ZERO-SHOT HUMAN POSE ESTIMATION USING DIFFUSION-BASED INVERSE SOLVERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Pose estimation refers to tracking a human's full body posture, including their head, torso, arms, and legs. The problem is challenging in practical settings where the number of body sensors are limited. Past work has shown promising results using conditional diffusion models, where the pose prediction is conditioned on both ⟨location, rotation⟩ measurements from the sensors. Unfortunately, nearly all these approaches generalize poorly across users, primarly because location measurements are highly influenced by the body size of the user. In this paper, we formulate pose estimation as an inverse problem and design an algorithm capable of zero-shot generalization. Our idea utilizes a pre-trained diffusion model and conditions it on rotational measurements alone; the priors from this model are then guided by a likelihood term, derived from the measured locations. Thus, given any user, our proposed `InPose` method generatively estimates the highly likely sequence of poses that best explains the sparse on-body measurements.

1 INTRODUCTION

Human pose estimation is a crucial piece to numerous applications, including medical rehabilitation, virtual and augmented reality (AR/VR), sports coaching, health monitoring, performing arts, etc. An ecosystem of pose-estimation tools already exists. In lab settings Stathopoulos et al. (2024), the environment is instrumented with visual/infrared cameras to track user-pose so long as they are in the camera's field of view. In un-instrumented settings, such as in homes or offices, virtual reality (VR) technologies are aiming to achieve full-body pose tracking using VR goggles and two handheld controllers Han et al. (2020). The results have steadily improved Dittadi et al. (2021); Pavlakos et al. (2019) with a recent boost from generative models (e.g., conditional diffusion models Castillo et al. (2023)) that predicted the user's full-body pose from just 3 sensors. Unfortunately, such proposed generative techniques have an important limitation; they don't generalize well across users with varying body sizes. A generative model trained on data from a single user can't be used by a user with a different body size without fine-tuning. Authors in Aliakbarian et al. (2022) try to overcome this issue by jointly training over both pose datasets and varying body shapes, but this increases model complexity, and there is no guarantee that all possible body sizes were accounted for during training. An algorithm that generalizes even to body shape outliers would be ideal.

In this paper, we propose `InPose`, a diffusion-based method that implicitly accounts for the user's body size without requiring any fine-tuning. Our core observation is that any human's full-body pose can be decomposed into a "scale-free pose" and a scale-dependent component. For human poses, the scale-free pose can be imagined as a template human body whose skeletal joints (e.g., shoulders, elbows, hip, knees, etc.) are rotated appropriately to create a given pose. The scale-dependent component is the location of the joints in 3D space. Forward kinematics relates the scale-free pose, along with the body size, to the scale-dependent component. Since the sensors give ⟨rotation, location⟩ measurements from 3 body joints, it is possible to estimate a distribution of scale-free poses from rotational measurements alone. Then, the location measurements can be used to sharpen this distribution to poses that best explain the measurements. This decomposition lends itself to an inverse problem formulation, shown visually in Fig. 1a. Using ⟨rotation, location⟩ measurements from 3 body joints—head and two wrists—`InPose` aims to track the locations of all 22 body joints, necessary to fully define the full 3D pose of a human.

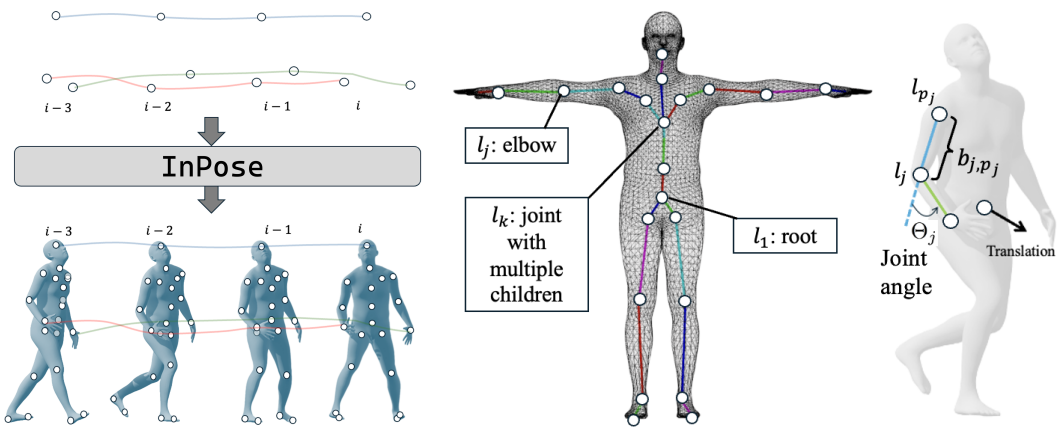


Figure 1: (a) `InPose`'s input and output visualized over 4 time frames. (b) "T" pose. (c) Pose with depiction of rotation angle and root translation.

`InPose`'s inverse problem formulation can be sketched as follows. We train a Diffusion model conditioned on *rotational measurements* from existing datasets; this gives us a conditional prior on scale-free poses. When inferring a specific user's pose, we use the user's body size to scale up/down the scale-free pose, and compare against *location measurements* to estimate a likelihood of the pose. This likelihood term requires propagating a Gaussian random variable through a nonlinear operator. We prove this propagation can be approximated as a Gaussian, and use the likelihood as an inverse kinematics guidance term to guide the diffusion denoising process. The denoised result is a sequence of full-body poses—samples from the posterior—that best explains the 3-point measurements for that specific user. Through extensive experiments, we show promising generalization results on the AMASS Mahmood et al. (2019) dataset across a wide range of body sizes and shapes.

2 Model and Measurement

Body Model: Following the conventional SMPL framework Loper et al. (2015), we model the human body as a graph (Fig. 1b). The vertices of this graph are the 22 main joints in the human skeleton; the edges are the bones connecting these joints. The 3D coordinates of the joints (in a global reference frame) are denoted as $l_j \in \mathbb{R}^3, j \in \{1, ..22\}$. The bone that connects adjacent joints l_j, l_k are denoted by a vector $b_{j,k} \in \mathbb{R}^3$ of fixed length $|b_{jk}|$. Every joint l_j has a unique parent, l_{p_j}. The whole joint-tree has a root joint l_1 located at the pelvis.

The *global pose* of a body is fully defined by the 22 joint locations in a global reference frame. Fig.1b shows a "T" pose and Fig.1c shows a running pose. Intuitively, a global pose can be computed in three steps. (1) Start with a standard "T" pose with the human located at the origin of a global reference frame. (2) Move the root joint l_1 to bring the human to it's correct location; the human is still in the "T" pose but the whole body is displaced. (3) Now, starting from joint l_1, rotate each joint based on the rotational measurements. Perform this *sequentially down the joint tree* ensuring a parent joint p_j has been rotated before rotating joint j. These 3 steps brings us to the human's global pose.

Eq. 1 models the above steps to compute joint j's global location at time frame i.

$$l_j(i) = l_{p_j}(i) + R_{p_j}(i) \cdot b_{j,p_j} \tag{1}$$

Here $R_{p_j}(i)$ is a global 3D rotation matrix of the parent joint. Note that the global 3D rotation matrix for any joint j is computed as $R_j(i) = R_{p_j}(i) \cdot \Theta_j(i)$, where $\Theta_j(i)$ is the *local* 3D rotation matrix shown in Fig.1c. Since $\Theta_j(i)$ is represented as 3D rotation matrices[1], all the joint angles in the "T" pose are identity matrices. As the human performs different poses, `InPose` aims to track the root location $l_1(i)$ and global rotation $R_j(i)$ for each of the joints.

Joint Angle Representation: While representing $R_j(i)$ using 3D rotation matrices makes it easy to compute joint locations, Zhou et al. (2019) has shown that representing them instead using the

[1]Other representations are possible, including axis-angle, Euler, or quaternion representation.

6DoF parameterization $r_j(i) \in \mathbb{R}^6$ is better for neural network training. This is due to the continuity properties of this representation, which, unlike most other representations, do not require any form of normalization[2]. The forward mapping for vector $r_j(i)$ is computed as:

$$r_j(i) = [R_j^{(1,1)}(i) \quad R_j^{(2,1)}(i) \quad R_j^{(3,1)}(i) \quad R_j^{(1,2)}(i) \quad R_j^{(2,2)}(i) \quad R_j^{(3,2)}(i)]^\top$$

where $R_j^{(k,l)}(i)$ is the $\{k,l\}^{\text{th}}$ element of the corresponding 3D rotation matrix. There also exists a non-linear differentiable inverse $\bar{\mathcal{D}} : \mathbb{R}^6 \rightarrow \mathbb{R}^{3\times3}$ that maps the 6DoF representation to rotation matrices (defined in Appendix A). Hence, Eq.1 becomes: $l_j(i) = l_{p_j}(i) + \bar{\mathcal{D}}(r_{p_j}(i)) \cdot b_{j,p_j}$. We extend $\bar{\mathcal{D}}$ to map all $|M|$ rotations from 6DoF to rotation matrices, and term this function $\mathcal{D}$.

Measurements: To align with recent work in this area Zheng et al. (2023a); Castillo et al. (2023), we use the same AMASS dataset that contains locations and rotation angles of the head and two wrists. These measurements are from VR goggles and handheld controllers, which use a combination of ego-centric cameras, IMU sensors, visual SLAM algorithms, and dead reckoning methods to estimate m locations and rotations in the global reference frame, where $m \subset M = \{1, ...22\}$ and $|m| = 3$.

3 INPOSE: INVERSE ZERO-SHOT POSE ESTIMATION

3.1 FORMULATION

We denote the noisy signal measurements from the 3 sensor joints as $y_m(i) = [l_m(i), r_m(i)]$. Here i indexes the measurement time-frames and is dropped throughout the rest of the paper unless specified. $l_m(i) = l_m^+(i) + \sigma_l v(i)$ is the noisy location measurement, where l_m^+ is the noise-free joint location, and $v(i)$ is iid Gaussian noise. Similarly, the rotation $r_m(i)$ is also noisy. Our goal is to predict r_M, which are the 22 global joint rotations and l_1, the root's translation. We are also provided the user's bone lengths b_{j,p_j}. With such sparse measurements, this is an ill-posed pose estimation problem.

