 (Ours)	ResNet50	93.3	59.5	**5.0**
	EvaPose-ViT-L (Ours)	ViT-Large	**49.5**	**38.4**	7.6
EMHI: P2	UnrealEgo [2]	ResNet18	244.9	148.5	40.4
	EgoPoseFormer [48]	ResNet18	160.7	107.3	60.2
	FRAME [7]	ResNet50	112.7	106.9	9.8
	EvaPose-ResNet50 (Ours)	ResNet50	103.6	57.4	**4.7**
	EvaPose-ViT-L (Ours)	ViT-Large	**45.9**	**36.0**	6.6

Table 6. Cross-dataset evaluation results. In this experiment, all methods are trained on the Eva-3M dataset and tested on the EMHI dataset.

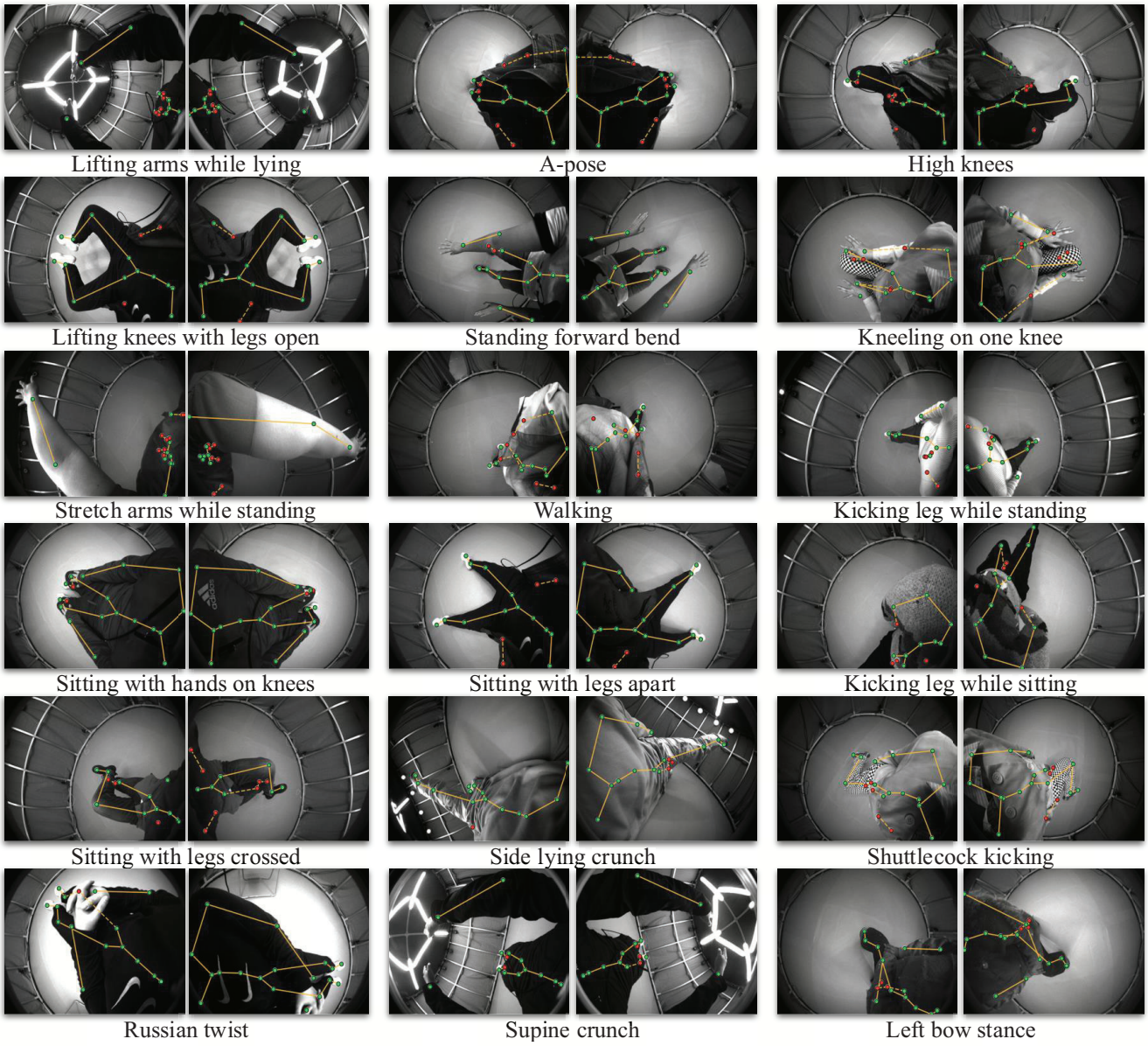

Figure 5. Representative samples of the keypoint visibility labels in our Eva-3M dataset.

8. Implementation Details

In this section, we provide additional implementation details of our EvaPose model.

8.1. Visibility-Aware 3D Pose Estimation

Image Encoder. Recall that our EvaPose uses two image backbones to extract visual features: ResNet50 [20] and ViT-Large [13]. For the ResNet50 image encoder, following previous works [7], we use the weights pretrained on ImageNet [12] dataset. For the ViT-Large image encoder, we use the pretrained ViT-L/14 weights from DINOv2 [33] to benefit from the self-supervised pretraining on large quantities of data. To lower the computational load, the input images are resized to 448×336 for the ViT-Large backbone.

Heatmap Estimation. Motivated by ViTPose [47], we use a lightweight decoder to process the features extracted from the backbone network and localize the keypoints. It consists of three stacks of deconvolution blocks and one 2D convolution layer. Each deconvolution block contains a 2D transposed convolution layer which upsamples the image feature maps by 2 times, followed by batch normalization [23] and ReLU [1] activation. The convolution layer with the kernel size 1×1 is utilized to get the heatmaps for the keypoints.

Visibility Prediction. For simplicity, we use a convolution

layer and a MLP network to capture the visibility status of individual keypoints. First, the image feature maps are fed into a 2D convolution layer, reducing its channel dimension to the number of keypoints. Then, the 2D image feature maps are flattened and passed through a three-layer MLP network to predict a visibility score for each keypoint.

