This WACV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

# XR-MBT: Multi-modal Full Body Tracking for XR through Self-Supervision with Learned Depth Point Cloud Registration

Denys Rozumnyi Nadine Bertsch Othman Sbai Filippo Arcadu Yuhua Chen
Artsiom Sanakoyeu Manoj Kumar Catherine Herold Robin Kips

Meta Reality Labs Zurich

## Abstract

*Tracking the full body motions of users in XR (AR/VR) devices is a fundamental challenge to bring a sense of authentic social presence. Due to the absence of dedicated leg sensors, currently available body tracking methods adopt a synthesis approach to generate plausible motions given a 3-point signal from the head and controller tracking. In order to enable mixed reality features, modern XR devices are capable of estimating depth information of the headset surroundings using available sensors combined with dedicated machine learning models. Such egocentric depth sensing cannot drive the body directly, as it is not registered and is incomplete due to limited field-of-view and body self-occlusions. For the first time, we propose to leverage the available depth sensing signal combined with self-supervision to learn a multi-modal pose estimation model capable of tracking full body motions in real time on XR devices. We demonstrate how current 3-point motion synthesis models can be extended to point cloud modalities using a semantic point cloud encoder network combined with a residual network for multi-modal pose estimation. These modules are trained jointly in a self-supervised way, leveraging a combination of real unregistered point clouds and simulated data obtained from motion capture. We compare our approach against several state-of-the-art systems for XR body tracking and show that our method accurately tracks a diverse range of body motions. XR-MBT tracks legs in XR for the first time, whereas traditional synthesis approaches based on partial body tracking are blind.*

## 1. Introduction

The creation of immersive XR experiences requires bringing realistic body representations for users to AR/VR devices. The full body motions of the users must be accurately represented to create authentic social experiences or interactions between users and their environment. This is also required to create a smooth user experience by accurately reproducing the whole body behavior in AR/VR applications, showing the virtual body as close as possible to the real one. Current XR body tracking methods focus on the "3-point" task, where signals from the head and controller tracking systems are leveraged to predict full body poses [11, 38, 47, 57]. Due to the absence of lower-body sensors in standalone XR devices, these approaches heavily rely on synthesis and generative models to reconstruct a plausible full-body motion. While recent methods explore egocentric vision-based body tracking [17, 36, 52], they require datasets collection with high-quality ground-truth, making them prohibitively expensive to execute at a large scale. This calls for the creation of novel methods to relax the need for labeled data for lower body tracking.

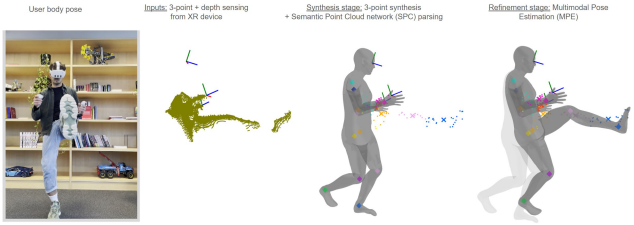


Figure 1. XR-MBT uses 3-points (head/wrists) and depth sensing from the XR device to learn real-time multi-modal body tracking. First, the synthesis stage generates a plausible body pose using the 3-Point signal. Then, the semantic point cloud network registers the body point cloud, and our self-supervised residual multi-modal pose estimation network predicts the refined body pose.

Due to the rapid development of augmented reality systems, depth sensing has become a core component of modern XR devices. In fact, real-time depth estimation is at the center of multiple essential systems for XR devices, from Passthrough rendering [7, 48] to sensing user environments for safety and solving occlusion when rendering virtual objects in the user environment. Figs. 1 and 3 show such egocentric point cloud data obtained from an XR device.

However, these *egocentric* point clouds present charac-

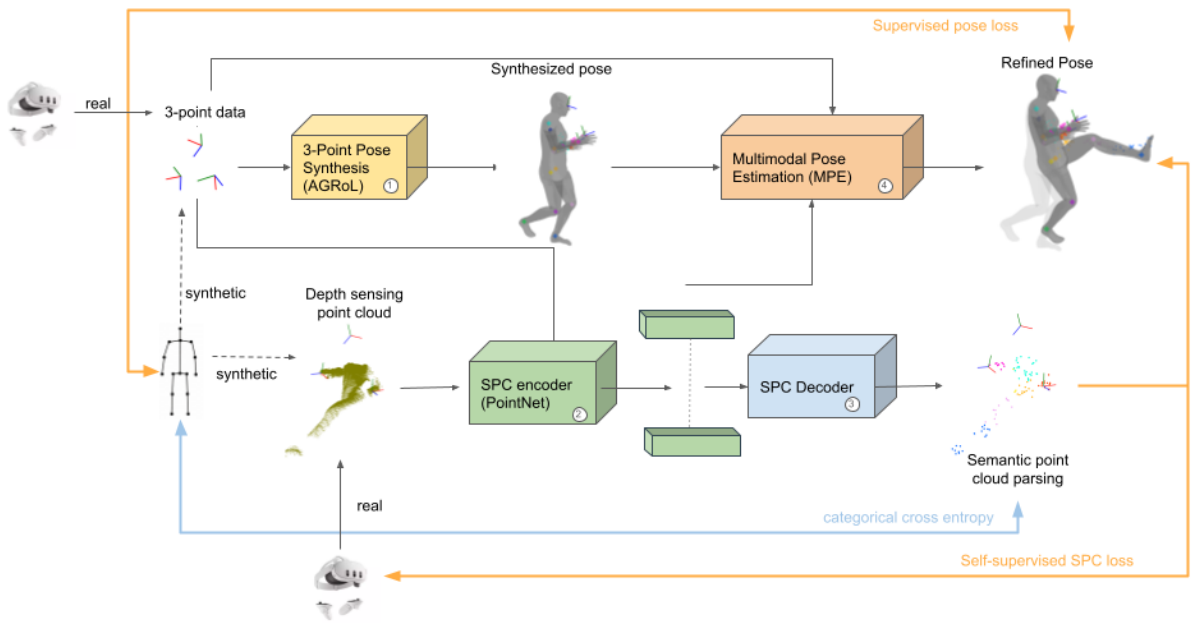


Figure 2. **XR-MBT architecture.** First, we use AGRoL [11] to synthesize the initial pose (yellow). Second, we process the point cloud by the Semantic Point Cloud (SPC) encoder (green) to generate point features. Third, the SPC decoder (blue) generates the probability of mapping of every point to a body joint, which is used for self-supervised learning. Last, the Multi-modal Pose Estimation (MPE) network (orange) estimates the final body pose by combining all modalities. MPE and SPC networks are jointly trained with a combination of Mocap and unlabeled depth data.

