MoBind: Motion Binding for Fine-Grained IMU–Video Pose Alignment

Supplementary Material

In this supplementary material, we provide additional implementation details, extended experiments, ablation studies, and qualitative results. Sec. 6 details the datasets, MoBind, and baseline implementations; Sec. 7 presents additional experiments; Sec. 8 reports further ablation studies; and Sec. 9 provides qualitative visualizations. All code and benchmarking scripts will be released upon acceptance.

6. Implementation Details

In this section, we provide details on the datasets used in the paper, the preprocessing steps, and the architecture and training setups for MoBind and all baselines.

6.1. Datasets

mRi. The mRi dataset is a multimodal collection of synchronized RGB-D video and IMU recordings at 10 fps captured from 20 subjects performing repetitive rehabilitation exercises, totaling roughly 4.5 hours of data. Although the original setup includes six IMUs placed on the left wrist, right wrist, left knee, right knee, head, and torso, the accompanying documentation does not explicitly map each raw IMU stream to its corresponding body part, and we found it non-trivial to infer this mapping reliably from the data alone. To avoid ambiguous supervision, we discard the head- and torso-mounted sensors and retain only the four limb-mounted IMUs. We use a subject-wise split throughout the experiments, assigning subjects 16 and 18 to validation, subjects 15, 17, 19, and 20 to testing, and the remaining subjects to training.

TotalCapture. Compared to mRi's repetitive rehabilitation activities, where subjects remain mostly stationary, the TotalCapture dataset presents a more challenging setting with a top-down camera view and subjects freely moving through the scene while performing highly dynamic actions. TotalCapture consists of 5 subjects performing 4 distinct activities, recorded at 60 fps for both cameras and IMUs, with a total duration of one hour. A total of 13 wearable IMU sensors are available; we select 6 representative sensors located on the left wrist, right wrist, left knee, right knee, head, and torso. Similar to mRi, we adopt a subject-wise split: subject 4 is used for validation, subject 5 for testing, and the remaining subjects for training.

EgoHumans. The EgoHumans dataset introduces a more complex setting than mRi and TotalCapture, with multi-person scenes in diverse indoor and outdoor environments where subjects interact with one another. Although EgoHumans does not originally include IMU data, we follow common practice of generating synthetic IMU signals from the SMPL 3D body model [9, 12, 18, 22, 49, 50, 54], placing six virtual sensors at the same locations as in TotalCapture (left wrist, right wrist, left knee, right knee, head, and torso). Both video and synthetic IMU streams are recorded at 20 fps. The dataset features approximately 2–4 people per scene and covers 7 activity classes, yielding about 2 hours of synchronized multimodal data. For EgoHumans, we adopt a non-overlapping sequence split: for each activity, the first four sequences are used for testing, the fifth for validation, and the remaining sequences for training.

	mRi	TotalCapture	EgoHumans
Training			
Train	4719	1082	3710
Val	678	134	228
Retrieval, Localization, Recognition			
Test	1309	150	956
Synchronization			
Test	1539	1539	1539

Table 7. **Number of sequences in the train, validation, and test splits** used throughout our experiments, each sequence is a 5s synchronized IMU–video pair.

Sensor	Keypoints index
Left Wrist	5, 7, 9
Right Wrist	6, 8, 10
Left Knee	11, 13, 15
Right Knee	12, 14, 16
Head	0, 1, 2

Table 8. **Locations of the worn IMU sensors on the body and their corresponding body-part keypoints.**

6.2. Data Preprocessing

We extract all 2D keypoint sequences using mmpose [7], following the COCO [29] format with 17 joints. The IMU locations and their corresponding body-part joint indices are defined in Table 8. Before partitioning the skeleton into body parts, we normalize all joint coordinates to $[0, 1]$ with respect to the image size. For missing, occluded, or out-of-camera joints, we set the corresponding body-part joints to zero. Since the 2D keypoints are detected from video, their frame rates follow the original video FPS: 10 for mRi, 60 for TotalCapture, and 20 for EgoHumans. For the IMU signals, we use 7 input channels, comprising 3D acceleration (in units of g) and rotation represented as quaternions.

