PHD: Personalized 3D Human Body Fitting with Point Diffusion

Supplementary Material

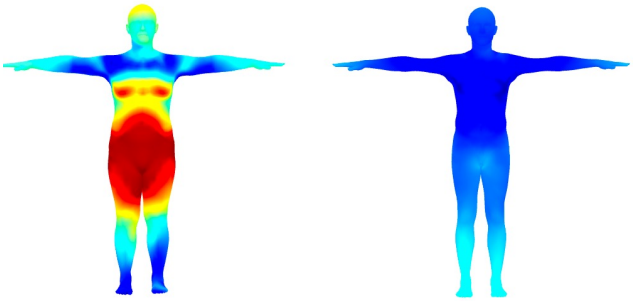

Figure 7. Importance of weight regularization. While the body height is correct, without weight regularization, the fitting is prone to converge at high BMI shape parameters.

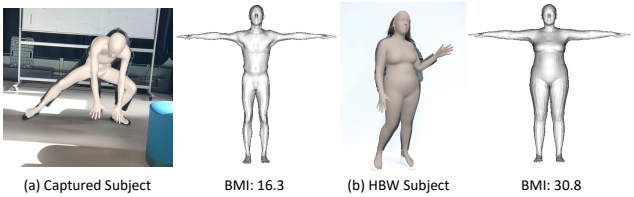

Figure 8. SHAPify is also capable of estimating shapes of subject with high and low BMI values, which allows us to perform customized pose estimation for diverse human subjects.

6. Implementation Details

6.1. SHAPify Details

In the optimization of SHAPify, we initialize the pose parameters θ as the rest pose (T-/I-pose) and the pelvis position $\mathbf{p}$ to:

$$\mathbf{p} = [\frac{x_p - c_x}{f} Z, \frac{y_p - c_y}{f} Z, Z], \tag{11}$$

where f is the camera focal length and $(x_p, y_p), (c_x, c_y)$ are the pelvis pixels and camera center on the image, respectively. Z is the depth of the pelvis, which we approximate as $Z = f \cdot SW_{SMPL}/SW_{kp}$. Note that SW_{SMPL} is the shoulder width of the SMPL mean shape, and SW_{kp} is the length of shoulder keypoints on the 2D image. This approximation holds because the horizontal line on the image is not affected by the pitch angles of global orientation, and we can assume that the roll and yaw angles of the pelvis orientation ϕ are typically small in the frame of rest poses. We also initialize the roll and yaw angles to 0 and update them with small learning rates.

In the regularization term of SHAPify, we calculate body heights and weights from the SMPL body meshes. The heights ($H(\cdot)$) are calculated as the distance between the top of the head (Vertex# 411) to the center of feet (Vertex# 3439, 6839). The heights ($W(\cdot)$) are the volume of human body meshes multiplied by the body density (985 kg/m^3). We use $\lambda_\beta = 0.1$, $\lambda_h = 100$, and $\lambda_w = 10$ if the body measurements are available, and $\lambda_\beta = 1$, $\lambda_h = 1$, and $\lambda_w = 1$ if not. Without using the body measurements, our method achieved a 14mm joint error and a 13mm vertex error (Tab. 3), which is only slightly higher than using the body measurements. Moreover, we found the body weight regularization term crucial. Without such a constraint, the fitting is prone to converge to shapes with large bellies. (See Fig. 7)