Like Jiang et al. (2022), we simplify this problem by first assuming the root joint is stationary. We estimate the scale-free pose defined by r_M; then scale to the correct pose defined by l_M; and then *drag* this pose until the head's location matches the measured head location, l_{head}. From this, we infer the root translation l_1. Thus, the core question boils down to sampling from the posterior $p(r_M|y_m)$.

Diffusion models have recently found remarkable success for these types of posterior sampling problems. They were originally proposed as a tool for sampling from a prior distribution $p_0(x^0)$. This is done by first defining a noising process $p_t(x^t)$ by injecting iid Gaussian noise of standard deviation σ_t into it, where $t \in \{0 : T\}$. Diffusion models aim to reverse this noising process by learning the score function $\nabla_{x_t} log\ p_t(x_t)$.

In our scenario, we require the conditional score $\nabla_{r_M^t} log\ p_t(r_M^t|y_m)$. One method is to use Classifier-Free Guidance (CFG) proposed by Ho & Salimans (2021). In this formulation, a conditional diffusion model is trained to accept all the inputs $y_m = [l_m, r_m]$ for conditioning. Most previous work Castillo et al. (2023); Van Wouwe et al. (2024) use this approach and are unable to support zero-shot generalization. This is because the (noisy) location measurements l_m vary based on body size—if two people are in the same pose, their joints would share identical rotation angles, *but because of differences in bone lengths, we see from Eq. 1 that the joint locations l_j will be different.* Thus, a conditional model trained on one user's data does not generalize well to another. With a sequence of poses through time, small prediction errors accumulate resulting in greater degradation.

To overcome this, we split the conditional score $\nabla_{r_M^t} log\ p_t(r_M^t|y_m) = \nabla_{r_M^t} log\ p_t(r_M^t|\{l_m, r_m\})$ using Bayes' rule:

$$\nabla_{r_M^t} log\ p_t(r_M^t|\{l_m, r_m\}) = \nabla_{r_M^t} log\ p_t(r_M^t|r_m) + \nabla_{r_M^t} log\ p_t(l_m|r_M^t, r_m) + 0 \tag{2}$$

$$= \nabla_{r_M^t} log\ p_t(r_M^t|r_m) + \nabla_{r_M^t} log\ p_t(l_m|r_M^t) \tag{3}$$

where we have assumed l_m and r_m are **conditionally independent**.

The conditional score $\nabla_{r_M^t} log\ p_t(r_M^t|r_m)$ is scale-free, and can be learned by a CFG-based conditional diffusion model. The scale-dependent likelihood score $\nabla_{r_M^t} log\ p_t(l_m|r_M^t)$ can be utilized as a

[2]Rotation matrices and quaternions need to satisfy unitary conditions, affecting the quality of output poses

guidance to the prior Chung et al. (2023); Kawar et al. (2022). This guidance is performed during inference and *does not require any training or fine-tuning of the generative neural network.* We use the Pseudoinverse-Guidance for Diffusion Models (ΠGDM) Song et al. (2023) framework, but propose a mechanism to propagate a Gaussian random variable through the non-linear inverse $\mathcal{D}(.)$ function inside the likelihood term (discussed soon). This mathematically enables the decomposition of the user's pose into a general scale-free pose (from the prior) and a user-specific scaling factor (captured in the likelihood). Thus, our main contribution over past work—and the key to enabling zero-shot pose-prediction—is to perform CFG using *only* the measured rotations, and use *only* the joint locations as a pseudoinverse guidance to that CFG.

3.2 Designing the Likelihood, Prior, and Posterior Terms

We now describe the various score terms used in our algorithm during every diffusion timestep.

CFG Prior: We train a CFG-based score model $\epsilon_\theta(r_M^t, t, r_m)$ to determine the conditional score $\nabla_{r_M^t} log\ p_t(r_M^t|r_m)$ as a function of the noisy rotation inputs r_m from the 3-point sensors. This is then used to derive a conditionally denoised estimate $\hat{r}_M^t$ using Tweedie's formula Efron (2011):

$$\hat{r}_M^t = \frac{r_M^t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(r_M^t, t, r_m)}{\sqrt{\bar{\alpha}_t}} \tag{4}$$

We adapt the same DiT Peebles & Xie (2022) transformer architecture used in BoDiffusion Castillo et al. (2023) but modify it appropriately (details in Section 4) since we are only conditioning on rotation r_m, while BoDiffusion conditioned on both r_m and l_m.

Likelihood score: To compute the likelihood score $\nabla_{r_M^t} log\ p_t(l_m|r_M^t)$, we need to relate the joint rotations to joint locations. Mathematically, we aim to minimize the likelihood $||l_m - \mathcal{A} \circ \mathcal{D}(\hat{r}_M^t)||_2$ where $\mathcal{A} \circ \mathcal{D}(\cdot)$ is the *measurement* operator. Recall, $\mathcal{D}(.)$ converts 22 joint rotations from the 6DoF vectors to rotation matrices, and $\mathcal{A}$ is a linear function that uses these rotation matrices to determine the joint location estimates. Eq. 1 had earlier shown the operation of $\mathcal{A}$ for a single joint, where $R_j(i) = \mathcal{D}\big(r_j(i)\big)$.

Unfortunately, two issues stand in the way of estimating the likelihood. ❶ Since r_M^t is a noisy estimate of r_M, we cannot pass it through the measurement operator to obtain $\hat{l}_m$. To mitigate this, we adopt ideas from ΠGDM Song et al. (2023) to help approximate r_M^t as a Gaussian distribution. ❷ Since $\mathcal{D}(.)$ is a non-linear function, propagating the approximated Gaussian random variable through $\mathcal{D}(.)$ is problematic. We will prove that this propagation can be approximated with a Gaussian as well, giving us a pathway to the final solution. Let us briefly review ΠGDM first and then visit the second step in `InPose`.

ΠGDM Recap: Consider the general problem where we are given observations $z = Ax^0 + \sigma_z n$, where A is the measurement model, n is unit Gaussian noise, and σ_z is the noise variance. Say we would like to estimate x^0, for which we will guide a diffusion model that is denoising a noisy x^t at each diffusion time step. This guidance needs to use the likelihood score $\nabla_{x^t} \log p_t(z|x^t)$, which needs to be computed through an intermediate step of marginalization over x^0 as follows:

$$p_t(z|x^t) = \int p(z|x^0)p_t(x^0|x^t)\mathrm{d}x^0 \tag{5}$$

If A is a linear function and the noise n is Gaussian, then $p(z|x^0) \sim \mathcal{N}(Ax^0, \sigma_z \mathrm{I})$. For the second term $p_t(x^0|x^t)$, ΠGDM proposes to approximate this distribution as $\mathcal{N}(\hat{x}^t, w_t^2 \mathrm{I})$, where the mean comes from a regular diffusion step. Hence, the distribution $p_t(z|x^t)$ and the corresponding likelihood score can both be approximated by a Gaussian as follows:

$$p_t(z|x^t) \approx \mathcal{N}(A\hat{x}^t, w_t^2 AA^\top + \sigma_z \mathrm{I}) \tag{6}$$

$$\nabla_{x^t} \log p_t(z|x^t) \approx ((z - A\hat{x}^t)^\top (w_t^2 AA^\top + \sigma_z^2 \mathrm{I})^{-1} A \frac{\partial \hat{x}^t}{\partial x^t})^\top \tag{7}$$

Let us now return to `InPose`. We cannot directly apply ΠGDM to our likelihood score $\nabla_{r_M^t} log\ p_t(l_m|r_M^t)$ since our measurement operator contains the $\mathcal{D}(.)$ function. But if $\mathcal{D}(.)$ is ignored—meaning that the rotation matrix R_M^t is somehow available—then the measurement opera-

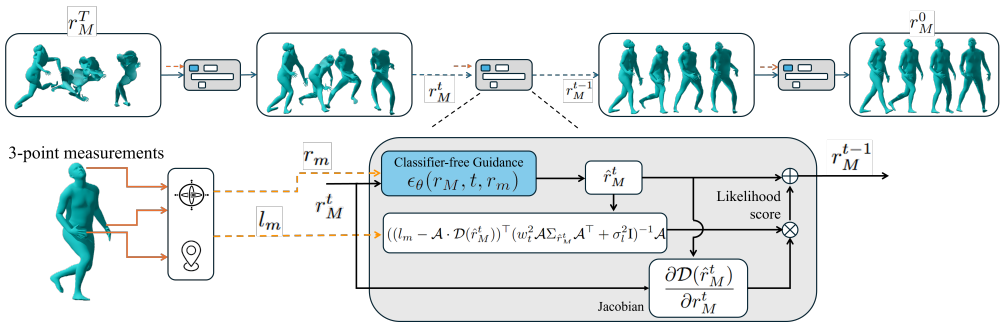


Figure 2: InPose pipeline: 3-sensor rotation + location measurements are inputs. Rotations fed as conditions to CFG which outputs conditional prior; location measurements estimate the likelihood, which steers denoising.

tor in Eq. 1 becomes linear. When $l_1 = 0$, the joint location l_j becomes a matrix-vector product, $C\kappa$, as follows:

$$[R_1 ... R_{p_j}] \cdot [b_{2,1}^\top ... b_{j,p_j}^\top]^\top = l_j \tag{8}$$

where $C = [R_1 ... R_{p_j}]$, and $\kappa = [b_{2,1}^\top ... b_{j,p_j}^\top]^\top$. The matrix-vector product can be rearranged to form $(\mathrm{I}_3 \otimes \kappa^\top) \cdot \mathrm{vec}(C)$ where $\otimes$ is the Kronecker product. We can thus obtain our linear function $\mathcal{A} \coloneqq \mathrm{I}_3 \otimes \kappa^\top$. Plugging this $\mathcal{A}$ into Eq 7 gives us the likelihood score.

