SOMA: Solving Optical Marker-Based MoCap Automatically **Supplementary Material**

Nima Ghorbani Michael J. Black
Max Planck Institute for Intelligent Systems, Tübingen, Germany
{nghorbani,black}@tuebingen.mpg.de

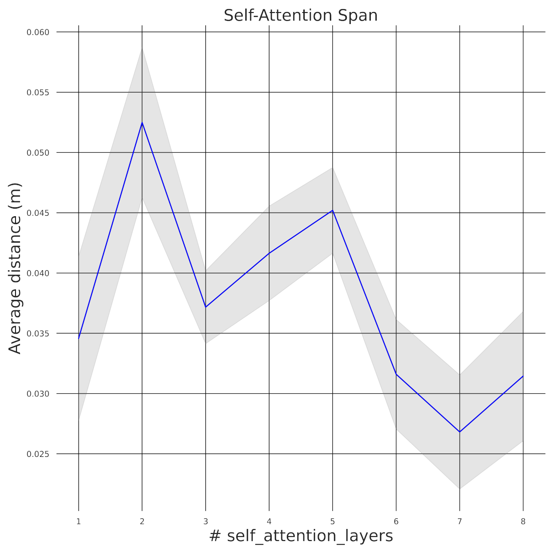


Figure 1: Attention span as a function of layer depth in meters. The grey area indicates 95% confidence interval.

SOMA labels raw and noisy "mocap point clouds" at scale, without requiring subject calibration, and across various capture technologies. Here we provide more details as referenced in the main paper. Additionally, we encourage the reader to watch the **supplementary video**. Since mocap is inherently about motion, it is very difficult to convey the quality of the results in a static format. The video provides a much clearer picture of what SOMA does and the quality of the results. All supplementary material can be accessed in the project website https://soma.is.tue.mpg.de/

1. Self-Attention Span

As explained in Sec. 4.1 of the main paper, to increase the capacity of the network and learn rich point features at multiple levels of abstraction, we stack multiple self-attention residual layers. Following [11], a transformer self-attention layer, Fig. 3 of the main paper, takes as input two vectors, the query (Q), and the key (K), and computes a weight vector $W \in [0, 1]$ that learns to focus on different regions of the input value (V), to produce the final output. In self-attention, all the three vectors (key, query and value) are projections of the same input; i.e. either 3D points or their features in deeper layers. All the projection operations are done by 1D-convolutions, therefore the input and the output features only differ in the last dimensions (number of channels). Following notation of [11]:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_{model}}})V. \quad (1)$$

In a controlled experiment on the validation dataset, HDM05 with marker layout presented in Fig. 9d, we pass the original markers (without noise) through the network and keep track of the attention weights at each layer; i.e. output after Softmax in Eqn. 1. At each layer, the tensor shape for the attention weights is $\#batch \times \#heads \times \#points \times \#points$. We concatenate frames of 50 randomly selected sequences, roughly 50000 frames, and take the maximum weight across heads and the mean over all the frames to arrive at a mean attention weight per layer; $(\#points \times \#points)$. In Fig. 4 of the main paper, the weights are visualized on the body with a color red intensity for 3 markers. In the first layers, the attention span is wide and covers the entire body. In deeper layers, the attention becomes gradually more focused on the marker of the interest and its neighboring markers on the body surface. Fig. 2 shows the attention span for more markers.

To make this observation more concrete, we compute the euclidean distance of each marker to all others on a A-Posed body to create a distance discrepancy matrix of $(\#points \times \#points)$, and multiply the previous mean attention weights with this distance discrepancy matrix to arrive at a scalar for attention span in meters. On average we observe a narrower focus for all markers in deeper layers; Fig. 1.

2. Implementation Details

Through model selection, Sec. 3, we choose 35 iterations for Sinkhorn and $k = 8$ as optimal choices and we empirically pick $c_l = 1, c_{reg}$ =5e-5, $d_{model} = 125, h = 5$. The model contains 1.44 M parameters and full training on 8 Titan V100 GPUs takes roughly 3 hours. We implement SOMA in PyTorch [9]. We benefit from the log-domain stable implementation of Sinkhorn released by [10]. We use ADAM [7] with a base learning rate of $1e-3$ and reduce it by a factor of 0.1 when validation error plateaus with patience of 3 epochs and train until validation error does not drop anymore for 8 epochs. The training code is implemented in PyTorch Lightning [1] and easily extendable to run on multiple GPUs. For the LogSoftmax experiment, we replace the optimal transport layer and everything else in the architecture remains the same. In this case, the score matrix, S in Fig. 3 of the main paper, will have an extra dimension for the null label. Fig. 3 shows a detailed architecture of the SOMA model.

3. Hyper-parameter Search

To choose the optimum number of attention layers and iterations for Sinkhorn normalization we exploit the validation dataset HDM05 to perform a model selection experiment. We produce synthetic training data following the prescription of Sec. 4.4 of the main paper for the marker layout of HDM05 (Fig. 9d) and evaluate on real markers with synthetic noise as explained in Sec. 5 of the main paper. For hyperparameter evaluation, we want to eliminate random variations in the network weight initialization so we always use the same seed. In Fig. 4, we train one model per given number of layers. Guided by this graph we choose k =8 layers as a sensible choice for adequate model capacity, i.e. $1.44M$, and generalization to real markers. In Fig. 5, we repeat the same process, this time keeping the number of layers fixed as 8, and varying the number of Sinkhorn iterations. We choose 35 iterations that seem a good trade-off between computation time vs performance.

