

Ground Reaction Inertial Poser: Physics-based Human Motion Capture from Sparse IMUs and Insole Pressure Sensors

Supplementary Material

6. Supplementary Video

The supplementary video presents an overview of the GRIP method described in Sec. 3, including its input data, intermediate network outputs, a visualization of the PRISM dataset, qualitative comparisons with baseline methods (Sec. 4.2), and representative failure cases.

We first visualize the sparse wearable sensor inputs to KinematicsNet and DynamicsNet, including IMU measurements from the wrists and feet and plantar pressure signals from the insole sensors. We then show intermediate outputs of KinematicsNet, including leaf-joint positions, full-body joint positions and joint angles, and key-joint velocities, followed by physically plausible whole-body motion reconstructed by DynamicsNet in the Isaac Gym simulation environment. The video also demonstrates the fall-recovery mechanism using a HistoryBuffer that temporarily switches to kinematic predictions after fall detection to safely resume simulation. In addition, we illustrate the sensor configuration used to construct the PRISM dataset and visualize the calibrated and synchronized multimodal sensor data and motion annotations.

Finally, we present qualitative comparisons with baseline methods (PIP [60], GlobalPose [63], MobilePoser [57], and FoRM [65]), highlighting differences in global pose accuracy, object interaction, and foot-ground contact naturalness, followed by representative failure cases such as instability during object interaction and challenging motions.

7. PRISM Dataset Details

7.1. Data Acquisition

In constructing the PRISM dataset, it is necessary to integrate the various sensor measurements obtained from multiple wearable devices and an optical MoCap system into a unified global coordinate frame and to synchronize all recordings under a consistent temporal reference. This section describes the sensing devices and data acquisition protocol used during collection, the coordinate-frame calibration procedures for each device, and the temporal synchronization process.

Sensors and Capture Protocol. Data were collected using an optical motion-capture system (OptiTrack), IMU-equipped smartwatches (Apple Watch), IMU-equipped insole-type pressure sensors (Moticon OpenGo), a head-mounted multimodal sensing device (Meta Aria Glass), and IMU devices (Xsens MTw). OptiTrack provides high-

precision 3D marker trajectories, and the captured marker data were fitted with Mosh++ [33] to reconstruct SMPL meshes and high-fidelity body motion. OpenGo includes a 6DoF IMU (accelerometer and gyroscope) and sixteen pressure sensors on each foot, enabling the measurement of vertical forces, pressure distribution, and center of pressure (CoP). The Apple Watch includes a 9DoF IMU (accelerometer, gyroscope, and magnetometer). Aria Glass includes two 9DoF IMUs and simultaneously records forward-facing video and two additional SLAM camera views. Using these videos and IMU data, the Project Aria Machine Perception Service (MPS) was used to obtain environmental point clouds and device trajectories. Furthermore, Xsens MTw units (9DoF IMUs) were attached to the knees and pelvis.

At the beginning of each recording, the subject maintained a T-pose for approximately three seconds while wearing all sensors. This static posture is used to determine the relative orientations of each IMU with respect to the SMPL body model. Starting each capture with the same protocol also reduces ambiguity in the initial pose during real-world operation. After the T-pose, the subject returned to an upright stance and performed three consecutive vertical jumps. The characteristic acceleration peaks observed during the jump motions are later used as robust temporal landmarks for precise synchronization across all sensor streams. The subject then performed the instructed motion sequence.

IMU Calibration. IMU data were collected from eight body locations using four types of wearable devices: Apple Watch, OpenGo, Aria Glass, and Xsens MTw. Because each device defines its measurements in its own coordinate system, all IMU observations must be transformed into a unified global coordinate frame. We use the coordinate system of the SMPL motion reconstructed from the optical MoCap data using Mosh++ as the global coordinate frame, adopting a z-up convention. At the beginning of each recording session, the subject holds a T-pose facing the positive y -axis of the global frame. This protocol enables consistent initialization of all IMU orientations and establishes the reference posture used for the subsequent calibration steps. The following sections describe how raw IMU orientations and accelerations are aligned with the global coordinate frame and further mapped to the corresponding SMPL joint frames.