To create synchronized pairs for training MoBind, we partition each original video into 5 s windows with a stride of 2 s. The synchronized IMU–pose window serves as the positive example, and all other windows act as negatives. These windows are used for training as well as for retrieval, localization, and action recognition experiments. For synchronization, we construct a separate test set: from the test split, we partition video into 20 s segments and simulate misalignment by introducing random lags uniformly sampled from $[-7, 7]$ s between the modalities. The number of sequences from all three datasets used in our experiments is summarized in Table 7.

6.3. MoBind

Here, we describe the detailed architecture of MoBind. Both the IMU encoder $\mathcal{E}_{\text{imu}}$ and the part encoder $\mathcal{E}_{\text{part}}$ share the same backbone and differ only in the input frames and channels. Given an input with F frames (with F depending on the modality frame rate) and C channels (C=7 for IMU and C=6 for parts, corresponding to three 2D joints), we first apply four 1D convolutional blocks with kernel size 5 and padding $5/2$. Each block consists of a 1D convolution, batch normalization, and ReLU activation, with feature dimensions $[32, 64, 128, 128]$, and we apply dropout with probability 0.3 after the second and fourth blocks. The resulting feature map of size $F \times 128$ is then partitioned along the temporal dimension into T non-overlapping patches (with T=25 in our setup). Each patch is flattened and linearly projected to a D=256-dimensional embedding and fed to a Transformer encoder with 4 layers and 8 heads. The Transformer outputs a sequence of temporal tokens $\mathbf{Z} \in \mathbb{R}^{T \times D}$, from which we obtain a local-level representation $\overline{\mathbf{Z}}$ by average pooling over time.

The local-level representations from all N sensors are concatenated and passed through an aggregation block consisting of layer normalization followed by a linear layer that maps the resulting $N \times D$ features to a global embedding $\mathbf{G}$ of dimension $D' = 256$. Before applying the contrastive objectives, both local- and global-level features are further processed by modality-specific projection heads to map them into a shared latent space.

We further introduce a masked token prediction (MTP) auxiliary task to encourage the IMU encoder to retain class-semantic information instead of over-focusing on features purely beneficial for fine-grained alignment. MTP operates on the temporal tokens from all N sensors: we randomly mask a fraction $\alpha = 0.75$ of the tokens and replace them with a learnable query vector, yielding $\mathbf{Z}^{\text{mask}}$. We then add learnable positional and sensor-identity embeddings and feed the sequence into a Transformer with 4 layers and 8 heads. The output tokens are normalized and passed through a prediction head that reconstructs the masked tokens from the unmasked context. This module is optimized jointly with the alignment objectives and discarded at inference time.

```
# P[B, F_p, 2J]    - minibatch of pose sequences
# I[B, S, F_i, C] - minibatch of IMU sequences
# t               - learned log-temperature

# extract feature representations
G_p = pose_encoder(P)
G_p = l2_normalize(G_p, axis=-1)      # [B, D]
G_i = imu_encoder(I)
G_i = l2_normalize(G_i, axis=-1)      # [B, S, D]

# Pose -> IMU (multi-positive)
G_i_flat   = G_i.reshape(B * S, D)
logits_p2i = exp(t) * dot(G_p, G_i_flat.T)

blocks     = logits_p2i.reshape(B, B, S)
pos_logits = blocks[arange(B), arange(B), :]

num   = logsumexp(pos_logits, axis=-1) - log(S)
denom = logsumexp(logits_p2i, axis=-1)
loss_p2i = -mean(num - denom)

# IMU -> Pose (single-positive)
logits_i2p = exp(t) * dot(G_i_flat, G_p.T)
labels     = repeat(arange(B), repeats=S)
loss_i2p   = CE_loss(logits_i2p, labels)

# symmetric loss
loss = (loss_p2i + loss_i2p)/2
```

Figure 8. **Pseudocode for the multi-positive contrastive loss for baselines**, follow CLIP style [36].