4. Standard Deviations

In Tab. 1, we report accuracy standard deviation of Tab. 2 of the main paper as complementary material. We observe lower variation for the model trained on synthetic data using AMASS bodies. The model trained on synthetic data of limited number of bodies shows the largest variation.

Method	Number of Exact Per-Frame Occlusions						
	0	1	2	3	4	5	5+G
SOMA-Real	1.76	1.90	2.03	2.22	2.44	2.73	3.08
SOMA-Synthetic	4.68	2.89	3.25	3.54	4.10	4.62	5.92
SOMA *	1.59	1.76	1.91	2.12	2.34	2.59	2.85

Table 1: Accuracy standard deviation corresponding to Tab. 2 of the main paper.

5. Marker Layout Variation of HDM05

In Fig. 8 we visualize the marker layout modifications for the experiment in Sec. 5.3 of the main paper.

6. Stability of the Training Process

We consistently observe stable runtime and training processes. In Fig. 6, we provide training curves for more “extreme” ghost point distributions. Specifically, we add a uniform distribution in a cubic volume in the range of $[-2, 2]$ meters and skewed Gaussian with a mean location sampled uniformly from the same random volume and a random covariance matrix. We also drastically increase the number of ghost points to up to 60 per-frame. As suggested by the figure, training is stable and converges from early iterations on.

7. Processing Real MoCap Data

Here we elaborate on Sec. 4.5 of the main paper, namely on real use-case scenarios of SOMA. The marker layout of the test datasets, Sec. 5 of the main paper, are obtained by running MoSh on a single random frame chosen from the respective dataset. Fig. 9 demonstrates the marker layout used for training SOMA for each dataset.

In addition to test datasets with synthetic noise, presented in Sec. 4.5 of the main paper, we demonstrate the real application of SOMA by automatically labeling four real mocap datasets captured with different technologies; namely: two with passive markers, SOMA and CMU-II [4], and two with active marker technology, namely DanceDB [2] and Mixamo [3]; for an overview refer to Tab. 6 of the main paper.

For proper training of SOMA we require one labeled frame per significant variation of the marker layout throughout the dataset. Most of the time one layout is utilized to capture the entire dataset, yet as we see next, this is not always the case, especially when the marker layout is adapted to the target motion. To reduce the effort of labeling the single frame we offer a semi-automatic bootstrapping technique. To that end, we train a general SOMA model with a marker layout containing 89 markers selected from the MoSh dataset [8], visualized in Fig. 7; this is a marker super-set. We choose one sequence per each of representative layouts and run the general SOMA to *prime the labels;*

Name	# Frames	# Motions	Acc.	F1	$V2V_{mm}^{mean}$	$V2V_{mm}^{median}$
Clap	7572	6	100.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.08	0.00
Dance	15023	8	99.78 ± 0.87	99.68 ± 1.29	0.15 ± 1.76	0.00
Jump	9621	6	99.99 ± 0.13	99.99 ± 0.25	0.03 ± 0.72	0.00
Kick	10787	6	99.59 ± 1.18	99.48 ± 1.50	0.75 ± 6.92	0.00
Lift	16932	6	100.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.06	0.00
Random	19617	7	100.00 ± 0.05	100.00 ± 0.09	0.00 ± 0.21	0.00
Run	9356	6	100.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.06	0.00
Sit	9829	6	100.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.09	0.00
Squat	11287	6	100.00 ± 0.00	100.00 ± 0.00	0.01 ± 0.13	0.00
Throw	9292	6	99.99 ± 0.15	99.99 ± 0.22	0.00 ± 0.09	0.00
Walk	12264	6	100.00 ± 0.00	100.00 ± 0.00	0.00 ± 0.11	0.00
	131580	69	99.94 ± 0.47	99.92 ± 0.64	0.08 ± 2.09	0.00

Table 2: Per-motion-class statistics of the SOMA dataset and performance of the SOMA model.

we choose one frame per auto-labeled sequence and correct any incorrect labels manually. The label priming step significantly reduces the manual effort required for labeling mocap datasets with diverse marker layouts. After this step, everything stays the same as before.

Labeling Active Marker Based MoCap should be the easiest case since the markers emit a frequency-modulated light that allows the mocap system to reliably track them. However, often the markers are placed at arbitrary locations on the body so correspondence of the frequency to the location on body is not the same throughout the dataset, hence these archival mocap datasets cannot be directly solved. This issue is further aggravated when the marker layout is unknown and changes drastically throughout the dataset. It should be noted that, for the case of active marker mocap systems, such issues could potentially be avoided by a carefully documented capture scenario, yet this is not the case with the majority of the archival data.

As an example, we take DanceDB [2], a publicly released dance-specific mocap database. This dataset is recorded by active marker technology from PhaseSpace [5]. The database contains a rich repertoire of dance motions with 13 subjects on the last access date. We observe a large variation in marker placement especially on the feet and hands, hence we manually label one random frame per each significant variation; in total 8 frames. We run the first stage of MoSh independently on each of the selected 8 frames to get a fine-tuned marker layout; a subset is visualized in Fig. 10. It is important to note that we train only one model for the whole dataset while different marker layouts are handled as a source of noise. As presented in Tab. 6, manual evaluation of the solved sequences reveals an above 80% success rate. The failures are mainly due to impurities in the original data, such as excessive occlusions or large marker movement on the body due to several markers coming off (e.g. the headband).

The second active marker based dataset is Mixamo [3], which is widely used by the computer vision and graphics community for animating characters. We obtained the original unlabeled mocap marker data used to generate the animations. We observe more than 50 different marker layouts and placements on the body, of which we pick 19 key variants. The automatic label priming technique is greatly helpful for this dataset.

