FRAME: Floor-aligned Representation for Avatar Motion from Egocentric Video

Supplementary Material

A. Model Details

A.1. Camera Model

In this project we use the camera model introduced by Kannala et Al. [3]. In the following paragraph, we summarize its main characteristics and describe how we employed it in this project.

For a normalized undistorted pixel (u, v), let θ represent the angle between the incoming ray and the optical axis, and r the radial distance in the image plane. By definition, in the pinhole model the following relation holds $r = \tan(\theta)$.

The distorted radial coordinate is given by:

$$r_d = \theta \cdot (1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8), \tag{1}$$

where k_1, k_2, k_3, k_4 are specific to each the lens.

The relationship between distorted and undistorted coordinates is given by

$$u = \left(\frac{r}{r_d}\right) u_d, \quad v = \left(\frac{r}{r_d}\right) v_d \tag{2}$$

where the $_d$ suffix stands for the distorted quantity.

A.1.1. Unprojection

We call unprojection the process of obtaining the 3D coordinates given a 2D pixel and its distance from the camera. To unproject a point, it is necessary to undistort it and subsequently obtain its 3D coordinates via the standard pinhole model. To undistort pixels we need to solve Equation 1 for θ. Since it contains higher-order terms in θ, direct inversion is intractable. Therefore, we follow OpenCV [1] in using a fixed point algorithm, starting with $\theta_0 = r_d$ and iterating $\theta_{n+1} = f(\theta_n)$. In practice, we empirically observe that 5 steps are enough to find an accurate solution.

Once θ converges, it's possible to compute $r = \tan(\theta)$ and obtain the undistorted pixel coordinates. Given the undistorted pixels and their associated depth, we can obtain their 3D coordinates directly by inverting the pinhole model equation.

$$x' = \frac{u - c_x}{f_x}, \quad y' = \frac{v - c_y}{f_y} \tag{3}$$

Where f is the focal length and c the coordinates of the optical axis intersection with the image plane The 3D coordinates (X, Y, Z) are then obtained by scaling the normalized coordinates by the corresponding depth value d at each pixel:

$$X = d \cdot x', \quad Y = d \cdot y', \quad Z = d \tag{4}$$

A.2. SoftArgmax

In order to obtain the u_k, v_k pixel coordinates of each joint k and their depth d_k we define the weight matrices $\mathbf{Q}^x, \mathbf{Q}^y$ as

$$Q^x_{ij} = \frac{j}{W-1} \quad Q^y_{ij} = \frac{i}{H-1} \tag{5}$$

This allows us to perform a *soft-argmax* operation [4] to obtain the normalized 2D coordinates in a differentiable way.

$$\begin{aligned} u_k &= \sum_{i,j} (\mathbf{Q}^x \otimes \hat{\mathbf{H}}_k)_{ij}, \\ v_k &= \sum_{i,j} (\mathbf{Q}^y \otimes \hat{\mathbf{H}}_k)_{ij}, \end{aligned} \tag{6}$$

Where $\otimes$ denotes the Hadamard product

B. Metrics

PA-MPJPE We adopt the definition established in prior work [2]. PA-MPJPE quantifies the similarity between predicted and ground-truth 3D poses by computing the mean per-joint Euclidean distance after Procrustes alignment. This alignment removes pelvis translation and applies a similarity transform (rotation, translation, and scale) to minimize the discrepancy between poses.

3D-PCK 3D-PCK quantifies the percentage of predicted 3D keypoints within 10 cm of the ground-truth. For each joint, the Euclidean distance is calculated, and predictions below the threshold are deemed correct.

Jitter Jitter quantifies temporal smoothness by comparing frame-to-frame changes in predicted and ground-truth 3D joint positions. Although there are multiple ways to quantify it, we follow Physcap [6] and compute:

$$\frac{1}{N \cdot J} \sum_{n=1}^{N} \sum_{j=1}^{J} \left| \|\mathbf{v}^j_{n,\text{pred}}\| - \|\mathbf{v}^j_{n,\text{gt}}\| \right|$$

where N is the number of sequences, J is the number of joints, $\mathbf{v}_{n,\text{pred}}$ and $\mathbf{v}_{n,\text{gt}}$ are the predicted and ground-truth joint velocity at frame n for joint j, respectively.

Non-penetration Percentage This metric measures the fraction of all poses where all predicted 3D joints remain above the ground plane $(y > 0)$

Mean Penetration Error (MPE) MPE measures the average penetration depth of joints below the ground plane $(y \leq 0)$.

Foot Sliding Velocity This metric evaluates foot velocity discrepancies between prediction and ground truth when feet are in contact with the ground. The sliding velocity error is computed as:

$$\frac{1}{N_c}\sum_{n=1}^{N_c}\sum_{j=1}^{4}\|(\mathbf{v}^j_{n,\text{pred}}-\mathbf{v}^j_{n,\text{gt}})\cdot\mathbf{1}_{\text{xz}}\| \quad \text{if joint } j \text{ is in contact}$$

where N_c is the number of contact frames, 4 is the number of feet joints, and $\mathbf{1}_{\text{xz}}$ is a vector that projects the velocity onto the xz-plane.

C. Data Collection

C.1. Recording Rig

The recording rig is built on the Meta Quest 3 [5], chosen for its lightweight design and comfort, allowing for prolonged use. The Quest 3 includes an RGB outward-facing camera with high-quality passthrough capabilities, enabling users to interact naturally with their surroundings. The headset computes on-device 6D head pose data, streamed continuously via HTTP from the Quest 3 to a Raspberry Pi worn by the user, with a custom-built Unity [7] application handling data transmission.

Egocentric Video Synchronization The fisheye cameras mounted on the VR headset record at 30Hz as the studio camera array, but their clocks are not hardware synchronized. To address this, we use a simple visual cue: toggling the studio lights on and off at the beginning and end of each session. This provides a temporal reference, allowing us to align the fisheye camera frames with the studio's frame of reference. Manual intra-frame adjustments are applied to account for any residual offsets or rolling shutter artifacts, ensuring tight alignment across all frames.

References

[1] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 1

[2] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[3] J. Kannala and S.S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1335–1340, 2006. 1

[4] Diogo C Luvizon, Hedi Tabia, and David Picard. Human pose regression by combining indirect part detection and contextual information. *Computers & Graphics*, 85:15–22, 2019. 1

[5] Meta. Meta quest 3, 2023. Accessed: 2024-10-09. 2

[6] Soshi Shimada, Vladislav Golyanik, Weipeng Xu, and Christian Theobalt. Physcap: physically plausible monocular 3d motion capture in real time. *ACM Trans. Graph.*, 39(6), 2020. 1

[7] Unity Technologies. *Unity: Real-time development platform*, 2024. Version 2024.1.0. 2