Probabilistic Modeling for Human Mesh Recovery

Nikos Kolotouros[1], Georgios Pavlakos [2], Dinesh Jayaraman [1] Kostas Daniilidis[1]
[1] University of Pennsylvania [2] UC Berkeley

In this Supplementary Material we provide additional details that were not included in the main paper due to space constraints. In Section 1 we extend the discussion about probabilistic pose models. In Section 2 we describe in detail the architecture of our model. Section 3 includes all training details whereas Section 4 describes the datasets used for training and evaluation. FInally, in Section 5 we report additional quantitative and qualitative evaluations.

1. Additional discussion

Heatmap-based methods One additional class of non-parametric probabilistic models for 3D human pose estimation is the heatmap-based methods [14]. Heatmap-based methods allow for pose sampling (for each joint, we first sample a voxel and then sample again uniformly inside each voxel), as well as likelihood evaluation for a given pose. However the main issue with heatmap-based methods is that by design the probability distribution is factorized, *i.e.*,

$$p(\boldsymbol{\theta}|I) = \prod_i p(\boldsymbol{\theta}_i|I), \tag{1}$$

for all joints $\boldsymbol{\theta}_i = (x_i, y_i, z_i)$. This type of model is problematic for sampling because it can lead to unrealistic poses, *e.g.*, in ambiguous cases where the output heatmaps are not unimodal, since each joint location is sampled independently of the other joints. Moreover, it is not possible to extend heatmap-based methods to non-Euclidean output spaces such as the space of SMPL parameters.

2. Architecture details

In this Section we describe in detail the architecture of the proposed model. In Figure 1 we show the design of the proposed flow model and the information flow both in forward and reverse mode. The implementation of f_{coupl} is shown in Figure 2, whereas Figure 3 depicts the architecture of the residual block used in f_{coupl}. The number of channels in the residual block is $n = 1024$. For the partitioning of $x = (x_1, x_2)$, we alternate between the odd and even dimensions of x in each successive coupling layer to allow sufficient information propagation. The flow architecture is the same for both ProHMR and the probabilistic version of Martinez *et al*. [11]. The only difference is that for ProHMR the latent dimension is $d = 6 \cdot 24 = 144$, whereas for the 2D skeleton lifting is $d = 16 \cdot 3 = 48$. We use 16 joints instead of 17, since the pelvis location is always at $(0, 0, 0)$.

3. Training details

We implemented our model using PyTorch [13]. Our Normalizing Flow implementation was based on the `nflows` package [3].

ProHMR We trained our model for 500K iterations with a batch size of 64 using the Adam optimizer [6] with learning rate 0.0001 and weight decay 0.0001. Training takes about 2.5 days on a NVIDIA RTX2080Ti using mixed precision. For the final loss and loss weights, we use a more fine-grained distinction than the generic described in the main manuscript. The final training loss is written as:

$$\begin{aligned} L = & \lambda_{nll} L_{nll} \\ & + \lambda_{exp,2D} L_{exp,2D} + \lambda_{exp,adv} L_{exp,adv} \\ & + \lambda_{mode,2D} L_{mode,2D} + \lambda_{mode,adv} L_{mode,adv} \\ & + \lambda_{mode,\theta} L_{mode,\theta} + + \lambda_{mode,\beta} L_{mode,\beta} \\ & + \lambda_{mode,3D} L_{mode,3D} + \lambda_{orth} L_{orth}, \end{aligned}$$

where $L_{mode,*}$ and $L_{exp,*}$ refer to the corresponding terms included in the loss functions. More specifically, $L_{mode,2D}$ and $L_{exp,2D}$ are the reprojection loss for the mode and the expected reprojection loss respectively, $L_{mode,\theta}$ and $L_{mode,\beta}$ and $L_{mode,3D}$ are penalties on the SMPL parameters and 3D joint locations and $L_{mode,adv}$ and $L_{exp,adv}$ are the adversarial priors. We use ℓ_1 penalty for the losses on the 2D and 3D keypoints and ℓ_2 penalty for the loss on the SMPL parameters. For all losses we sum over all dimensions and then divide by the batch size. The loss weights are: $\lambda_{prob} = 0.001$, $\lambda_{exp,2D} = 0.001$, $\lambda_{exp,adv} = \lambda_{mode,adv} = 0.0005$, $\lambda_{mode,2D} = 0.01$, $\lambda_{mode,3D} = 0.05$, $\lambda_{mode,\theta} = 0.001$, $\lambda_{mode,\beta} = 0.0005$ and $\lambda_{orth} = 0.1$.