6.4. Baselines

6.4.1. Contrastive Baselines

IMU2CLIP was originally proposed to align IMU signals with egocentric video in the Ego4D dataset by projecting both modalities into a shared latent space through the CLIP text encoder [36]. However, when moving to third-person video datasets such as mRi, TotalCapture, and EgoHumans, we found that directly aligning IMU with RGB video is challenging under limited data (e.g., roughly 1k training pairs in TotalCapture versus 528k in Ego4D), making it difficult to learn meaningful representation for alignment. For fairness and consistency with MoBind, we therefore replace RGB video with 2D pose sequences. Concretely, we reuse the MoBind pose encoder backbone but, instead of encoding body parts, we flatten the full 2D pose at each frame and feed it as an input of shape $F \times 34$ (17 joints with 2D coordinates). The IMU encoder and the overall training pipeline follow their original implementation. Similarly, for DeSPITE, we keep the original IMU encoder and training pipeline, but instead of aligning IMU with 3D skeletons as in the original work, we align it with flattened 2D keypoints using the same pose encoder design as in our IMU2CLIP adaptation.

SyncNet was first proposed for audio–video alignment

Method	mRi						TotalCapture						EgoHumans					
	IMU→Video			Video→IMU			IMU→Video			Video→IMU			IMU→Video			Video→IMU		
	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
IMU2CLIP	0.67	0.88	0.92	0.38	0.69	0.81	0.06	0.20	0.33	0.07	0.17	0.27	0.29	0.51	0.59	0.29	0.51	0.60
DeSPITE	0.57	0.85	0.91	0.32	0.70	0.82	0.03	0.15	0.24	0.03	0.11	0.25	0.54	0.73	0.80	0.54	0.74	0.80
SyncNet	0.77	0.94	0.97	0.75	0.92	0.95	0.51	0.87	0.94	0.54	0.86	0.93	0.74	0.90	0.93	0.71	0.89	0.93
MoBind	**0.94**	**0.99**	**1.00**	**0.92**	**0.99**	**0.99**	**0.87**	**0.96**	**0.98**	**0.68**	**0.91**	**0.97**	**0.83**	**0.93**	**0.96**	**0.83**	**0.93**	**0.95**

Table 9. **Cross-modal retrieval performance on thethree datasets using the concatenated-IMU variants of the baselines.** Overall, concatenating all IMU sensors improves retrieval performance, but—as shown later—produces representations that are weaker for temporal synchronization and person/body-part localization.

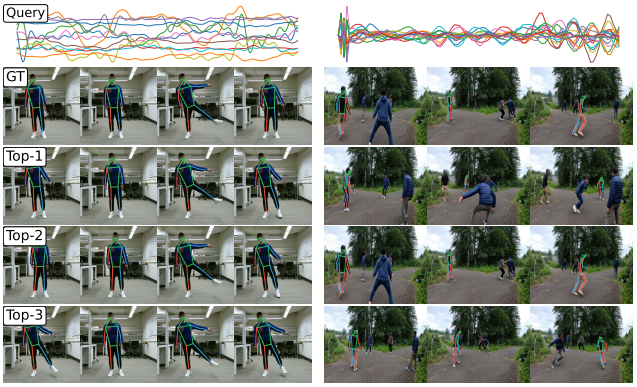

Figure 9. **Failure cases of IMU→Video retrieval on mRi and EgoHumans** where the ground-truth match does not appear at top-1 of MoBind prediction due to nearly identical, highly repetitive motions; the retrieved sequence is nevertheless visually almost indistinguishable from the ground truth.

using an identity loss and a content loss between the two modalities. Building on this idea, we adapt SyncNet to the IMU-pose setting by applying our encoder architecture to IMU signals and 2D pose sequences. To handle multi-sensor input, a straightforward approach would be to follow DeSPITE and concatenate IMU signals from all sensors; however, we find that this yields suboptimal representations for fine-grained alignment. Instead, we adopt a multi-positive contrastive scheme Fig. 8 in which each individual IMU stream is aligned with the full pose sequence, and the resulting per-sensor IMU embeddings are then averaged to form the final IMU representation.

6.4.2. IMUSync

Inspired by recent offset-classification methods [20, 21], we develop an offset-classification head that operates directly on the pose and IMU token sequences produced by the MoBind encoders. Given pose and IMU tokens $\mathbf{Z}^{\text{pose}}, \mathbf{Z}^{\text{imu}} \in \mathbb{R}^{T\times D}$, we first apply modality-specific linear projections followed by ℓ_2 normalization. We then enumerate a discrete set of candidate lags $\Delta\tau \in \{-7.0, -6.5, \ldots, 6.5, 7.0\}$ at 0.5 s resolution. For each lag, we temporally shift one sequence relative to the other and compute the elementwise product between the two token sequences; the product is then averaged over time and feature dimensions to obtain a single similarity score for that lag. Stacking the scores over all candidate lags yields a 29-dimensional logit vector, corresponding to the 29 discrete offsets in $[-7, 7]$ s, which is scaled by a learnable temperature and trained with a cross-entropy loss to predict the ground-truth offset.

