# QuestEnvSim: Environment-Aware Simulated Motion Tracking from Sparse Sensors

SUNMIN LEE, Seoul National University, South Korea
SEBASTIAN STARKE, Reality Labs Research, Meta, United States of America
YUTING YE, Reality Labs Research, Meta, United States of America
JUNGDAM WON*, Seoul National University, South Korea
ALEXANDER WINKLER*, Reality Labs Research, Meta, United States of America

Fig. 1. From only the pose of a headset and controllers, our method reconstructs a matching full-body pose that interacts naturally with various objects in a simulated environment.

Replicating a user's pose from only wearable sensors is important for many AR/VR applications. Most existing methods for motion tracking avoid environment interaction apart from foot-floor contact due to their complex dynamics and hard constraints. However, in daily life people regularly interact with their environment, e.g. by sitting on a couch or leaning on a desk. Using Reinforcement Learning, we show that headset and controller pose, if combined with physics simulation and environment observations can generate realistic full-body poses even in highly constrained environments. The physics simulation automatically enforces the various constraints necessary for realistic poses, instead of manually specifying them as in many kinematic approaches. These hard constraints allow us to achieve high-quality interaction motions without typical artifacts such as penetration or contact sliding. We discuss three features, the environment representation, the contact reward and scene randomization, crucial to the performance of the method. We demonstrate the generality of the approach through various examples, such as sitting on chairs, a couch and boxes, stepping over boxes, rocking a chair and turning an office chair. We believe these are some of the highest-quality results achieved for motion tracking from sparse sensor with scene interaction.

CCS Concepts: • **Computing methodologies** → **Motion capture**; **Physical simulation**.

---

*co-corresponding authors

Additional Key Words and Phrases: Character Animations, Motion Tracking, Reinforcement Learning, Environment Interaction

## 1 INTRODUCTION

AR/VR (Augmented, Virtual Reality) has the potential to create entirely new social experiences. For example, instead of 2D video calls, users could interact in a virtual 3D space. To create the feeling of presence, a user's avatar must accurately replicate their movement and body language as well as naturally interact with the environment. Marker-based motion tracking is inconvenient for such an experience, since it requires dozens of expensive cameras, a special suit and tedious calibration procedures. To create a lower friction experience, recent works have investigated *markerless* solutions for motion tracking such as phone video, Inertial-Measurement Units (IMUs), or Head Mounted Displays (HMDs).

We aim for creating a motion tracking system that allows environment interaction and relying only on the pose of the consumer VR device and environment information as input. Synthesizing full-body motions from sparse sensors is challenging because many different poses could potentially match a given sensor input and generation of the lower body motions is even more challenging due to the absence of information. In addition to these challenges,

Sunmin Lee, Sebastian Starke, Yuting Ye, Jungdam Won, and Alexander Winkler

generating plausible object-interaction motion requires special care. When users interact with their environments (e.g. sitting on a couch or leaning on a desk), the system should generate motions using the environments, which introduces complex physical constraints. Also, such motions are different from locomotion since the lower body is not always fully constrained by balancing, so there is more ambiguity. For example, when sitting on a couch, many different poses could potentially match a given sensor input.

In this paper, we develop a motion tracking algorithm that takes as input the headset and controller pose as well as a representation of the environment and generates full-body motions that match both sensor inputs and its surrounding environment. More specifically, we use a physically simulated avatar and learn a control policy to generate torques to drive the simulated avatar via deep reinforcement learning, where the goal is to track the user's headset and controller pose as close as possible. Similar to our approach, several motion tracking systems using physics-based avatars have been proposed, however, environment interactions apart from foot-floor contact were not demonstrated [Winkler et al. 2022; Ye et al. 2022]. Other methods [Luo et al. 2022] incorporated artificial forces to deal with the complex contact dynamics, however these forces can produce unnatural motions. Instead of using artificial forces, our control policy is trained to actively use the environment to generate the appropriate external forces to drive the simulated avatar, where the strategy is learned from mocap data that includes environment interaction. As a result, our system generates motions that are physically accurate and more believable within their environment. For example, an HMD close to a chair likely implies a user has sit down compared to just being in a crouching position.

As contributions, we first demonstrate that sparse-upper body input, if combined with physics simulation and environment observations, can generate realistic full-body motions in highly constrained environments without using any artificial force. We also show a non-trivial combination of key technical components are crucial for achieving high tracking performance and generalization to unseen real user inputs: Environment representation fed into the policy, contact reward during learning control policies, and scene randomization. To show the capability of our system, various examples, such as sitting on chairs, a couch and boxes, stepping over boxes, rocking a chair and turning an office chair are demonstrated where all motions are generated from unseen real user inputs without using any cleanup or post-processing (e.g. inverse-kinematics, contact resolving, smoothing, and etc). We believe these are some of the highest-quality results achieved for motion tracking from sparse sensor with scene interaction. We also show the capability of a policy tracking users with scene interaction from the HMD alone without controllers and a policy without future observations. Finally, we run ablation studies for the key design choices adopted in our system to understand how they affect performance and generalization of our system.

## 2  RELATED WORK

Motion tracking/reconstruction from a specific sensor is a research topic that has a long history in computer graphics and computer vision, so we review previous studies that are most closely relevant to

our approach, which includes *synthesizing motions with interaction*, *kinematic motion tracking from sparse sensor input*, and *physics-based motion tracking*.

### 2.1  Motion Synthesis with Interaction

Synthesizing plausible full-body motions with interaction is regarded as one of the notorious problems because the difficulty grows exponentially as environments get more complex (e.g. the number of objects in the scene, the types of interactions). Several data-driven methods have been proposed for solving the problem. Safonova et al. [2007] optimized transitions in the pre-constructed motion graph where interactions were modeled as spatio-temporal constraints. Its extension to multi-character interaction was also demonstrated [Won et al. 2014] via combining stochastic sampling and Laplacian motion editing [Ho et al. 2010; Kim et al. 2009]. Due to the complexity of optimization, patch-based methods have been proposed, which preserve interaction in a near-fixed state [Henry et al. 2012; Lee et al. 2006; Shum et al. 2008]. Although complex and large-scale human-object and human-human interactions have been demonstrated, they are computationally expensive and offline. Recently, many deep learning methods have been proposed, where a mapping from environment states to motions is constructed by a deep neural network in a supervised manner. Holden et al. [2017] demonstrated a locomotion controller autoregressively generating walking motions that adapt to uneven terrain. This idea was extended for human-scene interaction [Starke et al. 2019] and hand-object interaction [Zhang et al. 2021] where spatial representations such as voxel occupancy and proximity sensors were incorporated to describe the interactions. Our method is also a data-driven method and adopts spatial representations similar to the methods above to represent the current state of the environment, however, we synthesize such motions with sparse signals only and less supervision for the dataset.