teristics that prevent them from being used directly in body tracking, which we address in this work. First, the point cloud is *unregistered* as correspondences with body parts cannot be immediately identified to drive the body pose estimation. Second, the egocentric body point cloud is *incomplete*. As the depth is captured by headset sensors located on the user's head, most body parts are only partially visible. Visibility is limited through the dependency of the sensor's field-of-view (FOV) on the user's head pose and also through self-occlusions. This signifies that body tracking still needs to rely on synthesis for body parts not captured by the sensing signal. Finally, egocentric point clouds have a *characteristic noise* due to the sensor placement specificity and the depth-sensing method, which require the use of real data to train body tracking models to learn sensor domain adaptation and generalize to diverse body poses.

For the first time in the XR body tracking field, we propose leveraging depth sensing of XR devices to introduce multi-modal full-body tracking. The **contributions** can be summarized as follows:

- We introduce XR-MBT (Fig. 2), a new multi-modal architecture to extend 3-point-based motion synthesis models to leverage depth sensing signal available in XR devices, allowing full body tracking in real-time.
- We propose a novel self-supervised method to train a multi-modal motion estimation network from unregistered point cloud data using a Semantics Point Cloud (SPC) loss leveraging a trained SPC network. This allows us to learn the robustness of the depth sensing system from real data while relaxing the need to collect expensive and complex ground truth data.
- We collect an egocentric body depth dataset with head and wrist tracking information, which can be used to explore a new research area around multi-modal body tracking for XR and increase the diversity of tracked motions.
- We demonstrate that our method can be applied to VR devices, achieving real-time tracking of the full body for XR for the first time, and we demonstrate the different elements of our method in an ablation study under a controlled synthetic environment, as well as using real data.

## 2. Related work

### 2.1. Motion synthesis for XR

Human motion synthesis is a key area in computer vision, with applications spanning animation, AR/VR and digital humans. The goal is to synthesize motions that are not only realistic but also expressive and diverse, catering to various tasks such as motion interpolation/prediction [26], denoising [21], and text-to-motion generation [13, 54]. Given the temporal nature of human motion, models like RNNs have been successfully employed for this task [12, 15, 21]. In recent years, generative models like VAEs [4, 26] and GANs [14] have been extensively explored for their

ability to generate high-quality, plausible motion data. The latest advancements have seen the emergence of diffusion models in motion synthesis [39, 53, 54], offering a new paradigm for generating motion by gradually transforming random noise into structured motion data.

Motion synthesis for XR has long been recognized as a key component of immersive XR experiences. The XR user wears a head-mounted device and hand controllers equipped with inertial measurement units (IMUs) that are used together with a vision system to track the position and orientation of the head and hands. This signal is typically referred to as the "3-Point" input. Predicting the full body pose from a 3-Point sequence represents an under-constrained problem with sparse inputs. [16, 19, 50] propose to address the motion synthesis problem with a bidirectional RNN by adding IMUs placed on the pelvis and lower body on top of the 3-Point input. [3, 10, 18] tackled the problem of egocentric pose estimation from just 3-Point input using VAEs. In [47], a deep reinforcement learning model with physics simulation was proved capable of synthesizing full body poses from either 3-Point input or the device pose alone. A method to generate a full body pose from 3-Point was also proposed using physics simulations in [51]. Some egocentric pose datasets with depth maps have been proposed such as xR-EgoPose [40, 41] and UnrealEgo [2]. A different direction to increase the number of tracked joints consists of estimating full-body pose from egocentric cameras mounted on the head [2, 6, 17, 17, 20, 28, 36, 43, 52, 56]. However, none of these studies focused on the specificity of the XR use case, where the availability of the 3-Point signal, in addition to the headset camera and depth sensors, requires multi-modal body tracking models.

Building multi-modal body tracking models requires large-scale ground-truth data collection to train models that can generalize to diverse body poses and environments for each sensor set, which leads to prohibitive costs. Unfortunately, such datasets dedicated to XR body tracking do not yet exist. To the best of our knowledge, no existing study addresses the problem of estimating full-body pose by fusing multi-modal inputs provided by an XR device.

### 2.2. Depth sensing for XR

Depth on AR/VR devices can be obtained based purely on vision, *e.g*. from stereo cameras [7]. Nevertheless, recent advances in machine-learning-based depth estimation, *e.g*. CRE-Stereo [23], made ML-based depth on the device a standard approach [1]. Alternative but complementary approaches are physical depth sensors such as dToF, which are also available on most smartphones today. For most of the applications using depth on XR devices, a strong constraint is to provide high-frequency and low-latency depth, which matches body-tracking needs. Thus, the choice of the depth sensing approach is determined given an aggregation of constraints to balance (such as power, hardware, frequency, quality, resolution, *etc*.) and can vary between different devices. In this work, we propose a method that is independent of the depth sensing technology. We only postulate that the XR device contains depth sensing technology that returns an unstructured point cloud. We also assume that the depth sensor is attached to the head-mounted device with a limited field of view of the human body.

### 2.3. 3D Human pose estimation from depth

The rapid evolution of deep learning techniques for processing 3D point cloud data has significantly enhanced the capabilities in various fields such as robotics, autonomous vehicles, and computer vision applications. Recent studies have delved into key tasks like 3D object classification, tracking, pose estimation, segmentation, and point cloud completion, showcasing the strides made in the development and application of these methods. [9, 49] provide a comprehensive review of these advancements, highlighting the historical progression, model architectures, learning algorithms, and the challenges that still persist. Focusing on real-time performance, [5, 37] propose a multi-stage architecture by first estimating a 2D pose from RGB or depth maps followed by a 3D pose estimation.