The Mixamo dataset contains many sequences with markers on objects, i.e. props, which SOMA is not specifically trained to deal with. However, we observe stable performance even with challenging scenarios with a guitar close to the body; see the third subject from the left of Fig. 1 of the main paper. A large number of solved sequences were rejected mostly due to issues with the raw mocap data; e.g. significant numbers of markers flying off the body mid capture.

Labeling Passive Marker Based MoCap is a greater challenge for an auto-labeling pipeline. For these systems, markers are assigned a new ID on their reappearance from an occlusion, which results in small tracklets instead of full trajectories. The assignment of the ID to markers is random.

For the first use case, we process an archived portion of the well-known CMU mocap dataset [4] summing to 116 minutes of mocap which has not been processed before, mostly due to cost constraints associated with manual labeling. It is worth noting that the total amount of available data is roughly 6 hours of which around 2 hours is pure MPC. Initial inspection reveals 15 significant variations in marker layouts, with a minimum 40 markers and a maximum 62; a sample of which can be seen in Fig. 11. Again we train one model for the whole dataset that can handle variations of these marker layouts. SOMA shows stable performance across the dataset even in presence of occasional object props as seen in Fig. 1 of the main paper; the second subject from the left is carrying a suitcase.

Due to extreme variation of marker layouts throughout the dataset we notice failure cases where many points could

Symbol	Description
MPC	MoCap Point Cloud
L	set of labels including the null label
l	a single label
$\mathbf{v}$	vector of marker layout body vertices corresponding to labels not including null
$\tilde{\mathbf{v}}$	vector of varied marker layout vertices
M	number of markers
P	set of all points
$G^{'}$	ground-truth augmented assignment matrix
A	predicted assignment matrix
$A^{'}$	augmented assignment matrix
S	score matrix
W	class balancing weight matrix
X	markers
V	body vertices
d	marker distance from the body along the surface normal
J	body joints
h	number of attention heads
k	number of attention layers

Table 3: List of Symbols

not be assigned to a marker on the body, most probably due to variation from the expected placement. As studied in Sec. 5.3 of the main paper, this deteriorates the labeling performance and could result in a failure in solving the body mainly because of introduced occlusions.

In the second case, we record our own dataset with two subjects for which we pick one random frame and train SOMA for the whole dataset. In Tab. 2 we present details of the dataset motions and per-motion-class performance of SOMA. For this dataset, we manually label it to have ground truth and then we fit the labeled data with MoSh. This provides ground truth 3D meshes for every mocap frame. The V2V error measures the average difference between the vertices of the solved body using the ground truth and using the SOMA labels. Mean V2V errors are under one mm and usually by an order of magnitude. Sub-millimeter accuracy is what users of mocap systems expect and SOMA delivers this.

8. List of Symbols

In Tab. 3, we provide a table of mathematical symbols used throughout the paper.

References

[1] Pytorch lightning, 2019. 2

[2] Andreas Aristidou, Efstathios Stavrakis, Margarita Papaefthimiou, George Papagiannakis, and Yiorgos Chrysanthou. Style-based motion analysis for dance composition. *The Visual Computer*, 34:1725–1737, 2018. 2, 3

[3] Adobe Mixamo MoCap Dataset, 2019. 2, 3

[4] Carnegie Mellon University (CMU) MoCap Dataset, 2019. 2, 3

[5] PhaseSpace, Inc., 2019. 3

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, page 448–456, 2015. 6

[7] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[8] 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, 7

[9] 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. pages 8024–8035. Curran Associates, Inc., 2019. 2

[10] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, pages 4937–4946, 2020. 2

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPs*, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. 1

