

Ego4o: Egocentric Human Motion Capture and Understanding from Multi-Modal Input

Supplementary Material

6. Implementation Details

In this section, we describe the implementation details of our method.

6.1. Part-Aware VQ-VAE

6.1.1. Network Structure

We first introduce the network structure of the encoder \mathcal{E} for each human body part. For each joint encoder, we utilize a codebook containing 4096 code vectors, each with a dimension of 64. The input human motion for a specific body part is $J_i \in \mathbb{R}^{T \times D}$, where T represents the motion length and D denotes the dimension of HumanML3D [16] representation. The input first traverses through a 1D convolutional layer (kernel size=3, stride=1, padding=1), followed by a ReLU activation function, producing a feature with 512 channels.

The motion feature then passes through two down-sampling blocks. Each down-sampling block comprises a 1D convolutional layer (kernel size=4, stride=2, padding=1) and three Resnet blocks. A Resnet block consists of a sequential structure: a convolutional layer, followed by a ReLU activation function, and another convolutional layer. The output from this sequence is combined with the input through addition to form the Resnet block's output.

A final 1D convolutional layer (kernel size=3, stride=1, padding=1) is applied to generate the feature $Q_i \in \mathbb{R}^{T' \times d}$, where d equals 64 (matching the code dimension) and $T' = T/4$. Before quantization, the encoded feature undergoes normalization. The full-body latent code \hat{Q}_i is constructed by combining the quantized codes from all six joint encoders.

The decoder mirrors the encoder's architecture, with one key modification: convolutional layers having stride=2 are replaced with upsampling layers using nearest neighbor interpolation. This process finally yields the reconstructed human motion \hat{J}_{recon} .

6.1.2. Training Details

For training the part-aware VQ-VAE, we use standard loss terms including quantization, commitment, and reconstruction losses.

$$\mathcal{L} = \sum_i \left(\beta \|\text{sg}[\hat{Q}_i] - Q_i\|_2 + \|\hat{Q}_i - \text{sg}(Q_i)\|_2 \right) + \|J - \hat{J}_{\text{recon}}\|_2 \quad (4)$$

where β is a balancing term, $\text{sg}[\cdot]$ denotes the stop-gradient operator. During training, we employ the Adam optimizer [32] with a batch size of 128 and a learning rate of 1×10^{-4} .

6.2. Multi-Modal Encoder

6.2.1. Network Structure

In implementing the masked trajectory transformer, we utilize a pre-trained CLIP-ViT-B/32 [53] model to extract features from the egocentric image and motion description. The IMU signals (A, R) are grouped into single feature tokens, with each token spanning 4 time steps. We do this grouping since it aligns with our downsampling rate of 4 in the part-based VQ-VAE framework. Each token transforms into a 512-dimensional feature through a linear projection layer.

The image features, motion description features, and IMU tokens are then concatenated and processed through a 4-layer transformer encoder to obtain the latent space representation. In each transformer encoder layer, the attention head number is 4, the dimension of the feed-forward network is 2048, and the dropout rate is 0.1. Subsequently, a 3-layer transformer encoder transforms this latent space into a sequence of logits. In each transformer encoder layer, the attention head number is 4, the dimension of the feed-forward layer is 1024 and the dropout rate is 0.1. We employ GumbelSoftmax [24] to convert these logits into motion code indices δ_i for each possible IMU location i . The final motion features \hat{Q}_i are obtained by selecting from the corresponding VQ-VAE codebook C_i .

6.2.2. Training Details

For training the multi-modal encoder, we optimize the encoder network while keeping the VQ-VAE decoder and CLIP [53] model frozen. The network is trained for 25 epochs using the Adam optimizer [32] with a learning rate of 1×10^{-4} and a batch size of 128. During training, the weighting parameter λ in Eq. (1) is set to 0.001.