Previous point cloud-based human pose estimation methods have proposed to combine point clouds with IMU for pose estimation [34, 35, 46]. Furthermore, [42] proposes a multi-person 3D pose estimation system with 3D CNNs to overcome the limitation of using depth maps as single-channel information. The advancement of sequential 3D human pose and shape estimation from point clouds highlights the integration of spatial-temporal features to recover dynamic 3D body models accurately. [8] introduces novel multi-stage architecture RefiNet to progressively use 2D keypoints, depth maps and point clouds refining 3D human pose. These approaches tackle the inherent challenges of 3D pose estimation, including the ambiguity and complexity of human poses, the difficulty of annotating 3D joints in point clouds, and the demand for multi-person scenes. The introduction of hierarchical regression frameworks for detailed parametric model estimation [44, 45] from partial point clouds addresses the challenges related to the variability in human shapes and clothing details. Although not from egocentric views, [55] proposes a weakly supervised adversarial learning approach to generate 3D human pose leveraging both depth maps and point clouds. To the best of our knowledge, predicting full-body pose from egocentric depth and point clouds remains an understudied problem.

## 3. Method

We propose for the first time to formulate the XR body tracking problem as a multi-modal task where the inputs are both the conventional 3-Point data, *e.g*. from the headset

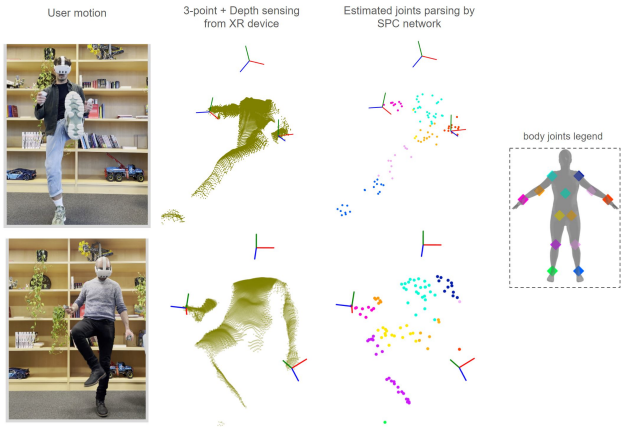


Figure 3. An example of real 3-Point and point cloud data accessible from XR devices and the corresponding output of our semantic point cloud (SPC) network. Even though the body point cloud is partial due to self-occlusions and outside the field of view, our SPC network learns to correctly identify the various body point joints by leveraging the 3-Point information. This indicates that the SPC network learns a meaningful registration of the body point cloud to supervise the MPE network training.

and controllers, and an unstructured point cloud obtained from the depth sensing system of the XR device. Such real inputs for diverse user motions are illustrated in Fig. 3.

In this section, we describe all components of XR-MBT, the proposed self-supervised multi-modal body tracking method. The Semantic Point Cloud (SPC) network learns to register an unstructured body point cloud to identify body joint information, while the Multi-modal Pose Estimation (MPE) residual network learns to refine body pose based on 3-Point data and SPC network embedding. First, we train both networks jointly on Mocap data with simulated synthetic point clouds. Using a novel self-supervised loss with learned semantics, the multi-modal residual network is then fine-tuned on a dataset of unlabeled depth data to learn robustness to sensor noise.

**Inputs.** We assume the input is a sequence of length $N = 196$ of 3-point inputs $x^{1:N}$ and sampled point clouds $p^{1:N}$. The 3-Point inputs consist of position, acceleration, rotation and rotational acceleration of each point. We represent rotations as 6D rotations as introduced in [58]. Thus, $x^t \in \mathbb{R}^{54}$. Due to computational reasons and since point clouds at run-time have different numbers of points, we sample $P = 100$ points from each point cloud with probability proportional to their depth value to reinforce the presence of body parts such as feet that cover a smaller surface than the closer torso. Hence, $p^t \in \mathbb{R}^{P\times 3}$.

**Outputs.** We denote the output of the synthesis network as

$$y^{1:N} = \text{synthesis}(x^{1:N}), \tag{1}$$

with $y^t \in \mathbb{R}^{J\times 6}$, where for each of $J$ joints from a skinned body model representation (*e.g.* SMPL [24]), we predict local joint rotations. The SPC network registers the point cloud by predicting joint class probabilities $l^{1:N}$, where $l^t \in \mathbb{R}^{P\times(J+1)}$ represents the probability of each point in the point cloud to be assigned to one of $J$ joints or the background/outlier label, *e.g.* for each point $i$, the following holds: $\sum_{j=1}^{J+1} l^t(i,j) = 1$. Additionally, the SPC network predicts point cloud features $f^{1:N}$, where $f^t \in \mathbb{R}^{128}$, thus:

$$(f^{1:N}, l^{1:N}) = \text{spc}(x^{1:N}, p^{1:N}). \tag{2}$$

Finally, the multi-modal pose estimation residual network predicts a final pose as an offset from local rotations $y^{1:N}$ of the synthesis network:

$$z^{1:N} = y^{1:N} + \text{mpe}(x^{1:N}, y^{1:N}, f^{1:N}). \tag{3}$$

### 3.1. Semantic point cloud network

As the egocentric point cloud obtained by the depth sensing system of the XR device is unlabeled, it cannot be directly leveraged to learn a multi-modal body tracking model. The goal of the *Semantic Point Cloud (SPC)* network is two-fold. First, it learns a compact encoding $f^{1:N}$ of point cloud features that are leveraged by a body tracking model. Second, it provides a semantic registration $l^{1:N}$ of the point cloud to identify the different body joints and label the point cloud. This registration is used to formulate a self-supervised loss described in Section 3.3.

The SPC network is composed of a PointNet [30, 31] encoder-decoder architecture, and its training is illustrated in Fig. 2. For each timestamp $t$ separately, it takes the sampled point cloud and the 3-Point data as input. Indeed, the head and wrist poses provide an important motion trajectory context for interpreting the semantics of the point cloud. Furthermore, the point cloud positions are normalized with respect to the position of the depth sensor attached to the XR device. To represent the temporal context of the point cloud, the SPC decoder takes as input the local PointNet features from the SPC encoder and an average pooling of the global PointNet features over the previous timestamps of the sequence. The 3-Point data with rotations are also passed to the decoder through a skip connection to ensure that the headset and controller information is preserved. For each of the sampled points in the point cloud, the SPC decoder outputs the probability of mapping to a body joint.