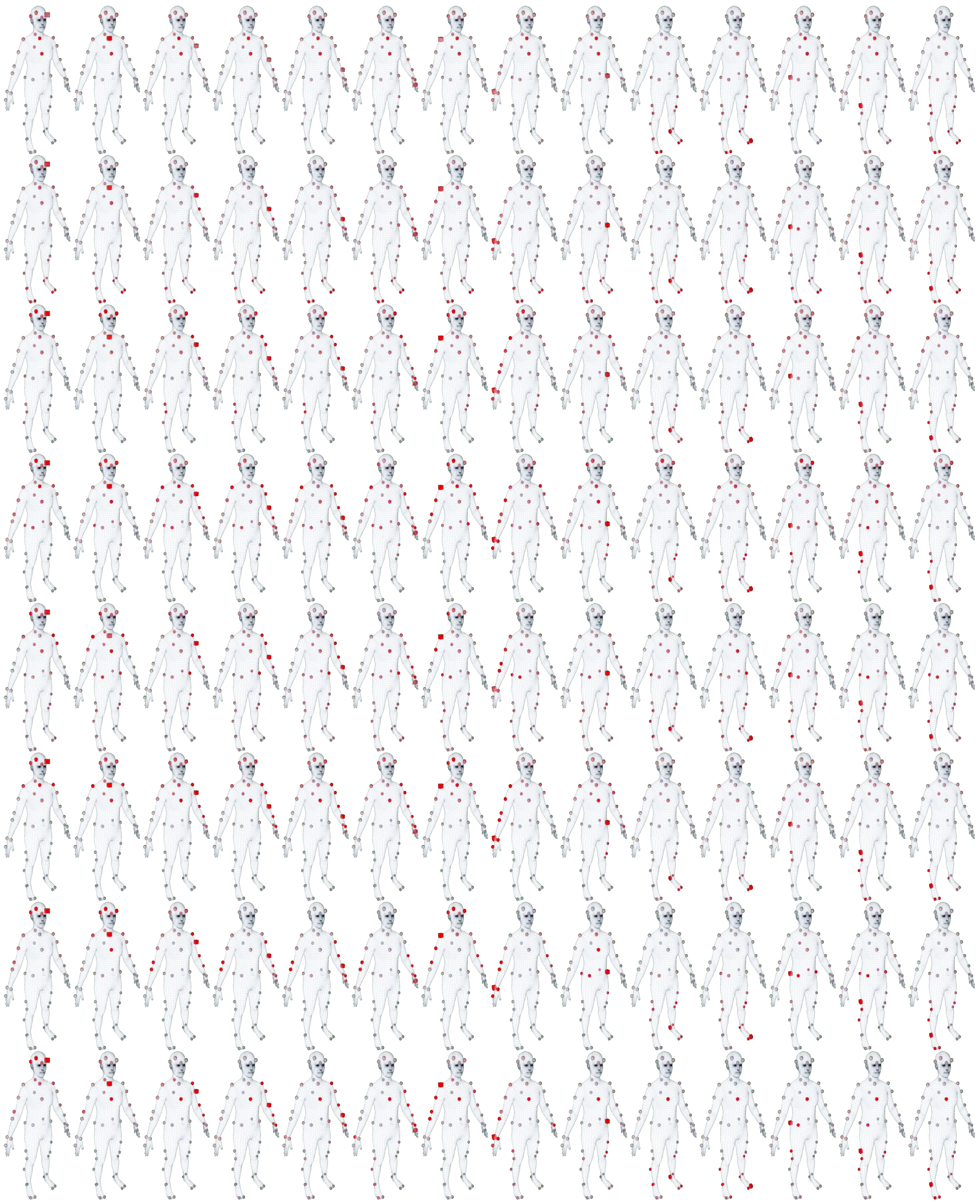
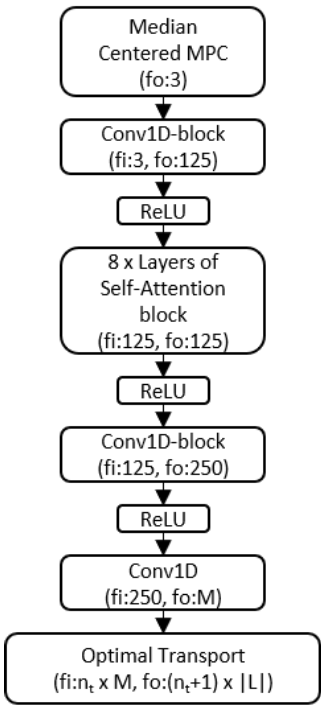
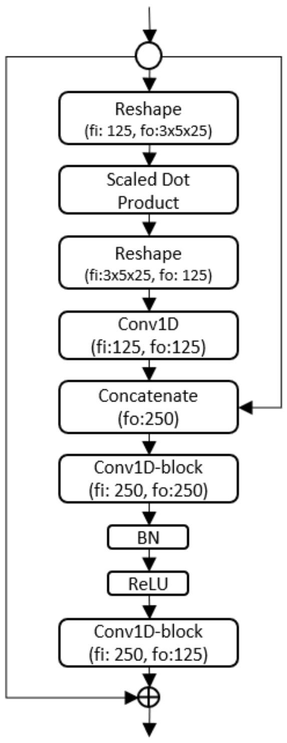


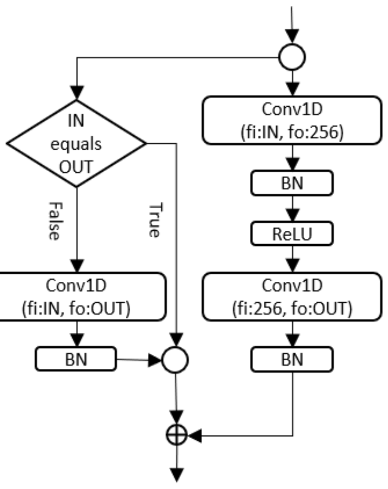
Figure 2: Attention span for 14 markers, across all layers. Each row corresponds to a layer in ascending order, with bottom most row showing the last layer.


Median Centered MPC (fo:3)
Conv1D-block (fi:3, fo:125)
ReLU
8 x Layers of Self-Attention block (fi:125, fo:125)
ReLU
Conv1D-block (fi:125, fo:250)
ReLU
Conv1D (fi:250, fo:M)
Optimal Transport (fi:n_t x M, fo:(n_t+1) x |L|)

(a) Detailed SOMA Architecture


Reshape (fi: 125, fo:3x5x25)
Scaled Dot Product
Reshape (fi:3x5x25, fo: 125)
Conv1D (fi:125, fo:125)
Concatenate (fo:250)
Conv1D-block (fi: 250, fo:250)
BN
ReLU
Conv1D-block (fi: 250, fo:125)

(b) Self-Attention block



(c) Conv1D-block

Figure 3: Detailed components of SOMA model. fi and fo show the number of input and output features of the layer. n_t is the number of points in a frame of data and $|L|$ is number of all labels including null. IN and OUT in (c) show the number of input and output features of the block. All convolutions are one dimensional. BN stands for batch normalization [6].