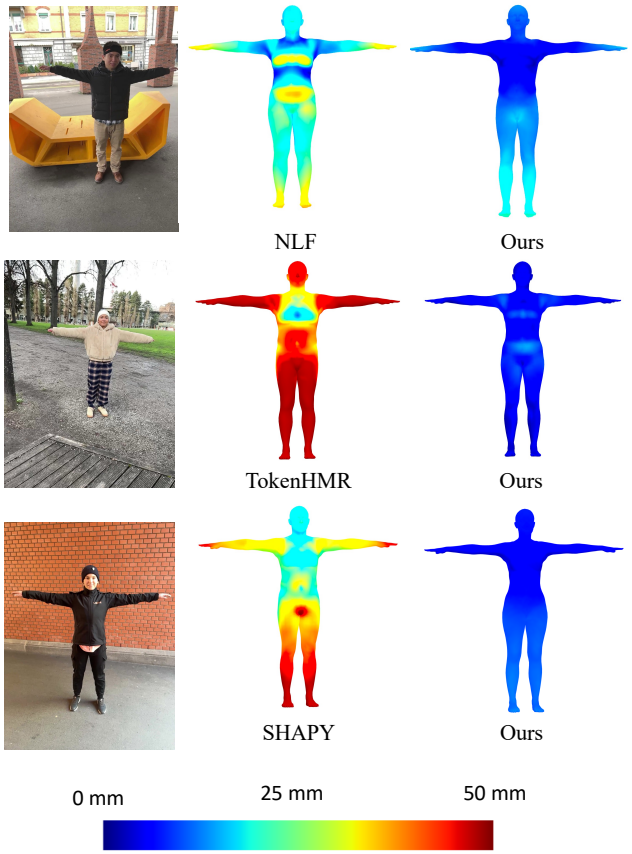

Figure 9. **Visualization of shape estimation errors** on sequences in EMDB. See also Tab. 3 for quantitative results.

6.2. PointDiT Architecture and Training

We show the detailed network architecture of PointDit in Fig. 11. More specifically, the PointDiT model contains 20 DiT blocks and is operated at a dimension of 512. We use the frozen ViTBackbone [77] to extract 256×256 image

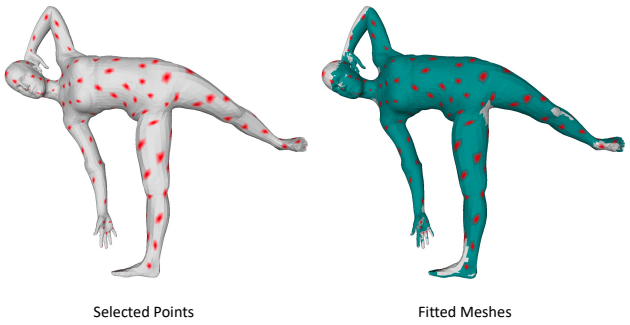

Figure 10. Visualization of selected body surface points. *Left*: We use the 238 vertices and 45 joints to fit the SMPL parameters. *Right*: The fitted body mesh (in green color) is highly aligned with the original ground-truth body mesh.

features of the shape of $16 \times 16 \times 1280$ and heatmaps of the shape of $64 \times 64 \times 17$. We add the image features, heatmaps, and positional embeddings together to obtain the final conditional features $\mathbf{c}$. Our body point clouds are made up of 45 SMPL joints and 238 vertices from the SMPL surface (see Fig. 10). This was chosen by the accuracy of the Point Fitter in NLF [61]. The point clouds used for diffusion are with the shapes of 283×3, and we normalize the points to zero mean and unit variance before adding noise. We use the rectified flow formulation [13, 44] in the diffusion model to reduce the number of denoising steps during inference, as described in the main paper. Both conditional features and point clouds are projected to 512-dimensional tokens and fed into the transformer. In the output layer, we project the tokens back to 3-dimensional points and de-normalize them. The image conditional tokens are only used for self-attention conditioning and are discarded in the final layer.

We train our PointDiT model using the synthetic BEDLAM dataset. We apply standard data augmentations [2, 12] to the provided image crops and ground-truth annotations. Our training consists of two stages. In the first stage, we set all conditional shape parameters β to zero, allowing the model to focus on sampling body point clouds corresponding to the conditional images. In the second stage, we learn the correct body shape of point clouds using the ground-truth shape parameters as conditions. For training, we utilize a batch size of 512 images and set the learning rate to 10^{-5} with the AdamW optimizer. The training takes approximately 1 day for the first stage, with 12K iterations, and another 2 days for the second stage, with 30K iterations, on 8 NVIDIA V100 GPUs. The training scheduler and reweighting factors follow the same configuration as in Stable Diffusion 3 [13]. We employ a dropout rate of 0.05 for the image and shape conditioning.

6.3. Point Distillation and Body Fitting Details

In the body fitting stage, we can initialize pose θ_0 and global orientation ϕ_0 with either our sampled points $\mathbf{x}_0$ or the results from a regressor. To initialize the pelvis position $\mathbf{p}_0$, we solve the weighted least squares problem by plugging in θ_0, ϕ_0 to Eq. (10). Afterward, we optimize 100 iterations per image with a learning rate of $3e-3$ for optimizing θ and $1e-3$ for optimizing $\phi, \mathbf{p}$ with the AdamW optimizer. The λ_{data} is set to 1.0 and the λ_{prior} is set to 100, (λ_p, λ_ϕ, λ_θ) are set to (0.1, 0.1, 1.0). Every 10 iterations, we resample the points again from the fitted meshes.