### 2.2  Motion Tracking from Sparse Sensors

Using wearable sparse sensors for human motion tracking received a lot of attention due to its ease of use and wide applications in AR/VR (e.g. daily and outdoor mocap). Several systems based on inertial measurement units (IMUs) have been proposed. Marcard et al. [von Marcard et al. 2017] takes an offline approach optimizing pose parameters for the entire frame so that they match with the input signals. The more popular approach is to use data-driven algorithms, where many proposed systems rely on a mocap database from which those systems search motion clips (or short segments) that match the input signals [Andrews et al. 2016; Liu et al. 2011; Ponton et al. 2022; Riaz et al. 2015; Tautges et al. 2011]. Deep neural networks (DNNs) have shown promising results on handling high-dimensional and large data. DNNs learn a mapping from the sparse input signals to full-body pose through supervised learning with paired data. Real-time systems predicting local poses with negligible delay were demonstrated [Huang et al. 2018; Nagaraj et al. 2020] and global root motions have been improved [Guzov et al. 2021; Jiang et al. 2022; Yi et al. 2021] . Other than using IMUs, a system based on wearable electromagnetic field sensors was also proposed [Kaufmann et al. 2021]. Because these methods learn complex mappings
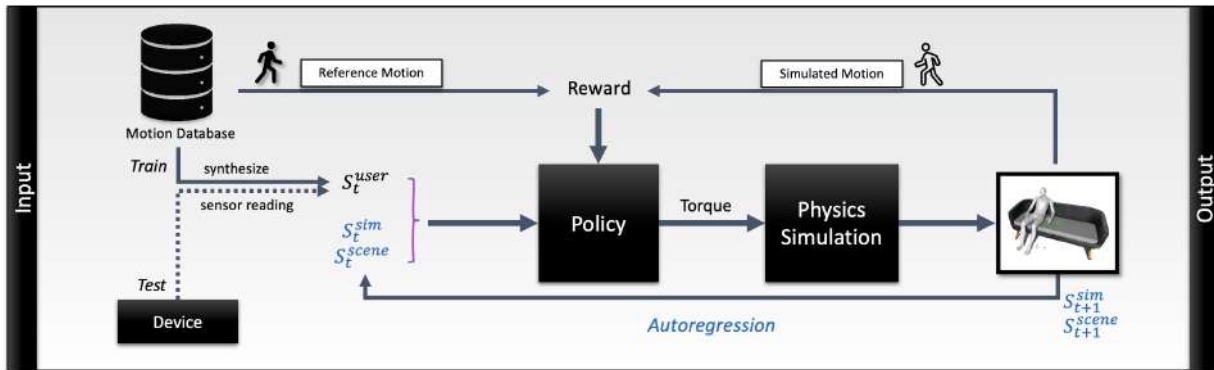
Fig. 2. System Overview.

directly from data while considering only kinematics it is crucial to have well-prepared and sufficient mocap data for training. In it's absence, artifacts such as foot sliding or jittery motions can appear. Furthermore, motion tracking from sparse sensors that allows full-body interactions has not been demonstrated yet in this research direction.

## 2.3 Physics-based Motion Tracking

Respecting physical laws when tracking user motions can improve the quality of generated motions. For example, foot sliding or penetration with the ground can be resolved by accurate contact modeling and forward simulation. One approach could be applying physics separately as soft constraints to refine estimated motions [Yi et al. 2022]. However, the most popular approach is to learn a control policy (a.k.a. controller) minimizing the discrepancy between the physically simulated characters and user inputs via deep reinforcement learning (Deep RL). Motion tracking from full-body mocap data have been proposed for a single clip [Merel et al. 2017; Peng et al. 2018a], interactive kinematic motion controllers [Bergamin et al. 2019; Park et al. 2019], and large datasets [Chentanez et al. 2018; Fussell et al. 2021; Won et al. 2020; Yuan and Kitani 2020]. Similar systems using video clips as input (either ego-centric or 3rd-person view) have also been demonstrated [Luo et al. 2021; Peng et al. 2018b; Yuan and Kitani 2019; Yuan et al. 2021]. They have also generated motions with contact-rich interaction such as sitting on chairs [Luo et al. 2022]. We also generate motions with contact-rich interactions as demonstrated in [Luo et al. 2022]. However, our system only requires the pose of the headset and controllers as input, with no information about the lower body. We also don't use artificial forces to stabilize the simulated character.

This setup is similar to recent approaches [Winkler et al. 2022; Ye et al. 2022]. However, both approaches demonstrated only minimal environment interactions. In this paper we explicitly incorporate scene observations into the policy, which allows the policy to learn how to use the environment for motion tracking. The policy learns that sitting on a chair produces external forces that allows it to lift the leg, or expects an external force when stepping on a box raised above the floor. It also learns to manipulate external objects like a tilting chair through the appropriate contact forces. In general, this policy has a much better understanding of its environment and how external forces affect its pose.

## 3 METHOD

As input our method takes a sequence of poses (i.e. 6D transformations) from a user's VR headset and two hand controllers, where the user is interacting with daily-life objects (e.g. chairs, table, and etc). Our system generates a full-body motion that tracks the user and their environment interactions in a physically plausible manner. We assume the 3D scene geometry is known, and can be obtained by scanning the scene in advance. Figure 2 shows an overview of our method. We use a physically simulated avatar to enforce the naturalness in both generated motions and interactions. We train our policy using mocap data that includes environment interactions. Training dataset is explained in Section 4.1.

