FLAG: Flow-based 3D Avatar Generation from Sparse Observations (Supplementary Material)

Sadegh Aliakbarian Pashmina Cameron Federica Bogo Andrew Fitzgibbon Thomas J. Cashman

Mixed Reality & AI Lab, Microsoft

https://microsoft.github.io/flag

Here, we provide implementation details of our model, statistics on the datasets we used in the main paper, additional qualitative and quantitative results, as well as further discussion on one of the baselines, VAE-HMD [6], as mentioned in the main paper. We also provide supplementary video of our approach which we highly recommend readers to watch[1].

1. Implementation Details

In this section, we provide implementation details of different components of our approach.

Flow-based model f_θ. Our flow-based model, f_θ is a RealNVP [5] with 16 transformation blocks. Each block is an affine coupling constructed with two MLPs representing the scale and translation networks. Each MLP is composed of three fully-connected layers with hidden size 512. Note that for scale network, we consider the `Tanh` non-linearity whereas for the translation network we use `ReLU`. For both networks we use no non-linearity for the last layer. In order to make the RealNVP conditional, we incorporate the conditioning signal $[x_{\mathbb{H}}, \beta]$ into both scale and translation networks, therefore, the output of each affine coupling layer becomes

$$\begin{aligned} y_m &= x_m \\ y_{\bar{m}} &= x_{\bar{m}} \odot \exp\big(s(x_m, [x_{\mathbb{H}}, \beta])\big) + t(x_m, [x_{\mathbb{H}}, \beta]) \end{aligned} \tag{1}$$

where s is the scale network, t is the translation network, m are the indices of the input representation that are masked, and $\bar{m}$ are the remaining indices. After each block, we swap mask and unmask indices to avoid information loss through identity mapping (as shown in the first line of Eq. 1). During training, intermediate supervision is applied to blocks $IS = \{2, 4, 6, 8, 10, 12, 14\}$ with negative log-likelihood weights $w = \{\frac{2}{16}, \frac{4}{16}, \frac{6}{16}, \frac{8}{16}, \frac{10}{16}, \frac{12}{16}, \frac{14}{16}\}$ correspondingly. During training we use $\lambda_{\text{NLL}} = 1$. After the flow is trained, for generation and likelihood estimation tasks, we drop all connections used for intermediate supervision. Since intermediate supervision is only applied during training, the model requires only one flow at test time. At the cost of increasing the *training* time $\sim 70\%$, intermediate supervision improves the performance considerably.

Transformer model. Following standard practice, our input to the transformer model is first mapped to learned embedding space with an MLP, consisting of a fully-connected layer that maps the data dimension to 256, followed by a `LeakyReLU`. The transformer encoder consists of 3 encoder layers, each with an embedding size of 256, 8 attention heads, and a feed-forward network with the hidden size of 512. Before passing the embedded representation to the transformer encoder, we add positional encoding to retain the joints order. The output of the transformer encoder is then pooled down to the representation of head and hands. The auxiliary task of masked joint prediction is done by a single fully-connected layer. Mapping to the categorical latent space is done via another MLP which gets as input the pooled representation of the transformer encoder and the conditioning signal $C = [x_{\mathbb{H}}, \beta]$. This MLP is constructed with a fully-connected layer that gets this input and maps it to a 256 dimensional embedding space, followed by a `LeakyReLU`, followed by another fully-connected layer that outputs a $G \times M$ matrix, representing the categorical latent representation. In our experiments, we found $G = 64$ and $M = 128$ to be sufficiently expressive for our task. From this latent representation, we sample a discrete latent variable (one-hot vector) via Gumbel-Softmax and use it as the input to the other auxiliary task of reconstructing the full pose, which is done with another MLP with two fully-connected layers and a `LeakyReLU` non-linearity in between. Finally, to compute $\mu_{\mathbb{H}}$ and $\Sigma_{\mathbb{H}}$, we use two identical MLPs, each with two fully-connected layers and a `LeakyReLU` non-linearity in between, getting as input the softmax-normalized latent representation and the conditioning signal C. Note that the two auxiliary tasks of masked

[1] Visit https://github.com/microsoft/flag.

joint prediction and full pose reconstruction are only used during training. For the loss terms weights during training, we use $\lambda_{\text{mjp}} = 1$, $\lambda_{\text{rec}} = 1$ and $\lambda_{\text{lra}} = 1$. For weights within $\mathcal{L}_{\textbf{lra}}$, we use $\alpha_{\text{nll}} = 1$, $\alpha_{\text{rec}} = 0.5$, and $\alpha_{\text{reg}} = 0.25$.