In practice, we find it critical that both modalities share the same frame rate, so we resample the IMU streams to match the pose FPS before encoding. The model is trained for 500 epochs using the Adam optimizer with a learning rate of 3×10^{-4} and a batch size of 256. The training data are generated in the same manner as described in Sec. 6.2: from the training split, we partition videos into 20 s segments and simulate misalignment by sampling offsets uniformly in $[-7, 7]$ s at 0.5 s steps for 29 offset classes.

7. Additional Experiments

This section presents additional experiments on the concatenation variants of the contrastive baselines, along with further analyses of MoBind on downstream tasks.

7.1. Cross-Modal Retrieval

Table 9 reports retrieval performance of the contrastive baselines on mRi, TotalCapture, and EgoHumans when IMU signals from all sensors are concatenated into a single input, rather than aligning each sensor individually with the full pose sequence. Overall, this setting improves baseline performance on mRi and TotalCapture compared to Table 1, with particularly large gains for SyncNet, but degrades performance on EgoHumans. Moreover, while concatenation helps retrieval, it consistently hurts synchronization and localization, as we show later.

We illustrate typical failure cases of MoBind for the IMU→Video retrieval task on mRi and EgoHumans in Fig. 9. Although MoBind does not always retrieve the exact ground-truth moment in mRi, the top-ranked results are often nearly indistinguishable from it, even for human observers, due to many visually similar repetitive patterns in the video. A similar trend appears in EgoHumans: while the ground-truth clip is not always ranked first, it usually lies within the top-3 results, and the remaining candidates

```
# P[P, S, F_p, 2J] - Pose sequence for P persons
# I[S, F_i, C]     - IMU sequence
# k                - top-k for retrieval

# window the input streams
imu  = unfold(I, win_len, stride_len)
part = unfold(P, win_len, stride_len)

# extract feature representations
imu_feat, part_feat = MoBind(imu, part)
imu_feat  = L2_norm(imu_feat)       # (W,  D)
part_feat = L2_norm(part_feat)      # (PW, D)

# compute similarity
sim = imu_feat @ part_feat.T        # (W, PW)

# compute offsets via voting
offset_cum = []
score_cum  = []

val_i, idx_i = retrieve(sim,   k)   # (W,  k)
val_p, idx_p = retrieve(sim.T, k)   # (PW, k)
idx = vector[0 .. W-1]

offsets_i = -(idx_i%W - idx)
offsets_p =  (idx_p   - idx)
offset_cum.extend([offsets_i, offsets_p])
score_cum.extend([val_i, val_p])

offset_all = concat(offset_cum).ravel()
score_all  = concat(score_cum).ravel()
uniq_offsets, inv = np.unique(offset_all)
score_sums = np.bincount(inv, score_all)
best_offset_bin = uniq_offsets[argmax(score_sums)]

# convert bin to time offset
pred_offset = int(best_offset_bin * stride_len)
```

Figure 10. **Pseudocode for synchronizing an IMU stream with a multi-person video.**

are visually similar clips from the same sequence that are merely shifted in time.

7.2. Synchronization & Localization

Pseudo-code for temporal offset estimation is shown in Fig. 10, illustrating how we synchronize an IMU stream with a video that may contain multiple people. Table 10 reports temporal synchronization results for the concatenation variants of the baselines. Although concatenating IMU signals improves retrieval performance, it substantially degrades synchronization, increasing the MAE by nearly 2 s on mRi. This suggests that the resulting representations are less stable and less sensitive to fine-grained, repetitive motion. A similar trend appears for person localization: as shown in Table 11, the concatenated representations consistently reduce baseline performance on EgoHuman dataset compared to Table 3. In contrast, MoBind outperforms all baselines under both input configurations, highlighting the benefits of our hierarchical multi-sensor modeling, where each IMU is first aligned with its corresponding body part