We train a torque-based control policy to simulate the avatar via Deep Reinforcement Learning (DRL). At each time step, an agent (i.e. the simulated avatar) observes the state $s_t$ and performs an action $a_t$. The state is updated to $s_{t+1}$ while receiving a reward $r_t$ that represents the desirability of the transition, the updated state, and the action. The goal of DRL is to learn a control policy $\pi_\theta(a_t|s_t)$ (i.e. a tracking controller represented as a deep neural network with parameters $\theta$), which maximizes the expected sum of rewards $J(\pi_\theta) = E[\sum_{t=1,\ldots,\infty} \gamma^{t-1} r_t]$ over the entire trajectory, where $\gamma \in (0, 1)$ is the discount factor. We treat the policy output $\pi(a_t|s_t)$ as the mean of a Gaussian distribution with a fixed, diagonal covariance matrix and use Proximal Policy Optimization (PPO) algorithm [Schulman et al. 2017] to find an optimal policy.

## 3.1 Simulation Environment

Our simulated character (Figure 3) has 32 degrees of freedom and 18 links and is driven by joint torques. We do not allow self-collision between the character's body links. The objects that our mocap actors interact with must also be replicated in our physics simulation. We use primitive geometries like boxes or cylinders to approximate the real collision shapes. Figure 3 shows an example scene that includes the simulated character and the environment objects.

We label when and where contacts happen between the actor and their environment. This information is used as a supervision

Fig. 3. For accurate contact behavior we approximate all the real shapes (right) with geometric primitives (left): The character's mesh is approximated with 18 collision geometries (capsules, boxes and spheres). The environment objects are similarly approximated with geometric primitives. As long as the contact surfaces match, the visualization will look accurate. We calculate masses and inertias of the character and objects based on their collision geometry and assuming uniform density.

signal during training. The policy learns to leverage contacts with environment objects to track a user's motion. The contact labels are computed semi-automatically by using the simulator's collision detection algorithm on the mocap reference motions. Some manual correction is performed in difficult environments.

## 3.2 State & Action

The policy outputs action in the range of [-1,1], which we scale to appropriate torques using each joints maximum torques. The state $s_t = (s_t^{sim}, s_t^{user}, s_t^{scene})$ at time $t$ consists of the simulated avatar state $s_t^{sim}$, the pose of the users headset and controllers $s_t^{user}$, and the scene observation $s_t^{scene}$ representing the surrounding environment and objects. These are discussed in the following.

### 3.2.1 Simulated State. The simulated state

$$s_t^{sim} = \{q, \dot{q}, p, \dot{p}, R, \dot{R}, f\}$$

is composed of joint angles $q$, link positions $p$, link orientations $R$, their corresponding velocities and contact forces $f$ acting on the links. All values are represented in the avatar-centric coordinate system (i.e. facing frame) as used in [Winkler et al. 2022; Won et al. 2020]. This frame is defined by the pelvis orientation and position projected onto the ground. We use the first two columns of the rotation matrix to represent a link orientation.

### 3.2.2 Sensor State. The sensor state

$$s_t^{user} = (o_{t-l}^{user}, \cdots, o_t^{user}, \cdots, o_{t+k}^{user})$$

is a time-window of user observations $o^{user} = (R_h, p_h, p_l, p_r)$ which includes the orientation $R_h$ and the position $p_h$ of the headset and the positions $p_l$, $p_r$ from the left and right controllers.

We only use the position of the controllers, not their orientation. This is because the controllers orientations proved to be noisy and less reliable, due to fast motion and user-dependent styles of holding controllers. Similarly to the simulated character state, the sensor state is represented in the same avatar-centric coordinate system. In all our experiments, unless otherwise noted, we use a fixed window size of 1s of past and future information.

### 3.2.3 Scene State. In our method, information of the environment is crucial to make the policy understand and generate motions with environment interaction. The scene observation $s_t^{scene} = (h_1, h_2, \cdots)$ uses a height map to observe the geometrical features of the current environment. This height map is created from a circular grid centered around the avatar-centric coordinate and samples the height at each grid point $h_i$. The radius of the heightmap is 0.48m and it includes 120 grid points with 0.08m and 1/20 radian interval.

## 3.3 Reward

The behavior of the simulated avatar varies depending on the used rewards. In addition to an imitation reward that has been used in many other works [Bergamin et al. 2019; Peng et al. 2018a; Winkler et al. 2022; Won et al. 2020], we also add rewards that are crucial for natural-looking motions with environment interaction. Our reward function

$$r_t = r_{\text{imitation}} + r_{\text{contact}} + r_{\text{regularization}} \tag{1}$$

consists of three terms discussed in the following.

### 3.3.1 Imitation Reward. The imitation reward encourages to imitate the movement of the reference motion

$$\begin{aligned} r_{\text{imitation}} = {} & w_q e^{-k_q \|q_{\text{sim}} - q_{\text{ref}}\|^2} + w_{\dot{q}} e^{-k_{\dot{q}} \|\dot{q}_{\text{sim}} - \dot{q}_{\text{ref}}\|^2} \\ & + w_p e^{-k_p \|p_{\text{sim}} - p_{\text{ref}}\|^2} + w_{\dot{p}} e^{-k_{\dot{p}} \|\dot{p}_{\text{sim}} - \dot{p}_{\text{ref}}\|^2} \\ & + w_R e^{-k_R \|log(R_{\text{sim}}, R_{\text{ref}})\|^2}, \end{aligned} \tag{2}$$

where we measure the difference between our simulated avatar and the reference motion by joint angles $q$, joint angle velocities $\dot{q}$, link positions $p$, link linear velocities $\dot{p}$, and link orientations $R$. The subscript sim, ref refers to simulation and reference, respectively and $w_q$, $w_{\dot{q}}$, $w_p$, $w_{\dot{p}}$ and $w_R$ are the corresponding weights. In theory, this reward is over-specified because joints and links are simply transferable via forward kinematics. However, this over-specified formulation often performs better for many motion reconstruction tasks, not only in physics-based but also in kinematics-only settings [Holden et al. 2017; Peng et al. 2018a; Starke et al. 2019].