The SPC network is trained on the Mocap data, where synthetic point clouds are generated by sampling points on the body mesh surface. The body shape is randomized to represent diverse users. To increase robustness in real-world scenarios, we add Gaussian noise with a standard deviation of 2 cm to all points. Then, we create outliers by adding Gaussian noise with a standard deviation of 20 cm to 2% of all points in the point cloud. The ground truth label for each point in the point cloud is defined as the closest joint

in the Mocap ground truth body pose. If the closest joint is further away than a threshold of 10 cm, we label such points as background/outliers to represent sensor noise. The SPC encoder and decoder modules are jointly trained with all other modules using categorical cross-entropy loss between the predicted labeling $l^{1:N}$ and ground-truth labeling $\hat{l}^{1:N}$:

$$\mathcal{L}_{ce} = \text{cross-entropy}(l^{1:N}, \hat{l}^{1:N}) \,. \tag{4}$$

### 3.2. Multi-modal pose estimation network

The proposed pose estimation method is composed of two stages: synthesis and refinement, and its training is described in Fig. 2. Because there is no guarantee that any body parts are in the field-of-view of the depth sensor, we initialize the body tracking system with a 3-Point pose synthesis to guarantee the plausibility of the pose. For this purpose, we reuse the AGRoL [11] generative diffusion-based model architecture for the synthesis stage. The role of the Multi-modal Pose Estimation residual network is to refine the synthesized pose using the point cloud information available from the depth sensing. The MPE network takes as inputs the 3-Point signal $x^{1:N}$, the point cloud information, represented as embedding $f^{1:N}$ of the SPC encoder, and the synthesis network output $y^{1:N}$. Then, it predicts the final pose $z^{1:N}$ as an offset from the synthesis output.

The MPE network is trained on synthetic data, where point clouds are generated using a procedure similar to the SPC model training. The Mocap ground-truth pose $\hat{z}^{1:N}$ is used for supervision, with joints local rotation mean-squared error (MSE) and global position MSE, defined as:

$$\mathcal{L}_{rot} = \text{rot-mse}(z^{1:N}, \hat{z}^{1:N}) \,, \mathcal{L}_{pos} = \text{pos-mse}(z^{1:N}, \hat{z}^{1:N}) \,. \tag{5}$$

### 3.3. Self-supervised loss

Due to the nature of our input data, we could directly compute a self-supervised loss using the provided point cloud. The most straightforward Point Cloud loss (PC-loss) is defined as an average distance between the point cloud and the closest face of the predicted body pose mesh, *i.e.* we have to assume a default body shape *in this simplest case*. However, mapping a pose to an unlabeled point cloud is too ambiguous and there is no guarantee that the right body parts would move toward the correct points. The predicted pose could simply deform to collide with the center of the point cloud while minimizing the PC-loss.

Instead, we propose to leverage the SPC decoder to predict a semantic registration of the point cloud to formulate a novel Semantic Point Cloud Loss (SPC-loss). We define support for each joint $j$ as the sum of predicted probabilities from all points in the point cloud:

$$\text{support}_j(l^t) = \sum_{i=1}^{P} l^t(i,j) \,. \tag{6}$$

For each body joint with support larger than $0.05P$, which form a set $J^*$, we compute a spatial center that represents a noisy estimate of the joint position:

$$\text{cen}_j(p^t, l^t) = \frac{1}{\text{support}_j(l^t)} \sum_{i=1}^{P} p_i^t \cdot l^t(i,j) \,. \tag{7}$$

This joint estimate is too noisy to be directly used by a run-time optimization methods (*e.g.* inverse kinematics) as it lies on the surface of the body in most cases, and is largely affected by the sampling of the point cloud. Thus, we introduce a threshold $\theta = 10$ cm and penalize only distances over this threshold. The predicted joints are extracted by $\text{pos}_j(z^t)$, and the final SPC-loss is $\mathcal{L}_{spc} =$

$$\frac{1}{N|J^*|} \sum_{t=1}^{N} \sum_{j \in J^*} \max(0, \|\text{pos}_j(z^t) - \text{cen}_j(p^t, l^t)\|_2 - \theta) \,. \tag{8}$$

As an example in Fig. 1, the noisy estimate of the joint position is represented as a cross, while the predicted joint position by the model is illustrated by a diamond. To preserve the realism of the initially synthesized pose, the body joints that are not visible to the sensors also need to be moved to a position that is consistent with the visible joints, for instance, to preserve a realistic balance.

**Fine-tuning on real data.** Training only on synthetic point clouds does not capture the full, characteristic noise of the depth sensor and cannot fully generalize to real depth data, as demonstrated in Section 4, and an additional form of supervision is beneficial. Even though it is unlabeled, the real point cloud data can be used to supervise the training of the MPE network and learn an adaptation to the specific noise of the sensor. During self-supervision on real data, we use the same self-supervised pose loss $\mathcal{L}_{spc}$ (8) on unlabeled depth data and minimize the distance of predicted centroids to the pose estimated by our MPE model.

### 3.4. Training

The 3-Point synthesis model is pre-trained separately using the method described in [11]. Then, it is frozen, and the SPC and MPE networks are jointly trained by minimizing a weighted sum of all previously defined loss terms:

$$\mathcal{L} = w_{rot}\mathcal{L}_{rot} + w_{pos}\mathcal{L}_{pos} + w_{ce}\mathcal{L}_{ce} + w_{spc}\mathcal{L}_{spc} \,, \tag{9}$$

where weights are set empirically to $w_{rot} = 1$ (for error in radians), $w_{pos} = 0.01$ (for position error in meters), $w_{ce} = 0.1$, and $w_{spc} = 0.01$ (for error in meters).

The SPC and the MPE networks are jointly trained on a combination of synthetic Mocap and real data using ADAM optimizer [22] for 20k iterations. The batch size for Mocap data is set to 128, and for real data, it is set to 32. The SPC-loss $\mathcal{L}_{spc}$ is averaged for both real and Mocap data. All