Unfortunately, the $\mathcal{D}(.)$ function is non-linear in InPose, hence the conditional distribution $p_t(z|x^t)$ in Eq. 6 is no longer Gaussian. However, using the following Theorem, we show that it is possible to approximate $p_t(l_m|r_M^t)$ as a Gaussian and compute its covariance matrix with a well-trained score model ϵ_θ (proof in Appendix A).

Theorem 1. *We are given a well-trained error model ϵ_θ, that learns the error distribution $\epsilon_t \leftarrow \epsilon_\theta(r_M^t, t, r_m)$, and denoises $\hat{r}_M^t \leftarrow \frac{r_M^t - \sqrt{1-\bar{\alpha}_t}\epsilon_t}{\sqrt{\bar{\alpha}_t}}$. If the model ensures that $||\hat{r}_j^{t,1:3}|| = ||\hat{r}_j^{t,3:6}|| = 1,\ \langle \hat{r}_j^{t,1:3}, \hat{r}_j^{t,3:6} \rangle = 0,\ \forall j \in M$ then $p_t(\mathcal{D}(r_M^0)|r_M^t) \approx \mathcal{N}(\mathcal{D}(\hat{r}_M^t), w_t^2 \Sigma_{\hat{r}_M^t})$ where $\Sigma_{\hat{r}_M^t}$ is a positive definite matrix.*

From the proof of Theorem 1, we obtain the covariance matrix for $\mathcal{D}(\hat{r}_M^t)$ as $\widetilde{\mathrm{Cov}}(\mathcal{D}(\hat{r}_M^t)) = w_t^2 \Sigma_{\hat{r}_M^t}$. Using this approximation, we get $p_t(\mathcal{D}(r_M^0)|r_M^t) \approx \mathcal{N}(\mathcal{D}(\hat{r}_M^t), w_t^2 \Sigma_{\hat{r}_M^t})$, and thus

$$\nabla_{r_M^t} \log p_t(l_m|r_M^t) = ((l_m - \mathcal{A} \cdot \mathcal{D}(\hat{r}_M^t))^\top (w_t^2 \mathcal{A} \Sigma_{\hat{r}_M^t} \mathcal{A}^\top + \sigma_l^2 \mathrm{I})^{-1} \mathcal{A} \frac{\partial \mathcal{D}(\hat{r}_M^t)}{\partial r_M^t})^\top \tag{9}$$

3.3 Accounting for Translation: Differential Parameterization

From Eq. 1, we see that all joint locations at frame i have an additive dependence on $l_1(i)$ due to the kinematic chain:

$$l_j(i) = \sum_{k=3}^{j} (l_{p_k}(i) + R_{p_k}(i) \cdot b_{k,p_k}) + R_1(i) \cdot b_{2,1} + l_1(i) \tag{10}$$

The mapping $\mathcal{A}$ derived from Eq. 8 is only valid if $l_1(i) = 0$. To enable the linear inverse guidance formulation when $l_1(i) \neq 0$, we use the difference between the positional measurements from each of the 3 measured joints at every frame. Thus, the contribution of the root translation $l_1(i)$ for each of the measured joint locations gets canceled.

3.4 Model Pipeline

Figure 2 summarizes the InPose pipeline. Its objective can be summarized as performing Guided diffusion to infer a sequence of human poses r_M using a combination of conditioning inputs and Pseudoinverse guidance using a modified ΠGDM likelihood score. The inputs to the algorithm are the noisy joint rotations $r_{j \in m}$ and locations $l_{j \in m}$ of a subset of joints m of size 3.

At each diffusion step t, InPose's workflow can be summarized in the following steps:

- Use the CFG score function $\nabla_{r_M^t} log\ p_t(r_M^t|r_m)$ conditioned on noisy rotation inputs $\{r_m\}$ to generate a conditionally denoised estimate $\hat{r}_M^t$.
- Use ΠGDM to estimate the likelihood score $\nabla_{r_M^t} \log p_t(l_m|r_M^t)$.
- Combine the conditionally denoised estimate and the likelihood score using modified DDIM to generate the diffusion output for the next step r_M^{t-1}. The proposed `InPose` algorithm is described in Algorithm 1.

Algorithm 1 `InPose` Inference using modified ΠGDM

Require: $N, \epsilon_\theta, \eta \in [0,1], \mathcal{A}, \mathcal{D}(\cdot)$
Inputs: $y_m = [l_m, r_m], \sigma_l$
Find a sequence of timesteps $q_{i \in 0..N}$ with $q_0 = 0$ and $q_N = T$
Initialize $r_M \sim \mathcal{N}(0, \mathrm{I})$
for $i \in \{N..1\}$ **do**
  $t \leftarrow q_i, \quad s \leftarrow q_{i-1}$ ▷ Get start and end times
  $\bar{\alpha}_t \leftarrow \frac{1}{1+\sigma_t^2}$ ▷ Get α for VP-SDE
  $\epsilon_t \leftarrow \epsilon_\theta(r_M, t, r_m)$
  $\hat{r}_M^t = \frac{r_M - \sqrt{1-\bar{\alpha}_t}\epsilon_t}{\sqrt{\bar{\alpha}_t}}$ ▷ Denoised output at current iteration
  $c_1 \leftarrow \eta\sqrt{(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_s})\frac{1-\bar{\alpha}_s}{1-\bar{\alpha}_t}}$ ▷ Constants for DDIM
  $c_2 \leftarrow \sqrt{1 - \bar{\alpha}_s - c_1^2}$
  $w_t^2 \Sigma_{\hat{r}_M^t} \leftarrow \widetilde{\mathrm{Cov}}(\hat{r}_M^t)$
  $g \leftarrow ((l_m - \mathcal{A} \cdot \mathcal{D}(\hat{r}_M^t))^\top (w_t^2 \mathcal{A} \Sigma_{\hat{r}_M^t} \mathcal{A}^\top + \sigma_l^2 \mathrm{I})^{-1} \mathcal{A} \frac{\partial \mathcal{D}(\hat{r}_M^t)}{\partial r_M^t})^\top$ ▷ Likelihood score
  Sample $\epsilon \sim \mathcal{N}(0, \mathrm{I})$
  $r_M \leftarrow \sqrt{\bar{\alpha}_s}\hat{r}_M^t + c_1\epsilon + c_2\epsilon_t + \sqrt{\bar{\alpha}_t} g$ ▷ Posterior update
end for
return r_M ▷ Return Estimated Pose sequence

A pertinent question one may ask is as follows. Given that the user's body size parameters are available during inference, why not scale the default-body dataset with these body size parameters? Said differently, applying Eq. 1, the scale-free joint rotations r_M can be scaled—using the available body size parameters—to regenerate locations l_M. This new dataset can then be used to train a CFG model, obviating the need for inverse solvers like `InPose`. Unfortunately, this is possible only if l_1 was known (or equal to 0). Otherwise, the mapping between r_M to l_1 is non-trivial. As an illustration, consider that the user jumps. Without modeling the dynamics of the human body, it is hard to determine the user's displacement while they are airborne. This is why pure CFG models are unable to generalize across body sizes, while `InPose`'s inverse guidance formulation does not require new datasets from new users.

4 Experiments

Datasets: All our experiments were performed on AMASS Mahmood et al. (2019), which is an aggregate of multiple human-pose datasets and the de facto standard today for pose estimation/generation. The data is in the SMPL body model format. Each dataset within AMASS consists of multiple samples, each of which is a sequence of poses at 60, 100, or 120Hz; we resample all data to 60Hz. A fixed default body shape typical of the average male is used for model training. Our experiments follow two dataset protocols, as per our BoDiffusion baseline Castillo et al. (2023):

1. The Transitions Mahmood et al. (2019) and the HumanEVA Sigal et al. (2009) datasets within AMASS were used for testing, while others were used for training.
2. An approximately 90%/10% split for training and testing respectively, on the CMU Carnegie Mellon University, BMLrub Troje (2002), and the HDM05 Müller et al. (2007) datasets.

Baselines: We choose the two following SOTA algorithms as baselines. These models accept the 3-point joint locations $l_m(i)$, rotations $r_m(i)$, and the corresponding velocity and angular velocity

as inputs; they output the full-body pose of the user for which each was trained. Both the velocity and the angular velocity are computed as linear functions of l_m and r_m. The rotation and angular velocities are represented in the 6DoF representation.

- **AvatarJLM** Zheng et al. (2023a) is a conventional neural network-based approach.
- **BoDiffusion** Castillo et al. (2023) is the CFG-based diffusion model that we adopt for `InPose`. In the original BoDiffusion paper, it outputs the local joint angles $\Theta_M(i)$. The authors provide a pretrained model, which is denoted as BoDiffusion(Local) in all our results.
- **BoDiffusion(Global):** We modified BoDiffusion(Local) to output global joint angles r_M. This is also evaluated using both l_m and r_m for CFG and is termed BoDiffusion(Global).

Implementation Details: We fine-tune the neural network used in BoDiffusion, and perform inference using $N = 50$ steps. The original BoDiffusion algorithm implements a DiT Peebles & Xie (2022) based denoiser in a CFG diffusion framework. Additional details are provided in Appendix B.

Evaluation Metrics: We use 4 standard metrics from literature to evaluate the models:

- **Mean Per Joint Position (location) Error(MPJPE)** measures the mean joint location error, in cm, across all joints and poses in the sequence.
- **Mean Per Joint Rotation Error(MPJRE)** measures the mean joint rotation error, in degrees, again across all joints and poses. MPJPE captures the scale-dependent error, while MPJRE captures the scale-free error.
- **UPE, LPE:** We also report the joint position error, in cm, for the upper and lower body separately, respectively. These two tell us how well the model is able to infer the upper body versus leg movement, given that measurement sensors are all placed at the upper body.

4.1 Results

Zero-shot generalization across body sizes: Fig. 3(a,b) presents results when the models are trained on a default body size, and then tested for various body sizes (including the default). The body sizes are varied by changing the scaling factor on the X axis (a value greater than 1.0 on the X axis indicates a proportionally taller human, and vice versa). Note, all bones of the taller person (or shorter) human has been scaled up (or down) by the same factor (we will report separate results where different parts of the body are scaled differently). The root joint translation is also proportional to this scaling factor. The results are performed using Protocol 1. We report location and rotation errors (MPJPE and MPJRE). *Importantly, we divide the estimated MPJPE by the scaling factor*; The MPJRE is obviously scale-free.