### 3.3.2 Contact Reward. The contact reward encourages the simulated avatar to create the same contact state as the reference character

$$r_{\text{contact}} = w_c e^{-k_c \|c_{\text{sim}} - \hat{c}_{\text{ref}}\|}, \tag{3}$$

where the contact state $c = (c_{root}, c_{spine}, c_{feet}, c_{hands})$ is defined by multi-hot encoding which each element becomes 1 if the the link is currently in contact with any object in the scene, otherwise it becomes 0. We select feet, hands, pelvis and spine for the contact reward which are the links that take important roles in supporting the character's body and leave it open for other links. During training, the contact state of the simulated character can be noisy, so we ignore contacts of magnitudes lower than $50N$ in the gravity (i.e. upward) direction. Note that this ground truth contact state is necessary only for training. In our all experiments, we observed that the absence of this contact reward significantly degrades overall motion quality (tracking performance and the naturalness).

*3.3.3 Regularization Reward.* The regularization reward greatly improves the naturalness of the generated motions.

$$r_{\text{regularization}} = w_a e^{-k_a \|a_t\|^2} + w_s e^{-k_s \|a_t - a_{t-1}\|^2}, \quad (4)$$

where $a_t$, $a_{t-1}$ are the current and previous actions. Because the action in our simulation setup is a set of torques applied to all joints, the regularization reward prevents the controller from using excessive forces or changing forces abruptly.

## 3.4 Scene Randomization

We randomly vary the object placement during training to prevent the control policy from overfitting to the training data. Although larger object variation could further improve generalization, we randomized the scene in a small range so that the perturbed environment does not hugely contradict ground truth motion. It avoids the simulation becoming unstable during initialization due to the penetration and does not require motion editing, yet effective in generalization to unseen real-data In our experiment, the position of objects is randomly perturbed up to 8cm in all directions including height, and the orientation is randomly rotated up to 1 radian ($\approx$ 5.7 degrees) around the up-axis.

## 4 RESULTS

### 4.1 Training

We implement our system using IsaacGym [Liang et al. 2018] for physics simulation and Torch [Paszke et al. 2019] for deep learning model. In every iteration of deep reinforcement learning, 61440 transition tuples are collected from 4096 simulated environments that are running in parallel on GPUs. We use Proximal Policy Optimization (PPO) algorithm [Schulman et al. 2017] where we use clip ratio 0.2, learning rate 1.2e-4, gamma 0.97, lambda 0.95, and minibatch size 7680, respectively. We use feed-forward deep neural networks with *Tanh* activation for both actor (i.e. control policy) and critic (i.e. value function network), where the width and depth of those networks are [300, 200, 100] and [400, 400, 300, 200], respectively.

We create an in-house dataset (Table 1) that includes motions such as sitting on various objects and stepping over boxes, as well as motions without interaction such as walking, gesturing, and squatting. We captured the dataset with 5 subjects, except for some motions (50 subjects for walking, gestures, and writing in the air). We modeled our character with a single body scale according to the subject's height. We mirror all the data to double the data size. We also captured object placement and motion if it moves.

To demonstrate various examples, we used a relevant subset of the total dataset to train the policy depending on the task. We included non-interactive motions (walking, gestures, writing in the air) in all examples to enhance stable locomotion and hand tracking.

### 4.2 Evaluation

To demonstrate the performance of our framework we show a variety of environment interactions, such as sitting on objects, getting up from the floor, manipulating objects as well as mismatches of real and virtual worlds that require environment observations. An

Table 1. Training Data Composition

|                              | Time(Min) | Used Examples  |
| ---------------------------- | --------- | -------------- |
| Walking                      | 140       | all            |
| Gestures                     | 140       | all            |
| Writing in air               | 116       | all            |
| Squat                        | 4.6       | all            |
| Bench Interaction            | 5.62      | Living Room    |
| Couch Interaction            | 16.4      | Living Room    |
| Sitting in Box, Chair, Stool | 20.02     | Living Room    |
|                              |           | Tilting Chair  |
|                              |           | Rotating Chair |
| Tilting Chair                | 6.98      | Tilting Chair  |
| Rotating Chair               | 3.27      | Rotating Chair |
| Stepping over Boxes          | 10.8      | Stepping Box   |
| Sit on and get up from a floor | 10.54   | Getting up     |

overview of all the demonstrations can be seen in Figure 4. We encourage readers to watch the accompanying video as it best demonstrates the physical reasoning our policy has learned and the quality of the motion tracking.

All evaluations were conducted in unseen object arrangement and user input. For instance, in the living room example, we combined a couch, a stool, and boxes that were in different training clips in different locations. We measured the object placement and replicated them in the simulation environment. We did not apply scene randomization during the evaluation. Sensor inputs were directly fed from the real VR device. The simulated avatar is initialized in a default A-pose. A user does not have to start in a strict A-pose, and as long as they start standing, the system robustly catches up.

We also evaluate our system quantitatively by tracking error, jerk and success ratio in Table 2. When the average tracking error of the headset and controller is higher than 0.8m, we assume the character has drifted and consider it a failure. We measure the tracking error and jerk when successfully tracking the sensor and do not take them into account after the failure. We evaluate each test data multiple times by initializing in all possible frames. Success ratio [20s/30s] is computed as the ratio of number of episodes that did not fail for 20s and 30s. Success ratio [frame] indicates the average number of frames successfully tracked during 30s.

### 4.3 Deliberate environment contact

Most existing approaches are trained only on flat-ground and foot contact. Our framework is able to deliberately generate contact forces with the environment to influence its root motion to track the sensors. One example is the getting up from the ground, where the policy uses the hands to generate appropriate contact forces.