2D pose lifting We trained our model for 300K iterations with a batch size of 64 using the Adam optimizer [6] with learning rate 0.0001 and no weight decay. Training takes

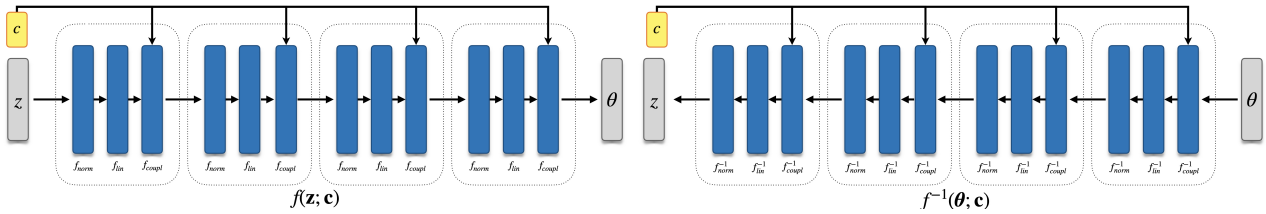

Figure 1: **Normalizing Flow architecture**. The figure shows the implementation of $f(\boldsymbol{\theta};\mathbf{c})$ and its inverse using Normalizing Flows. Left: Behavior of our model in the sampling phase (map $\mathbf{z} \rightarrow \boldsymbol{\theta}$). Right: Behavior of our model in the likelihood evaluation phase (map $\boldsymbol{\theta} \rightarrow \mathbf{z}$).

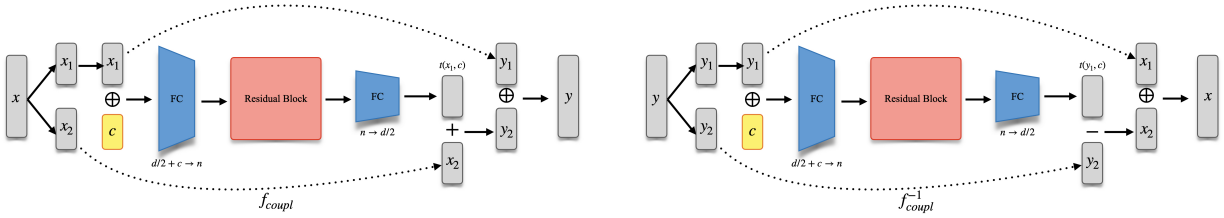

Figure 2: **Coupling layer architecture**. The figure shows the implementation of f_{coupl} and its inverse. Left: Behavior of our model in the forward phase. Right: Behavior of our model in the inverse phase.

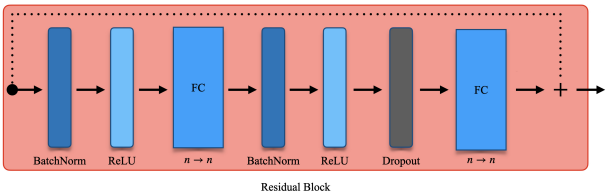

Figure 3: **Residual architecture**. The figure shows the implementation of the residual block used in the coupling layers f_{coupl} and its inverse.

about 7 hours on a NVIDIA RTX2080Ti. Since we have full 3D supervision, the final loss function is:

$$L = \lambda_{nll} L_{nll} + \lambda_{mode,3D} L_{mode,3D},$$

with loss weights $\lambda_{prob} = 0.001$ and $\lambda_{mode,3D} = 1.0$.

