A. Architecture

The architecture of our MAPConNet is based on that of 3D-CoreNet [33] and is shown in Table 3. We use F, C, and R to denote the feature extraction, correspondence, and refinement modules, respectively. We let N_{id} and N_{pose} be the number of vertices in the input identity and pose meshes, respectively. One major difference between our method and 3D-CoreNet is that the output of our F module is split into identity and pose parts, each with a size of $N_{id} \times 128$ (for identity input) or $N_{pose} \times 128$ (for pose input), whereas 3D-CoreNet outputs a single feature of size $N_{id} \times 256$ or $N_{pose} \times 256$. In addition, we only feed the identity part of the latent feature to the refinement module, resulting in a slightly lower total number of trainable parameters (23.8 million compared to 24.5 million in 3D-CoreNet).

Module	*Layer*	*Output shape(s)*
F	Conv1D, 1	$N_{id} \times 64$, $N_{pose} \times 64$
	Conv1D, 1	$N_{id} \times 128$, $N_{pose} \times 128$
	Conv1D, 1	$N_{id} \times 256$, $N_{pose} \times 256$
	RB $\times 4$	$N_{id} \times 128$, $N_{id} \times 128$ $N_{pose} \times 128$, $N_{pose} \times 128$
C	Conv1D, 1	$N_{id} \times 256$
	Conv1D, 1	$N_{pose} \times 256$
	OT matrix	$N_{id} \times N_{pose}$
	Warped	$N_{id} \times 3$
R	Conv1D, 3	$N_{id} \times 1024$
	Conv1D, 1	$N_{id} \times 1024$
	ElaIN RB	$N_{id} \times 1024$
	Conv1D, 1	$N_{id} \times 512$
	ElaIN RB	$N_{id} \times 512$
	Conv1D, 1	$N_{id} \times 256$
	ElaIN RB	$N_{id} \times 256$
	Conv1D, 1	$N_{id} \times 3$

Table 3. **The layers of MAPConNet in detail.** We use "Conv1D, K" denote a 1D convolutional layer with kernel size K, "RB" for a residual block, and RB $\times 4$ means the same RB architecture repeated 4 times in succession. When a layer has multiple output shapes, that layer is shared between those outputs.

Algorithm 1 *MAPConNet in Supervised Learning.*

Require: Dataset of triplets $\{(\mathbf{x}^{A1}, \mathbf{x}^{B2}, \mathbf{x}^{B1}), \ldots\}$, wherein poses must be aligned across identities.

Initialise: Generator G with trainable weights θ_G, feature extractor F (which is a part of G), time step $t = 0$, initial learning rate η_0, maximum time step t_{max}.

repeat

Retrieve samples $\mathbf{x}^{A1} \in \mathbb{R}^{N_{pose} \times 3}$ and $\mathbf{x}^{B2}, \mathbf{x}^{B1} \in \mathbb{R}^{N_{id} \times 3}$ whose vertices are randomly re-ordered so that $\mathbf{x}^{B1}$ is aligned with $\mathbf{x}^{B2}$ but *not* $\mathbf{x}^{A1}$.

With $\mathbf{x}^{A1}$ as the pose input, $\mathbf{x}^{B2}$ as the identity input, compute OT matrix $\mathbf{T} \in \mathbb{R}^{N_{id} \times N_{pose}}$, and $\mathbf{B}$ from $\mathbf{T}$ (Equation 10).

Generate warped output $\mathbf{w}^{B1} = \mathbf{T}\mathbf{x}^{A1}$ and final output $\hat{\mathbf{x}}^{B1} \in \mathbb{R}^{N_{id} \times 3}$.

Compute $\mathcal{L}_s$ (Equation 3):

$$\lambda_{rec}\mathcal{L}_{rec}(\hat{\mathbf{x}}^{B1}; \mathbf{x}^{B1}) + \lambda_{edge}\mathcal{L}_{edge}(\hat{\mathbf{x}}^{B1}; \mathbf{x}^{B2}), \tag{17}$$

where $\mathbf{x}^{B1}$ is the ground truth.