Estimating 3D Poses in the Camera Coordinate System. Given the multi-view visibility-aware heatmaps as input, we first embed the heatmaps into tokens via a patch embedding layer. Then, a set of learnable positional encodings is added to the tokens, resulting in the transformer input. After that, the embedded tokens are processed by three transformer layers, each of which is consisted of a multi-head self-attention (MHSA) layer and a feed-forward network (FFN). Each transformer layer has a feedforward dimension of 512 and 8 attention heads. Finally, the output tokens corresponding to individual keypoint heatmaps are concatenated and projected through a three-layer MLP network to output the 3D positions in the camera coordinate system.

8.2. Iterative Intra-and Inter-Frame Attention

Stereo Transformer Decoder. The Stereo Transformer Decoder (STD) network is used for multi-view feature fusion within each frame. First, it applies two transformer decoders to let the query feature interact with the image features of each viewpoint independently. Each transformer decoder consists of 4 transformer decoder layers, each with a feedforward dimension of 512 and 8 attention heads. Then, the multi-view features are concatenated and fed into a one-layer MLP network for fusion.

Temporal Transformer Encoder. Having the estimated per-frame multi-view fused features, we use a Temporal Transformer Encoder (TTE) for temporal fusion of all information within a time window of $T = 24$. Specifically, TTE is structured with 8 transformer encoder layers, each having a feedforward dimension of 512 and 8 attention heads.

9. Coordinate Systems Transformation

As shown in Fig. 6, there are three coordinate systems used in this paper: the camera coordinate system which is defined as the left camera coordinate system, the world coordinate system, and the canonical coordinate system. In this section, we provide the relative transformations among these coordinate systems.