We first describe the calibration procedure for the Apple Watch and Xsens IMUs, following the notation used in previous works [27, 29]. The relationship between the joint-frame orientation and the raw IMU measurements is

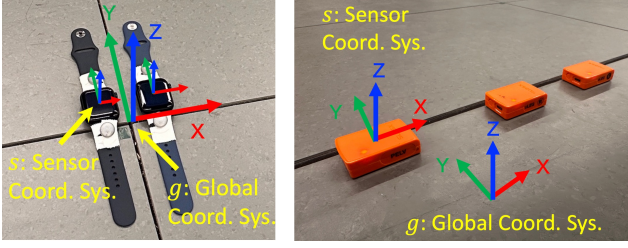


Figure 6. Calibration setup for initial orientation. Left: Apple Watch. Right: Xsens MTw sensors.

expressed as follows:

$$\mathbf{R}_g^j = \mathbf{R}_g^r \mathbf{R}_r^s \mathbf{R}_s^j, \quad \mathbf{a}_g = \mathbf{R}_g^r \mathbf{R}_r^s \mathbf{a}_s, \quad (1)$$

where r denotes the raw IMU reference frame, and g denotes the global coordinate frame. To obtain the joint rotations and accelerations in the global coordinate system at each IMU attachment site, the IMU orientation must be converted into the corresponding joint orientation \mathbf{R}_g^j , and the raw accelerations defined in the sensor frame must likewise be transformed into the global frame \mathbf{a}_g , where \mathbf{R}_r^s denotes the raw IMU orientation expressed in the reference sensor frame. Thus, we first estimate the orientation of this reference frame relative to the global frame. Following prior work [27, 29], before being attached to the user, each IMU sensor is placed on the floor in a specified orientation aligned with the global frame and kept still for 3 seconds (Fig. 6). By averaging the static IMU signal, we obtain a stable estimate of its orientation at that moment, where $\mathbf{R}_r^s = \mathbf{R}_g^s$ holds. The inverse $(\mathbf{R}_r^g)^{-1}$ is then applied to compute \mathbf{R}_g^r . In the next step, we compute the joint orientation \mathbf{R}_g^j by combining the IMU-to-global rotation with the joint-to-sensor rotation as $\mathbf{R}_g^s \mathbf{R}_s^j$. To determine the joint-to-sensor transform \mathbf{R}_s^j , the subject takes a T-pose while facing the global y -axis. The relationship is given by

$$\mathbf{R}_{sT}^{jT} = (\mathbf{R}_r^{sT})^{-1} \mathbf{R}_r^g \mathbf{R}_g^{jT} \quad (2)$$

where the subscript T indicates values obtained during the T-pose calibration. Assuming the IMU remains rigidly attached to the body during capture, the relative transform \mathbf{R}_s^j is constant, allowing us to treat $\mathbf{R}_{sT}^{jT} = \mathbf{R}_s^j$. Because the joint rotation in the T-pose, \mathbf{R}_g^{jT} , is determined from the SMPL model, the T-pose raw IMU measurements \mathbf{R}_r^{sT} provide a reliable estimate of \mathbf{R}_s^j .

Next, we describe the calibration procedure for the OpenGo device. Unlike the other IMU devices, the sensor embedded in OpenGo is a 6DoF IMU without a magnetometer, and therefore cannot utilize magnetic-field information for orientation estimation. Following prior work [2], we use the Versatile Quaternion-based Filter (VQF) [28] to estimate the sensor orientation based on the gravity vector. Let the raw sensor coordinate frame be s and the SMPL

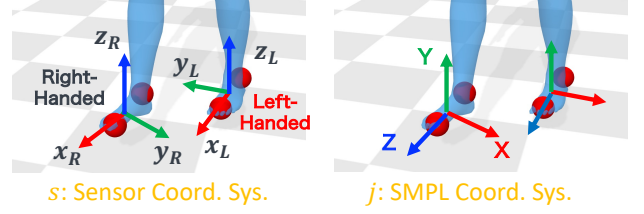


Figure 7. Sensor and global coordinate frames for the OpenGo devices, illustrating the handedness difference between sensors.