The box-stepping examples demonstrate how the policy learned to step on a box to achieve a higher head position to track the real user. If there were no simulated box, the policy would not attempt the step. On the other hand our policy also learns to avoid environment contact if this contact is expected to interfere with the tracking (Figure 5). We show a user walking on flat ground, whereas the simulation contains boxes at arbitrary positions. Since contact

Sunmin Lee, Sebastian Starke, Yuting Ye, Jungdam Won, and Alexander Winkler



(a) Living Room



(c) Tilting Chair



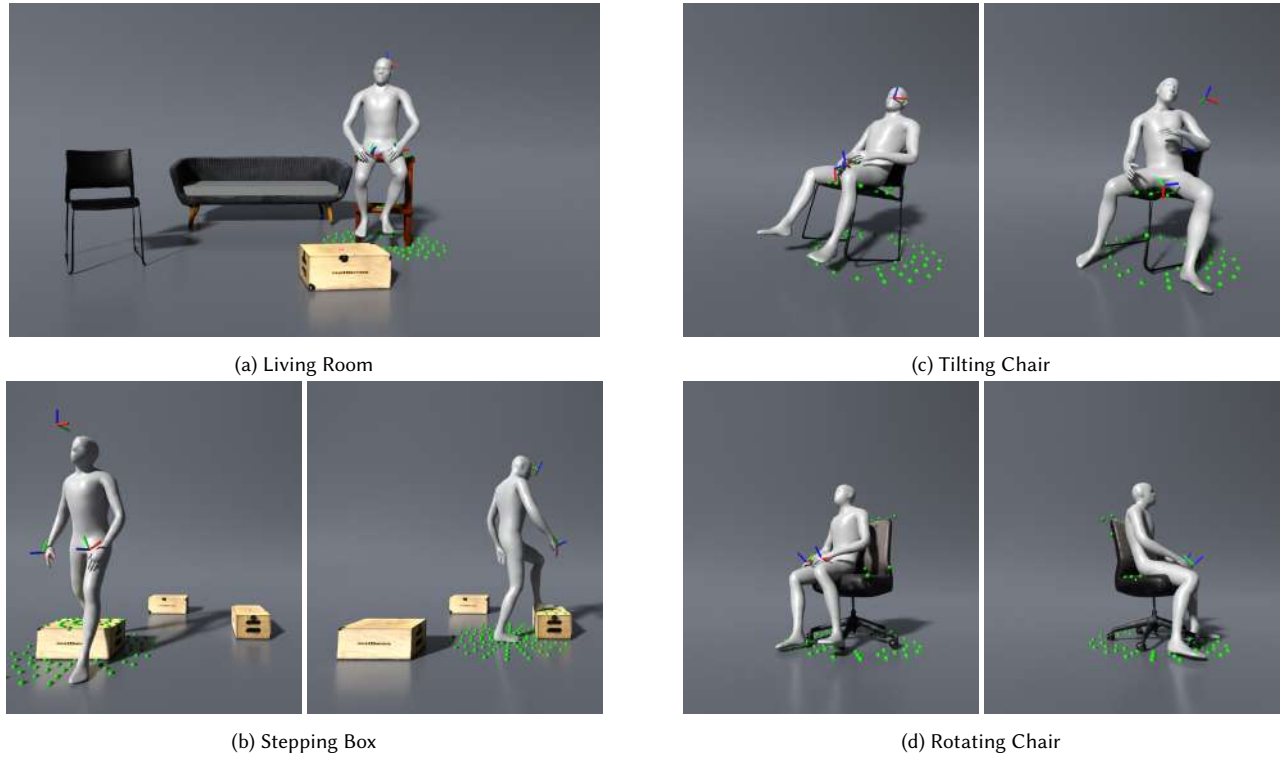(b) Stepping Box



(d) Rotating Chair

Fig. 4. Examples Snapshot. The green spheres indicate height map around the simulated character.

Table 2. Performance for various scenarios using real sensors.

| | Tracking Error Headset [cm] | Tracking Error Headset [deg] | Tracking Error Controller [cm] | Jerk [km/$s^3$] | Success ratio [20s] | Success ratio [30s] | Success ratio [frame] |
|---|---|---|---|---|---|---|---|
| Living Room | 5.6855 | 10.977 | 5.1963 | 0.3267 | 0.7689 | 0.6099 | 0.8239 |
| Stepping Box | 4.1150 | 7.1939 | 3.6289 | 0.4800 | 0.6683 | 0.4860 | 0.7422 |
| Tilting Chair | 6.4191 | 8.2919 | 5.3416 | 0.2303 | - | - | 0.7115 |
| Rotating Chair | 5.3879 | 14.0383 | 5.1379 | 0.2512 | - | - | 0.7623 |

Table 3. Impact of specific rewards, observations and training procedures on the living room example.

| | Tracking Error Headset [cm] | Tracking Error Headset [deg] | Tracking Error Controller [cm] | Jerk [km/$s^3$] | Success ratio [20s] | Success ratio [30s] | Success ratio [frame] |
|---|---|---|---|---|---|---|---|
| Ours | 5.6855 | 10.977 | 5.1963 | 0.3267 | **0.7689** | 0.6099 | **0.8239** |
| w/o Scene Observation | 5.7672 | 11.045 | 5.1422 | **0.3017** | 0.73620 | **0.6243** | 0.8185 |
| w/o Contact Reward | 5.6992 | **9.8969** | 5.4907 | 0.4418 | 0.6060 | 0.3762 | 0.7147 |
| w/o Scene Randomization | **5.5769** | 10.8448 | **4.9798** | 0.4207 | 0.6010 | 0.4145 | 0.7202 |

Table 4. Impact of future sensor observations in the living room example. 3-point uses HMD and controllers, 1-point uses headset only.

| | Tracking Error Headset [cm] | Tracking Error Headset [deg] | Tracking Error Controller [cm] | Jerk [km/$s^3$] | Success ratio [20s] | Success ratio [30s] | Success ratio [frame] |
|---|---|---|---|---|---|---|---|
| 3-point with 1s future | **5.6855** | **10.977** | **5.1963** | **0.3267** | **0.7689** | **0.6099** | **0.8239** |
| 3-point with no future | 6.8037 | 12.3596 | 5.9104 | 0.4759 | 0.5948 | 0.4845 | 0.7844 |
| 1-point with 1s future | 6.1253 | 11.6988 | 12.1518 | 0.3527 | 0.6952 | 0.4001 | 0.7049 |

with these boxes will likely decrease headset and controller tracking, the policy learns to lift the leg higher than usual to avoid them.

## 4.4 Manipulating Objects to improve tracking

Tilting or rotating a chair is a very common behavior people do in daily life but it involves complex dynamics. With our simulated approach and environment observations, our approach learns to manipulate objects in a way to best track a user's motion.