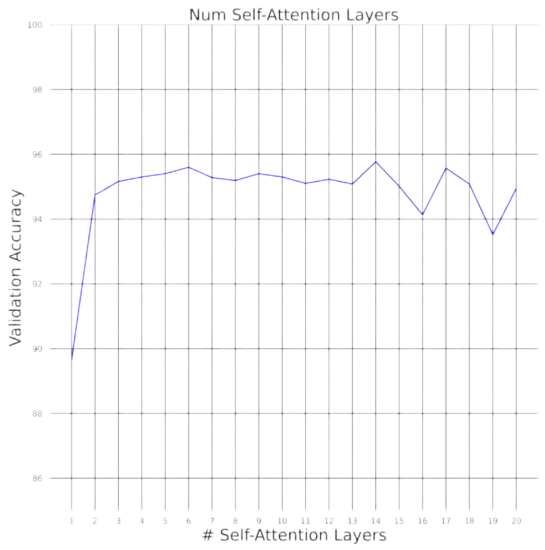


Figure 4: Validation accuracy as a function of number of attention layers

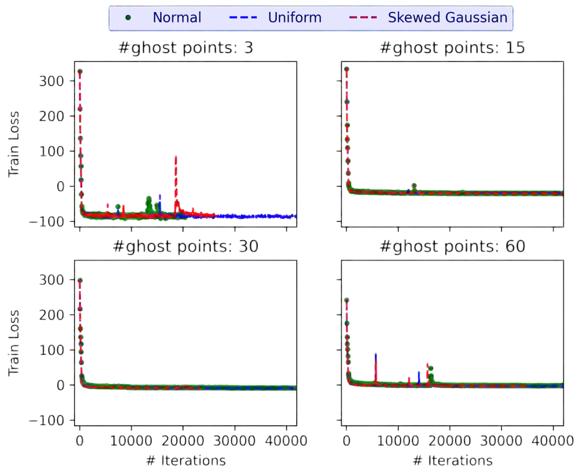


Figure 6: Training convergence with extreme ghost point distributions on the validation dataset for 40 training epochs; i.e. HDM05.

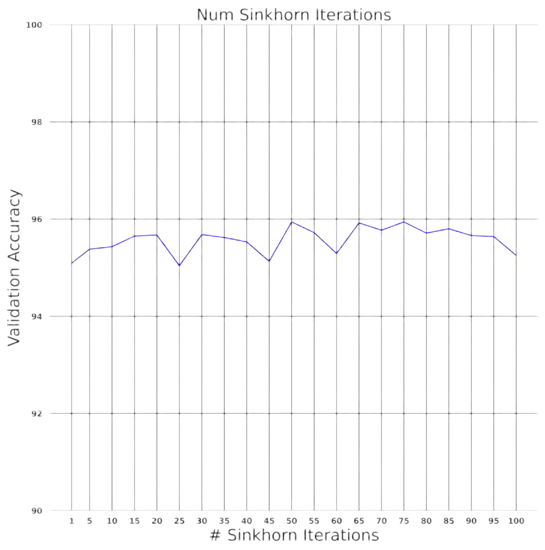


Figure 5: Validation accuracy as a function of number of Sinkhorn normalization steps.

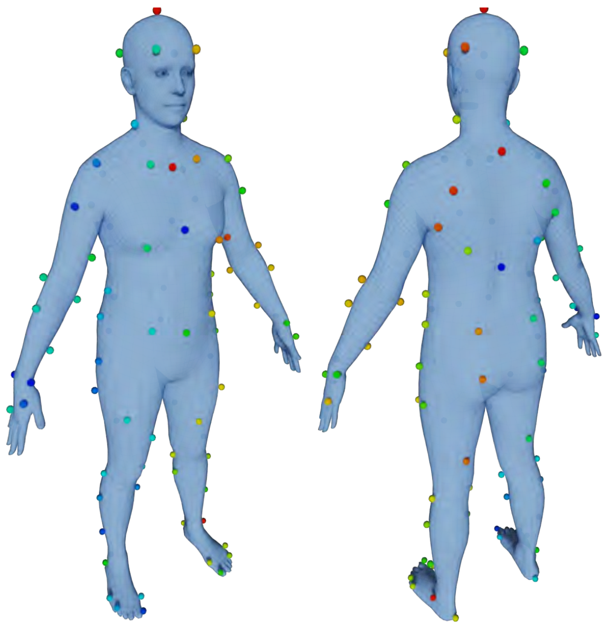
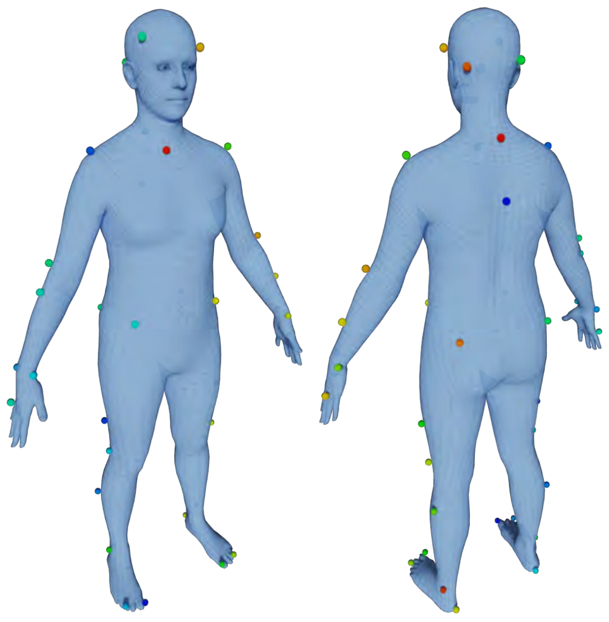
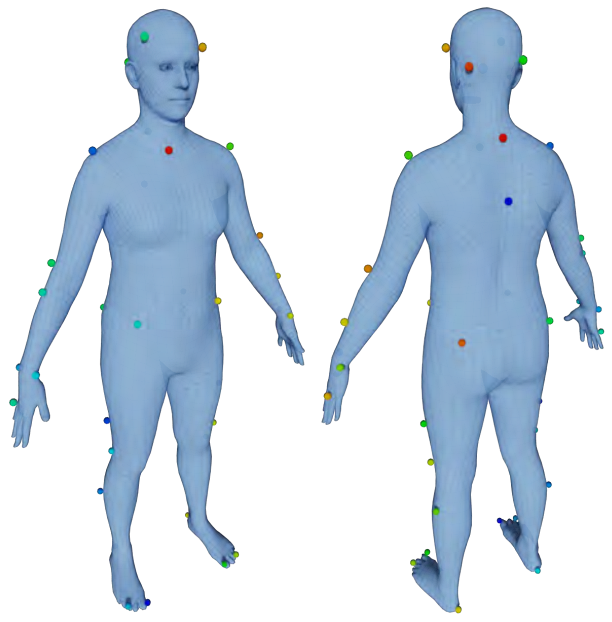
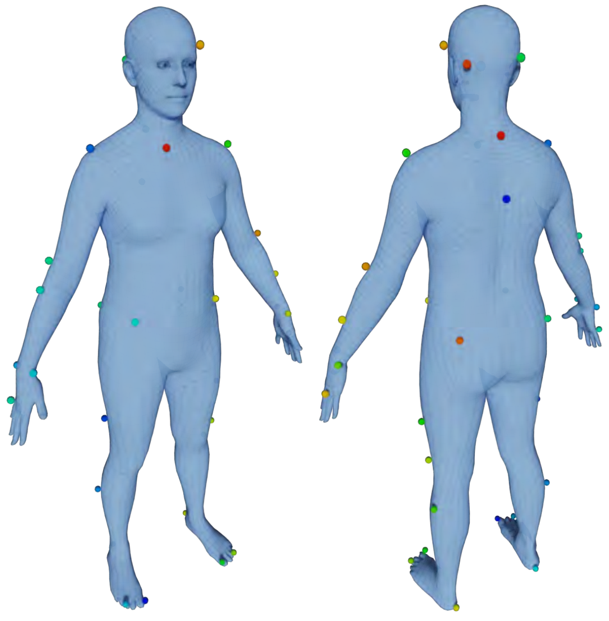