foot-joint coordinate frame be j . Because the left and right OpenGo sensors follow different handedness conventions (Fig. 7), axis-sign corrections are applied to unify their coordinate systems. The measured angular velocity ω_s and acceleration \mathbf{a}_s are mapped into the joint frame using the fixed sensor-to-joint rotation \mathbf{R}_j^s as

$$\begin{aligned} \omega_{j,\text{Right}} &= \mathbf{R}_j^s \omega_s, \\ \omega_{j,\text{Left}} &= \mathbf{R}_j^s (\omega_s \odot [-1, 1, -1]), \\ \mathbf{a}_{j,\text{Right}} &= \mathbf{R}_j^s \mathbf{a}_s, \\ \mathbf{a}_{j,\text{Left}} &= \mathbf{R}_j^s (\mathbf{a}_s \odot [1, -1, 1]), \end{aligned} \quad (3)$$

where \odot denotes the element-wise product. VQF provides the foot orientation in its gravity-aligned frame, denoted as $\mathbf{R}_{\text{vqf}}^j$. To express this orientation in the global coordinate frame, we align it with the SMPL foot-joint orientation in the T-pose. Let \mathbf{R}_g^{jT} be the global SMPL foot-joint rotation in the T-pose, and let $\mathbf{R}_{\text{vqf}}^{jT}$ be the VQF-estimated sensor orientation at the same moment. The global joint-frame orientation is obtained by directly aligning the VQF-estimated sensor orientation with the calibrated joint orientation in the T-pose:

$$\mathbf{R}_g^j = \mathbf{R}_g^{jT} \left(\mathbf{R}_{\text{vqf}}^{jT} \right)^{-1} \mathbf{R}_{\text{vqf}}^j. \quad (4)$$

The global acceleration is obtained by rotating the raw sensor acceleration using the joint's global orientation:

$$\mathbf{a}_g = \mathbf{R}_g^j \mathbf{a}_j. \quad (5)$$

Finally, we describe the calibration procedure for the Aria Glass. Aria Glass is equipped with two IMUs (1000 Hz and 800 Hz), and in our dataset we use the 1000 Hz unit, which is downsampled to 100 Hz to match the other sensors. The Aria system defines several coordinate frames, including the IMU frame i , the device frame d , the SLAM world frame w estimated by MPS, and the Central Pupil Frame (CPF) c , which is used to approximate the effective IMU attachment location. The global MoCap coordinate system is denoted as g . MPS provides the device orientation as \mathbf{R}_w^d in the SLAM coordinate system, and factory calibration further provides the fixed extrinsic rotations \mathbf{R}_d^i (IMU to device) and \mathbf{R}_i^c (IMU to CPF). The transformation from

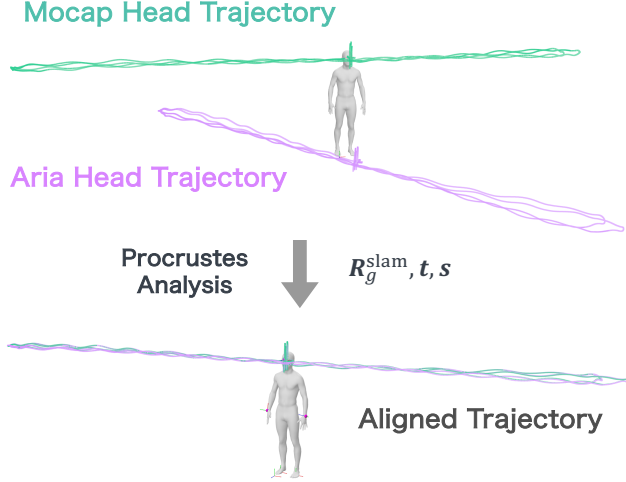


Figure 8. Alignment of the Aria Glass SLAM trajectory to the MoCap-based head trajectory using Procrustes analysis.