| Model | MPJPE (cm) up \| low | MPJRE (radians) up \| low | MPJVE ($m^2/s^3$) up \| low | jitter (pred/gt) up \| low | PC-loss |
|---|---|---|---|---|---|
| All Motions | | | | | |
| AGRoL [11] (retrained) | 1.87 \| 11.21 | 1.79 \| 2.30 | 10.27 \| 34.36 | **2.55** \| **6.19** | 0.61 |
| + MPE | 1.78 \| 9.62 | 1.75 \| 2.23 | 9.94 \| 31.89 | 3.22 \|12.66 | 0.39 |
| + MPE + SPC-decoder | 1.77 \| 9.30 | 1.75 \| **2.21** | 9.95 \| 32.41 | 3.11 \| 12.98 | 0.34 |
| + MPE + SPC-decoder + PC-loss | 1.78 \| 10.25 | 1.76 \| 2.25 | 9.96 \| 32.97 | 2.84 \| 9.43 | 0.48 |
| + MPE + SPC-decoder + SPC-loss | **1.76** \| **9.27** | **1.74** \| **2.21** | **9.78** \| **31.40** | 2.88 \| 9.47 | **0.32** |
| Kicking | | | | | |
| AGRoL [11] (retrained) | 1.58 \| 10.21 | 1.63 \| 2.05 | 10.53 \| 41.10 | **1.87** \| **3.32** | 0.80 |
| + MPE | 1.49 \| 8.27 | 1.61 \| 2.01 | 10.22 \| 36.56 | 2.31 \| 7.6 | 0.39 |
| + MPE + SPC-decoder | **1.48** \| 7.97 | 1.61 \| 1.99 | 10.24 \| 37.76 | 2.23 \| 7.88 | 0.33 |
| + MPE + SPC-decoder + PC-loss | 1.50 \| 8.91 | 1.62 \| 2.02 | 10.26 \| 38.98 | 2.08 \| 5.70 | 0.51 |
| + MPE + SPC-decoder + SPC-loss | **1.48** \| **7.78** | **1.59** \|**1.98** | **10.11** \| **36.42** | 2.18 \| 5.93 | **0.28** |

Table 1. Ablation study on Mocap data test set with synthetically generated point clouds. We compare the state-of-the-art AGRoL method to our MPE (Sec. 3.2) network, SPC decoder (Sec. 3.1), and various self-supervised losses (Sec. 3.3). Metrics are reported separately for the upper (up) and lower (low) body. Additional results for diverse motion protocols can be found in the supplementary material.

| | AGRoL | SPC | MPE | Total |
|---|---|---|---|---|
| Compute [GFLOPs] | 122 | 89 | 2.5 | 213.5 |
| Runtime [ms] | 35 | 2.48 | 3.93 | 41.41 |

Table 2. Compute and runtime of each step in our pipeline.

other losses are computed only for the Mocap batch. The learning rate is set to $0.0003$. As the synthesis network is a diffusion model, we randomly sample new predictions for each batch during training of the PSC and MPE models to learn better robustness to the noise and diversity of the pose synthesis stage. We implement the MPE network as a simple MLP with 2M parameters, and the models are trained for 1 day on a cluster with 8 GPUs NVIDIA P100.

**Compute.** AGRoL backbone compute is 122 GFLOPs for diffusing 5 DDIM steps. In addition, the SPC network compute is 89 GFLOPs, using the original PointNet implementation, but could be replaced by an efficient architecture for on-device integration such as PointNeXt [32] reporting 7.2 GFLOPs for 3D semantic segmentation. The MPE network compute is 2.5 GFLOPs. The inference time on a single P100 GPU is 2.48 ms for the SPC network and 3.93 ms for the MPE network. Runtime of the full pipeline is 41.41 ms (Table 2), which corresponds to 25 fps, making our approach compatible with real-time usage on the device.

# 4. Experiments

## 4.1. Datasets

**Mocap dataset.** To ensure sufficient representation of lower body motions and to break down metrics by motion type, we replace the conventionally used AMASS dataset [25] with our own Mocap dataset manually labeled with action types. Active lower body motions, where the tracked lower body is beneficial, are under-represented in the AMASS dataset [25] and do not suit the task we propose to address in this paper. To illustrate this, we extract BABEL [29] action labels for AMASS and compute that it has only 2.7 hours of lower body active motions (out of 43 hours, 6%). In contrast, our Mocap dataset has 16 hours of such lower body active motions (out of 65 hours, 25%). The dataset entails 2964 motion capture sequences collected using a Vicon system from 330 subjects, 50% male and 50% female, with a high diversity of height and weight. Each sequence lasts on average 60 seconds and was designed to include a single action label such as "walking" or "knee strikes". The marker data was then fitted using a skeleton of 89 body joints. We retrain the AGRoL synthesis model on our dataset and skeleton model to account for these changes and report the metrics in Table 1.

**Egocentric depth dataset.** We construct a real dataset of egocentric depth data taken from a real XR device. We collect a total of 203 sequences from 11 subjects executing diverse body motion protocols while using a Quest 3 VR headset. For each sequence, we record data from the 3-Point tracking signal of the controllers and headset tracking, as well as the point clouds from the native depth sensing system of the device. We additionally collect 20 test sequences that are not used during training, and where user motions are additionally captured from a mobile phone camera for qualitative evaluation. As the collected point cloud does not only cover the body but the entire scene, we filter the body point cloud using simple proximity and density as-