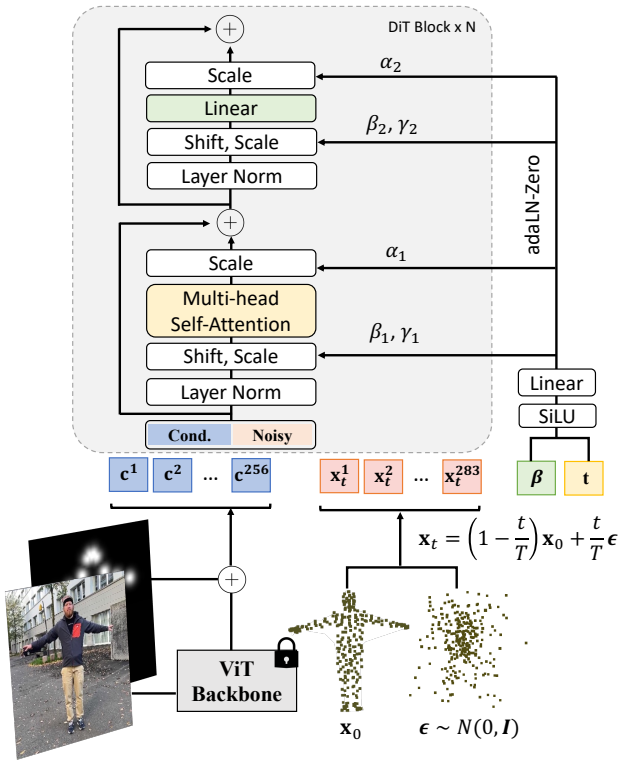

Figure 11. **Network architecture of PointDiT**. Our point clouds are made up of 45 SMPL joints and 238 vertices from the SMPL surface. This was chosen in accordance with the accuracy of the SMPL Fitter in NLF [61]. Hence, the point clouds are of 283-dimensional. We use the rectified flow formulation [13, 44] in the diffusion model to reduce the number of denoising steps as described in the main paper.

6.4. Inference

We extract 2D keypoints using Sapiens [32], and we preprocess keypoints and bounding boxes (image crops) before inference. SHAPify is a lightweight optimization algorithm that can be run on a CPU and takes roughly 1 second for each image. For body fitting inference, we tested our model on a single NVIDIA RTX 3090 GPU. We set the number of denoising steps to $T = 5$, and it takes approximately 1 second per frame without any parallelization (batch size $= 1$). As a reference, ScoreHMR [67] requires $T = 20$ denoising steps and takes about 3 seconds per frame under the same data setup.

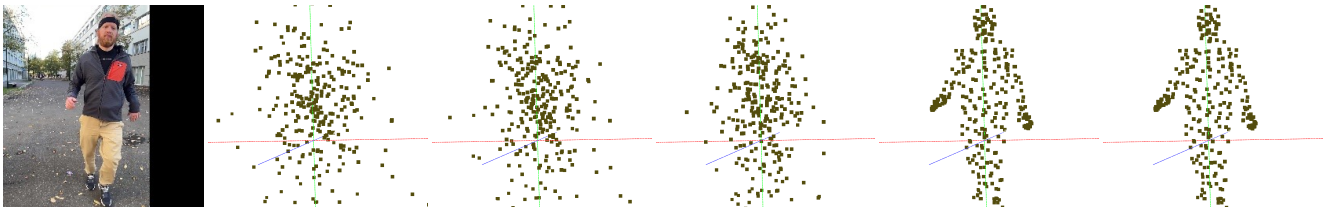

Figure 12. **Example of the body points at each denoising time step.** Using the rectified flow formulation allows us to efficiently sample body point clouds in as few as 5 denoising steps.