6.2.3. Optimization Details

In the energy function Eq. (2) in Sec. 3.2.2, we set the weights $\lambda_a = 0.01$, and $\lambda_r = 1$, respectively. We use smaller weights for the IMU accelerations since they are noisy. During the run-time optimization stage, we first freeze the VQ-VAE decoder \mathcal{D} and then optimize the VQ-VAE latent vector Q by employing the L-BFGS [38] method with a learning rate of 1 and a convergence tolerance of 1×10^{-6} . The optimization process runs for a max-

Setups	IMUPoser [46]		Ego4o-IMU		Ego4o	
	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE
H	90.34	152.5	85.45	123.8	72.60	99.68
LP	73.65	103.6	73.73	98.78	65.06	86.94
LP+H	69.05	97.02	71.44	93.26	65.00	83.84
LP+RP	66.08	94.27	68.62	92.27	65.90	84.56
LW	84.80	126.9	82.59	118.5	67.05	92.75
LW+H	79.27	138.3	76.20	107.4	64.80	88.19
LW+LP	66.05	95.31	67.41	94.05	62.18	84.73
LW+LP+H	63.12	92.06	62.16	83.09	59.66	82.25
LW+LP+RP	65.63	100.2	63.73	86.36	57.48	77.36
LW+RP	73.21	100.1	65.07	86.71	60.12	81.98
LW+RP+H	67.66	100.1	58.49	77.98	58.95	79.17
LW+RW	71.74	117.1	73.02	110.4	64.38	90.74
LW+RW+H	68.32	105.0	68.42	101.9	59.78	82.73
LW+RW+LP	67.93	99.80	59.01	82.62	53.25	74.61
LW+RW+RP	79.26	121.6	56.96	80.71	54.10	77.05
RP	77.55	97.16	74.18	97.54	69.37	91.18
RP+H	76.66	104.6	69.48	92.30	66.08	88.33
RW	76.01	117.3	75.01	110.2	69.21	97.98
RW+H	79.06	115.7	73.65	106.8	64.69	91.40
RW+LP	67.06	95.73	63.34	90.16	61.73	84.58
RW+LP+H	65.12	98.58	64.24	88.39	58.81	78.35
RW+LP+RP	70.46	105.9	65.51	88.67	57.89	78.87
RW+RP	65.37	93.78	64.18	88.72	60.28	82.38
RW+RP+H	67.24	100.1	65.19	88.46	58.89	80.82

Table 5. Result of the IMU-based human motion capture on the Nymeria Dataset under different IMU setups. H, LP, RP, LW, and RW indicate the IMU located on different body parts. H: head, LP: left hip, RP: right hip, LW: left wrist, RW: right wrist. The results are shown in millimeters.

imum of 1,000 iterations, maintains a history size of 200, and utilizes the strong Wolfe [48] conditions for line search.

6.3. Training Details of Multi-Modal LLM for Motion Understanding

During the pre-training phase, we train only the motion embedding layer E_M while keeping all other modules frozen. The embedding layer is trained for 1 epoch using the Adam optimizer with a learning rate of 1×10^{-3} and a batch size of 16. In the multi-modal fine-tuning phase, we keep the CLIP model and multi-modal encoder frozen while fine-tuning both the image and motion embedding layers along with the large language model. We employ LoRA [22] with a rank of 128 and an alpha value of 256. The language model is fine-tuned for 4 epochs using the Adam optimizer with a learning rate of 2×10^{-5} and a batch size of 16.

7. Results on Different IMU Setups

In this section, we present results for human motion capture across different IMU configurations in Table 5. The intuitive results can be seen in Figure 5.

From the results, we observe that human motion capture accuracy decreases when fewer IMU inputs are used. This is expected, as a lower number of IMUs provide less information, leading to greater ambiguity in the motion capture process.

Additionally, we find that using lower-body IMUs generally leads to higher motion capture accuracy compared to upper-body IMUs, a trend that is particularly evident in the Ego4o-IMU results. A possible explanation is that lower-body movements provide essential kinematic constraints for motion analysis, such as differentiating between standing and sitting postures—something a wrist-mounted IMU, for instance, cannot reliably capture. However, this pattern is less observed in the Ego4o method, likely because the text-based motion descriptions and egocentric image inputs provide contextual motion cues, enabling the model to infer postural states (e.g., standing or sitting) more effectively, thereby reducing reliance on IMU data alone.