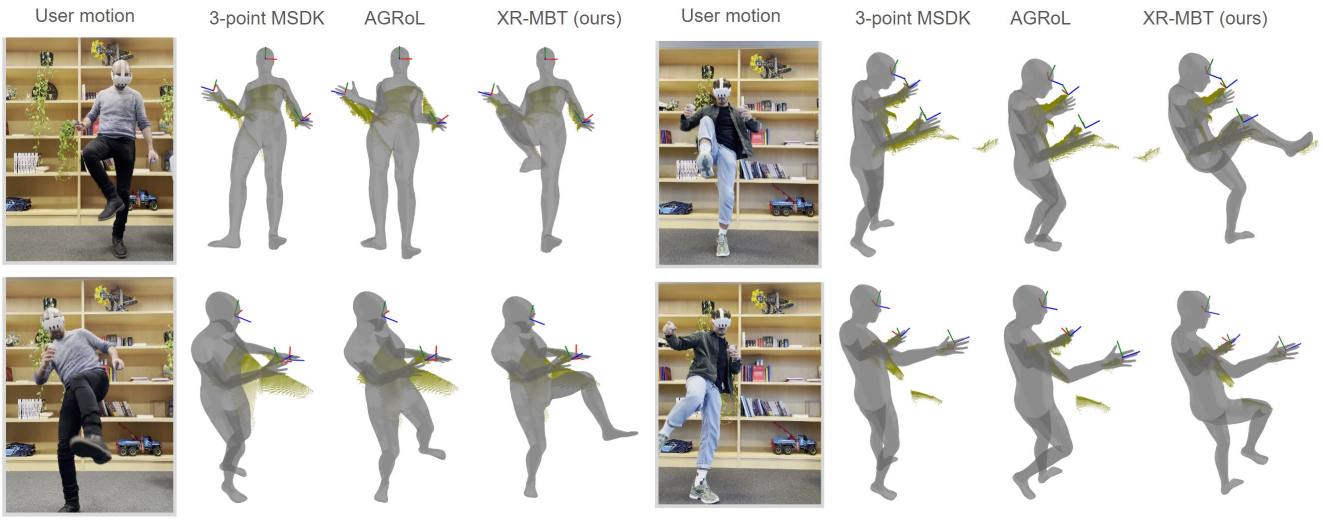


Figure 4. Comparison of our XR-MBT method to other XR body tracking methods on real data. By leveraging multi-modal inputs in addition to the 3-Point data, our method is able to cover a large variety of lower body motions while preserving the plausibility of the pose for body parts outside the field of view of the depth sensor.

sumptions to retrieve only the body parts. Examples of the obtained point clouds are visible in Figs. 3 and 4.

### 4.2. Experiments on synthetic data

To conduct an ablation study in a controlled environment, we first evaluate each component of our method on Mocap data with simulated point clouds. Such evaluation is commonly encountered in the XR body tracking literature as there is no available dataset of real 3-Point inputs with body GT. We individually train and evaluate different components of our method (Table 1). We reuse the metrics definition from [11], with the exception of the jitter metric that we report as the ratio between the predicted and ground-truth jitter for better readability. Additionally, we report the PC-loss (the point cloud loss without registration).

Several observations can be made from these results. First, using an SPC decoder module improves the lower body accuracy compared to training the MPE network using only a PointNet encoder. This demonstrates the importance of learning a semantically meaningful point cloud registration as input to the MPE network. Second, using the simple PC-loss for self-supervised training of the MPE network leads to an increased error compared to using only pose supervision. Indeed, minimizing the distance between the predicted pose and an unregistered point cloud is too ambiguous as the closest body pose vertex might belong to a different body part than the one visible to the depth sensor. However, the introduction of the SPC-loss $\mathcal{L}_{spc}$ (8) leveraging the SPC decoder outputs allows to improve the quality of self-supervision, decreasing the lower body pose error for kicking motions from 8.91 cm to 7.79 cm. Note that our motion labels are attributed at the sequence level and not frames. As a result, the kicking frames are still underrepresented in the kicking sequences (10%). Thus, these metrics underestimate the margin in quality between our lower body estimate and conventional 3-Point methods, as shown by the qualitative evaluation in the following section. We also report the distribution of the lower body pose error for different motions in Fig. 5 and show that the self-supervised SPC-loss generalizes well to diverse types of action labels. Even though it only brings a marginal improvement in such synthetic experiments, formulating self-supervision to learn adaptation to the depth sensor characteristics is key when training and evaluating with real data, as we demonstrate in the following section. Finally, computing the PC-loss on the synthetic point clouds allows to verify that the SPC-loss learns to leverage the point cloud data better.

The improvement in walking metrics can intuitively be explained by the following factors: **(1)** Due to the large field of view of the depth sensor, the lower body and, in particular, the upper leg and knee are frequently visible in the point cloud (even when not looking down), providing strong information to improve leg position accuracy. **(2)** When a leg is not visible to the depth sensors, this also constitutes a relevant signal to the model that this leg should not be stepping forward, which can improve the stepping cycle accuracy.

Compared to the 3-Point synthesis task, our method leads to a higher jitter level. While the jitter increase is controlled for the upper body, going from 2.55 to 2.88 on average, the increase in the lower body jitter is substantially more significant, going from 6.19 to 9.47. To some extent, this is expected as the lower body is no longer synthesized but tracked. The leg joints are activated more often, particularly for the motion types that we use. An excessive jitter

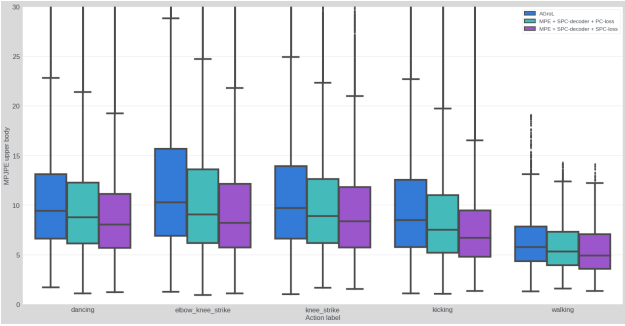


Figure 5. MPJPE for lower body computed on the Mocap test set with different action labels. We compare AGRoL [11] to our method trained with PC-loss or SPC-loss (8).

could be corrected by using a temporal model or IK constraints at run-time to regularize body joint activations.

### 4.3. Experiments on real data

Finally, we evaluate our approach against several state-of-the-art solutions in the field of XR body tracking. In particular, we use our unlabeled 3-Point egocentric depth dataset to compare the proposed XR-MBT architecture against AGRoL [11] and the commercially available 3-Point MSDK [27]. We could not include the MSDK model in the synthetic experiments, as it is only available as a Unity package, making it impossible to control the training set.

For qualitative evaluation of the SPC network, we use real point cloud data from our test sequences and visualize the body classification of the SPC decoder. These results are visible in Fig. 3 and in supplementary materials. Even when the body point cloud obtained from real depth data is partial due to the self-occlusion of the body and the limited field of view of the depth sensor, the proposed SPC decoder learns to classify various body joints accurately. This indicates that the SPC network learns a meaningful registration of the body point cloud to supervise the MPE network training.