We model the revolving chair with a revolute joint between the seat and the bottom. Our training data also includes moving objects such as the office chair, so the character learns that using feet to generate a frictional contact force with the ground creates a torque on the root. This torque can be used to rotate the chair in order to better follow the users HMD and controller positions. For the tilting chair, the character understands that pushing its feet into the ground also exerts a force on the backrest of the chair causing it to tilt. And it learned that tilting the chair in such a situation is beneficial for more closely tracking the user's sensor pose.

## 4.5 Ablation on design choices

Three crucial components are required to achieve the demonstrated performance: The scene observation in an appropriate representation, the contact reward during training which includes not just feet but various other body parts, and finally the random variation of the position of objects during training. In Table 3 we quantitatively show how each component improves the performance. We notice that especially leaving out the contact reward significantly reduces the success ratio [30s]. A similar drop in performance can be seen without scene randomization. Scene observations slightly increase the success ratio [20s] in this example. For this sitting example, the policy might be able to infer sitting from the quest pose alone. The scene observations show more usefulness in the box stepping videos, where the character raises its feet to avoid contact with a virtual box. Finally, even though some design choices don't significantly impact the tracking errors, the success corresponds more to perceived visual quality and is therefore prioritized.

## 4.6 Ablation Headset only vs Headset and Controllers

We also demonstrate tracking users with scene interaction from the HMD alone, without controllers (Figure 6 middle and bottom rows). Even with such limited input, the character is still able to interact with various objects, albeit with slightly worse metrics (Table 4 bottom row). In some situations, the model utilizes the arms in different ways than the user, such as when sitting on a chair (Figure 6 (14)), in order to better stabilize the body. Due to the ambiguity of poses, the character may also produce different upper-body poses (Figure 6 (11)), e.g. where the arms are hovering over the sofa while the real user is resting the arms on it.

## 4.7 Ablation real-time vs future

Our framework can be used to track users in real-time, as is shown in the video. However, the quality of the generated poses degrades compared to if the policy has access to one future observation. This degradation is reflected in Table 4 in the lower success ratio, as well as higher tracking errors. Nonetheless, as can be seen in the

video, qualitatively the motions look reasonable, and the learned environment interaction of sitting and getting up is still possible. Access to future information will likely provide the biggest benefit for motions that require future planning, like jumping over a gap. However, to track day-to-day movements, real-time tracking provides acceptable results.

## 5 CONCLUSION AND FUTURE WORK

We demonstrated a physics-based motion tracking framework from sparse sensors that actively utilizes physical interactions with the environment to generate natural-looking motions. The policies were trained using Reinforcement Learning, where our non-trivial combination of the components in the system design (e.g. environment observation, contact rewards, and scene randomization) enabled the policy to learn accurate and robust control strategies. We believe these are some of the highest-quality results achieved for motion tracking from sparse sensors with scene interaction.

Although we showed the solid performance of our motion system over a variety of scenarios, there exist several limitations that we want to address in the future. First, each type of interaction requires a specialized motion tracker in our current system. It would be ideal if a single tracker can be learned, which covers wider repertoires. This might need a more complex neural network model such as a mixture-of-experts [Won et al. 2020] or longer training time with larger datasets in general.

We also had a difficulty in reliably producing motions like getting up from the floor or generally more complex interaction motions. Since we don't use any artificial root forces, such behaviors that require careful coordination of contacts still seem difficult to learn (getting up from the floor is also a difficult motion to perform for older people). The simulated avatar used in our system can also fail (i.e. losing balance) as other physics-based characters, which incurs the failure of motion tracking of the user. Automatic failure detection followed by applying external force can be a reasonable compromise, however, physical realism would be degraded.

Another promising future direction would be extending our system for unknown scenes that include dynamically moving objects. In this case, the ability to infer physical properties (e.g. inertia, fiction coefficients) of surrounding objects in a scene would become more critical for successful motion tracking. Online system identification [Feng et al. 2022; Yu et al. 2017] could be combined as a part of the system.

## REFERENCES

Sheldon Andrews, Ivan Huerta, Taku Komura, Leonid Sigal, and Kenny Mitchell. 2016. Real-Time Physics-Based Motion Capture with Sparse Sensors. In *Proceedings of*

the 13th European Conference on Visual Media Production (CVMP 2016) (CVMP 2016). Article 5.

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: Data-driven Responsive Control of Physics-based Characters. *ACM Trans. Graph.* 38, 6, Article 206 (2019). http://doi.acm.org/10.1145/3355089.3356536

Nuttapong Chentanez, Matthias Müller, Miles Macklin, Viktor Makoviychuk, and Stefan Jeschke. 2018. Physics-based motion capture imitation with deep reinforcement learning. In *Motion, Interaction and Games, MIG 2018*. ACM, 1:1–1:10. https://doi.org/10.1145/3274247.3274506

Gilbert Feng, Hongbo Zhang, Zhongyu Li, Xue Bin Peng, Bhuvan Basireddy, Linzhu Yue, Zhitao Song, Lizhi Yang, Yunhui Liu, Koushil Sreenath, et al. 2022. Genloco: Generalized locomotion controllers for quadrupedal robots. *arXiv preprint arXiv:2209.05309* (2022).

Levi Fussell, Kevin Bergamin, and Daniel Holden. 2021. SuperTrack: Motion Tracking for Physically Simulated Characters using Supervised Learning. *ACM Trans. Graph.* 40, 6, Article 197 (2021). https://dl.acm.org/doi/10.1145/3478513.3480527

Vladimir Guzov, Aymen Mir, Torsten Sattler, and Gerard Pons-Moll. 2021. Human poseitioning system (hps): 3d human pose estimation and self-localization in large scenes from body-mounted sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4318–4329.

Joseph Henry, Hubert P. H. Shum, and Taku Komura. 2012. Environment-Aware Real-Time Crowd Control. In *Proceedings of the 11th ACM SIGGRAPH / Eurographics Conference on Computer Animation (EUROSCA'12)*.

Edmond S. L. Ho, Taku Komura, and Chiew-Lan Tai. 2010. Spatial Relationship Preserving Character Motion Adaptation. In *ACM SIGGRAPH 2010 Papers (SIGGRAPH '10)*. Article 33, 8 pages. https://doi.org/10.1145/1833349.1778770

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-Functioned Neural Networks for Character Control. *ACM Trans. Graph.* 36, 4 (jul 2017). https://doi.org/10.1145/3072959.3073663

Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. 2018. Deep Inertial Poser: Learning to Reconstruct Human Pose from Sparse Inertial Measurements in Real Time. *ACM Transactions on Graphics* 37, 6 (12 2018).