Method	MPJPE ↓ (mm)	PA-MPJPE ↓ (mm)	MVE ↓ (mm)	PA-MVE ↓(mm)
IK (24 joints) [81]	67.8	49.3	81.9	60.3
25% Points	63.8	45.3	78.1	59.0
50% Points	63.2	44.7	73.7	53.6
100% Points	**62.6**	**44.4**	**72.9**	**53.1**

Table 6. **Effectiveness of Point Fitter.** We report the fitting accuracy on the EMDB "P1-14" sequence using IK and Point Fitter. We also analyze the robustness of Point Fitter by dropping out surface points during fitting.

7. More Experimental Results

7.1. Shape Estimation

We visualize the results of SHAPify against existing methods on EMDB in Fig. 9. Even on a simple T-pose image, existing methods predict shapes with significant errors, especially when subjects wear loose clothing like jackets.

EMDB mainly contains subjects with moderate body shape. Since real-world data of extreme body shapes is limited, in Fig. 8, we visualize the SHAPify and PHD's fitting results on a newly recorded slim subject and another high-BMI subject from the HBW [7] dataset. Because PointDiT is trained on BEDLAM, which contains body shapes across a wide BMI range (17.5 to 42.5), our method is able to generalize to human subjects with diverse body shapes.

7.2. Effectiveness of Point Fitter

In Tab. 6, we analyze the effect of the point cloud size by retaining 25%, 50% points for fitting. Furthermore, Inverse Kinematics (IK) [48, 81] is commonly used for fitting 3D joints back to the parameter space. We also design a joint-only baseline by replacing the Point Fitter with an open-source IK solver [81]. The joint-only IK is less accurate than Point Fitter and incurs expensive optimization loops that slow down the method (15 seconds per frame). In contrast, the Point Fitter is faster but requires more points to ensure accuracy. When only 25% of points are retained, we observe a clear increase in MVE. Overall, our design is both efficient and accurate.

Method	MPJPE ↓	MPJPE-PA ↓	MVE ↓	MVE-PA ↓
ScoreHMR [67] Sample init.	94.3	66.6	122.3	94.7
PHD (Ours) Sample init.	84.2	51.0	100.4	67.6
HMR2.0b [15] init.	81.7	54.2	93.5	67.7
w/ SMPLify [3]	-	60.1(+5.9)	-	-
w/ ScoreHMR [67]	-	51.1(-3.1)	-	-
w/ ScoreHMR* [67]	75.7(-6.0)	51.2(-3.0)	87.5(-6.0)	65.4(-2.3)
w/ PHD (Ours)	80.5(-1.2)	44.4(-9.8)	93.3(-0.2)	57.3(-10.4)
CameraHMR [54] init.	62.1	38.5	72.9	-
w/ ScoreHMR* [67]	59.6(-2.5)	38.0(-0.5)	71.5(-1.4)	51.1
w/ PHD (Ours)	**59.4**(-2.7)	**37.5**(-1.0)	**71.3**(-1.6)	**50.9**

Table 7. **Pelvis-aligned local pose accuracy** on 3DPW (14 joints). *Top*: Body fitting using sampled pose from the body priors. *Middle*: Body fitting using the HMR2.0b predictions as initialization, which is originally used in the ScoreHMR [67] paper. *Bottom*: Body fitting using the CameraHMR initialization. * denotes re-run with the same input shape and focal length.

7.3. Results on 3DPW

We conduct similar evaluation in Sec. 4.2 on the 3DPW dataset [73]. Note that for this evaluation benchmark, we don't have access to the body measurements of the subjects, and we use the estimated shapes for evaluation. Overall, we observed similar performance improvements as on EMDB, but less pronounced. We identified two major issues with this benchmark.

First, as noted in EMDB [31], performance on 3DPW has saturated due to its limited pose diversity, which motivates the need for EMDB as a diverse and challenging benchmark. Many recent methods report trends consistent with ours, *i.e.*, smaller improvements on 3DPW compared to EMDB. Second, unlike EMDB, which uses EM sensors and SLAM to reconstruct accurate 3D camera trajectories and body poses, 3DPW estimates them through a joint optimization process based on 2D keypoints and IMU sensors. This approach introduces systematic errors of approximately 26 mm, as reported in the original paper, resulting in a biased pose distribution. Since this distribution deviates from that of synthetic training data, the learned prior becomes less effective during the fitting stage. Consequently, recent methods [61, 65] often finetune on the 3DPW training set to better align with its data distribution. Overall, we believe that consistent performance gains on EMDB without finetuning are more reliable indicators of robustness to