Camera-to-World Transformation. As described in the main paper, the SLAM system of the Pico4 Ultra VR-MR HMD device can provide the camera poses in the world coordinate system at each timestamp t, including camera rotation $\boldsymbol{R}^t_{world,cam}$ and position $\boldsymbol{T}^t_{world,cam}$ with millimeter-level accuracy. Formally, the camera-to-world transformation matrix can be computed as:

$$\boldsymbol{M}^t_{world,cam} = (\boldsymbol{R}^t_{world,cam}, \boldsymbol{T}^t_{world,cam}) \in \mathrm{SE}(3) \quad (10)$$

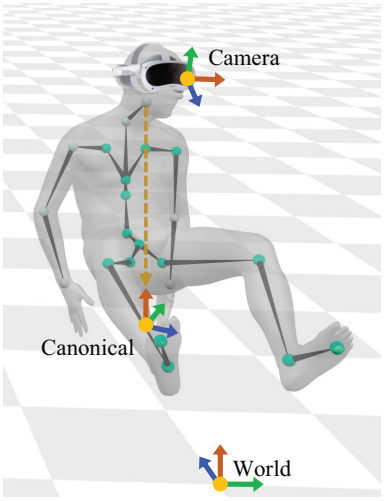

Figure 6. Overview of the coordinate systems.

Canonical-to-World Transformation. As illustrated in Fig. 6, the canonical coordinate system is obtained by projecting the head joint to the floor plane, aligning y axis to be vertical, and using the projection of the horizontal axis of head orientation on the floor plane as the horizontal axis. Formally, given the estimated head joint position in the world coordinate system $\boldsymbol{T}^t_{world,head}$, we can compute the canonical-to-world translation matrix:

$$\boldsymbol{T}^t_{world,can} = [\boldsymbol{e}_x, \vec{0}, \boldsymbol{e}_z]^T \boldsymbol{T}^t_{world,head} \quad (11)$$

For rotation matrix, we align the canonical frame's z-axis toward the "forward" direction of the head pose.

$$\vec{v}^t = \boldsymbol{R}^t_{world,head} \boldsymbol{e}_y \quad (12)$$

$$\boldsymbol{R}^t_{world,can} = \boldsymbol{R}_y(-arctan2(\boldsymbol{e}_x^T \vec{v}^t, \boldsymbol{e}_z^T \vec{v}^t)) \quad (13)$$

where $\boldsymbol{R}_y : \mathbb{R} \rightarrow \mathrm{SO}(3)$ constructs a y-axis rotation. Then, the canonical-to-world transformation matrix can be computed as:

$$\boldsymbol{M}^t_{world,can} = (\boldsymbol{R}^t_{world,can}, \boldsymbol{T}^t_{world,can}) \in \mathrm{SE}(3) \quad (14)$$

10. Cross-Dataset Performance Evaluation

In egocentric human pose estimation, cross-dataset generalization remains a problem. Given that EMHI and Eva-3M were captured using PICO4 and PICO4 Ultra HMD devices, respectively, these two datasets have similar camera settings. The Eva-3M dataset is collected in a controlled laboratory environment with green screen backgrounds. Differently, the EMHI dataset is captured in real indoor environments. To simulate real-world usage, we conduct this cross-generalization experiment using the Eva-3M dataset as the training set and the EMHI dataset as the test set. Tab. 6

shows the results. It can be concluded from these results that: (1) Using a larger backbone network can achieve better generalization capabilities. (2) Compared to FRAME [7], our EvaPose-ResNet50 uses the same ResNet50 backbone and achieves better results. This demonstrates that our approach has better generalization performance. (3) Our EvaPose-ViT-L significantly outperforms all other methods in both MPJPE and PA-MPJPE metrics. It shows that the ViT-Large backbone from DINOv2 [33], pretrained on large quantities of data, can improve the generalization performance of the model.