4. Datasets

In this part we give a short description of the datasets used for training and evaluation. The dataset that we use are Human3.6M [4], MPI-INF-3DHP [12], 3DPW [15], MPII [1], COCO [10] and Mannequin Challenge [10].

Human3.6M It is a studio-captured benchmark for 3D human pose estimation. It includes several different actions performed by various subjects We follow standard practices in the literature and use subjects S1, S5, S6, S7, S8 for training and subjects S9 and S11 for evaluation.

MPI-INF-3DHP It contains data captured primarily in indoor studio environments and the 3D pose data is captured using a marker-less setup. We use the predefined training and testing splits for training and evaluation respectively.

3DPW It is a dataset captured in a variety of indoor and outdoor locations and uses IMU sensors combined with a 2D pose detector to compute pose and shape ground truth. Following standard practice, we use this dataset only for evaluation in the predefined test split.

MPII It is a dataset containing images of people annotated with 2D keypoint locations. We use this dataset for training.

COCO It is a large scale dataset used among other applications for object detection, segmentation and pose estimation. We use 2D keypoint annotations from the `train2014` split to train our model.

Mannequin Challenge It is a dataset of videos of people staying frozen in diverse natural poses. We use the SMPL annotations generated by [9] and employ the entire dataset (train, test, validation) for evaluation.

5. Additional evaluations

5.1. Evaluation metrics

Here we give an outline of the metrics used for evaluation. To evaluate the 3D pose we use the Mean Per Joint Position Error (MPJPE) which computes the mean Euclidean error between the predicted and ground truth joints, after aligning the two poses at the pelvis. With PA-MPJPE we refer to the error after aligning the prediction with the ground truth pose by performing Procrustes alignment.

ProHMR Consistently with HMR [5], CMR [8] and SPIN [7], for evaluating the models that predict SMPL parameters we report the error on the 14 common LSP joints, except for MPI-INF-3DHP where we use all 17 joints provided by the dataset. For Human3.6M with the exception of

the multiview experiment (Table 4 of the main manuscript) we evaluate using the frontal camera, whereas for the multiview experiment we use all available cameras.

2D pose lifting To evaluate on Human3.6M we report MPJPE and PA-MPJPE computed on the 17 body joints and use frames from all available cameras.

5.2. Additional details

We clarify that for the model fitting experiment (Table 3 of the main manuscript) "H36M (OP)" refers to fitting the SMPL model to the OpenPose detectioins, whereas for "H36M (GT)" we use the ground truth 2D keypoints. Also, when we refer to the *minimum* error of samples from the learned distribution (Table 2 of the main manuscript), we use $n = 4096$ samples. For results with smaller n, we sample $n = 4096$ random poses from the posterior and following [2], we "quantize" them using K-Means. We would also like to highlight that theoretically our model can generate an arbitrary number of samples from the posterior, whereas for the multi-head architecture of Biggs *et al*. [2], increasing the maximum number of proposals requires a linear increase in the model size as well as retraining a new model.

5.3. Quantitative evaluation

It is interesting to study how the minimum error of our method scales with the number of samples. Figure 4 shows the minimum PA-MPJPE with respect to the number of samples drawn for the 2D pose lifting network. We can see that the minimum error for our method decreases almost linearly in log-scale. At the same time we show that sampling a large number of poses from our learned posterior achieves significantly lower error than sampling the same number of poses from the training distribution.

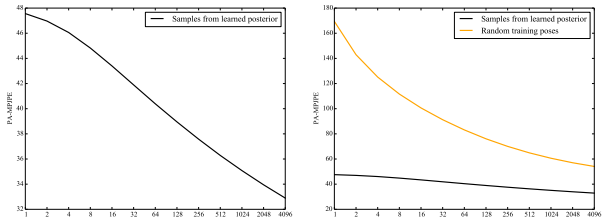

Figure 4: **The effect of sampling on 3D errors**. We report the minimum PA-MPJPE on Human3.6M for different number of samples. To eliminate the effect of extra data, we report results for the 2D pose lifting network [11] trained on Human3.6M. Left: Error vs number of samples from the learned posterior $p(\boldsymbol{\theta}; \mathbf{c})$. Right: Comparison with drawing an equal number of random poses from the training set.