In addition, we run different XR body tracking models on real test data sequences and visually compare the results with reference videos. The results are visible in Fig. 4, and demo figures and videos are accessible in the supplementary materials. It can be observed that the refinement step successfully learns to transition from a synthesized pose to a tracked pose that matches the point cloud signal. This transition is more complex than moving detected body joints toward the point cloud, and we can see in Figs. 1 and 4 that the MPE network successfully learns to move the leg not visible to the depth sensor to maintain the character balance and the overall plausibility of the motion. It can also be observed that using the point cloud information allows a better pose of the upper body joints visible to the sensor, such as the torso and elbows. By leveraging multi-modal inputs in addition to the 3-Point data, our method can cover a largely

| Motion | MSDK [27] | AGRoL [11] | XR-MBT | + real |
|---|---|---|---|---|
| walking | 7.16 | 7.44 | 4.67 | 4.39 |
| kicking | 7.60 | 6.41 | 5.01 | 3.74 |
| football | 1.70 | 1.90 | 1.79 | 1.22 |
| beatsaber | 6.77 | 5.39 | 4.70 | 2.90 |
| knee strikes | 1.16 | 1.13 | 1.04 | 0.93 |
| lift legs | 2.45 | 2.19 | 1.67 | 1.64 |
| average | 4.12 | 3.74 | 3.02 | 2.58 |

Table 3. Point cloud error (PC-loss, in cm) computed on real test data for different actions. These results demonstrate that training with self-supervision on real data allows to learn adaptation to the real depth sensors, leading to a reduced error compared to training on synthetic point cloud data.

increased range of motions, such as kicking, knee strike and football, while preserving the realism of the motions when body parts are outside the field of view.

**Limitations.** One apparent limitation of our approach is the absence of a representation of the user's body shape. Indeed, even though we cannot assume that the user's body shape has been previously registered, an explicit representation of the user's shape could be beneficial to use as input to our approach. To that extent, solving the body shape of the user at run-time given the partial point cloud data could constitute a natural extension of our method and help disambiguate the point cloud and refine the pose further. Another limitation is that we assume that the user does not interact with any large objects, which could create interference in the point cloud registration. Moreover, motion quality could be improved by using human motion priors [33].

## 5. Conclusion

In this work, we introduced for the first time the use of depth sensors in the XR body tracking task. While these sensors provide a useful signal to drive the body, the unregistered point cloud is ambiguous and cannot be directly used to drive body tracking. We proposed a solution by learning to predict a semantic registration of the egocentric point cloud. We showed that this learned point cloud registration could be leveraged to formulate a self-supervised loss, which in turn enabled training a multi-modal body tracking model on real unlabeled depth data. As a result, our architecture is capable of automatically transitioning between synthesis and tracking depending on the body parts visible to the sensors in order to track the lower body and always guarantee a plausible pose and an authentic body presence in XR. We believe that this work opens novel perspectives in the field of XR body tracking by exploiting readily available depth data in an efficient and effective manner.

## References


[1] Meta blog. https://www.meta.com/en-gb/blog/quest/ai-powered-technologies-quest-3-pro-ray-ban-meta-smart-glasses/. Accessed: 2024-07-09. 3

[2] Hiroyasu Akada, Jian Wang, Soshi Shimada, Masaki Takahashi, Christian Theobalt, and Vladislav Golyanik. Unrealego: A new dataset for robust egocentric 3d human motion capture. In *ECCV*, 2022. 3

[3] Sadegh Aliakbarian, Pashmina Cameron, Federica Bogo, Andrew Fitzgibbon, and Thomas Cashman. Flag: Flow-based 3d avatar generation from sparse observations. In *CVPR*, 2022. 3

[4] Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Lars Petersson, and Stephen Gould. A stochastic conditioning scheme for diverse human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5223–5232, 2020. 2

[5] Abhijat Biswas, Henny Admoni, and Aaron Steinfeld. Fast on-board 3d torso pose recovery and forecasting. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 20–24, 2019. 3

[6] Young-Woon Cha, True Price, Zhen Wei, Xinran Lu, Nicholas Rewkowski, Rohan Chabra, Zihe Qin, Hyounghun Kim, Zhaoqi Su, and Yebin et al. Liu. Towards fully mobile 3d face, body, and environment capture using only head-worn cameras. In *IEEE TVCG*, 2018. 3

[7] Gaurav Chaurasia, Arthur Nieuwoudt, Alexandru-Eugen Ichim, Richard Szeliski, and Alexander Sorkine-Hornung. Passthrough+ real-time stereoscopic view synthesis for mobile mixed reality. *ACM TOG*, 3(1):1–17, 2020. 1, 3

[8] Andrea D'Eusanio, Stefano Pini, Guido Borghi, Roberto Vezzani, and Rita Cucchiara. Refinet: 3d human pose refinement with depth maps. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2320–2327. IEEE, 2021. 3

[9] Zifeng Ding, Yuxuan Sun, Sijin Xu, Yan Pan, Yanhong Peng, and Zebing Mao. Recent advances and perspectives in deep learning techniques for 3d point cloud data processing. *Robotics*, 12(4):100, 2023. 3

[10] Andrea Dittadi, Sebastian Dziadzio, Darren Cosker, Ben Lundell, and Tom Cashman. Full-body motion from a single head-mounted device: Generating smpl poses from partial observations. In *ICCV*, 2021. 3

[11] Yuming Du, Robin Kips, Albert Pumarola, Sebastian Starke, Ali Thabet, and Artsiom Sanakoyeu. Avatars grow legs: Generating smooth human motion from sparse tracking inputs with diffusion model. In *CVPR*, 2023. 1, 2, 5, 6, 7, 8

[12] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE international conference on computer vision*, pages 4346–4354, 2015. 2

[13] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5152–5161, 2022. 2

[14] Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7134–7143, 2019. 2

[15] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)*, 35(4):1–11, 2016. 2

[16] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. In *ACM TOG*, pages 1–15, 2018. 3

[17] Hao Jiang, Hao Jiang Grauman, and Kristen Grauman. Seeing invisible poses: Estimating 3d body pose from egocentric video. In *CVPR*, 2017. 1, 3

[18] Jiaxi Jiang, Paul Streli, Huajian Qiu, Andreas Fender, Larissa Laich, Patrick Snape, and Christian Holz. Avatarposer: Articulated full-body pose tracking from sparse motion sensing. In *ECCV*, 2022. 3

[19] Yifeng Jiang, Yuting Ye, Deepak Gopinath, Jungdam Won, Alexander Winkler, and Karen Liu. Transformer inertial poser: Real-time human motion reconstruction from sparse imus with simultaneous terrain generation. In *CVPR*, 2022. 3

[20] Taeho Kang, Kyungjin Lee, Jinrui Zhang, and Youngki Lee. Ego3dpose: Capturing 3d cues from binocular egocentric views. 2023. 3

[21] Seong Uk Kim, Hanyoung Jang, and Jongmin Kim. Human motion denoising using attention-based bidirectional recurrent neural network. In *SIGGRAPH Asia 2019 Posters*, pages 1–2. 2019. 2