Setups	w/o optim		only gt text		only image		w/ gen text		image & gen text	
	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE
H	99.69	72.60	95.09	70.52	106.62	78.69	104.92	75.24	96.53	71.56
LP	86.02	65.11	94.56	70.67	92.82	73.83	91.60	69.34	89.30	66.33
LP+H	85.92	65.10	84.66	67.52	87.47	67.50	85.93	65.49	86.41	66.64
LP+RP	85.59	65.95	87.82	66.25	88.50	66.87	87.41	66.43	92.53	67.57
LW	93.76	67.06	98.22	69.93	105.46	72.68	99.22	69.62	102.66	72.69
LW+H	89.24	66.81	84.50	61.30	96.04	67.86	96.04	69.23	90.30	64.79
LW+LP	85.73	64.21	86.83	65.24	88.02	63.25	83.79	61.25	82.68	61.52
LW+LP+H	83.25	61.76	78.71	56.10	83.20	61.32	81.27	61.88	84.40	62.98
LW+LP+RP	78.45	59.53	76.78	57.25	80.98	61.06	81.55	62.33	79.40	58.54
LW+RP	83.03	62.19	82.08	61.53	88.56	65.89	89.21	69.14	84.13	63.20
LW+RP+H	80.18	61.04	79.53	58.29	86.23	63.73	78.52	58.45	80.57	60.19
LW+RW	91.78	66.39	86.97	60.79	100.60	70.83	97.83	66.98	95.32	66.37
LW+RW+H	83.75	61.86	85.38	59.33	88.71	62.52	85.05	59.64	87.50	62.34
LW+RW+LP	75.67	55.33	86.49	61.75	83.67	59.28	81.56	59.34	79.21	57.23
LW+RW+RP	78.08	56.14	82.35	58.28	83.31	56.80	82.68	56.54	79.83	57.62
RP	92.25	71.45	95.32	73.19	91.01	68.26	87.94	66.35	88.91	69.01
RP+H	89.33	68.12	84.98	64.51	84.52	64.21	90.82	69.30	83.56	65.76
RW	99.01	71.26	97.44	69.03	112.56	74.99	103.29	72.26	104.32	71.15
RW+H	92.42	66.75	95.17	66.01	99.42	69.64	90.04	64.11	94.27	66.67
RW+LP	85.64	63.77	80.35	59.87	91.61	65.31	90.86	66.36	81.71	60.36
RW+LP+H	79.42	60.85	79.57	57.55	85.03	64.21	74.61	56.42	83.25	59.26
RW+LP+RP	79.90	59.93	77.60	57.26	83.17	59.37	84.70	60.58	78.54	56.40
RW+RP	83.42	62.29	86.20	62.59	86.47	63.91	88.75	64.07	82.57	61.11
RW+RP+H	81.84	60.93	82.52	60.57	86.44	62.98	84.56	60.68	80.13	58.59

Table 6. Ablation study of the IMU-based human motion capture on the Nymeria Dataset under different IMU setups. H, LP, RP, LW, and RW indicate the IMU located on different body parts. H: head, LP: left hip, RP: right hip, LW: left wrist, RW: right wrist. The results are shown in millimeters.

8. Ablation Study on Different IMU Setups

In this section, we present ablation study results for human motion capture across different IMU configurations in Table 6.

9. Evaluation Metrics

We evaluate our method using three standard metrics for human motion capture accuracy: Mean Per Joint Position Error (MPJPE), Procrustes-aligned Mean Per Joint Position Error (PA-MPJPE) and Jitter. MPJPE measures the average Euclidean distance between predicted and ground truth joint positions. To compute PA-MPJPE, we first perform rigid alignment of the predicted pose to the ground truth using Procrustes analysis [31], then calculate the MPJPE. The Procrustes alignment helps evaluate pose accuracy independent of global position and orientation. Jitter [13] quantifies motion smoothness by measuring the mean jerk (third-time derivative of position) of all body joints in global space, expressed in km/s^3 .