Jiaxi Jiang, Paul Streli, Huajian Qiu, Andreas Fender, Larissa Laich, Patrick Snape, and Christian Holz. 2022. AvatarPoser: Articulated Full-Body Pose Tracking from Sparse Motion Sensing. In *Proceedings of European Conference on Computer Vision*. Springer.

Manuel Kaufmann, Yi Zhao, Chengcheng Tang, Lingling Tao, Christopher Twigg, Jie Song, Robert Wang, and Otmar Hilliges. 2021. EM-POSE: 3D Human Pose Estimation from Sparse Electromagnetic Trackers. In *International Conference on Computer Vision (ICCV)*.

Manmyung Kim, Kyunglyul Hyun, Jongmin Kim, and Jehee Lee. 2009. Synchronized Multi-Character Motion Editing. *ACM Trans. Graph.* 28, 3 (2009). https://doi.org/10.1145/1531326.1531385

Kang Hoon Lee, Myung Geol Choi, and Jehee Lee. 2006. Motion Patches: Building Blocks for Virtual Environments Annotated with Motion Data. In *ACM SIGGRAPH 2006 Papers (SIGGRAPH '06)*. 898–906. https://doi.org/10.1145/1179352.1141972

Jacky Liang, Viktor Makoviychuk, Ankur Handa, Nuttapong Chentanez, Miles Macklin, and Dieter Fox. 2018. GPU-Accelerated Robotic Simulation for Distributed Reinforcement Learning. *CoRR* abs/1810.05762 (2018). arXiv:1810.05762 http://arxiv.org/abs/1810.05762

Huajun Liu, Xiaolin Wei, Jinxiang Chai, Inwoo Ha, and Taehyun Rhee. 2011. Realtime Human Motion Control with a Small Number of Inertial Sensors. In *Symposium on Interactive 3D Graphics and Games (I3D '11)*. 133–140. https://doi.org/10.1145/1944745.1944768

Zhengyi Luo, Ryo Hachiuma, Ye Yuan, and Kris Kitani. 2021. Dynamics-regulated kinematic policy for egocentric pose estimation. *Advances in Neural Information Processing Systems* 34 (2021).

Zhengyi Luo, Shun Iwase, Ye Yuan, and Kris Kitani. 2022. Embodied Scene-aware Human Pose Estimation. *Advances in Neural Information Processing Systems* (2022).

Josh Merel, Yuval Tassa, Dhruva TB, Sriram Srinivasan, Jay Lemmon, Ziyu Wang, Greg Wayne, and Nicolas Heess. 2017. Learning human behaviors from motion capture by adversarial imitation. *CoRR* abs/1707.02201 (2017).

Deepak Nagaraj, Erik Schake, Patrick Leiner, and Dirk Werth. 2020. An RNN-ensemble approach for real time human pose estimation from sparse IMUs. In *Proceedings of the 3rd International Conference on Applications of Intelligent Systems*. 1–6.

Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. 2019. Learning Predict-and-simulate Policies from Unorganized Human Motion Data. *ACM Trans. Graph.* 38, 6, Article 205 (2019). http://doi.acm.org/10.1145/3355089.3356501

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. 8024–8035.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. 2018a. DeepMimic: Example-guided Deep Reinforcement Learning of Physics-based Character

Skills. *ACM Trans. Graph.* 37, 4, Article 143 (2018). http://doi.acm.org/10.1145/3197517.3201311

Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. 2018b. SFV: Reinforcement Learning of Physical Skills from Videos. *ACM Trans. Graph.* 37, 6, Article 178 (Nov. 2018).

Jose Luis Ponton, Haoran Yun, Carlos Andujar, and Nuria Pelechano. 2022. Combining Motion Matching and Orientation Prediction to Animate Avatars for Consumer-Grade VR Devices. *Computer Graphics Forum* (Sept. 2022). https://doi.org/10.1111/cgf.14628

Qaiser Riaz, Guanhong Tao, Björn Krüger, and Andreas Weber. 2015. Motion Reconstruction Using Very Few Accelerometers and Ground Contacts. *Graph. Models* 79, C (may 2015), 23–38.

Alla Safonova and Jessica K. Hodgins. 2007. Construction and Optimal Search of Interpolated Motion Graphs. *ACM Trans. Graph.* 26, 3 (jul 2007). https://doi.org/10.1145/1276377.1276510

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 http://arxiv.org/abs/1707.06347

Hubert P. H. Shum, Taku Komura, Masashi Shiraishi, and Shuntaro Yamazaki. 2008. Interaction Patches for Multi-Character Animation. *ACM Trans. Graph.* 27, 5 (dec 2008). https://doi.org/10.1145/1409060.1409067

Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. 2019. Neural State Machine for Character-Scene Interactions. *ACM Trans. Graph.* 38, 6, Article 209 (nov 2019). https://doi.org/10.1145/3355089.3356505

Jochen Tautges, Arno Zinke, Björn Krüger, Jan Baumann, Andreas Weber, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bernd Eberhardt. 2011. Motion Reconstruction Using Sparse Accelerometer Data. *ACM Trans. Graph.* 30, 3 (2011).

Timo von Marcard, Bodo Rosenhahn, Michael Black, and Gerard Pons-Moll. 2017. Sparse Inertial Poser: Automatic 3D Human Pose Estimation from Sparse IMUs. *Computer Graphics Forum 36(2), Proceedings of the 38th Annual Conference of the European Association for Computer Graphics (Eurographics)* (2017), 349–360.

Alexander Winkler, Jungdam Won, and Yuting Ye. 2022. QuestSim: Human Motion Tracking from Sparse Sensors with Simulated Avatars. In *SIGGRAPH Asia 2022 Conference Papers (SA '22)*. https://doi.org/10.1145/3550469.3555411

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. 2020. A scalable approach to control diverse behaviors for physically simulated characters. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 33–1.