As expected, the baselines are able to outperform `InPose` in the default case when scale equals 1.0. This is because they are trained for this default shape. However, both the scaled MPJPE and the MPJRE (Fig. 3a and 3b) remain almost flat for `InPose` regardless of body size. This demonstrates the zero-shot nature of our inverse solver in contrast to the significant degradation of the baselines.

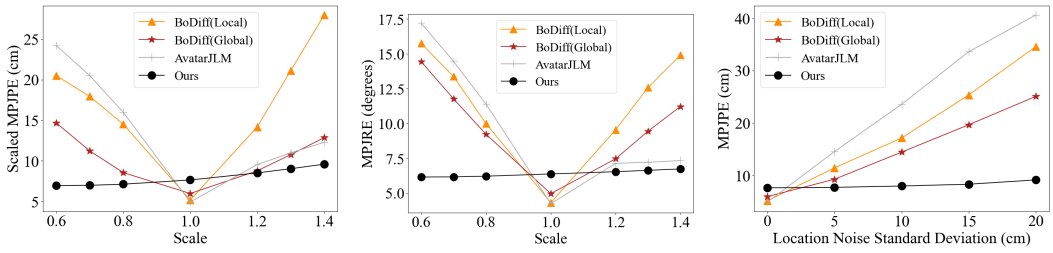


Figure 3: (a) Position error vs. body scale. (b) Rotation error vs. body scale. (c) Position error vs. location noise. All these tests were performed using Protocol 1

Robustness to measurement noise: `InPose` is designed to be implicitly robust to location measurement noise as well. We inject zero-mean i.i.d. Gaussian noise into the input location streams and compute the estimation errors, while maintaining the *default* body shape and the rotation measurements. This is an important test for practical applications since real-world wearable sensors—like watches and phones—have difficulty with measurement errors. Fig. 3c shows the location error under increasing Gaussian noise variance (the rotation error is reported in the Appendix). Evidently,

InPose stays flat while other baselines degrade with noise. This is expected because while BoDiffusion's model is sensitive to location noise, InPose uses the location only for inverse guidance, allowing the prior to play an important role in the final pose estimates. In our experiments, we also found that the velocity error in noisy conditions is lower in the case of InPose compared to BoDiffusion. The output pose sequences from the baselines have high jitter, indicating the estimated poses are out of distribution.

Qualitative results with scaling: Fig.4 presents qualitative comparisons between InPose and BoDiffusion(Global), for the default body size and two scaling factors of 0.6 and 1.4. BoDiffusion performs better for the default size, especially in the lower body, but degrades at the task of generalization. The errors are especially prominent in the 0.6 case, where BoDiffusion predicts the lower body to be in a squatted pose because the measurements are generated by a user of short stature. Since the priors were learnt by BoDiffusion on data generated by a user of the default body shape, there is no way of informing the model of this difference. For the 1.4 case too, BoDiffusion incurs higher torso error.

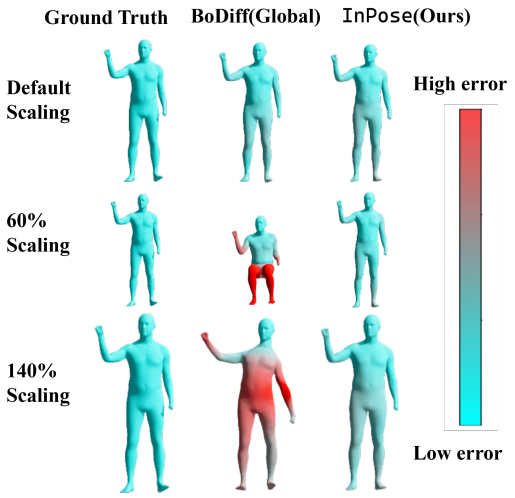


Figure 4: Qualitative results with scaling body size. The same pose is used for all scales.

Varying relative sizes of body parts: Table 1 reports results when the body parts are not scaled up or down uniformly; instead, limbs and torso are scaled with different scaling factors. This models even the outliers in human variations, e.g., basketball players with longer arms, or athletes with longer legs. To create the ground truth data, we scale bone lengths first, which are then used to recompute the joint locations l_M from the scale-free joint rotations r_M. Unfortunately, the root translation l_1 is a non-trivial function of r_M and the body shape. But in general, we observe that l_1 is similar across body shapes that share the same lower body bone lengths. Thus, we preserve both l_1 and the lower body shape, and only vary the upper body shape for these tests.

(a) Results with Upper Body shape variation (Protocol 1) (↓ is better)

Algorithm	Default shape				Upper body ×1.4				Arms ×1.4, Torso ×0.7			
	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE
AvatarJLM	**4.92**	**4.25**	**2.13**	9.94	26.09	7.02	25.46	27.47	18.89	9.33	14.76	25.95
BoDiffusion(Local)	5.16	4.32	2.36	**9.72**	25.69	15.35	22.79	30.21	9.98	9.24	8.05	13.33
BoDiffusion(Global)	5.97	4.97	2.35	11.96	13.40	11.48	10.91	17.93	7.61	7.24	5.15	**11.98**
InPose	7.64	6.38	3.36	14.74	**9.15**	**6.71**	**4.80**	**16.31**	**7.45**	**6.52**	**3.23**	14.6

(b) Results with Upper Body shape variation (Protocol 2) (↓ is better)

Algorithm	Default shape				Upper body ×1.4				Arms ×1.4, Torso ×0.7			
	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE
AvatarJLM	**3.54**	3.11	**1.49**	**6.92**	27.13	8.36	26.02	29.21	20.32	9.34	15.83	27.98
BoDiffusion(Local)	3.59	**2.68**	1.51	7.0	26.12	14.51	21.34	33.62	10.13	8.41	8.13	**13.63**
BoDiffusion(Global)	4.90	3.45	1.90	9.79	17.39	12.71	13.59	23.77	9.99	8.78	7.25	14.85
InPose	7.53	4.73	2.9	15.04	**8.94**	**4.73**	**3.97**	**16.85**	**6.96**	**4.77**	**2.60**	14.17

Table 1: Results across 4 metrics for non-uniform scaling of body sizes.

Evident from Table 1 (and more results in Table 2 in the Appendix), the results are aligned with previous graphs. With the default body shape, the baselines outperform InPose on all 4 metrics. This is because the respective neural networks are able to learn a complex non-linear mapping from both the joint rotation r_m and the location inputs l_m to the user's pose since the training body shape and the inference body shape are identical. In contrast, InPose uses linear constraints using l_m to steer the output towards an estimate of the pose sequence that best explains the input.

However, when the bone lengths change, the baselines are misguided by the input locations and, therefore, do not generalize. InPose outperforms both baselines in the MPJRE, MPJPE, and UPE metrics since inverse location guidance accounts for the change in body shape. Unfortunately, the LPE error remains higher for InPose in some cases.

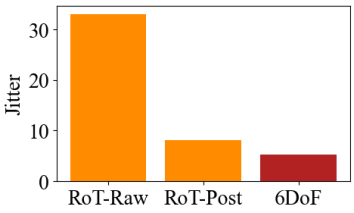


Figure 5: 6DoF vs. rotation matrix.

Ablation: 6DoF versus rotation matrices: Recall that `InPose` needed to tackle the non-linearity from the $\mathcal{D}(.)$ function, which was needed to convert 6DoF representation to matrices. A natural question is: was it necessary to use 6DoF at all? Figure 5 shows the importance of 6DoF over 3×3 rotation matrices. We train two unconditional UNET-based diffusion models—one using the 6DoF representation and the other using rotation matrices. These models generate 64-frame human pose samples as global joint angles in their respective representation. These joint angles are then used to generate sequences of body meshes. The raw rotation matrix sample meshes(labeled RoT-Raw) from the Rotation matrix UNET have high jitter. In contrast, the 6DoF meshes(labeled 6DoF) from the 6DoF UNET have much lower jitter. This is primarily because the raw output rotation matrices are not unitary. In fact, by postprocessing the rotation matrices using $\mathcal{D}(.)$ on the first 2 columns(labeled RoT-Post), we considerably lower the jitter. More results and animations are available here: `https://iclrinpose-crypto.github.io/ICLRInPose/`

5 Related Work

Deep-learning based methods for pose-tracking: Deep learning has found much success in determining pose from a sparse set of measurements. Aliakbarian et al. (2023); Du et al. (2023); Zheng et al. (2023b); Yuan et al. (2023) all use HMD-based location and rotation sensors to estimate pose and translation. Nearly all these works focus on using sensor information from the head and the two wrists. Jiang et al. (2022) estimates pose by first using a Transformer encoder to estimate local joint angles, and then estimates translation by fitting the generated head translation to the head location sensor input. Castillo et al. (2023) uses a CFG diffusion-based approach to estimate pose. Most of the above-mentioned approaches are specifically trained for a single user's body parameters, which comes at the cost of worse generalizability. One work that does generalize across users, Aliakbarian et al. (2022) jointly trains sensor inputs and bone length parameters in a flow-based generative model framework. But this algorithm requires jointly training pose and a large number of body shapes in order to generalize. In contrast, our work can directly accept any set of body bone parameters without requiring any bone shape generalization training.
A large number of works Huang et al. (2018); Mollyn et al. (2023) focus on pose and translation prediction using a sparse set of IMUs that provide acceleration and orientation data from the joints to which they are attached. Yi et al. (2021) uses a cascaded sequence of Neural networks to predict pose from 6 IMU sensors. Yi et al. (2022) incorporate physics-based dynamics constraints on the user's joint motion to improve pose estimation accuracy.