We evaluate our method with three metrics for motion description accuracy: BERT score [82], BLEU [49],

and ROUGE-L [37]. BLEU measures the precision of n-gram matches between generated and reference texts, indicating how well the generated descriptions align with ground truth at the phrase level. BERT score leverages pre-trained BERT embeddings to compute semantic similarity between generated and reference descriptions, providing a more contextually-aware evaluation than traditional n-gram based metrics. ROUGE-L computes the longest common subsequence between generated and reference descriptions, capturing the fluency and sequential consistency of the generated text.

In our experiments, we employ the Python “evaluate” package from the Huggingface to compute BERT, BLEU, and ROUGE-L scores. For the BERT score calculation, we enable IDF weighting and rescale with baseline, setting both parameters to “True”.

10. w/o Part-Aware VQ-VAE

In this section, we evaluate the effectiveness of our part-aware VQ-VAE by comparing its reconstruction accuracy with that of the traditional VQ-VAE.

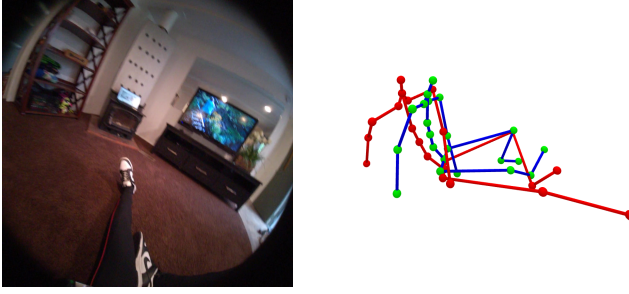


Figure 7. Failure Case. Left: input image; Right: output human body pose. Red skeleton is the ground truth pose.

Our part-aware VQ-VAE achieves a Mean Per Joint Position Error (MPJPE) of 44.93 mm and a Procrustes-aligned MPJPE (PA-MPJPE) of 32.72 mm. In contrast, the traditional VQ-VAE yields higher errors with an MPJPE of 47.73 mm and a PA-MPJPE of 36.71 mm. These results demonstrate that our part-aware approach reduces the reconstruction error, indicating superior performance in preserving motion details and overall pose structure.

11. Comparison with HMD-Poser

A direct comparison between our Ego4o method and HMD-Poser [7] would be unfair, as Ego4o supports an arbitrary number of IMUs and multi-modal inputs (e.g., text and images), whereas HMD-Poser relies on fixed 6DoF head and hand tracking data and cannot handle multi-modal inputs. Nonetheless, we retrained HMD-Poser and evaluated it on the DIP-IMU dataset under our experimental setup. The results—87.6 mm MPJPE, 66.9 mm PA-MPJPE, and 0.12 km/s^3 jitter—are inferior to ours.

12. Failure Case

Since IMUs track acceleration rather than position, our method may fail when the body remains still and the image lacks contextual information. This is shown in Figure 7, where the upper body and feet predictions are incorrect.

13. Efficiency and Resource Utilization

Our framework demonstrates real-time performance and low memory consumption across tasks:

- **Motion Capture Model**
 - Single image: **8.2 ms/frame** inference speed, **0.90 G** GPU memory
 - 10 images: **8.6 ms/frame** (+4.9% latency), **0.92 G** memory (+2.2% usage)
- **Motion Description Generation Model**
 - Single image: **37.3 ms/token** inference speed, **16.05 G** GPU memory

- 10 images: **38.6 ms/token** (+3.5% latency), **16.07 G** memory (+0.1% usage)

Both components show stable computational costs under increased input scales (1→10 images), demonstrating minimal computational overhead when scaling to multi-image inputs.