Jungdam Won, Kyungho Lee, Carol O'Sullivan, Jessica K. Hodgins, and Jehee Lee. 2014. Generating and Ranking Diverse Multi-Character Interactions. *ACM Trans. Graph.* 33, 6 (2014). https://doi.org/10.1145/2661229.2661271

Yongjing Ye, Libin Liu, Lei Hu, and Shihong Xia. 2022. Neural3Points: Learning to Generate Physically Realistic Full-body Motion for Virtual Reality Users. *Computer Graphics Forum* (Sept. 2022).

Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. 2022. Physical Inertial Poser (PIP): Physics-aware Real-time Human Motion Tracking from Sparse Inertial Sensors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xinyu Yi, Yuxiao Zhou, and Feng Xu. 2021. TransPose: Real-time 3D Human Translation and Pose Estimation with Six Inertial Sensors. *ACM Transactions on Graphics* 40, 4 (8 2021).

Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. 2017. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453* (2017).

Ye Yuan and Kris Kitani. 2019. Ego-Pose Estimation and Forecasting as Real-Time PD Control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 10082–10092.

Ye Yuan and Kris Kitani. 2020. Residual Force Control for Agile Human Behavior Imitation and Extended Motion Synthesis. In *Advances in Neural Information Processing Systems*.

Ye Yuan, Shih-En Wei, Tomas Simon, Kris Kitani, and Jason Saragih. 2021. SimPoE: Simulated Character Control for 3D Human Pose Estimation.

He Zhang, Yuting Ye, Takaaki Shiratori, and Taku Komura. 2021. ManipNet: Neural Manipulation Synthesis with a Hand-Object Spatial Representation. *ACM Trans. Graph.* 40, 4, Article 121 (jul 2021), 14 pages. https://doi.org/10.1145/3450626.3459830
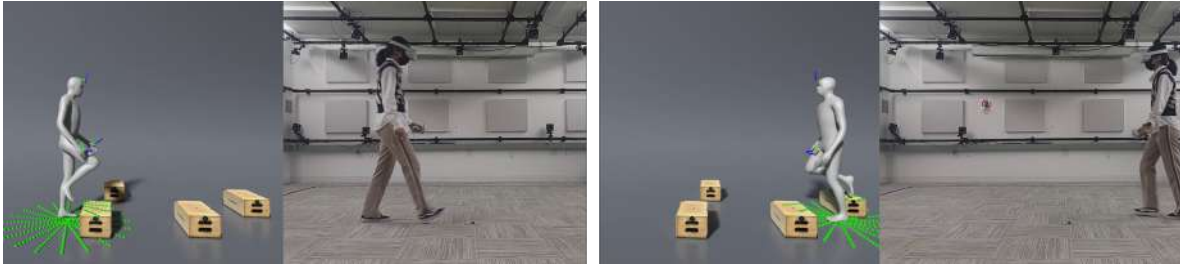
Fig. 5. The user walks on flat ground, whereas we place the virtual boxes in the simulation environment. The policy learns to observe its surrounding scene by height map and to lift the leg higher to avoid obstacles while tracking the user's sensor data.
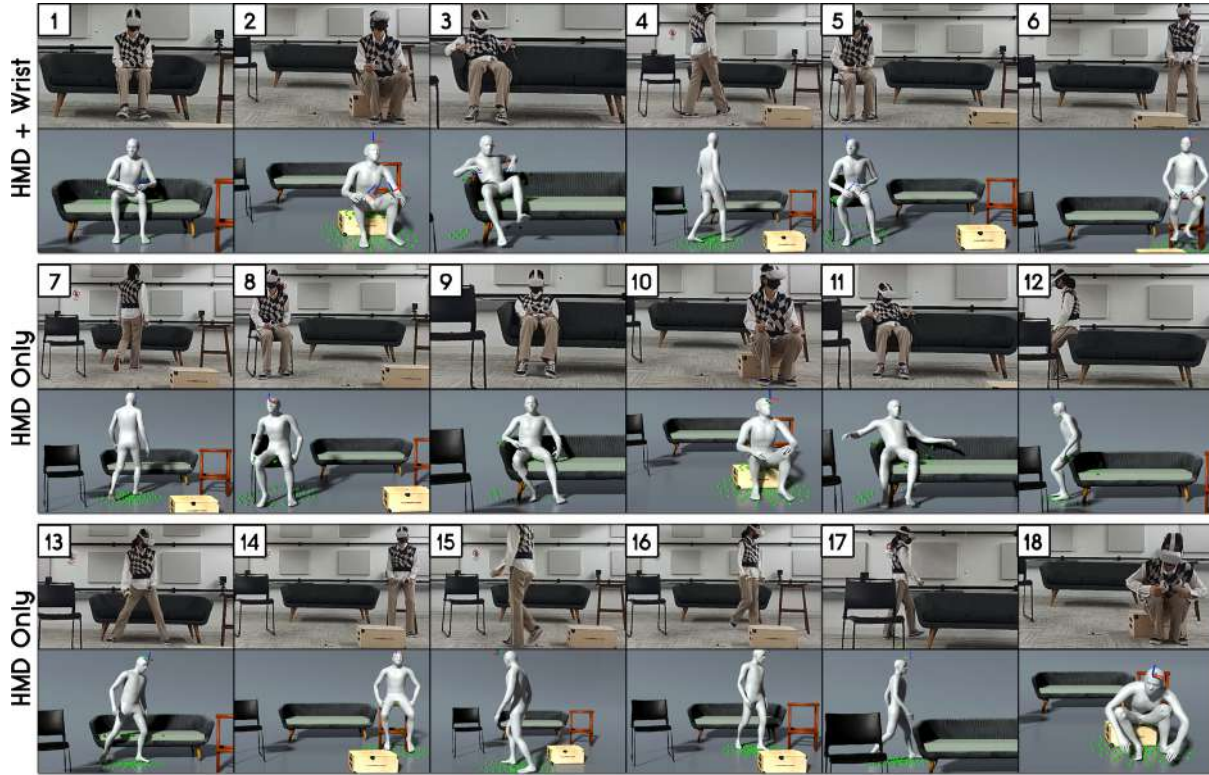


Fig. 6. Collection of real user poses and simulated avatar poses using HMD and wrist trackers (top row) and only HMD tracker (middle and bottom rows).