	EMDB [31]				3DPW [73]			
Method	MPJPE ↓	MPJPE-PA ↓	MVE ↓	MVE-PA ↓	MPJPE ↓	MPJPE-PA ↓	MVE ↓	MVE-PA ↓
HMR2.0b [15]	117.4	78.0	140.5	94.0	81.8	54.4	93.5	67.8
PARE [34]	113.9	72.2	133.2	85.4	74.5	46.5	88.6	-
HMR2.0a [15]	98.3	60.7	120.8	-	69.8	44.4	82.2	-
TokenHMR [12]	88.1	49.8	104.2	-	70.5	43.8	86.0	-
CameraHMR [54]	70.3	43.3	81.7	-	62.1	38.5	72.9	-
NLF [61]	<u>68.4</u>	**40.9**	<u>80.6</u>	**51.1**	**59.0**	**36.5**	**69.7**	**48.8**
LGD [66]	115.8	81.1	140.6	95.7	-	59.8	-	-
ReFit [74]	88.0	58.6	104.5	-	65.3	40.5	75.1	-
WHAM* [65]	79.7	50.4	94.4	-	**57.8***	**35.9***	**68.7***	-
ScoreHMR [67] (CameraHMR init.)	74.9	45.0	89.0	54.5	59.6	38.0	71.5	51.1
Ours (Sample init.)	73.6	49.2	86.4	59.1	84.2	51.0	100.4	67.6
Ours (HMR2.0b init.)	73.2	47.4	86.4	58.5	80.5	44.4	93.3	57.3
Ours (CameraHMR init.)	**62.5**	<u>42.4</u>	**74.6**	<u>51.6</u>	<u>59.4</u>	<u>37.5</u>	<u>71.3</u>	<u>50.9</u>

Table 8. **Comparisons of body fitting with learning-based methods on in-the-wild benchmarks**. * indicates fine-tuning on 3DPW, **bold** is best result, *underlined* second best. Our method, initialized with CameraHMR, either beats the state of the art or performs second best with a narrow margin. This is remarkable as NLF trained on multiple datasets while we only train on synthetic BEDLAM.

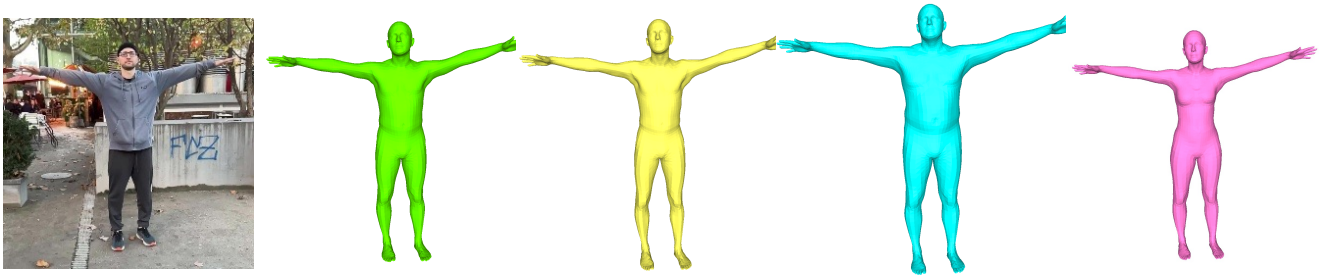

Figure 13. **Example of shape conditioning.** Give the same input image but different β for conditioning, PointDiT can sample point clouds in different body shapes. We use the SMPL fitter to convert the point clouds to body meshes for visualization.

diverse and challenging poses.

7.4. Comparison to Learning-Based Methods

We extensively compare our method against learning-based approaches in Tab. 8. Notably, our model is trained exclusively on the synthetic BEDLAM dataset (1M images), whereas other methods are trained on a combination of real-world datasets such as Human3.6M [23], MPI-INF-3DHP [49], and in the case of NLF [61], over 40 datasets.