Figure 7: Marker layout from MoSh [8] dataset with 89 markers. A model trained on this marker layout is used for rapid automatic label priming for labeling the single frame per significant marker layout variation.


(a)


(b)


(c)

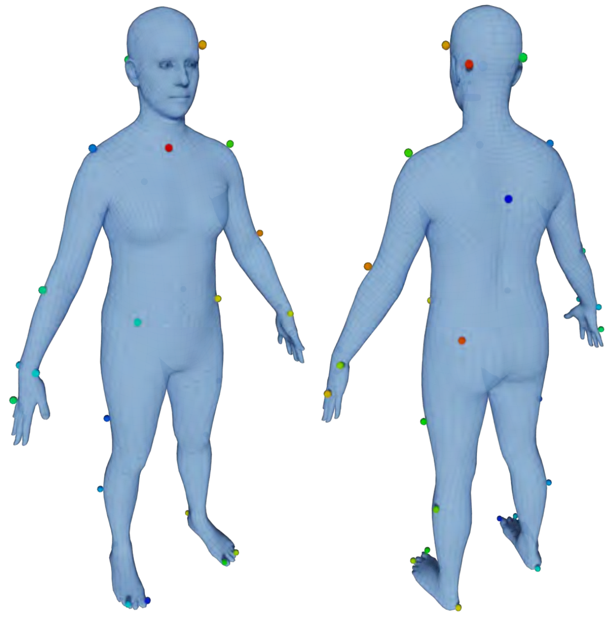
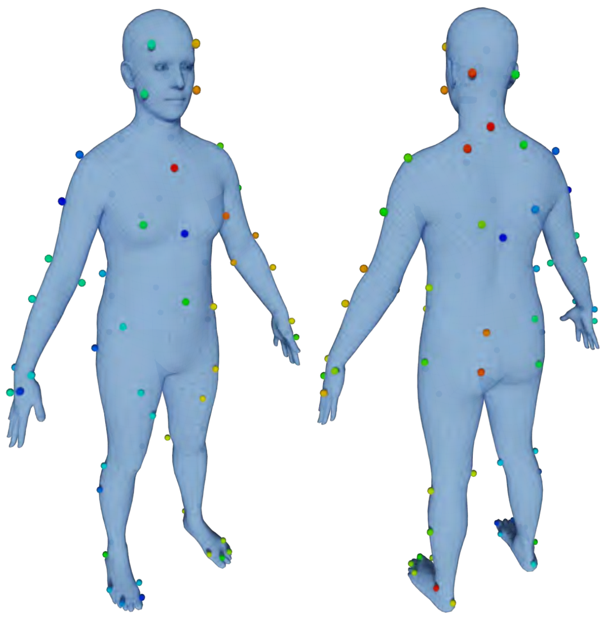
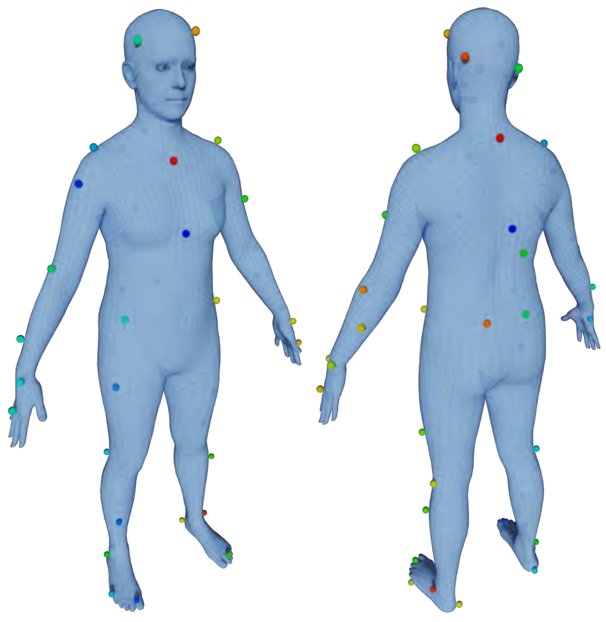
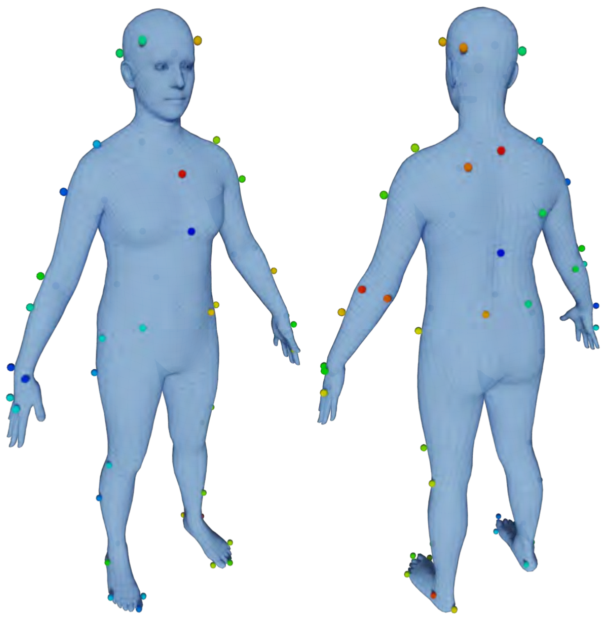