Compute $\mathcal{L}^{ss}_{mesh}$ (Equation 11):

$$\begin{aligned} &l\left(F_{pose}(\mathbf{w}^{B1}), \mathbf{B}F_{pose}(\mathbf{x}^{A1}), F_{pose}(\mathbf{x}^{B2})\right) \\ &+l\left(F_{id}(\mathbf{w}^{B1}), F_{id}(\mathbf{x}^{B2}), \mathbf{B}F_{id}(\mathbf{x}^{A1})\right). \end{aligned} \tag{18}$$

Compute $\mathcal{L}_{point}$ (Equation 12):

$$\frac{1}{N_{id}}\sum_{j=1}^{N_{id}}(m + d(F(\mathbf{w}^{B1})_j, F(\mathbf{x}^{B2})_j, F(\mathbf{x}^{B2})_k))^+. \tag{19}$$

Compute gradients and update:

$$\theta_G \leftarrow \theta_G - \eta_t \frac{\partial \mathcal{L}_L}{\partial \theta_G}, \tag{20}$$

where

$$\mathcal{L}_L = \mathcal{L}_s + \mathcal{L}^{ss}_{mesh} + \mathcal{L}_{point}. \tag{21}$$

$t \leftarrow t + 1$.

until $t = t_{max}$.

B. Additional results

We include additional qualitative comparisons on unseen input meshes from SMPL (see Figure 10), SMAL (see Figure 11), DFAUST (see Figure 12), and MG (see Figure 13). Similar to observations we obtained from Figures 5 and 6, our MAPConNet while under a fully supervised setting can generate more accurate outputs compared to 3D-CoreNet, particularly in the limbs, as shown by the PMD heatmaps. Our unsupervised and semi-supervised results are also comparable to the supervised ones.

In Figure 12, one can clearly observe that our unsupervised and semi-supervised models (*U*) and (*V*) produce more accurate transfer results on DFAUST inputs than supervised models (*C*) and (*D*) that can only be trained using labelled data SMPL and partial models (*S*) and (*T*). In Figure 13, the identity and pose inputs are chosen from different datasets but one of them is from MG. In this scenario, the model has to not only perform pose transfer on unseen meshes across different domains, but also handle the discrepancy in the numbers of vertices of both inputs. Our

Algorithm 2 *MAPConNet in Unsupervised Learning.*

Require: Dataset of triplets $\{(\mathbf{x}^{A1}, \mathbf{x}^{A2}, \mathbf{x}^{B3}), \ldots\}$, wherein poses do not have to align across identities.

Initialise: Generator G with trainable weights θ_G, feature extractor F (which is a part of G), time step $t = 0$, initial learning rate η_0, maximum time step t_{max}.

repeat

Retrieve meshes $\mathbf{x}^{A1}, \mathbf{x}^{A2} \in \mathbb{R}^{N_{pose} \times 3}$ and $\mathbf{x}^{B3} \in \mathbb{R}^{N_{id} \times 3}$ whose vertices are randomly re-ordered so that $\mathbf{x}^{A1}$ is aligned with $\mathbf{x}^{A2}$ but *not* $\mathbf{x}^{B3}$.

With $\mathbf{x}^{A1}$ as pose input, $\mathbf{x}^{A2}$ as identity input, compute OT matrix $\mathbf{T}_{cc} \in \mathbb{R}^{N_{pose} \times N_{pose}}$.

Generate warped output $\mathbf{w}^{A1} = \mathbf{T}_{cc}\mathbf{x}^{A1}$ and final output $\hat{\mathbf{x}}^{A1} \in \mathbb{R}^{N_{pose} \times 3}$.

Compute $\mathcal{L}_{cc}$ (Equation 4):

$$\lambda_{rec}\mathcal{L}_{rec}(\hat{\mathbf{x}}^{A1}; \mathbf{x}^{A1}) + \lambda_{edge}\mathcal{L}_{edge}(\hat{\mathbf{x}}^{A1}; \mathbf{x}^{A2}). \tag{22}$$

With $\mathbf{x}^{A1}$ as pose input, $\mathbf{x}^{B3}$ as identity input, compute OT matrix $\mathbf{T} \in \mathbb{R}^{N_{id} \times N_{pose}}$, and $\mathbf{B}$ from $\mathbf{T}$ (Equation 10).