For joint masking, we avoid masking the full body joints except head hands from the beginning, as the model finds it hard to predict masked joints. To ease the training, we proposed curriculum masking of joints, following the body kinematic tree. In this scheme, we start by masking the lower body, then torso joints, then neck and arms. By gradual masking, the model learns to reason about the full body given observed (unmasked) joints, and as the training progresses, we make the such task harder by progressively masking more joints, until we reach the real-world scenario where only head and hands are unmasked.

Training hyper-parameters. To train our models, we use the Adam optimizer [7], with the learning rate of 0.0001, batch size of 1024, for 100 epochs. For all other baselines, we follow their official implementation or the details in the original papers.

Optimization. To perform optimization in either pose space or latent space, we use limited-memory BFGS optimizer [8], with the history size of 10, learning rate of 1, and Strong-Wolfe line search function [12]. We perform optimization for the maximum 50 iterations, however, we evaluate the MPJPE after 2, 5, 10, 25, and 50 iterations. Note that when optimizing in the latent space, the optimizer has a freedom to move around the base distribution and find a latent code that matches the HMD signal. However, since HMD signal only contains information about the upper body, there is no constraint on the optimizer to stop it from going astray (into regions of low-likelihood lower body poses). To prevent the optimizer from straying into regions of the latent space that may correspond to undesirable lower body poses, we add a regularizer to implicitly guide the optimizer to search in regions of the latent space that modifies the upper body to minimize $\mathcal{C}_{data}$. This is done by adding $r = ||z_{opt} - \mu_{\mathbb{H}}||$, encouraging the solver to stick to the initial lower body guess.

Table 1. Detailed statistics of AMASS datasets we used in this paper.

Dataset	Subjects	Motions	Minutes
CMU	106	2083	551.56
MPI Limits	3	35	20.82
Total Capture	5	37	41.10
Eyes Japan	12	750	363.64
KIT	55	4232	661.84
BioMotionLab	111	3060	522.69
HumanEva	3	28	8.48
EKUT	4	349	30.71
ACCD	20	256	26.76
BMLMovi	86	1801	168.99
MPI Mosh	19	77	16.53
SFU	7	44	15.23
Transition	1	110	15.10
MPI HDM05	4	215	144.54

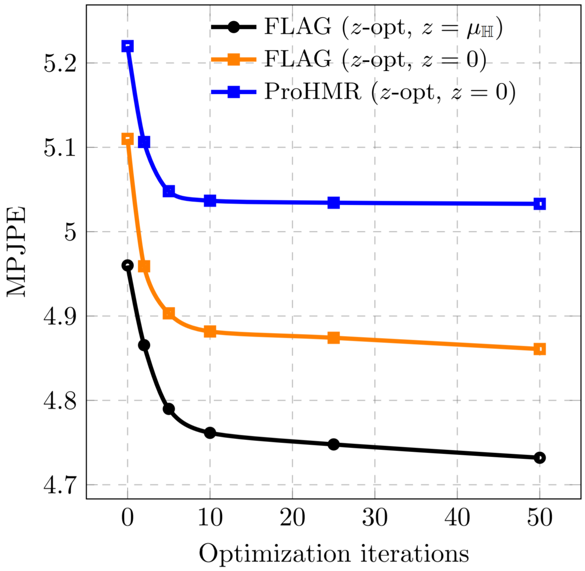

Figure 1. Full body MPJPE as a function of optimization iterations (in the latent space).

2. Dataset Details: AMASS

In this paper, we use AMASS [10] for training and evaluation. AMASS is one of the largest publicly available datasets of human motion (299,234 minutes of capture). It unifies different optical marker-based motion capture datasets by representing them within a common framework with consistent parameterization. AMASS comprises 20 datasets, from which We use the suggested training and test subsets. While we keep the original test set, we remove the dataset containing dance sequences from our dataset (similar to [13]). The train/test datasets splits we used are listed below:

Training datasets. CMU [2], MPI Limits [4], Total Capture [16], Eyes Japan [3], KIT [11], BioMotionLab [15], BMLMovi [10], EKUT [11], ACCAD [1], MPI Mosh [9], SFU [17], and MPI HDM05 [4]

Test datasets. HumanEva [14], Transition [10]

In Table 1, we provide details of each dataset in AMASS.