Human Motion synthesis: A closely related topic to pose estimation is pose synthesis Raab et al. (2023); Tevet et al. (2025). This usually involves training a generative model on a human motion dataset along with textual labels to generate motion based on a prompt or unconditionally. Shafir et al. (2024) uses a trained motion synthesis model to serve as a prior for more complex tasks such as motion blending and multi-person interactive motion.
Some of these textual models also accept other inputs to serve as guidance for the generated motion Tessler et al. (2024); Diller & Dai (2024). Xie et al. (2024) generate human motion from textual prompts using a diffusion model, but can also use a gradient-based inverse guidance method to specify motion trajectories of various joints. Their work is closely related to ours, but they require $(N = 1000)$ steps of inverse guidance (using DDPM) during inference, as well as a Transformer-based realism guidance module that encodes the joint location control signals.

6 Conclusions

We propose `InPose`, a diffusion-based model that estimates the user's 3D fully-body pose sequence from 3 sensor measurements. By decomposing poses into a scale-free and a scaling factor, we find a pathway to an inverse problem formulation, which in turn enables the zero-shot generalization. As a result, any new body size or shape need not be re-trained with personalized data; `InPose` is able to guide the diffusion-based prior by computing whether the samples from the prior are consistent with user's measurements. There is room for improvement at least in two fronts, namely in outperforming the baselines for the default sizes for which they are trained, and in improving lower body errors by better modeling the physics of leg movements. We believe these are rich problems for future research.

7 REPRODUCIBILITY STATEMENT

For reproducibility of our results: All AMASS data is available at `https://amass.is.tue.mpg.de/`, as long as the license agreements for each dataset are followed. The analysis codebase is available on our repository, linked from our website `https://iclrinpose-crypto.github.io/ICLRInPose/` with the dependencies documented in the repository as well. The implementation details and hardware requirements are provided in the Supplementary material B.

REFERENCES

Sadegh Aliakbarian, Pashmina Cameron, Federica Bogo, Andrew Fitzgibbon, and Thomas J. Cashman. FLAG: Flow-based 3D Avatar Generation from Sparse Observations . In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13243–13252, Los Alamitos, CA, USA, June 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.01290.

Sadegh Aliakbarian, Fatemeh Saleh, David Collier, Pashmina Cameron, and Darren Cosker. HMD-NeMo: Online 3D Avatar Motion Generation From Sparse Observations . In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9588–9597, Los Alamitos, CA, USA, October 2023. IEEE Computer Society. doi: 10.1109/ICCV51070.2023.00882.

Carnegie Mellon University. CMU MoCap Dataset. URL `http://mocap.cs.cmu.edu`.

Angela Castillo, Maria Escobar, Guillaume Jeanneret, Albert Pumarola, Pablo Arbeláez, Ali Thabet, and Artsiom Sanakoyeu. Bodiffusion: Diffusing sparse observations for full-body human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

Christian Diller and Angela Dai. CG-HOI: Contact-Guided 3D Human-Object Interaction Generation . In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19888–19901, Los Alamitos, CA, USA, June 2024. IEEE Computer Society. doi: 10.1109/CVPR52733.2024.01880.

Andrea Dittadi, Sebastian Dziadzio, Darren Cosker, Ben Lundell, Tom Cashman, and Jamie Shotton. Full-Body Motion from a Single Head-Mounted Device: Generating SMPL Poses from Partial Observations . In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11667–11677, Los Alamitos, CA, USA, October 2021. IEEE Computer Society. doi: 10.1109/ICCV48922.2021.01148.

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.

Bradley Efron. Tweedie's formula and selection bias. *J. Am. Stat. Assoc.*, 106(496):1602–1614, 2011.

George T. Gilbert. Positive definite matrices and sylvester's criterion. *The American Mathematical Monthly*, 98(1):44–46, 1991. ISSN 00029890, 19300972.

Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D. Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, Asaf Nitzan, Gang Dong, Yuting Ye, Lingling Tao, Chengde Wan, and Robert Wang. Megatrack: monochrome egocentric articulated hand-tracking for virtual reality. *ACM Trans. Graph.*, 39(4), August 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392452.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

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. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275108.

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 *Proceedings of European Conference on Computer Vision*. Springer, 2022.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

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, October 2015.

Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pp. 5442–5451, October 2019.

Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, January 2017. ISSN 2376-5992. doi: 10.7717/peerj-cs.103.

Vimal Mollyn, Riku Arakawa, Mayank Goel, Chris Harrison, and Karan Ahuja. IMUPoser: Full-body pose estimation using IMUs in phones, watches, and earbuds. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, volume 38, pp. 1–12, New York, NY, USA, April 2023. ACM.

M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, June 2007.

Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

Sigal Raab, Inbal Leibovitch, Peizhuo Li, Kfir Aberman, Olga Sorkine-Hornung, and Daniel Cohen-Or. Modi: Unconditional motion synthesis from diverse data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13873–13883, 2023.

Yoni Shafir, Guy Tevet, Roy Kapon, and Amit Haim Bermano. Human motion diffusion as a generative prior. In *The Twelfth International Conference on Learning Representations*, 2024.

Leonid Sigal, Alexandru O. Balan, and Michael J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(1–2):4–27, August 2009.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023.

Anastasis Stathopoulos, Ligong Han, and Dimitris Metaxas. Score-guided diffusion for 3d human recovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 906–915, June 2024.

Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. Maskedmimic: Unified physics-based character control through masked motion inpainting. *ACM Transactions on Graphics (TOG)*, 2024.

Guy Tevet, Sigal Raab, Setareh Cohan, Daniele Reda, Zhengyi Luo, Xue Bin Peng, Amit Haim Bermano, and Michiel van de Panne. CLoSD: Closing the loop between simulation and diffusion for multi-task character control. In *The Thirteenth International Conference on Learning Representations*, 2025.

Nikolaus F. Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of Vision*, 2(5):2–2, September 2002. doi: 10.1167/2.5.2.

Tom Van Wouwe, Seunghwan Lee, Antoine Falisse, Scott Delp, and C. Karen Liu. DiffusionPoser: Real-Time Human Motion Reconstruction From Arbitrary Sparse Sensors Using Autoregressive Diffusion . In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2513–2523, Los Alamitos, CA, USA, June 2024. IEEE Computer Society. doi: 10.1109/CVPR52733.2024.00243.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. Omnicontrol: Control any joint at any time for human motion generation. In *The Twelfth International Conference on Learning Representations*, 2024.

Xinyu Yi, Yuxiao Zhou, and Feng Xu. Transpose: real-time 3d human translation and pose estimation with six inertial sensors. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459786.

Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. 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)*, June 2022.

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 (ICCV)*, 2023.

Xiaozheng Zheng, Zhuo Su, Chao Wen, Zhou Xue, and Xiaojie Jin. Realistic full-body tracking from sparse observations via joint-level modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023a.

Xiaozheng Zheng, Zhuo Su, Chao Wen, Zhou Xue, and Xiaojie Jin. Realistic full-body tracking from sparse observations via joint-level modeling. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023b.

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.

A Proof of Theorem 1

Theorem 1. *We are given a well-trained error model ϵ_θ, that learns the error distribution $\epsilon_t \leftarrow \epsilon_\theta(r_M^t, t, r_m)$, and denoises $\hat{r}_M^t \leftarrow \frac{r_M^t - \sqrt{1-\bar{\alpha}_t}\epsilon_t}{\sqrt{\bar{\alpha}_t}}$. If the model ensures that $||\hat{r}_j^{t,1:3}|| = ||\hat{r}_j^{t,3:6}|| = 1,\ \langle \hat{r}_j^{t,1:3}, \hat{r}_j^{t,3:6} \rangle = 0,\ \forall j \in M$ then $p_t(\mathcal{D}(r_M^0)|r_M^t) \approx \mathcal{N}(\mathcal{D}(\hat{r}_M^t), w_t^2 \Sigma_{\hat{r}_M^t})$ where $\Sigma_{\hat{r}_M^t}$ is a positive definite matrix.*

Proof. (Sketch)We will prove the validity of the Gaussian approximation for a single joint rotation r_j^t. This result can then be naturally extended to r_M^t because each joint rotation is independent in the global reference frame. From ΠGDM, we approximate $p_t(r_M^T|r_M^t) \approx \mathcal{N}(\hat{r}_M^t, w_t^2 \mathrm{I})$. This implies that every element $r_j^{0,k},\ \forall k \in \{1:6\},\ \forall j \in M \sim \mathcal{N}(\hat{r}_j^{t,k}, w_t)$ are i.i.d. Gaussian Random variables.

The mapping $\bar{\mathcal{D}}(r_j^0) = R_j^0$ is defined as:

$$
\begin{aligned}
c^1 &= \left[r_j^{1:3}(t)\right]^\top, & \bar{c}^1 &= \frac{c^1}{||c^1||} \\
c^2 &= \left[r_j^{4:6}(t)\right]^\top - \text{proj}_{\bar{c}^1}\Big(\left[r_j^{4:6}(t)\right]^\top\Big), & \bar{c}^2 &= \frac{c^2}{||c^2||} \\
\bar{c}^3 &= \bar{c}^1 \times \bar{c}^2 \\
R_j(t) &= \begin{bmatrix}\bar{c}^1 & \bar{c}^2 & \bar{c}^3\end{bmatrix}
\end{aligned} \tag{11}
$$

By the definition of $\bar{\mathcal{D}}(r_j^0) = R_j^0$, the unit norm constraints $||\hat{r}_j^{t,1:3}|| = ||\hat{r}_j^{t,3:6}|| = 1$, and $\langle \hat{r}_j^{t,1:3}, \hat{r}_j^{t,3:6} \rangle = 0$, the elements of the first two columns, $R_j^{0,(1:2,1:3)} \sim \mathcal{N}(\hat{R}_j^{t,(l,m)}, w_t^2)$ are also Gaussian random variables that are uncorrelated. Now, the elements of the third column of each $R_j^{0,(3,1:3)}$ are the result of the cross-product $r_j^{0,1:3} \times r_j^{0,3:6}$. Let's first discuss $R_j^{0,(3,2)} = (r_j^{0,3} r_j^{0,4} - r_j^{0,1} r_j^{0,6})$. The mean of this random variable is:

$$
\begin{aligned}
\mathbb{E}[r_j^{0,3} r_j^{0,4} - r_j^{0,1} r_j^{0,6}] &= \mathbb{E}[r_j^{0,3} r_j^{0,4}] - \mathbb{E}[r_j^{0,1} r_j^{0,6}] && (12) \\
&= \mathbb{E}[r_j^{0,3}]\mathbb{E}[r_j^{0,4}] - \mathbb{E}[r_j^{0,1}]\mathbb{E}[r_j^{0,6}] && (13) \\
&= \hat{r}_j^{t,3} \hat{r}_j^{t,4} - \hat{r}_j^{t,1} \hat{r}_j^{t,6} && (14)
\end{aligned}
$$