(d)

Figure 8: Modified HDM05 marker layout. Number of markers removed: (a) 3 (b) 5 (c) 9 (d) 12.


(a) BMLmovi


(b) BMLrub


(c) KIT

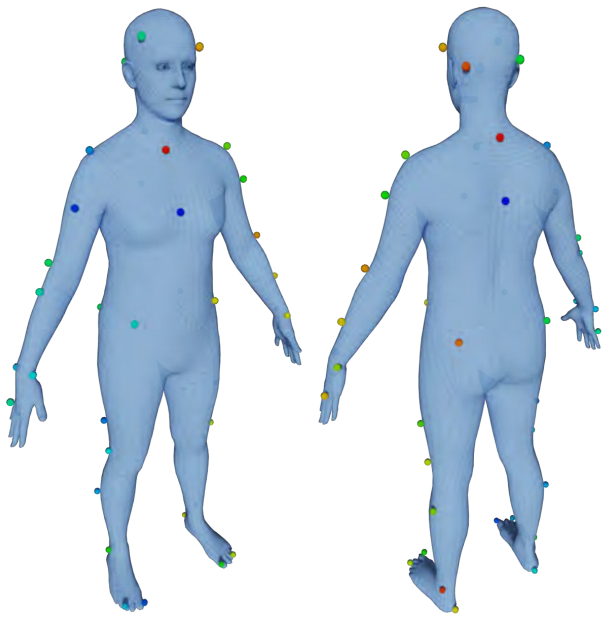

(d) HDM05

Figure 9: Marker layout of test and validation datasets.