Table 2. Best of K=10 samples (5 runs) MPJPE on AMASS. Sampling is based on baselines' prior.

Method	Sampling	Upper Body MPJPE (↓)	Full Body MPJPE (↓)
VPoser-HMD	$z \sim \mathcal{N}(0, \mathrm{I})$	1.58±0.02 cm	5.25±0.07 cm
HuMoR-HMD	$z \sim \mathcal{N}(\mu_{\text{prior}}, \Sigma_{\text{prior}})$	1.47±0.01 cm	4.83±0.03 cm
VAE-HMD	$z \sim \mathcal{N}(\mu_{\text{prior}}, \Sigma_{\text{prior}})$	3.16±0.02 cm	5.67±0.06 cm
ProHMR-HMD	$z \sim \mathcal{N}(0, \mathrm{I})$	1.62±0.02 cm	4.75±0.03 cm
FLAG	$z \sim \mathcal{N}(0, \mathrm{I})$	1.61±0.01 cm	4.65±0.03 cm
FLAG	$z \sim \mathcal{N}(\mu_{\mathbb{H}}, \Sigma_{\mathbb{H}})$	**1.29±0.0 cm**	**4.65±0.01 cm**

3. Further Evaluation of the Effect of Latent Variable Sampling

In addition to the experiments we conducted and provided the results in the main paper, here we also evaluate the effect of our latent variable sampling, i.e, $z = \mu_{\mathbb{H}}$, and compare it with the commonly used $z = \mathbf{0}$ when fed to the FLAG as the pose prior in optimization in the latent space. As illustrated in Fig. 1, our approach with $z = \mu_{\mathbb{H}}$ achieves considerably lower full body error compared to our approach with $z = \mathbf{0}$, showing the importance of latent variable sampling. Additionally, the importance of model design, the intermediate supervision in particular, is evident when looking at the performance of ProHMR (with $z = \mathbf{0}$) and FLAG (with $z = \mathbf{0}$).

4. Further Discussion on VAE-HMD

In our comparison to the existing approaches, we note relatively high MPJPE for VAE-HMD [6] on the AMASS test set, we argue this is due to imperfect utilization of the latent space resulting from the particular two-stage training used in VAE-HMD. To further investigate the behaviour, we followed the setup in the original paper and define a random train/test splits on a small dataset (MPI-HDM05 [4]). In this case, we observed that VAE-HMD is capable of achieving a very low full body MPJPE of 2.39, which is in line with the original paper, and this is due the fact that test poses are quite similar to the poses in the training set (as a result of random splitting of frames within sequences of a single dataset).

5. Additional Quantitative Results

Best of K metric. In addition to quantitative results presented in the main paper, we also report the MPJPE of the best of K samples in Table 2.

Contribution of auxiliary tasks. We empirically observed contributions of the two aux tasks. The masked joint prediction is required due to joint masking. The pose prediction auxiliary task helps the transformer's latent space to learn better representation of the full pose faster, as mentioned in in the main paper. We also observed that the performance of pose prediction auxiliary task is not competitive with f_θ (upper-body MPJPE of 3.89 vs 1.29 and full-body MPJPE of 6.49 vs 4.96).

6. Additional Qualitative Results

In this section, we provide additional qualitative results of our approach, as well as for all other baselines, in Fig. 2 to Fig. 4 shown in the next three pages. Note that the examples are not hand picked.

References

[1] OSU Advanced Computing Center for the Arts and Design. ACCAD. *https://accad.osu.edu/research/motion-lab/system-data*. 2

[2] CMU graphics lab. CMU graphics lab motion capture database. *http://mocap.cs.cmu.edu/*, 2000. 2

[3] Eyes, JAPAN Co. Ltd. Eyes. *http://mocapdata.com*, 2018. 2

[4] Ijaz Akhter and Michael J Black. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1446–1455, 2015. 2, 3

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. *International Conference on Learning Representations, ICLR*, 2017. 1

[6] Andrea Dittadi, Sebastian Dziadzio, Darren Cosker, Ben Lundell, Thomas J Cashman, and Jamie Shotton. Full-body motion from a single head-mounted device: Generating SMPL poses from partial observations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11687–11697, 2021. 1, 3

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations, ICLR*, 2015. 2

[8] Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989. 2

[9] Matthew Loper, Naureen Mahmood, and Michael J Black. MoSh: Motion and shape capture from sparse markers. *ACM Transactions on Graphics (TOG)*, 33(6):1–13, 2014. 2