Thus, all the elements of the third column $R_j^{0,(3,1:3)}$ have their mean given by the respective cross-product terms. Next, we can compute the variance as:

$$
\text{Var}[r_j^{0,3} r_j^{0,4} - r_j^{0,1} r_j^{0,6}] = \mathbb{E}[(r_j^{0,3} r_j^{0,4} - r_j^{0,1} r_j^{0,6})^2] - \mathbb{E}[r_j^{0,3} r_j^{0,4} - r_j^{0,1} r_j^{0,6}]^2 \tag{15}
$$

Treating the terms separately, we get

$$
\begin{aligned}
&\mathbb{E}[(r_j^{0,3} r_j^{0,4} - r_j^{0,1} r_j^{0,6})^2] \\
&= \mathbb{E}[(r_j^{0,3} r_j^{0,4})^2] + \mathbb{E}[(r_j^{0,1} r_j^{0,6})^2] - 2\mathbb{E}[(r_j^{0,3} r_j^{0,4} r_j^{0,1} r_j^{0,6})] && (16) \\
&= \mathbb{E}[(r_j^{0,3})^2]\mathbb{E}[(r_j^{0,4})^2] + \mathbb{E}[(r_j^{0,1})^2]\mathbb{E}[(r_j^{0,6})^2] - 2\mathbb{E}[(r_j^{0,3} r_j^{0,4} r_j^{0,1} r_j^{0,6})] && (17) \\
&= (w_t^2 + (\hat{r}_j^{t,3})^2)(w_t^2 + (\hat{r}_j^{t,4})^2) + (w_t^2 + (\hat{r}_j^{t,1})^2)(w_t^2 + (\hat{r}_j^{t,6})^2) - 2\hat{r}_j^{t,3} \hat{r}_j^{t,4} \hat{r}_j^{t,1} \hat{r}_j^{t,6} && (18) \\
&= 2w_t^4 + w_t^2((\hat{r}_j^{t,3})^2 + (\hat{r}_j^{t,4})^2) + (\hat{r}_j^{t,1})^2) + (\hat{r}_j^{t,6})^2)) + (\hat{r}_j^{t,3} \hat{r}_j^{t,4})^2 + (\hat{r}_j^{t,1} \hat{r}_j^{t,6})^2 - 2\hat{r}_j^{t,3} \hat{r}_j^{t,4} \hat{r}_j^{t,1} \hat{r}_j^{t,6} && (19)
\end{aligned}
$$

using the independence property and the definition of variance. Next,

$$\mathbb{E}[r_j^{0,3}r_j^{0,4} - r_j^{0,1}r_j^{0,6}]^2 = \mathbb{E}[r_j^{0,3}r_j^{0,4}]^2 + \mathbb{E}[r_j^{0,1}r_j^{0,6}]^2 - 2\mathbb{E}[(r_j^{0,3}r_j^{0,4}r_j^{0,1}r_j^{0,6})] \tag{20}$$

$$= (\hat{r}_j^{t,3}\hat{r}_j^{t,4})^2 + (\hat{r}_j^{t,1}\hat{r}_j^{t,6})^2 - 2\hat{r}_j^{t,3}\hat{r}_j^{t,4}\hat{r}_j^{t,1}\hat{r}_j^{t,6} \tag{21}$$

Substituting the above terms into Eqn. 15, we get

$$\text{Var}[r_j^{0,3}r_j^{0,4} - r_j^{0,1}r_j^{0,6}] = w_t^2(2w_t^2 + ((\hat{r}_j^{t,3})^2 + (\hat{r}_j^{t,4})^2) + (\hat{r}_j^{t,1})^2) + (\hat{r}_j^{t,6})^2)) \tag{22}$$

Finally, we compute each of the covariances of the third column elements $R_j^{0,(3,1:3)}$. To compute $\text{Cov}[R_j^{0,(3,2)}, r_j^{0,1}]$:

$$\begin{aligned}
&\text{Cov}[r_j^{0,3}r_j^{0,4} - r_j^{0,1}r_j^{0,6}, r_j^{0,1}] \\
&\quad = \text{Cov}[r_j^{0,3}r_j^{0,4}, r_j^{0,1}] - \text{Cov}[(r_j^{0,1})^2 r_j^{0,6}] && (23)\\
&\quad = \mathbb{E}[r_j^{0,3}r_j^{0,4}r_j^{0,1}] - \mathbb{E}[r_j^{0,3}r_j^{0,4}]\mathbb{E}[r_j^{0,1}] - \mathbb{E}[(r_j^{0,1})^2 r_j^{0,6}] + \mathbb{E}[r_j^{0,1}r_j^{0,6}]\mathbb{E}[r_j^{0,1}] && (24)\\
&\quad = 0 - (w_t^2 + (\hat{r}_j^{t,1})^2)\hat{r}_j^{t,6} + (\hat{r}_j^{t,1})^2\hat{r}_j^{t,6} && (25)\\
&\quad = -w_t^2\hat{r}_j^{t,6} && (26)
\end{aligned}$$

and $\text{Cov}[R_j^{0,(3,1)}, R_j^{0,(3,2)}]$:

$$\begin{aligned}
&\text{Cov}[r_j^{0,2}r_j^{0,6} - r_j^{0,3}r_j^{0,5}, r_j^{0,3}r_j^{0,4} - r_j^{0,1}r_j^{0,6}] \\
&\quad = 0 - \text{Cov}[r_j^{0,3}r_j^{0,5}, r_j^{0,3}r_j^{0,4}] + 0 - \text{Cov}[r_j^{0,2}r_j^{0,6}, r_j^{0,1}r_j^{0,6}] && (27)\\
&\quad = \mathbb{E}[r_j^{0,3}r_j^{0,5}]\mathbb{E}[r_j^{0,3}r_j^{0,4}] - \mathbb{E}[(r_j^{0,3})^2 r_j^{0,5}r_j^{0,4}] + \mathbb{E}[r_j^{0,2}r_j^{0,6}]\mathbb{E}[r_j^{0,1}r_j^{0,6}] - \mathbb{E}[(r_j^{0,6})^2 r_j^{0,1}r_j^{0,2}] && (28)\\
&\quad = 0 - w_t^2\hat{r}_j^{t,4}\hat{r}_j^{t,5} + 0 - w_t^2\hat{r}_j^{t,1}\hat{r}_j^{t,2} && (29)\\
&\quad = -w_t^2(\hat{r}_j^{t,4}\hat{r}_j^{t,5} + \hat{r}_j^{t,1}\hat{r}_j^{t,2}) && (30)
\end{aligned}$$

The list of variances is the following:

$$\begin{aligned}
\text{Var}[R_j^{0,(3,1)}] &= w_t^2(2w_t^2 + (\hat{r}_j^{t,2})^2 + (\hat{r}_j^{t,6})^2 + (\hat{r}_j^{t,5})^2 + (\hat{r}_j^{t,3})^2) \\
\text{Var}[R_j^{0,(3,2)}] &= w_t^2(2w_t^2 + (\hat{r}_j^{t,3})^2 + (\hat{r}_j^{t,4})^2 + (\hat{r}_j^{t,1})^2 + (\hat{r}_j^{t,6})^2) \\
\text{Var}[R_j^{0,(3,3)}] &= w_t^2(2w_t^2 + (\hat{r}_j^{t,1})^2 + (\hat{r}_j^{t,5})^2 + (\hat{r}_j^{t,4})^2 + (\hat{r}_j^{t,2})^2)
\end{aligned}$$

and the covariances are:

$$\begin{aligned}
&\text{Cov}[R_j^{0,(3,1)}, r_j^{0,2}] = w_t^2\hat{r}_j^{t,6} \quad \text{Cov}[R_j^{0,(3,2)}, r_j^{0,1}] = -w_t^2\hat{r}_j^{t,6} \quad \text{Cov}[R_j^{0,(3,3)}, r_j^{0,1}] = w_t^2\hat{r}_j^{t,5} \\
&\text{Cov}[R_j^{0,(3,1)}, r_j^{0,3}] = -w_t^2\hat{r}_j^{t,5} \quad \text{Cov}[R_j^{0,(3,2)}, r_j^{0,3}] = w_t^2\hat{r}_j^{t,4} \quad \text{Cov}[R_j^{0,(3,3)}, r_j^{0,2}] = -w_t^2\hat{r}_j^{t,4} \\
&\text{Cov}[R_j^{0,(3,1)}, r_j^{0,5}] = -w_t^2\hat{r}_j^{t,3} \quad \text{Cov}[R_j^{0,(3,2)}, r_j^{0,4}] = w_t^2\hat{r}_j^{t,3} \quad \text{Cov}[R_j^{0,(3,3)}, r_j^{0,4}] = -w_t^2\hat{r}_j^{t,2} \\
&\text{Cov}[R_j^{0,(3,1)}, r_j^{0,6}] = w_t^2\hat{r}_j^{t,2} \quad \text{Cov}[R_j^{0,(3,2)}, r_j^{0,6}] = -w_t^2\hat{r}_j^{t,1} \quad \text{Cov}[R_j^{0,(3,3)}, r_j^{0,5}] = w_t^2\hat{r}_j^{t,1}
\end{aligned}$$

$$\begin{aligned}
\text{Cov}[R_j^{0,(3,1)}, R_j^{0,(3,2)}] &= -w_t^2(\hat{r}_j^{t,1}\hat{r}_j^{t,2} + \hat{r}_j^{t,4}\hat{r}_j^{t,5}) \\
\text{Cov}[R_j^{0,(3,2)}, R_j^{0,(3,3)}] &= -w_t^2(\hat{r}_j^{t,2}\hat{r}_j^{t,3} + \hat{r}_j^{t,5}\hat{r}_j^{t,6}) \\
\text{Cov}[R_j^{0,(3,3)}, R_j^{0,(3,1)}] &= -w_t^2(\hat{r}_j^{t,1}\hat{r}_j^{t,3} + \hat{r}_j^{t,4}\hat{r}_j^{t,6})
\end{aligned}$$

while the terms that have been omitted are all 0.