the SLAM frame to the global MoCap frame, R_g^w , is estimated by leveraging the high accuracy of the SLAM trajectory produced by MPS. Specifically, as illustrated in Fig. 8, we align the Aria Glass trajectory to the MoCap trajectory of the SMPL-based CPF (central pupil) position across the entire sequence using Procrustes analysis, which yields the rotation R_g^w , translation t , and scale s . Using these transformations, the CPF orientation and IMU accelerations can be expressed in the global coordinate system as

$$\begin{aligned} R_g^c &= R_g^w R_w^d R_d^i R_i^c, \\ a_g &= R_g^w R_w^d R_d^i a_i, \end{aligned} \quad (6)$$

where a_i denotes the raw acceleration measured in the IMU frame. These quantities are used as the head IMU’s global orientation and global acceleration in the dataset.

Time Synchronization. All sensors used in this study (OptiTrack, Apple Watch, OpenGo, Aria Glass, and Xsens MTw) record data at 100 Hz. Since the measurements on each device are started manually, their recording start times are not aligned. Therefore, a synchronization procedure is required to align all sensor streams to a common temporal reference. In our dataset, we use the vertical acceleration peaks generated during the three vertical jumps performed immediately after the T-pose as temporal landmarks for synchronization. As described in the previous section, all IMUs are transformed into a unified global coordinate system through calibration. This allows the vertical acceleration components derived from each IMU’s measurements to be compared directly. Meanwhile, the SMPL mesh obtained from MoCap provides only positional information; therefore, the corresponding reference acceleration is computed by applying finite differences to the time-series vertex

positions at each IMU attachment site, converting positions to velocities and subsequently to accelerations.

For synchronization, the vertical acceleration waveform of each IMU is slid over the reference SMPL-derived vertical acceleration waveform using a sliding-window approach, and similarity (cross-correlation) is evaluated at each offset. The temporal offset that achieves the highest similarity is taken as the time shift for that IMU stream, aligning all sensors to a common time axis. Fig. 9 shows an example of the vertical acceleration signals computed from MoCap-based SMPL at the eight IMU attachment locations (wrists, feet, knees, waist, and head), alongside those from the corresponding IMU sensors, illustrating how the characteristic peaks from the three jumps serve as reliable synchronization cues.

7.2. Dataset Composition.

The PRISM dataset consists of 1,275 sequences divided into 1,021 for training and 254 for testing, each 10 seconds long (1,000 frames). Data were collected from six subjects (four male and two female), totaling approximately 12,750 seconds (≈ 3.5 hours) of motion. Each sequence includes synchronized multimodal signals, such as MoCap-based reference poses, IMU and insole pressure data, environmental point clouds, camera trajectories, and object models for interaction.

The dataset covers a wide range of motion categories. First, Basic Locomotion includes walking along predefined paths and random trajectories (Walking Square / Walking Random), as well as jogging along predefined or random paths (Jogging Square / Jogging Random). Next, Slow Movements consist of stretching, two-foot and one-foot balancing, squatting, and lunge walking. Furthermore, Fast Sports Actions include whole-body athletic movements such as sidestepping, forward jumping, golf, baseball, tennis, soccer, and direction-switching movements. Finally, Interaction Behaviors involve movements that include physical interaction with objects, such as stepping onto boxes or stairs (Stepping Boxes / Stepping Stair), sitting on boxes (Sitting Boxes), and object carrying.

7.3. Comparison with Existing Datasets.

In recent years, several foot-sensing motion datasets leveraging insole pressure sensors have been released, contributing significantly to the understanding of human motion (Table 5). However, existing datasets exhibit inherent trade-offs across aspects such as IMU availability, environmental information, motion diversity, and annotation accuracy. As a result, none of them is sufficient for comprehensively modeling physically consistent full-body motion together with interactions involving the environment and objects.