In a similar setting, we compare the minimum error of ProHMR using 4096 samples with drawing an equal number of samples from a Gaussian distribution centered around the prediction. While ProHMR achieves a minimum PA-MPJPE of 29.9mm, drawing 4096 samples from $\mathcal{N}(\boldsymbol{\theta}^*, \sigma I)$ results in a minimum PA-MPJPE of $35.2, 41.1, 42.2$ for $\sigma = 1, 10, 20$ cm respectively. This is in part related to the fact that Gaussian posteriors are unimodal. Additionally, sampling directly from high dimensional Gaussians is known to be problematic.

	$n = 1$	$n = 25$
Biggs *et al*. [2]	67.8	64.2
ProHMR	67.3	60.1

Table 1: **Comparison with the approach of Biggs *et al*. [2]** on their AH36M dataset. Numbers are PA-MPJPE in mm.

Finally, we also provide another quantitative comparison. We use the AH36M dataset (ambiguous version of Human36M [4]) from Biggs *et al*. [2] and compare directly with the result of Table 1 from their paper, reporting the full results in Table 1. Again, our approach outperforms the baseline of [2] in this comparison, particularly when increasing the number of samples n. Furthermore, we assess the effect of our method in the downstream task of fitting on AH36M. Even under the truncations of this dataset, our image-based fitting outperforms SMPLify, with SMPLify achieving a PA-MPJPE of 67.8mm, while our fitting version achieves 61.4mm for the same metric.

5.4. Qualitative Results

In Figure 8 we show additional reconstructions of ProHMR. We use pink color for the mode of the posterior distribution. Moreover, in Figure 10 we include additional comparisons between our fitting method and SMPLify initialized by our regression. An important observation is that by using our image-based prior, the body orientation after the fitting is significantly more accurate compared to SMPLify, especially in cases with truncated people.

In Figure 5 we show how the optimization-based pose refinement is able to get more accurate pose estimates by fusing information from multiple views. In the first view the hands are mostly occluded and the recovered pose is not very accurate, however after the joint pose refinement the consolidated pose captures the true pose more faithfully.

Additionally, Figure 9 depicts examples of performing interpolation in the learned latent space. Starting from $\mathbf{z} = 0$, we pick two random directions in $\mathbb{R}^d$ and then move along those directions. Please note that there are no semantics or explicit disentanglement in the latent space.

Finally, we highlight some failure cases of our method. First, in Figure 6 we show failure cases for the regression network. Remarkably though, if we have access to accurate 2D keypoint detections, then it is possible to recover from such errors using our image-based model fitting. However, the model fitting can in turn fail when there are wrong key-

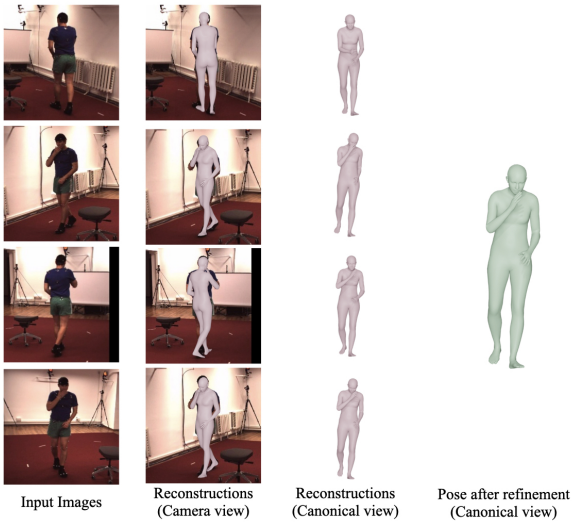

Figure 5: **Multiview refinement**. Pink: Regression. Green: Multiview refinement. Fitting with multiple views fixes the position of the right hand.