On the EMDB benchmark, when initializing our optimization with poses randomly sampled from PointDiT, we observe slightly higher errors compared to CameraHMR. This is likely due to domain gaps between the synthetic training images and real test-time images. However, when using CameraHMR for pose initialization, our method either outperforms the state-of-the-art (NLF [61]) or comes in a close second. On the 3DPW dataset, initializing with PointDiT-sampled poses results in suboptimal performance, primarily due to the biased pose distribution discussed in Sec. 7.3 and the domain mismatch between training and test images. To address this, we use CameraHMR for pose initialization and achieve the second-best performance on 3DPW. It is worth noting that the (PA-)MPJPE values on 3DPW have saturated and are close to the reported systematic error (26mm). As such, it is difficult to determine whether improvements reflect actual accuracy gains or overfitting to inherent dataset biases. Consequently, we argue that the evaluations on EMDB provide a more meaningful reflection of true pelvis-aligned 3D pose accuracy.

7.5. DiTPose Sampling

In Fig. 12, we visualize the point clouds denoising process of our PointDiT model. Our model leverages the rectified flow formulation to train a diffusion model, which allows for sampling body point clouds in as few as 5 denoising steps. In Fig. 13, we demonstrate the effectiveness of shape conditioning in our PointDiT model. Given the same input images but with different conditional shape parameters, our model samples diverse body shapes corresponding to the body pose described in the input images.

7.6. Effect of Shape on Pelvis Position

In Fig. 15, we illustrate how an incorrect body shape adversely affects pelvis positioning. Using incorrect shape parameters, such as mean shape, leads to using the wrong bone lengths for body fitting. This will not only affect the accuracy of the local pose but also the pelvis positions in

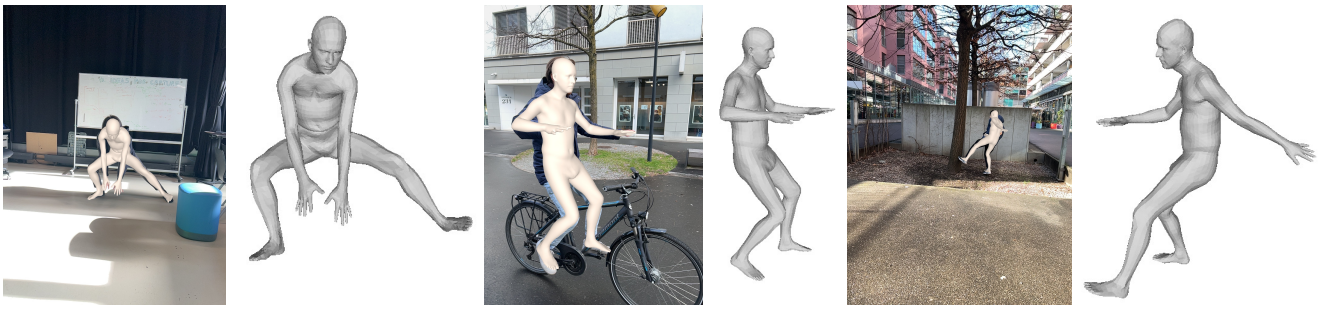

Figure 14. **Applications on in-the-wild capturing**. Given the body measurements of human subjects, our method allows for in-the-wild human performance capturing using a modern smartphone.

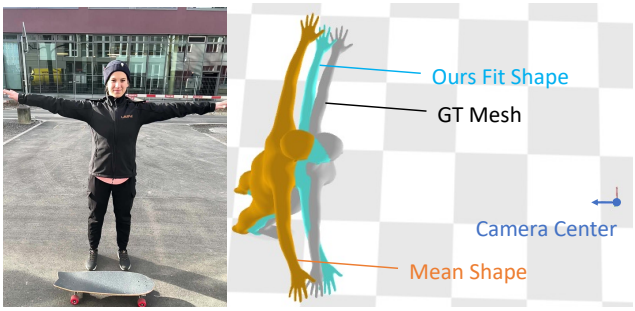

Figure 15. **Effect of shape errors on pelvis accuracy**. The mean shape of ScoreHMR's predictions is less accurate which leads to pelvis misalignments as the pose must correct for shape errors. In contrast, our shape estimate (from SHAPify) is more accurate and so a better pelvis position can be estimated.

the camera coordinate. Please also refer to our video for a better visualization of this effect.

7.7. More qualitative results

We present more qualitative results with ScoreHRM [67] in Fig. 17 and WHAM [65] in Fig. 18. In Fig. 14, we also showcase results of in-the-wild human performance capturing using our method with a modern smartphone.