[10] 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*, pages 5442–5451, Oct. 2019. 2

[11] Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The KIT whole-body human motion database. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 329–336. IEEE, 2015. 2

[12] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006. 2

[13] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. Humor: 3D human motion model for robust pose estimation. *International Conference on Computer Vision*, 2021. 2

[14] Leonid Sigal, Alexandru O Balan, and Michael J Black. HumanEva: Synchronized video and motion capture dataset and

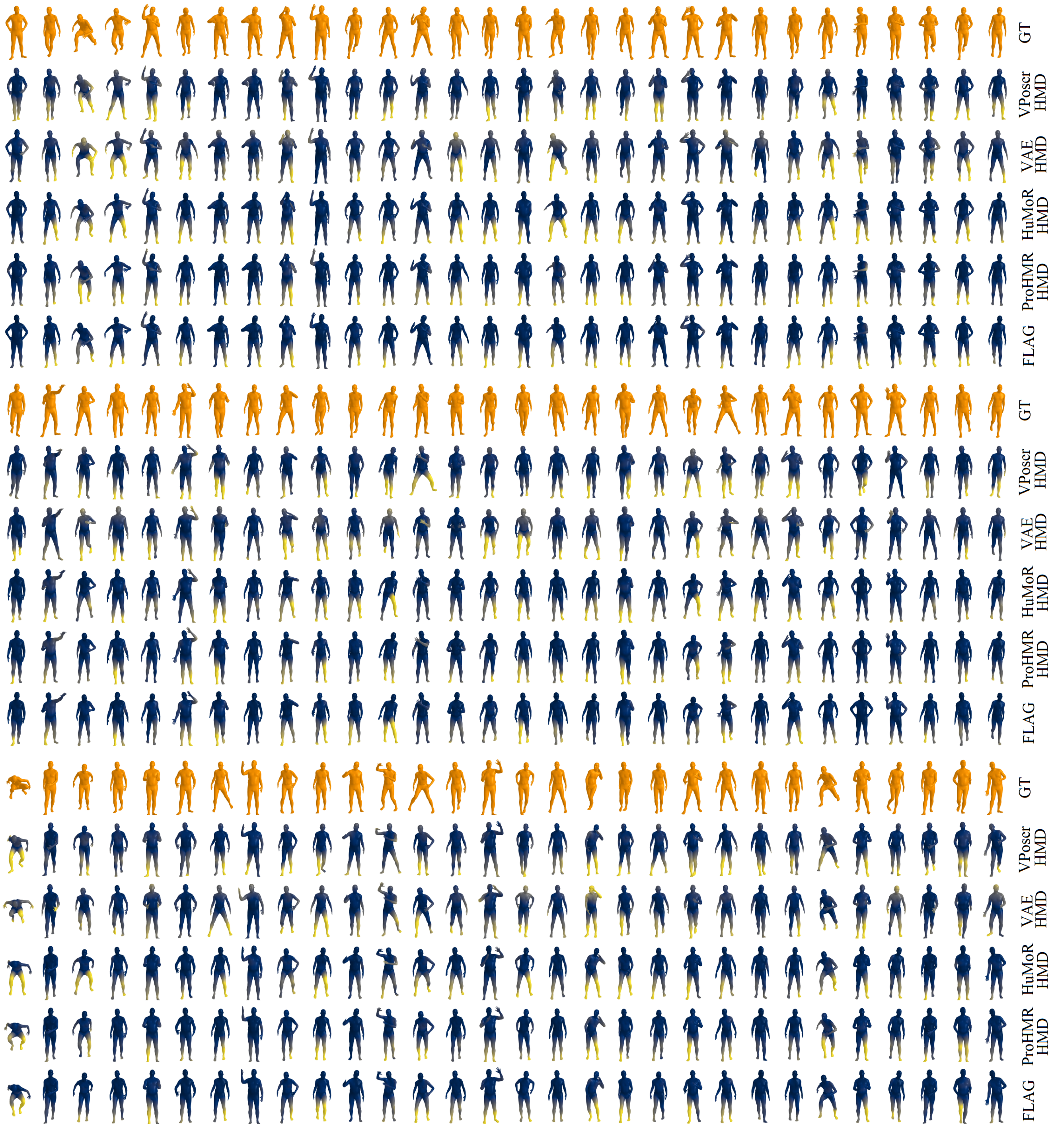

Figure 2. Additional qualitative results. Best seen zoomed in. Note, in each segment of results, the last row represents our approach.

baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1-2):4, 2010. 2

[15] Nikolaus F Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of vision*, 2(5):2–2, 2002. 2

[16] Matthew Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John P Collomosse. Total capture: 3D human pose estimation fusing video and inertial sensors. In *BMVC*, volume 2, pages 1–13, 2017. 2

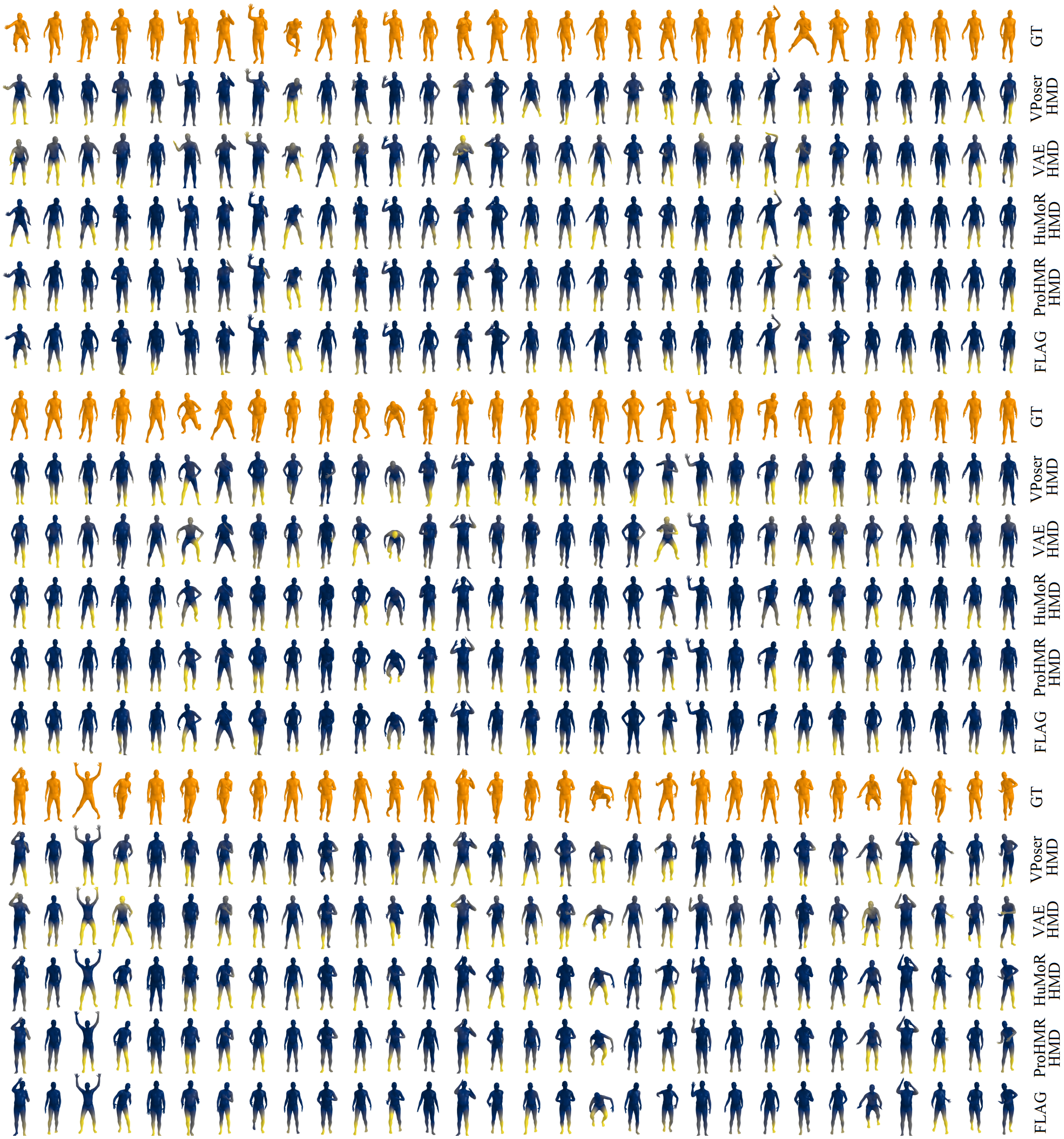

Figure 3. Additional qualitative results. Best seen zoomed in. Note, in each segment of results, the last row represents our approach.