PSU-TMM100 [45] is a dataset that combines insole-type pressure sensors with optical motion capture, con-

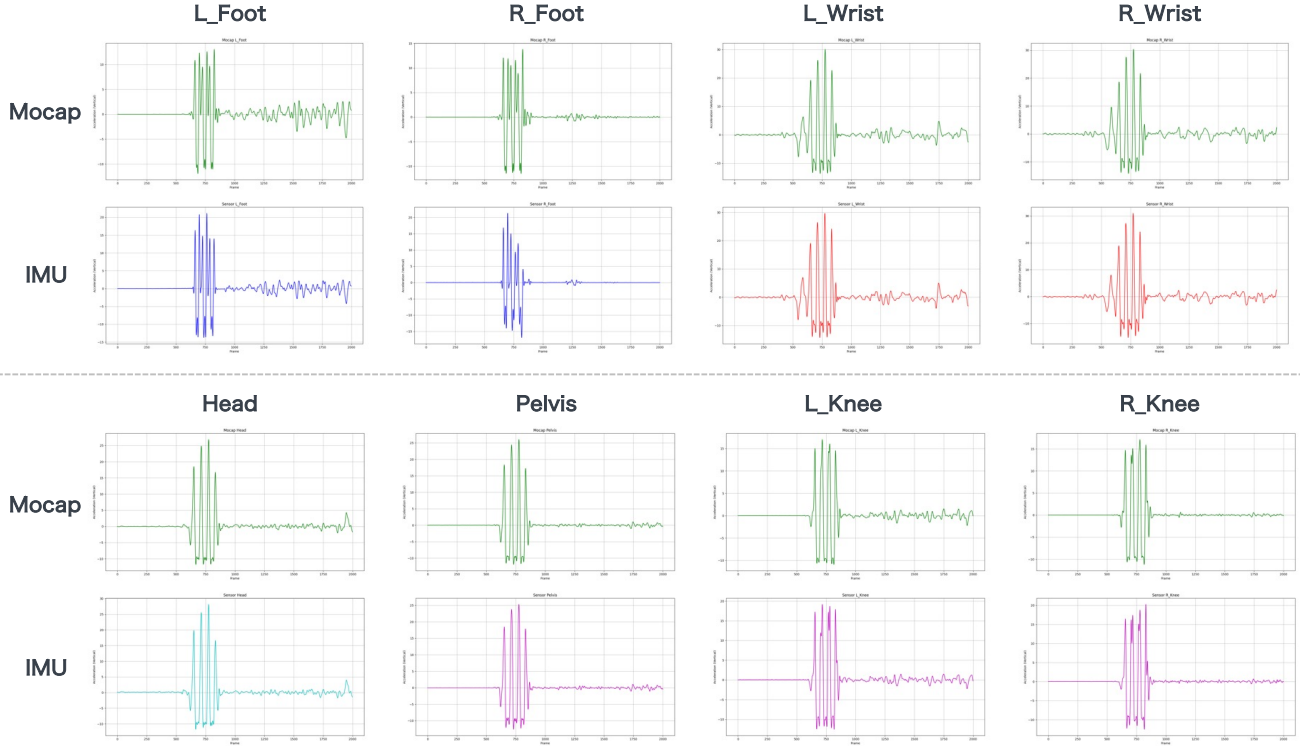


Figure 9. Vertical acceleration signals from MoCap data (top row) and from IMU sensors (bottom row) at the same set of body-mounted locations. The characteristic jump-induced peaks provide reliable cues for temporal synchronization across sensor streams.

Table 5. Comparison of human motion datasets utilizing insole-type foot pressure sensors.

Dataset	Year	Foot IMUs	Other IMUs	Env. Models	Motion Diversity	MoCap Method
PSU-TMM100 [45]	2020	No	No	No	Low (Tai Chi)	Optical
UnderPressure [41]	2022	Yes (2)	No	No	Medium (Locomotion)	Inertial
MMVP [68]	2024	No	No	No	Medium (Exercise)	RGBD
HAMPI [65]	2025	Yes (4)	No	No	High (Diverse Action)	Inertial + RGB
PRISM (Ours)	2025	Yes (2)	Yes (6)	Yes	High (Diverse Action)	Optical