Method	mRi		TotalCapture		EgoHumans	
	MAE ↓	Acc ↑	MAE ↓	Acc ↑	MAE ↓	Acc ↑
Random guess	10.72	0.00	10.06	0.00	10.18	0.01
IMU2CLIP	3.11	0.44	3.40	0.22	3.53	0.14
DeSPITE	3.60	0.39	4.40	0.06	3.24	0.28
SyncNet	3.60	0.26	1.34	0.49	4.01	0.19
MoBind	**0.47**	**0.88**	**0.05**	**0.98**	**0.04**	**1.00**

Table 10. **Synchronization results for the concatenated-IMU variants of the baselines.**. This configuration substantially degrades performance of the baselines, increasing MAE by nearly 2 s on the mRi dataset compared to Table 2, whereas MoBind's hierarchical three-level alignment consistently outperforms both baseline variants.

Methods	Acc
IMU2CLIP	0.89
DeSPITE	0.95
SyncNet	0.96
MoBind	**0.98**

Table 11. **The concatenated representation also degrades the localization performance of the baselines,** similar to synchronization,

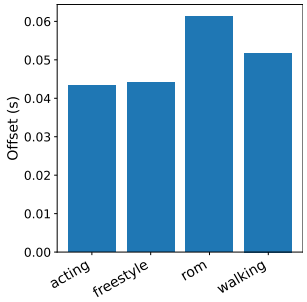

Figure 11. **Per-action synchronization performance on TotalCapture**, confirming MoBind's robustness under highly dynamic, repetitive actions.

before aggregating local features into a global representation.

In Table 11, we break down MoBind's synchronization performance on TotalCapture by action class. The synchronization error remains below 60 ms for all actions, with slightly higher values for rom and walking, which involve highly repetitive motion. Together with the findings in Table 4, this demonstrates the robustness of MoBind for fine-grained alignment across a wide range of action categories.

7.3. Human Action Recognition

The activity recognition results on EgoHumans are shown in Table 13. MoBind again achieves the best performance under both fine-tuning and 1-NN evaluation, indicating that its embedding space is well-suited for high-level semantic structure and offers strong class-level discriminability. These gains further highlight the contribution of the MTP auxiliary module. For consistency, fine-tuning results are reported at the 5th epoch.

Method	Performance		Params(M)	FLOPS(M)	Latency (ms)		
	I → V	V → I			t_{ex}	t_{sim}	Total
IMU2CLIP	0.30	0.24	2.27	38.42	1.10	0.031	1.13
DeSPITE	0.21	0.15	2.46	211.52	5.21	0.031	5.25
SyncNet	0.52	0.49	3.45	87.29	1.12	0.031	1.15
MoBind	**0.94**	**0.92**	4.76	239.09	1.16	0.031	1.19

Table 12. **Efficiency comparison between MoBind and contrastive baselines.** t_{ex} and t_{sim} denote the latencies of embedding extraction and similarity computation, respectively. Although MoBind has more parameters and FLOPs, its t_{ex} remains competitive with the lightweight baselines, making it suitable for real-world applications. t_{sim} is identical across all methods, as it corresponds to a dot product between IMU and pose feature matrices. All latencies are measured on a single RTX 5090 GPU.

Method	Finetune	1-NN
UniMTS	0.62	0.27
ImageBIND	0.68	0.40
EVI-MAE	0.37	0.33
Primus	0.43	0.57
IMU2CLIP	0.47	0.59
DeSPITE	0.48	0.53
SyncNet	0.53	0.62
MoBind	**0.78**	**0.67**

Table 13. **Human activity recognition on EgoHumans under fine-tuning and 1-NN evaluation.** MoBind outperforms all prior methods in both settings.

7.4. Computational Cost Analysis

Table 12 compares the number of parameters, floating-point operations (FLOPs), and latency of MoBind against the contrastive baselines. We report the average latency required to process a pair of IMU–video and compute similarity scores between their global embeddings. Although MoBind has more parameters and FLOPs than the other methods, it remains lightweight (around 4M parameters), making it practical for real-world deployment. Importantly, this modest increase in capacity translates into consistently superior performance not only for retrieval, but also for synchronization, localization, and activity recognition.