[17] KangKang Yin Goh Jing Ying, K Yin, KD Kumar, H Geng, SC Mahadevan, E Tanirgan, and K Hurley. SFU motion capture database. *URL http://mocap. cs. sfu. ca*, 2011. 2

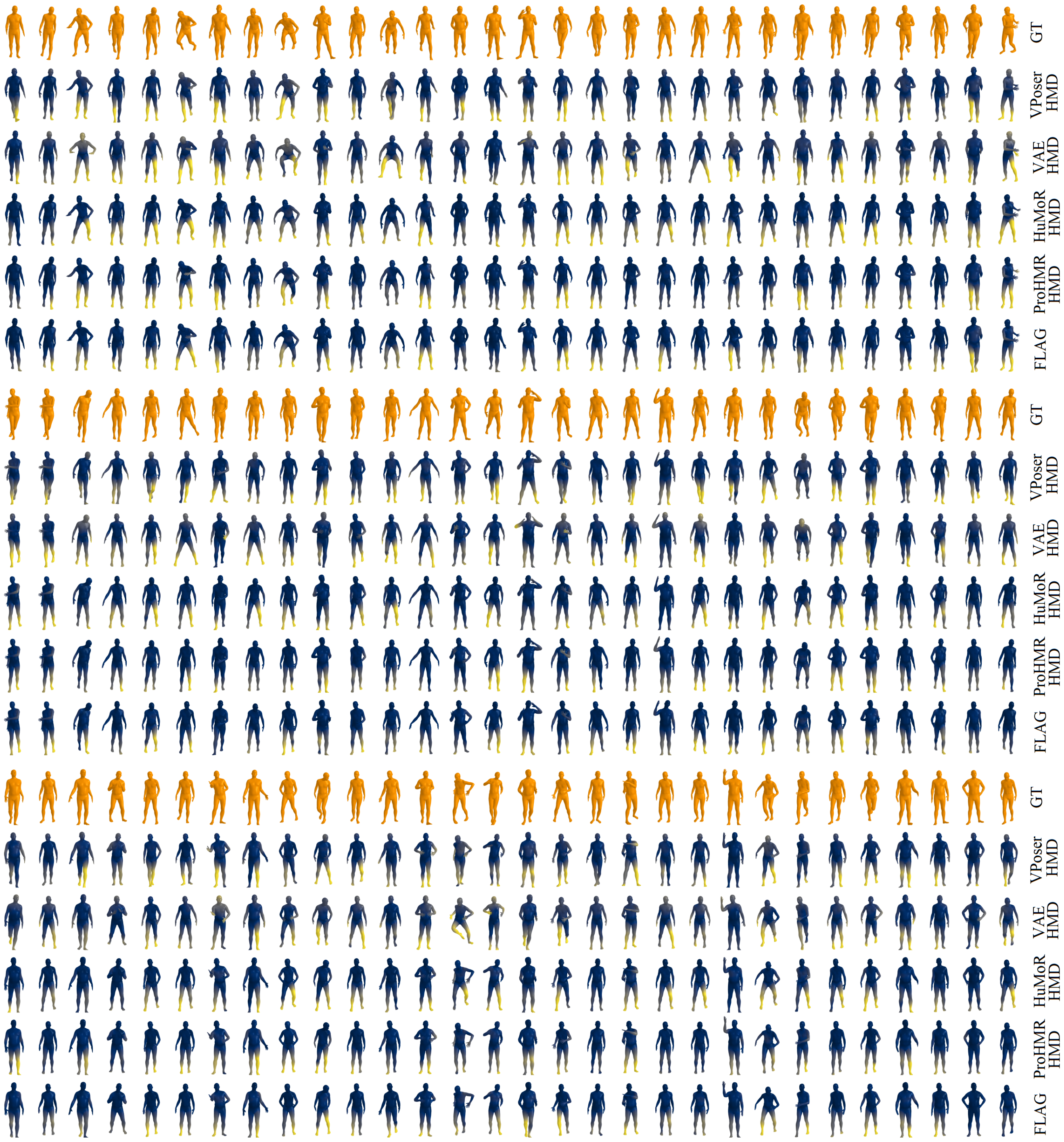

Figure 4. Additional qualitative results. Best seen zoomed in. Note, in each segment of results, the last row represents our approach.