Now that we have the respective variances and covariances, we can build the positive definite covariance matrix for $p_t(\bar{\mathcal{D}}(r_j^0) = \text{vec}(R_j^0)|r_j^t)$, which at diffusion step t is $w_t^2\Sigma_{\hat{r}_j^t}$. To show that $\Sigma_{\hat{r}_j^t}$ is a positive definite matrix, we use Sylvesters criterion Gilbert (1991). It states that a symmetric matrix $\Sigma \in \mathbb{R}^{N\times N}$ is positive definite if each upper left corner matrix of sizes $n \in \{1, \ldots, N\}$ has a positive determinant. Since $\Sigma_{\hat{r}_j^t}$ is a relatively large matrix of size 9×9, we use SymPy Meurer et al. (2017) to compute each corner matrix determinant.

We find that each determinant equals a positive number that depends on w_t^2. The first 6 corner matrices trivially have determinant 1 since they are all identity matrices. The determinant for $n = \{7, 8, 9\}$ (termed $\Sigma_{\hat{r}_j^t}^{n\times n}$) are the following:

$$\det(\Sigma_{\hat{r}_j^t}^{7\times 7}) = 2w_t^2 \qquad \det(\Sigma_{\hat{r}_j^t}^{8\times 8}) = (2w_t^2)^2 \qquad \det(\Sigma_{\hat{r}_j^t}) = (2w_t^2)^3$$

thus proving that $\Sigma_{\hat{r}_j^t}$ is positive definite.

Since we assume every rotation in r_M^t is independent of each other, when we combine the $w_t^2\Sigma_{\hat{r}_j^t}$ for each rotation, we get a large positive definite matrix $w_t^2\Sigma_{\hat{r}_M^t}$. Thus, we can approximate the distribution $p_t(\mathcal{D}(r_M^0)|r_M^t)$ as a Gaussian using the mean and the derived positive definite matrix.

□

B Implementation Details

As stated earlier, we use BoDiffusion as the base Diffusion model for `InPose`. The conditional inputs for CFG are the joint rotations, locations, the joint velocities, and the angular velocities from the 3 measured joints. The rotations are provided using the 6DoF representation. The output of the network is the 22 joint rotations Θ_M in the local reference frame, expressed in 6DoF. The model is designed to output a frame of 41 samples. For sequences larger than 41 samples, it uses an overlap of 20 samples between successive frames.

Since we require the model to output joint angles r_M in the global frame, we fine-tune the model using our training datasets to output global rotation angles. We use the weights provided by the BoDiffusion authors to initialise fine-tuning. We also tune the model to use only joint rotations and angular velocities for CFG by training the model on a subset of samples with zeroed out joint locations and velocities. During inverse-guidance-based inference, we can similarly zero out the location and velocity CFG inputs to the diffusion network.

This model has about 22M parameters. Finetuning is done using the DDPM framework for $N = 1000$ steps. Inference is done for $N = 50$ steps using DDIM, for both pure CFG-based guidance as well as `InPose`'s inverse guidance. We used an Nvidia RTX Titan GPU for training for 2 days with a batch size of 256.

Inverse Guidance: For the ΠGDM-based inverse guidance term $\nabla_{r^t} \log p_t(l_m|r_M^t)$, we used an additional scale parameter, which we found was useful in improving performance for all metrics. Increasing this term led to minor increases in MPJPE performance at the cost of worse MPJRE error. Secondly, since the positive definite matrix $\Sigma_{\hat{r}_M^t}$ is very large, its usage substantially slows down run-time when its inverse is computed. We found that setting $\Sigma_{\hat{r}_M^t}$ to simply the identity matrix doesn't degrade performance with the benefit of faster run-time.

C More Results

Performance with varying upper body(Part2): We show more results with varying upper body shape in Table 2. We once again see that `InPose` is able to generalize better to changes in relative bone lengths, but lags behind the other baselines in lower body pose estimation.

Performance with scaling body size on Protocol 2: Figures 6a and 6b are the performance results from scaling body size using Protocol 2. The tests were conducted in a manner similar to what is

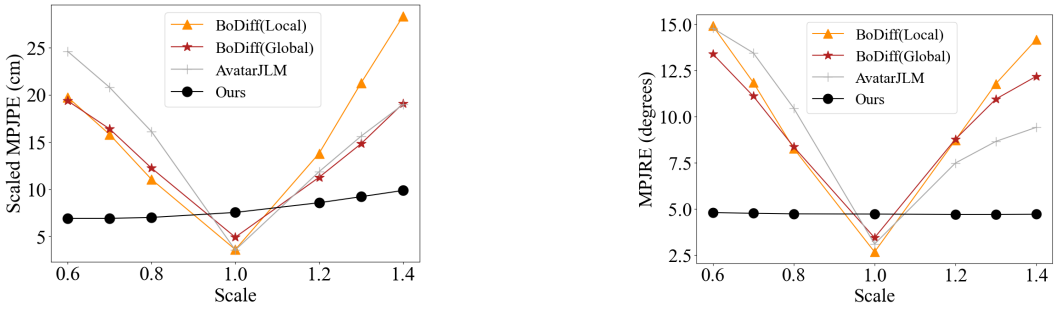
(a) Results with Upper Body shape variation (Protocol 1) (↓ is better)

Algorithm	Upper body ×0.7				Arms ×1.4				Arms ×0.7			
	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE
AvatarJLM	20.59	10.33	19.02	23.32	9.21	7.21	7.79	**11.84**	7.90	8.15	5.88	**11.46**
BoDiffusion(Local)	9.00	11.29	6.68	13.2	15.57	11.99	13.32	19.29	9.36	8.84	7.12	13.27
BoDiffusion(Global)	7.44	10.56	4.35	**12.81**	9.51	9.02	7.18	13.66	7.83	8.96	4.89	12.88
`InPose`	**6.67**	**6.17**	**2.42**	13.80	**8.25**	**6.67**	**3.97**	15.4	**7.22**	**6.20**	**2.92**	14.35

(b) Results with Upper Body shape variation (Protocol 2) (↓ is better)

Algorithm	Upper body ×0.7				Arms ×1.4				Arms ×0.7			
	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE	MPJPE	MPJRE	UPE	LPE
AvatarJLM	21.63	9.06	19.86	24.56	10.30	8.02	8.57	**13.36**	7.44	6.67	6.17	**9.65**
BoDiffusion(Local)	7.72	9.14	6.06	**10.67**	15.93	11.11	12.77	21.11	7.61	6.95	6.30	9.91
BoDiffusion(Global)	9.17	9.32	6.72	13.38	13.19	10.76	9.83	18.94	9.19	7.17	7.13	12.63
`InPose`	**6.53**	**4.75**	**2.11**	13.84	**7.80**	**4.74**	**3.21**	15.24	**7.34**	**4.78**	**2.65**	14.97

Table 2: Algorithm comparison for varying upper body shape. The metrics used are Mean Joint Position Error(MPJPE) in cm, Mean Joint Rotation Error(MPJRE) in degrees, Upper Joint Position Error(UPE) in cm, and Lower Joint Position Error(LPE) in cm. The lower body shape was kept the same, while the upper body bone lengths were scaled.



(a) MPJPE divided by Scale, vs body scale

(b) MPJRE vs body scale

Figure 6: Performance with body-size scaling using Protocol 2.

described in Section 4.1. We once again see that `InPose` performs worse than the baselines in the base case, where the same body shape that was used during training is used for testing. However, when the body size is scaled, `InPose` outperforms the baselines. MPJRE, which measures the scale-free performance, remains the same, while the scale-dependent MPJPE varies proportionally to the scale.

Robustness to measurement noise(cntd): Here are the rotation error results from the robustness study we performed in the Section 4.1. As stated earlier, we injected zero-mean i.i.d. Gaussian noise into the input location streams and computed the estimation errors, while maintaining the *default* body shape. Fig. 7 shows the rotation error(MPJRE) under increasing Gaussian noise variance in the location measurements. As with the location error, `InPose` stays flat while other baselines degrade with noise. Since the ΠGDM inverse guidance objective is formulated to be robust to noise, the prior is able to synthesize poses that are realistic while also obeying the guidance provided by the location inputs.

Errors in joint length estimates: Another study involves illustrating `InPose`'s sensitivity to errors in the joint length estimates, where we show how the MPJPE and MPJRE worsen when there is an error in our knowledge of the user's bone lengths. We add white Gaussian noise to the true bone lengths while constructing our measurement matrix $\mathcal{A}$. The body shape parameters are set to the defaults, and the measurements y_m are unaltered.

Figure 8 shows the results for this experiment. We see that `InPose` is quite sensitive to bone length estimation error. The algorithm can tolerate low errors within 1 cm, but begins to diverge any higher than that. Making the algorithm robust to bone length estimation errors will be looked at for future work.

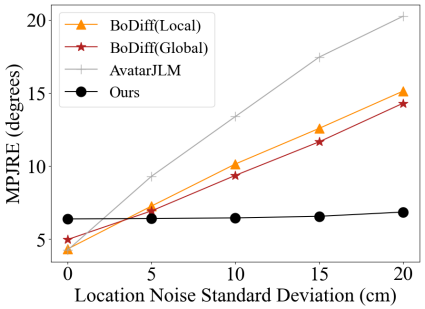


Figure 7: Rotation error vs location noise.