sisting of Tai Chi movements. While it provides plantar pressure distributions, it does not include any IMU measurements. UnderPressure [41] contains long recordings of basic locomotion such as walking, jogging, and skipping, and is characterized by the inclusion of both plantar pressure sensors and foot IMUs. It also includes a limited set of environment-interaction behaviors, such as stair ascent/descent and stepping onto objects, but neither environment geometry nor object models are provided. MMVP [68] records RGB-D video of body movements together with plantar pressure measurements, and features several fast exercise motions, but does not provide IMU data. HAMPI [65], the most recent foot-sensing dataset, includes plantar pressure, foot IMUs, and SMPL motions generated by combining inertial motion capture with monocular

video-based pose estimation. Although it covers a wide variety of actions, the IMUs are restricted to the feet, making the dataset unsuitable for studies that require wrist or full-body IMUs. Moreover, some motion-capture errors remain in the form of unnatural body tilts or elbow twisting, making the annotations not fully reliable for physics-based approaches such as GRIP.

Taken together, existing datasets do not simultaneously provide (i) detailed foot-sensing measurements, (ii) full-body inertial information, (iii) environment interaction data, (iv) a broad range of motions, and (v) accurate full-body mesh annotations within a single framework. PRISM addresses these gaps by combining SMPL motion reconstructed from optical MoCap with eight IMUs placed at common attachment sites (knees, pelvis, wrists, and head)

in addition to the feet, along with insole-type pressure sensors. PRISM also includes environmental point clouds and object models, and covers a wide variety of motions ranging from basic locomotion to slow movements, fast sports activities, and interactions with objects. Through this combination, PRISM serves as a unified dataset suitable for integrated analysis of full-body motion and foot-ground interaction dynamics in everyday scenarios, supporting research that requires physically consistent motion estimation.

8. Implementation Details

Reward Design in DynamicsNet. The reward function follows the Perpetual Humanoid Control (PHC) framework [37], and consists of three components: an Adversarial Motion Prior (AMP) [42], an imitation reward, and an energy penalty [12]. The total reward is formulated as:

$$r_t = 0.5r_t^{\text{amp}} + 0.5r_t^{\text{imit}} + r_t^{\text{energy}}, \quad (7)$$

where r_t denotes the total reward at timestep t .

The AMP reward r_t^{amp} encourages realistic and natural motions by training a discriminator D to distinguish generated motions from real human motion. The discriminator takes as input the self-observation $\mathbf{O}_t^{\text{self}}$, which consists of a temporally ordered sequence of motion features over W consecutive frames, maintained as a sliding window that is updated at every timestep. The AMP reward is computed as:

$$r_t^{\text{amp}} = -\log(1 - \sigma(D(\mathbf{O}_t^{\text{self}}))), \quad (8)$$

where $\sigma(\cdot)$ denotes the sigmoid function and $D(\mathbf{O}_t^{\text{self}})$ represents the discriminator logit output for the self-observation $\mathbf{O}_t^{\text{self}}$. This reward formulation encourages the policy to generate motions that the discriminator classifies as real, i.e., $\sigma(D(\mathbf{O}_t^{\text{self}})) \rightarrow 1$, thereby maximizing r_t^{amp} . The discriminator is trained using a standard Generative Adversarial Network (GAN) objective with gradient penalty regularization:

$$\begin{aligned} \mathcal{L}_D = & \mathbb{E}_{\mathbf{O}_\pi^{\text{self}}} [\log(1 - \sigma(D(\mathbf{O}_\pi^{\text{self}})))] \\ & + \mathbb{E}_{\mathbf{O}_{\text{ref}}^{\text{self}}} [\log \sigma(D(\mathbf{O}_{\text{ref}}^{\text{self}}))] \\ & + \lambda_{\text{gp}} \mathbb{E}_{\mathbf{O}_{\text{ref}}^{\text{self}}} [\|\nabla_{\mathbf{O}} D(\mathbf{O}_{\text{ref}}^{\text{self}})\|^2], \end{aligned} \quad (9)$$

where $\mathbf{O}_\pi^{\text{self}}$ denotes self-observations generated by the policy, $\mathbf{O}_{\text{ref}}^{\text{self}}$ denotes self-observations from the reference motion dataset, λ_{gp} is the gradient penalty coefficient, and $\nabla_{\mathbf{O}} D(\cdot)$ represents the gradient of the discriminator output with respect to the input observation.