[22] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015. 5

[23] Jiankun Li, Peisen Wang, Pengfei Xiong, Tao Cai, Ziwei Yan, Lei Yang, Jiangyu Liu, Haoqiang Fan, and Shuaicheng Liu. Practical stereo matching via cascaded recurrent network with adaptive correlation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16263–16272, 2022. 3

[24] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 4

[25] Naureen Mahmood, Nima Ghorbani, Nikolaus Troje, Gerard Pons-Moll, and Michael J. Black. Human motion prediction via spatio-temporal inpainting. In *ICCV*, pages 7134–7143, 2019. 6

[26] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2891–2900, 2017. 2

[27] Meta. Create more natural movements using inside-out body tracking and generative legs, 2023. Last accessed March 5th 2024. 8

[28] Evonne Ng, Donglai Xiang, Hanbyul Joo, and Kristen Grauman. You2me: Inferring body pose in egocentric video via first and second person interactions. 2020. 3

[29] Abhinanda R. Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J. Black. BABEL: Bodies, action and behavior with english labels. In *CVPR*, pages 722–731, June 2021. 6

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 4

[31] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 4

[32] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *NeurIPS*, 2022. 6

[33] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *ICCV*, pages 11468–11479, 2021. 8

[34] Yiming Ren, Xiao Han, Chengfeng Zhao, Jingya Wang, Lan Xu, Jingyi Yu, and Yuexin Ma. Livehps: Lidar-based scene-level human pose and shape estimation in free environment. In *CVPR*, pages 1281–1291, June 2024. 3

[35] Yiming Ren, Chengfeng Zhao, Yannan He, Peishan Cong, Han Liang, Jingyi Yu, Lan Xu, and Yuexin Ma. Lidar-aid inertial poser: Large-scale human motion capture by sparse inertial and lidar sensors. *IEEE TVCG*, 29(5):2337–2347, may 2023. 3

[36] Helge Rhodin, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt. Egocap: egocentric marker-less motion capture with two fisheye cameras. In *ACM TOG*, 2016. 1, 3

[37] Thomas Schnürer, Stefan Fuchs, Markus Eisenbach, and Horst-Michael Groß. Real-time 3d pose estimation from single depth images. In *VISIGRAPP (5: VISAPP)*, pages 716–724, 2019. 3

[38] Mingyi Shi, Sebastian Starke, Yuting Ye, Taku Komura, and Jungdam Won. Phasemp: Robust 3d pose estimation via phase-conditioned human motion prior. In *ICCV*, pages 14725–14737, 2023. 1

[39] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022. 3

[40] D. Tome, T. Alldieck, P. Peluse, G. Pons-Moll, L. Agapito, H. Badino, and F. De la Torre. Selfpose: 3d egocentric pose estimation from a headset mounted camera. *IEEE TPAMI*, pages 1–1, 2020. 3

[41] Denis Tome, Patrick Peluse, Lourdes Agapito, and Hernan Badino. xr-egopose: Egocentric 3d human pose from an hmd camera. In *ICCV*, pages 7728–7738, 2019. 3

[42] Manolis Vasileiadis, Christos-Savvas Bouganis, and Dimitrios Tzovaras. Multi-person 3d pose estimation from 3d cloud data using 3d convolutional neural networks. *Computer Vision and Image Understanding*, 185:12–23, 2019. 3

[43] Jian Wang, Zhe Cao, Diogo Luvizon, Lingjie Liu, Kripasindhu Sarkar, Danhang Tang, Thabo Beeler, and Christian Theobalt. Egocentric whole-body motion capture with fisheyevit and diffusion-based motion refinement. 2023. 3

[44] Kangkan Wang, Jin Xie, Guofeng Zhang, Lei Liu, and Jian Yang. Sequential 3d human pose and shape estimation from point clouds. In *CVPR*, pages 7275–7284, 2020. 3

[45] Kangkan Wang, Huayu Zheng, Guofeng Zhang, and Jian Yang. Parametric model estimation for 3d clothed humans from point clouds. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 156–165. IEEE, 2021. 3

[46] Zhenzhen Weng, Alexander S. Gorban, Jingwei Ji, Mahyar Najibi, Yin Zhou, and Dragomir Anguelov. 3d human keypoints estimation from point clouds in the wild without human labels. In *CVPR*, pages 1158–1167, June 2023. 3

[47] Alexander Winkler, Yuting Ye, and Jungdam Won. Questsim: Human motion tracking from sparse sensors with simulated avatars. In *ACM TOG*, pages 1–8, 2022. 1, 3

[48] Lei Xiao, Salah Nouri, Joel Hegland, Alberto Garcia Garcia, and Douglas Lanman. Neuralpassthrough: Learned real-time view synthesis for vr. In *ACM TOG*, pages 1–9, 2022. 1

[49] Tianxu Xu, Dong An, Yuetong Jia, and Yang Yue. A review: Point cloud-based 3d human joints estimation. *Sensors*, 21(5):1684, 2021. 3

[50] Dongseok Yang, Doyeon Kim, and Sung-Hee Lee. Lobstr: Real-time lower-body pose prediction from sparse upper-body tracking signals. 2021. 3

[51] Yongjing Ye, Libin Liu, Lei Hu, and Shihong Xia. Neural3points: Learning to generate physically realistic full-body motion for virtual reality users. pages 183–94, 2023. 3

[52] Ye Yuan and Kris Kitani. 3d ego-pose estimation via imitation learning. In *ECCV*, 2018. 1, 3

[53] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16010–16021, 2023. 3

[54] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 2, 3

[55] Zihao Zhang, Lei Hu, Xiaoming Deng, and Shihong Xia. Weakly supervised adversarial learning for 3d human pose estimation from point clouds. *IEEE transactions on visualization and computer graphics*, 26(5):1851–1859, 2020. 3

[56] Dongxu Zhao, Zhen Wei, Jisan Mahmud, and Jan-Michael Frahm. Egoglass: Egocentric-view human pose estimation from an eyeglass frame. In *Int. Conf. on 3D Vision*, 2021. 3

[57] Xiaozheng Zheng, Zhuo Su, Chao Wen, Zhou Xue, and Xiaojie Jin. Realistic full-body tracking from sparse observations via joint-level modeling. In *ICCV*, 2023. 1

[58] Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. On the continuity of rotation representations in neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4