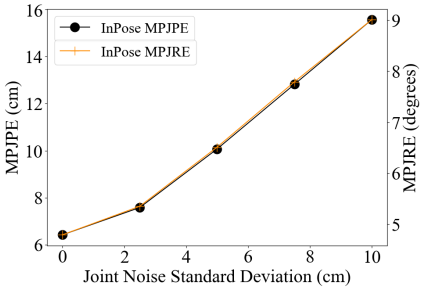
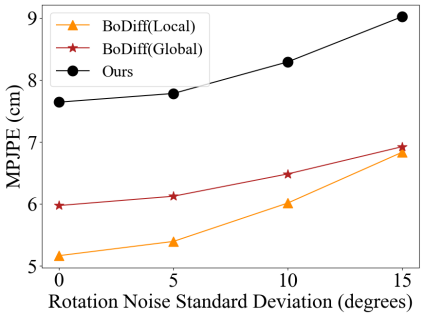
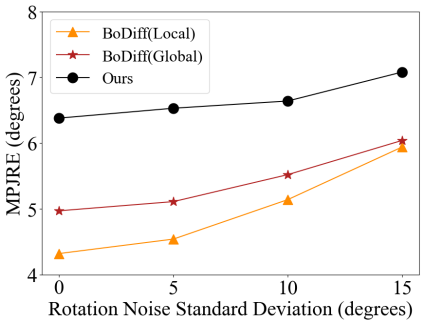


Figure 8: Performance with joint length error. The left axis is the MPJPE, and the right axis is the MPJRE.



(a) Position error vs rotation noise



(b) Rotation error vs rotation noise

Figure 9: Performance with additive white noise in rotation measurements.

Robustness to rotation noise: We conduct another robustness study with rotation measurements, similar to the location error study described in Section 4.1. Here, we add Gaussian noise to the rotation measurements while keeping the location measurements noise-free. We then compare the `InPose` with the two diffusion-based baselines. The results are summarized in Figure 9. We find that all the diffusion-based algorithms are relatively robust to rotation measurement noise, with both the rotation and position errors rising steadily as rotation noise increases.

Need for using location measurements: An important question is whether purely using the scale-free rotations r_m is better than including the scale-dependent location measurements l_m. Since r_m is scale-free, we could potentially use only r_m to estimate pose. As an ablation study, we compare BoDiffusion(Global) and `InPose`, against BoDiffusion(Global) without using l_m for CFG. Since BoDiffusion(Global) was trained to accept only r_m or both $\{r_m, l_m\}$ for CFG-based conditioning, this serves as an apples-to-apples comparison.

Table 3 shows the results of this comparison using Protocol 1, using the default body shape to generate l_m. As expected, BoDiffusion(Global) with l_m for CFG performs the best amongst the three algorithms. `InPose` is next, performing much better than BoDiffusion(Global) without l_m, illustrating the importance of l_m for pose estimation.

Metric	`InPose`	BoDiffusion(Global)	BoDiffusion(Global) no l_m
MPJPE(cm)	7.64	**5.97**	15.98
MPJRE(°)	6.38	**4.97**	8.71

Table 3: Comparison between `InPose`, BoDiffusion(Global) DPS and BoDiffusion(Global) with no l_m as input for CFG using Protocol 1

Using Inverse-Guidance on Local Joint Angle Representation: Considering that the BoDiffusion(Local) model outperforms the BoDiffusion(Global) model, an important question is why not use

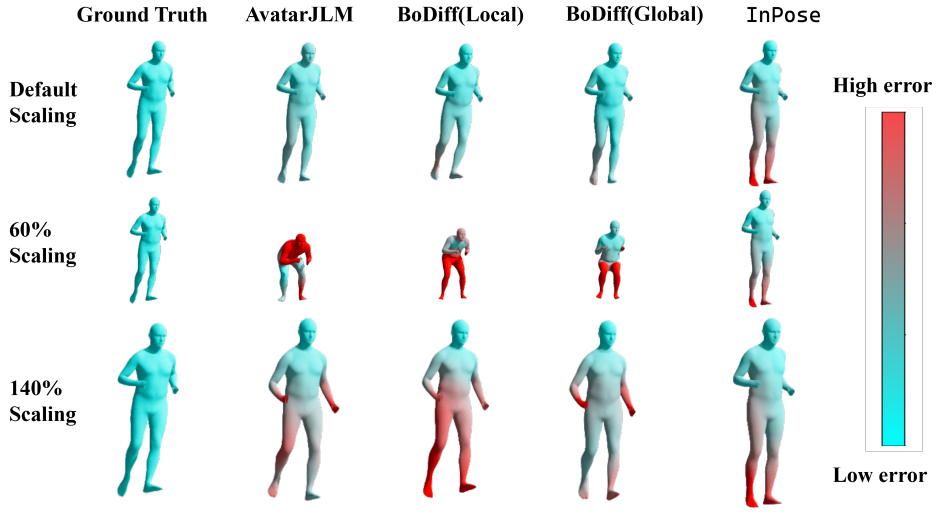


Figure 10: More qualitative results comparing InPose with the Baselines with varying body scale. Relative body shape and pose have been kept constant.

the local joint angle output Θ_M for inverse guidance. Firstly, the linear system $\mathcal{A}$ requires global joint angles R_M, hence we would have to transform the local joint angles Θ_M using the recursive equation described in Section 2. Because this equation is recursive, the transformation from $\Theta_M \rightarrow R_M$ is a higher-order polynomial function. We could use DPS Chung et al. (2023) as it allows for inverse guidance through differentiable nonlinear measurement functions. However, in our experiments, we found that this does not work well in practice, with the relatively low number of diffusion steps that we use $(N = 50)$ during inference. When we ran BoDiffusion(Local) with no joint position l_m conditioning, we saw no performance improvement when a DPS-based location inverse guidance term was added.

More Qualitative Results: We show another qualitative sample in Figure 10, comparing InPose with the baseline algorithms in a running pose. Once again, we keep the same relative body shape and modify the scale only. InPose falls behind the baselines when it comes to estimating lower body pose, especially the feet. However, it is able to generalize across all body scales tested, with the error at the arms being lower compared to the baseline algorithms.

D Limitations and Future Work

The biggest limitation of InPose is that the root translation isn't directly incorporated into the algorithm, which reduces its capability to infer lower-body pose. As described in Section 3.3, we remove the component of $l_1(i)$ present in the measurements to set up a linear system that maps the joint rotations r_M to the measured locations l_m. Thus, the algorithm can only use the prior to infer translation, and thereby infer lower body movement.

One technique that we could explore in future work is to incorporate foot contact constraints to set up another kinematic linear system to map joint rotations to root translation. The foot that is in contact with the ground is temporarily stationary with respect to the global coordinate frame, assuming no sliding takes place. As far as we know, inferring foot contact from just the 3 measured sensors at the head and the wrists is a difficult problem. Furthermore, in our preliminary experiments, we found that even when the foot contact is known, the kinematic system that we use for inverse guidance tends to drive the output towards stiff and unnatural motion.

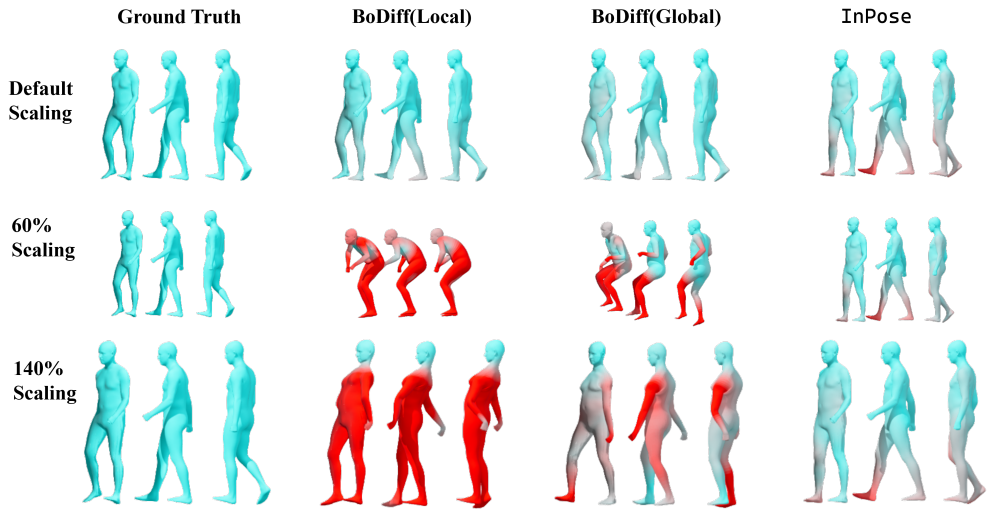


Figure 11: More scaling qualitative results comparing InPose with the Diffusion-based Baselines with varying body scale. InPose is able to infer lower body movement using the prior learnt from hand motion during walking.

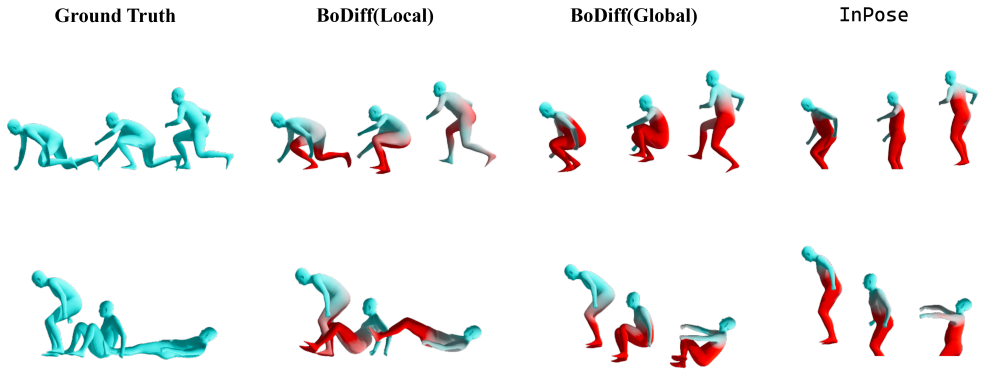


Figure 12: Some catastrophic failure cases of InPose. This occurs when the user gets extremely close to the ground. Without root translation information,InPose catastrophically fails, as it is unable to infer the user's posture