The imitation reward measures the consistency between the simulated and reference motion states. It is computed based on the differences in joint positions, rotations, linear

velocities, and angular velocities:

$$\begin{aligned} r_t^{\text{imit}} = & w_p e^{-100\|\hat{\mathbf{p}}_t - \mathbf{p}_t\|} + w_\theta e^{-100\|\hat{\boldsymbol{\theta}}_t \ominus \boldsymbol{\theta}_t\|} \\ & + w_v e^{-10\|\hat{\mathbf{v}}_t - \mathbf{v}_t\|} + w_\omega e^{-0.1\|\hat{\boldsymbol{\omega}}_t - \boldsymbol{\omega}_t\|}, \end{aligned} \quad (10)$$

where w_p , w_θ , w_v , and w_ω are weighting coefficients for each term, and \ominus denotes the rotation difference between the reference and simulated orientations. The hatted quantities (e.g., $\hat{\mathbf{p}}_t$, $\hat{\boldsymbol{\theta}}_t$) represent reference motion values, while the unaccented ones correspond to simulated states.

In addition, the energy penalty r_t^{energy} is computed as the negative mechanical power of the humanoid, obtained from the element-wise product of joint torques $\boldsymbol{\tau}_t$ and joint angular velocities $\boldsymbol{\omega}_t$. Specifically, the penalty is defined as

$$r_t^{\text{energy}} = -\alpha |\boldsymbol{\tau}_t \odot \boldsymbol{\omega}_t|, \quad (11)$$

where α is a power coefficient (default 0.0005). This term encourages energy-efficient control while excluding the first three frames to avoid initialization artifacts.

Networks and Training. The proposed framework consists of two stages: KinematicsNet, which estimates kinematic states, and DynamicsNet, which generates physically consistent motion within a physics simulator. Training is performed in two stages: first, KinematicsNet is trained independently, and then DynamicsNet is trained within a physics simulation environment. During the training of DynamicsNet, the parameters of KinematicsNet are frozen and used as a motion-state estimator.

KinematicsNet comprises four submodules: LP (leaf-joint position), FP (full-joint position), LV (leaf-joint velocity), and FA (full-joint angle). Each submodule is first trained independently to ensure stable estimation, and then all modules are jointly trained in an integrated manner. Following previous works [60, 62, 63, 70], we initialize the hidden states of each submodule RNN using the ground-truth data of the first frame, to avoid ambiguity caused by uncertain initial poses. This initialization is applied only to the first frame of each sequence. Training was conducted for 2000 epochs using the Adam optimizer (learning rate 1×10^{-5} , batch size 256). Following TIP [27], IMU signals were processed with a low-pass filter to remove high-frequency noise. All training was performed on data sampled at 100 Hz.

DynamicsNet is a policy network trained within a physics simulation environment to control a torque-driven humanoid model. The policy network is implemented as a multilayer perceptron (MLP) consisting of six layers with sizes [2048, 1536, 1024, 1024, 512, 512] and SiLU activations. The network outputs target joint angles, which are converted into joint torques through PD control. In the early training stage, an early termination condition is applied: an episode is terminated when any joint position error exceeds a threshold of $\tau_e = 0.25$ m, which accelerates