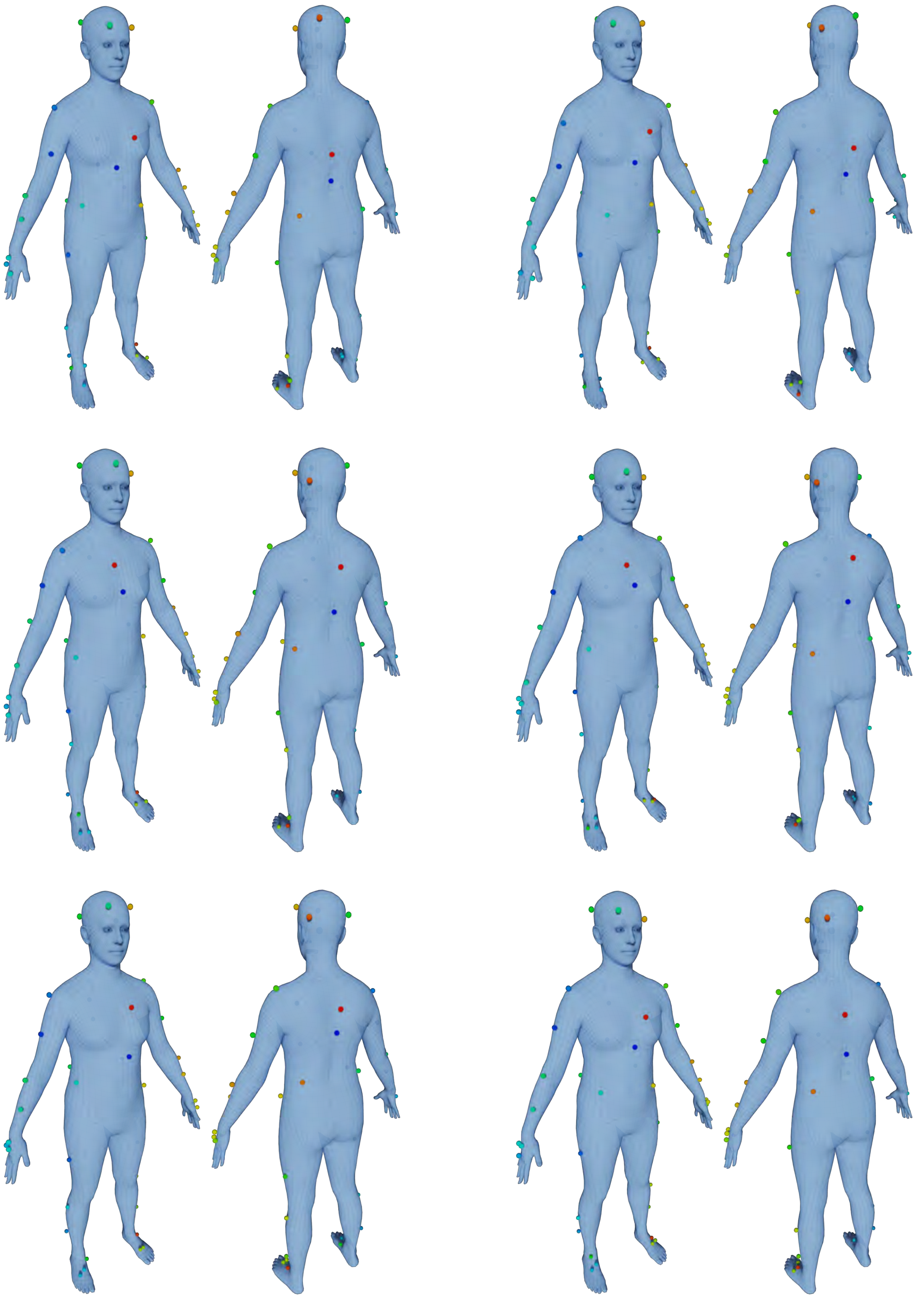


Figure 10: Significant variation of marker placement of DanceDB dataset on hands and foot.

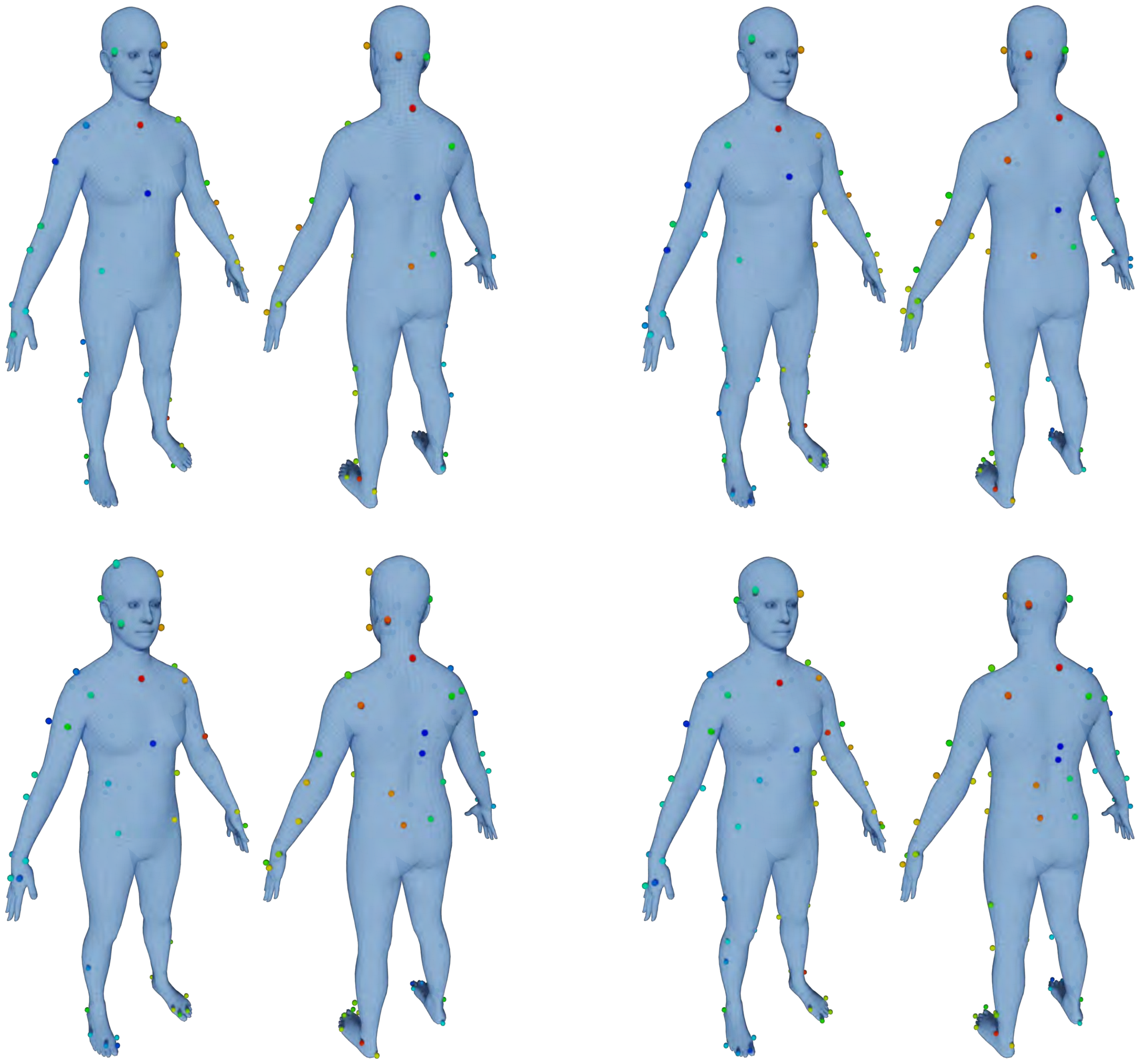


Figure 11: Sample of marker layouts used for training SOMA model for CMU-II dataset.