Despite the larger capacity, MoBind's feature-extraction latency is highly competitive with IMU2CLIP and SyncNet, requiring only 1.16 ms to process a 5 s input window—substantially faster than DeSPITE and well within real-time constraints. Note that t_{sim} is identical across all methods, as it corresponds to a dot product between two matrices. For synchronization, MoBind processes a 20 s IMU–pose window pair in 3.49 ms end-to-end (including windowing, feature extraction, similarity computation, histogram construction, and voting), and 10.2 ms for a 60 s window, which is comfortably real-time and suitable for practical applications. All latency measurements are obtained on a single RTX 5090 GPU, using 50 warm-up runs and averaging over 200 evaluation runs.

	Params(M)	FLOPS(M)
UniMTS	5.18	3555.00
ImageBIND	19.60	6355.00
EVI-MAE	21.63	1355.00
Primus	1.38	56.32
MoBind	2.51	153.24

Table 14. **Analysis of MoBind's IMU module efficiency compared to state-of-the-art IMU encoders.** Aside from Primus, MoBind's IMU module remains substantially smaller in both parameters and FLOPs, while achieving superior action recognition performance under both fine-tuning and 1-NN settings across all three datasets.

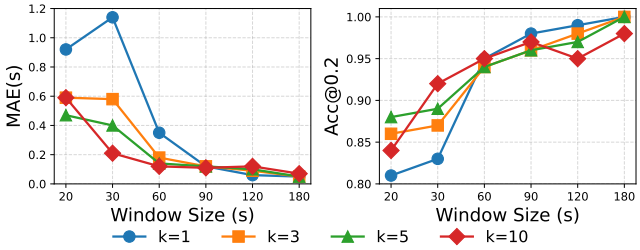

Figure 12. **Impact of top-k and input length on synchronization performance on the mRi.** Experiments show that $k = 5$ achieves the best trade-off between accuracy and computational cost.

We further assess the efficiency of MoBind by comparing its parameter count and FLOPs with several state-of-the-art IMU encoders, as shown in Table 14. MoBind again remains lightweight and computationally efficient compared to larger encoders. Note that all parameter and FLOP counts in the table are computed for the IMU encoder alone (for both baselines and ours), not for the full framework.

8. Additional Ablation Studies

This section presents additional ablation studies to quantify the impact of key hyperparameters and to evaluate the retrieval performance of local features from individual IMUs.

Mask ratio	Sync.	Action Recognition	
	MAE (s)	Finetune	1-NN
0.65	0.39	0.96	0.82
0.75	0.47	0.98	0.87
0.85	0.51	0.99	0.85

Table 15. **Effect of masking ratio on synchronization and action recognition performance.** A trade-off analysis across different masking ratios shows that $\alpha = 0.75$ offers the best balance between the two tasks.

8.1. Top-k and window size for synchronization

Recall that for synchronization, we first retrieve the top-k nearest neighbors and use their shifted index to build a weighted histogram. Fig. 12 evaluates the sensitivity of performance as k varies. Accuracy improves as k increases up to $k = 5$, after which gains become marginal and the results less stable, so we adopt $k = 5$ as the default trade-off between robustness and computational cost. The same figure also examines the effect of input window length: as the input duration grows, more windows contribute to the histogram, sharpening the peak at the true offset and further improving accuracy. With $k = 5$ and a 180 s input window, MoBind reaches 100% accuracy under a 200 ms tolerance and an MAE of only 0.05 s on mRi, demonstrating robust synchronization even for highly repetitive rehabilitation exercises.

8.2. Mask ratio of MTP

During training, we jointly optimize the alignment objective with the auxiliary Masked Token Prediction (MTP) task. Specifically, we randomly mask a fraction α of temporal tokens across all sensors and train the IMU module to reconstruct the masked tokens from the unmasked context. Table 15 evaluates different masking ratios α during training MoBind. A lower masking ratio of 0.65 yields the best synchronization performance but degrades action recognition, while a higher ratio of 0.85 harms synchronization without clear benefits for recognition. A ratio of 0.75 provides a good compromise, delivering strong synchronization and action recognition performance. We therefore adopt $\alpha = 0.75$ in our final model as the best trade-off between these objectives.