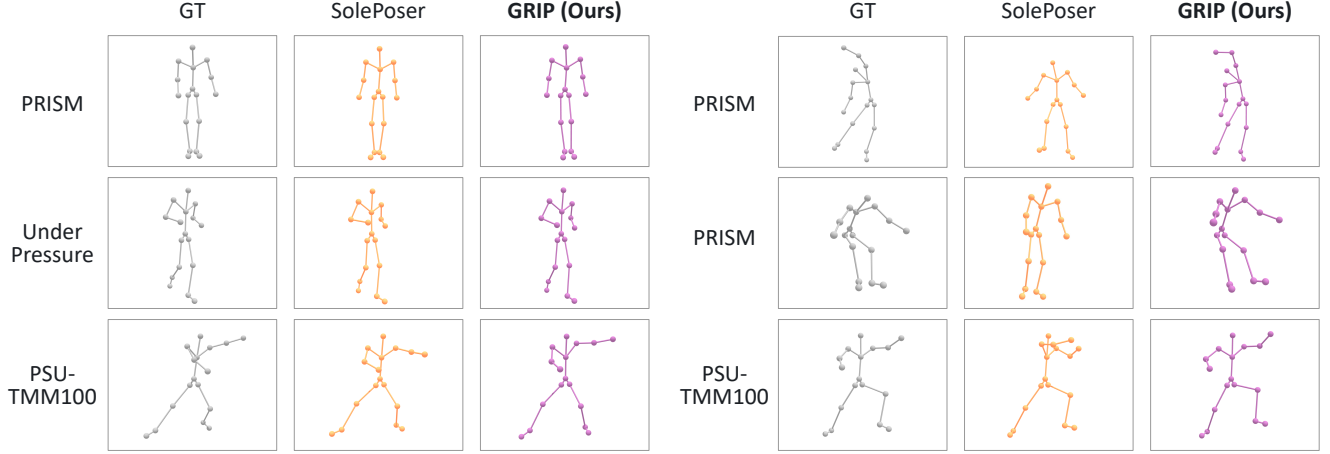


Figure 10. Comparison between SolePoser and our method using the same 17 joint locations extracted from the SMPL model.

convergence and stabilizes the learning of basic motion patterns. We train DynamicsNet for 500 epochs across 10 iterations under this early-termination regime using 4096 parallel environments, a policy minibatch size of 32,768, and an AMP minibatch size of 8192. Afterward, the early termination constraint is removed to enable training over long motion sequences, and an additional 500 epochs across 20 iterations are performed under the full-length setting. In addition, to enable fall-recovery mechanism during inference and allow the simulation to continue even after the humanoid falls, we introduce a fall-detection condition in which the humanoid is considered fallen when the root joint height drops below 0.30 m and the AMP discriminator probability ρ_t falls below 0.7.

Training takes about 24 hours for KinematicsNet and 48 hours for DynamicsNet. During inference, KinematicsNet and DynamicsNet require approximately 1.5 ms and 26.5 ms per frame, respectively. The sequential prediction in KinematicsNet is designed to improve accuracy, with limited computational overhead.

Computational Environment. Experiments were conducted on Ubuntu 20.04.6 LTS. The system is equipped with an AMD EPYC 7543 processor configuration providing 64 cores (128 threads), 251 GB RAM, and eight NVIDIA RTX A6000 GPUs with 48 GB VRAM each. All neural networks were implemented in PyTorch and optimized using the Adam optimizer. Physics simulations and reinforcement learning were performed using Isaac Gym in a CUDA 11.8-enabled environment with NVIDIA driver version 570.133.07.

9. Additional Experimental Results

Qualitative Comparison with SolePoser. In this section, we present a qualitative comparison between SolePoser [56]

and our proposed method, GRIP, which could not be included in the main paper due to space limitations. SolePoser estimates root-relative full-body pose using only the foot-mounted IMUs and foot pressure data obtained from insole sensors. Fig. 10 shows results on the PRISM, Under-Pressure, and PSU-TMM100 datasets.

SolePoser produces reasonable estimates for relatively simple poses and periodic locomotion such as walking and jogging. However, due to the limitation of using only foot sensors, its accuracy degrades significantly for motions involving substantial upper-body movement or complex postural changes, and several failure cases can be observed. In contrast, GRIP incorporates wrist-mounted IMUs in addition to foot IMUs and plantar-pressure sensing, allowing it to capture a broader range of whole-body inertial information. As a result, our method reconstructs full-body poses more accurately even in non-periodic and complex motions where SolePoser struggles. These observations indicate that GRIP can estimate whole-body dynamics that cannot be recovered from foot-only sensing, by combining IMUs at the extremities (hands and feet) with foot pressure information.