8. Discussion

8.1. Applications

Our method is beneficial for in-the-wild avatar reconstruction [17, 18, 51, 76]. Existing avatar reconstruction methods all require accurate pose fitting as a starting point, with the most common practice being to use the averaged shape from a regressor and refine the poses using 2D keypoints or by jointly optimizing poses with appearance cues [46]. However, these strategies are only effective when pose errors are small and correctable. As discussed earlier, generalized regressors can easily produce implausible 3D poses, which prevent the avatar from learning accurate pose-dependent appearances. Therefore, PHD offers a simple and effective solution by providing more accurate pose and shape inputs for in-the-wild avatar reconstruction. Furthermore, we believe that accurate personal shape information is a key factor for future avatar-centric applications, such as 3D virtual try-on and digital human interaction.

(a) Wrong Keypoint Detection (b) Left-right Keypoint Flipped

Figure 16. **Failure cases of our method**. (a) Significantly wrong 2D keypoint detections that our pose prior cannot counteract. (b) The keypoint detector swapped left and right, which is difficult to recover from.

8.2. Absolute Pose Metrics

In Sec. 4.2, we emphasized the importance of absolute pose accuracy in the camera coordinate system and showed that most state-of-the-art pose regressors failed to achieve desirable results. Recently, a new line of work [25, 64, 65, 78, 79] has pursued a similar idea in a slightly different setup by estimating body poses and trajectories in the world coordinate system. Their evaluation metric, **G-MPJPE**, is determined by two factors: (1) how well the human poses and motion over time are estimated, and (2) how accurately the camera pose trajectories are recovered. Consequently, these approaches usually involve full-video SLAM tracking and global optimization over body and camera poses. This metric is more suitable for video-based methods.

In contrast, our proposed camera-coordinate metric, **C-MPJPE**, is better suited for per-frame or online methods that can be used for on-device computations (like ours). C-MPJPE disentangles the error caused by camera poses and evaluates only the accuracy of the body pose. Because personal shape (scale) information is fixed in our problem setting, C-MPJPE is also correlated with 2D alignment accuracy, as the 2D projection is definitive when the body scale

Figure 17. **Quantitative comparisons on EMDB**. While the 2D reprojections seem to be correct, ScoreHMR frequently produces implausible body poses such as bending knees and self-penetration. PHD addresses these issues with a stronger 3D pose prior, PointDiT.

is fixed. It is also worth noting that C-MPJPE can be converted to G-MPJPE given the camera extrinsics over time.

8.3. Limitations and Future Work

Hands and faces. Currently, our work focuses on fitting SMPL parameters to videos due to the limited availability of evaluation benchmarks for in-the-wild human pose estimation. It is worth noting that both our method and the PoseDiT model are capable of being easily extended to any format of parametric models, such as SMPL-X. One of our future goals is to extend PointDiT to handle hand gestures and facial expressions. This can be readily achieved by selecting the corresponding data from the BEDLAM training set in SMPL-X format. We believe that for future perceptual AI systems, estimating holistic human body poses with precise hand and facial information will be indispensable.

2D keypoint detection. Although PointDiT is an effective 3D prior, our fitting method still depends on the accuracy of 2D keypoint detections. When these detections fail significantly (see Fig. 16), our method struggles to recover. An exciting direction for future work is to explore appearance-/feature-based fitting by leveraging powerful pretrained foundation models such as DINOv2 [53]. This approach could make the fitting process more robust to challenging poses that 2D keypoint detectors could fail.

Temporal Smoothness. Our method and the evaluations in this paper follow a per-frame setting. This setup is closer to real-world on-device computation and allows for a fair assessment of the model's performance without relying on future or past information. While the metrics appear promising, we observed noticeable jittering across frames. Additionally, our method does not hallucinate continuous body motion when parts of the body are occluded or unobserved. To address these issues, a promising direction is to extend PointDiT into a temporal model that incorporates motion priors rather than predicting a single pose at a time. However, training such a model would require even more data beyond what BEDLAM provides. We believe that video generative AI can help address this challenge by offering a wide range of realistic motion data.