8.3. Retrieval results with local features

MoBind adopts a hierarchical design that exploits spatial locality: each IMU sensor is first aligned with its corresponding body-part representation, and these local features are subsequently aggregated into a global embedding for cross-modal alignment. Extensive experiments show that this global representation is not only effective for fine-grained temporal alignment but also beneficial for coarse semantic tasks such as action recognition. Beyond the global embedding, the per-sensor local representations are also informative for body-part localization; here, we further assess their quality on cross-modal retrieval. As shown in Table 16, retrieval performance varies across limbs according to their motion characteristics: sensors on highly active regions (e.g., the wrists) yield stronger results than those on less active regions (e.g., the knees in mRi). These findings confirm that MoBind learns meaningful local embeddings, supporting the use of their aggregation into a powerful global representation for downstream tasks.

Limb	IMU → Video			Video → IMU		
	R@1	R@5	R@10	R@1	R@5	R@10
RWrist	0.79	0.89	0.92	0.73	0.86	0.88
LWrist	0.81	0.93	0.96	0.85	0.91	0.93
RKnee	0.15	0.30	0.34	0.11	0.22	0.27
LKnee	0.30	0.55	0.59	0.26	0.41	0.56
Global	**0.94**	**0.99**	**1.00**	**0.92**	**0.99**	**0.99**

Table 16. **Retrieval performance on the mRi dataset using local features.** Performance varies by body part, with dynamic limbs (e.g., wrists) performing best. Results highlight the framework's ability to capture and integrate motion cues from individual sensors.

9. Qualitative Results

In this section, we present additional qualitative results to further demonstrate the effectiveness of MoBind. Fig. 13 and Fig. 14 show examples of video retrieval using IMU queries across the mRi, TotalCapture, and EgoHumans datasets. Fig. 15 and Fig. 16 illustrate representative weighted offset histograms used for temporal synchronization. To avoid clutter and make the IMU streams easier to interpret, we visualize only a single IMU in these figures instead of all available sensors; nevertheless, the examples highlight MoBind's robustness in correcting offsets even under highly repetitive motion. Finally, Fig. 17 provides additional examples of body-part localization on EgoHumans.

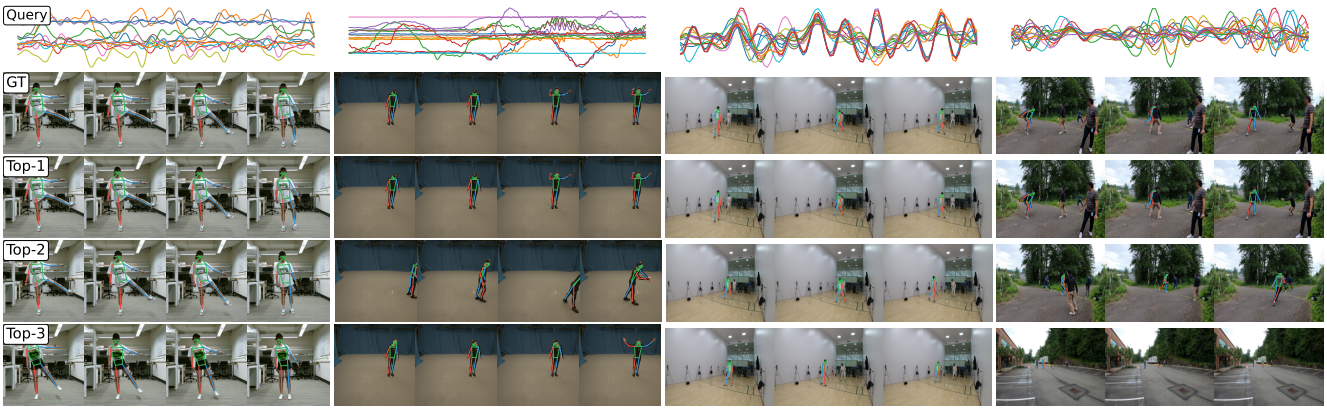

Figure 13. Qualitative examples of IMU → Video retrieval.