Figure 6: **Failure cases of pose regression.** Some failure cases of the regression (pink mesh) in challenging poses. In these examples, the model fitting (green mesh) is able to improve the pose reconstruction

Figure 7: **Failure cases for the model fitting.** The optimization can fail if there are wrong keypoint detections with high confidence (rows 1 and 2) or very few detected keypoints (row 3).

point detections with high confidence, or when there are too few detected keypoints as we show in Figure 7.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014. 2

[2] Benjamin Biggs, Sébastien Ehrhart, Hanbyul Joo, Benjamin Graham, Andrea Vedaldi, and David Novotny. 3D multibodies: Fitting sets of plausible 3D models to ambiguous image data. In *NeurIPS*, 2020. 3

[3] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. nflows: normalizing flows in PyTorch, Nov. 2020. 1

[4] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *PAMI*, 36(7):1325–1339, 2014. 2, 3

[5] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 2

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[7] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3D human pose and shape via model-fitting in the loop. In *ICCV*, 2019. 2

[8] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 2

[9] Vincent Leroy, Philippe Weinzaepfel, Romain Brégier, Hadrien Combaluzier, and Grégory Rogez. SMPLy benchmarking 3D human pose estimation in the wild. In *3DV*, 2020. 2

[10] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T. Freeman. Learning the depths of moving people by watching frozen people. In *CVPR*, 2019. 2

[11] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3D human pose estimation. In *ICCV*, 2017. 1, 3

[12] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3D human pose estimation in the wild using improved cnn supervision. In *3DV*, 2017. 2

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 1

[14] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3D human pose. In *CVPR*, 2017. 1

[15] Timo von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3D human pose in the wild using IMUs and a moving camera. In *ECCV*, 2018. 2

Figure 8: **Samples from the learned distribution.**. The pink colored mesh corresponds to the mode whereas we use purple and yellow for the additional samples.

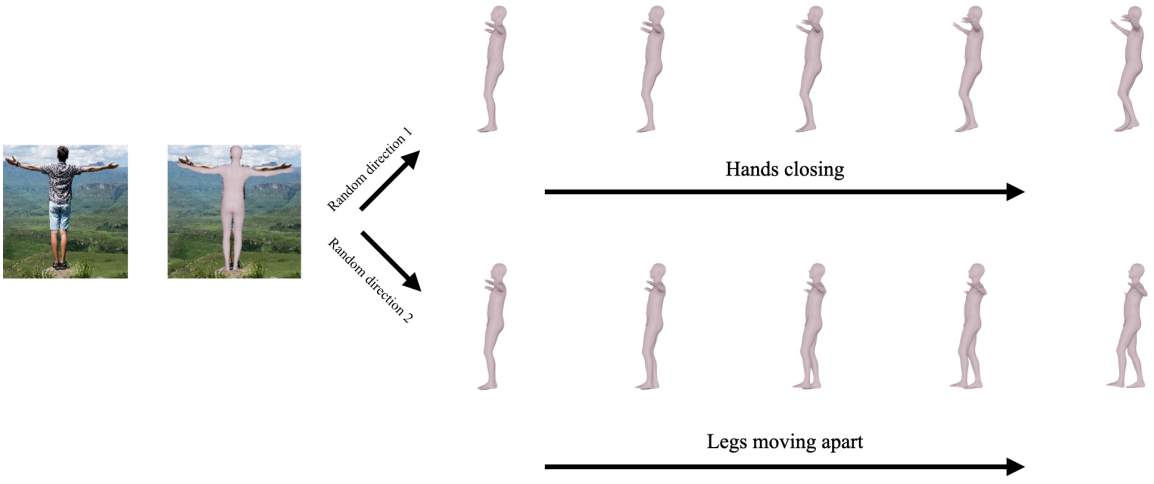

Figure 9: **Interpolation in the latent space.** We pick two random directions in the latent space and visualize the transformed samples on each direction from a side view.

Figure 10: **Model Fitting results**. Pink: Regression. Green: ProHMR + fitting. Grey: Regression + SMPLify