Learning-based optimization. Our method can run at 1 FPS on a common GPU (RTX 3090). While this is already faster than most existing pose fitting methods, it is still not yet suitable for real-time pose tracking. This limitation stems from the nature of optimization loops. Even though PointDiT provides strong 3D guidance during the fitting process, it still requires several iterations to converge. One way to mitigate these long optimization loops is to exploit a "learning to optimize" approach [8, 66], allowing the network to memorize optimization trajectories. Potentially,

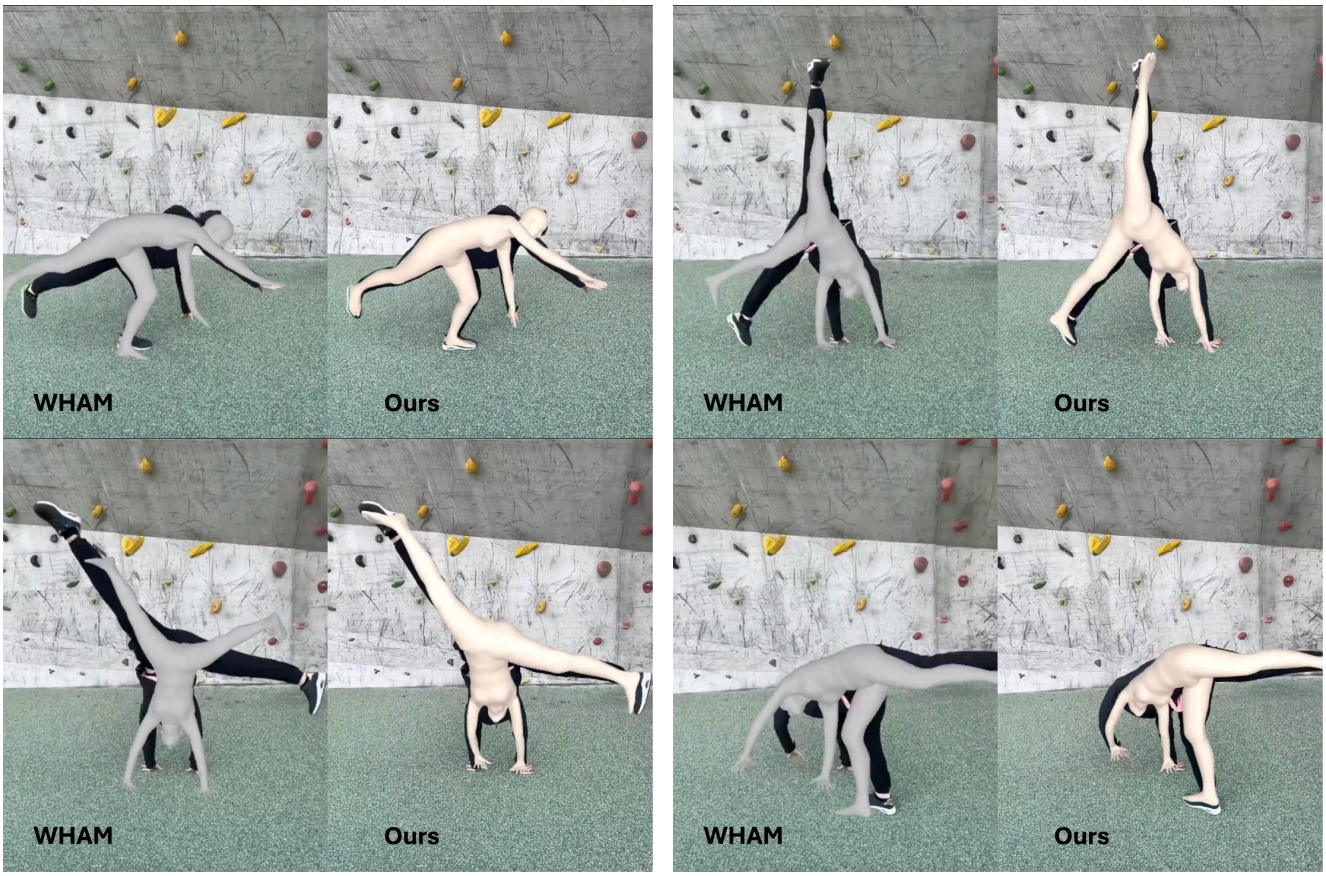

Figure 18. **Quantitative comparisons on EMDB**. When comparing with regression-based method, WHAM, our approach is more robust to challenging poses and achieves better 2D alignments.

it achieves faster inference speeds through a feed-forward network while retaining the accuracy of optimization.