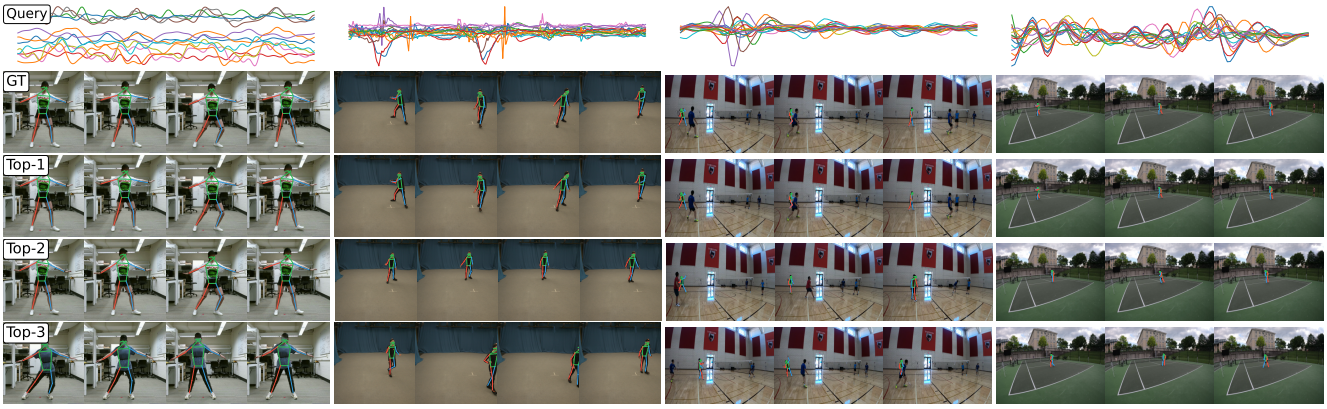

Figure 14. Qualitative examples of IMU → Video retrieval.

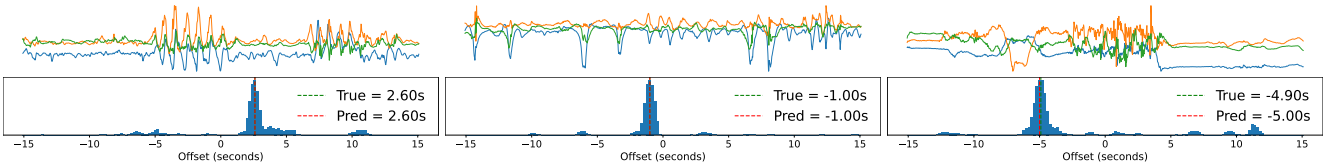

Figure 15. Weighted offset histograms constructed from the top-5 retrieved neighbors on TotalCapture.

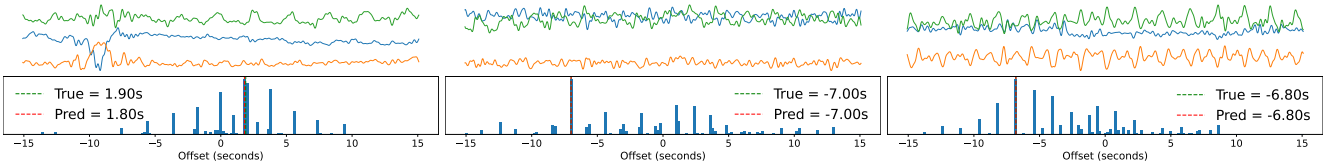

Figure 16. Weighted offset histograms constructed from the top-5 retrieved neighbors on mRi.

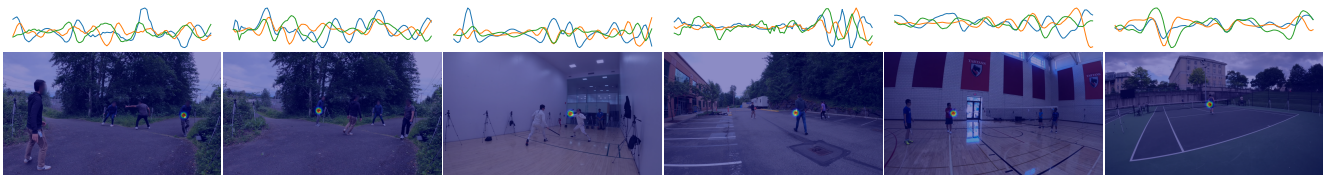

Figure 17. Body-part localization on the EgoHumans.