Generate warped output $\mathbf{w}^{B1} = \mathbf{T}\mathbf{x}^{A1}$ and final output $\hat{\mathbf{x}}^{B1} \in \mathbb{R}^{N_{id} \times 3}$.

With $SG(\hat{\mathbf{x}}^{B1})$ as pose input, $\mathbf{x}^{A2}$ as identity input, where SG stops the backward gradient flow, compute OT matrix $\mathbf{T}' \in \mathbb{R}^{N_{pose} \times N_{id}}$, and $\mathbf{B}'$ from $\mathbf{T}'$ (Equation 10).

Generate warped output $\tilde{\mathbf{w}}^{A1} = \mathbf{T}' SG(\hat{\mathbf{x}}^{B1})$ and final output $\tilde{\mathbf{x}}^{A1} \in \mathbb{R}^{N_{pose} \times 3}$.

Compute $\mathcal{L}_{sc}$ (Equation 5):

$$\lambda_{rec}\mathcal{L}_{rec}(\tilde{\mathbf{x}}^{A1}; \mathbf{x}^{A1}) + \lambda_{edge}\mathcal{L}_{edge}(\tilde{\mathbf{x}}^{A1}; \mathbf{x}^{A2}). \tag{23}$$

Compute $\mathcal{L}^{cc}_{mesh}$ (Equation 9):

$$l\left(F_{pose}(\mathbf{x}^{A1}), F_{pose}(\mathbf{w}^{A1}), F_{pose}(\mathbf{x}^{A2})\right) + l\left(F_{id}(\mathbf{x}^{A1}), F_{id}(\mathbf{x}^{A2}), F_{id}(\mathbf{w}^{A1})\right) \tag{24}$$

Compute $\mathcal{L}^{ss}_{mesh} = \mathcal{L}^{ss,1}_{mesh} + \mathcal{L}^{ss,2}_{mesh}$ (Equation 11), where:

$$\mathcal{L}^{ss,1}_{mesh} = l\left(F_{pose}(\mathbf{w}^{B1}), \mathbf{B}F_{pose}(\mathbf{x}^{A1}), F_{pose}(\mathbf{x}^{B3})\right) + l\left(F_{id}(\mathbf{w}^{B1}), F_{id}(\mathbf{x}^{B3}), \mathbf{B}F_{id}(\mathbf{x}^{A1})\right), \tag{25}$$

$$\mathcal{L}^{ss,2}_{mesh} = l\left(F_{pose}(\tilde{\mathbf{w}}^{A1}), \mathbf{B}'F_{pose}(SG(\hat{\mathbf{x}}^{B1})), F_{pose}(\mathbf{x}^{A2})\right) + l\left(F_{id}(\tilde{\mathbf{w}}^{A1}), F_{id}(\mathbf{x}^{A2}), \mathbf{B}'F_{id}(SG(\hat{\mathbf{x}}^{B1}))\right). \tag{26}$$

Compute $\mathcal{L}_{point} = \mathcal{L}^{cc}_{point} + \mathcal{L}^{sc}_{point}$ (Equation 12), where:

$$\mathcal{L}^{cc}_{point} = \frac{1}{N_{pose}} \sum_{j=1}^{N_{pose}} (m + d(F(\mathbf{w}^{A1})_j, F(\mathbf{x}^{A2})_j, F(\mathbf{x}^{A2})_k))^+, \tag{27}$$

$$\begin{aligned}\mathcal{L}^{sc}_{point} =& \frac{1}{N_{id}} \sum_{j=1}^{N_{id}} (m + d(F(\mathbf{w}^{B1})_j, F(\mathbf{x}^{B3})_j, F(\mathbf{x}^{B3})_k))^+ \\ &+ \frac{1}{N_{pose}} \sum_{j=1}^{N_{pose}} (m + d(F(\tilde{\mathbf{w}}^{A1})_j, F(\mathbf{x}^{A2})_j, F(\mathbf{x}^{A2})_k))^+.\end{aligned} \tag{28}$$

Compute gradients and update:

$$\theta_G \leftarrow \theta_G - \eta_t \frac{\partial \mathcal{L}_U}{\partial \theta_G}, \tag{29}$$

where

$$\mathcal{L}_U = \mathcal{L}_{cc} + \mathcal{L}_{sc} + \mathcal{L}^{cc}_{mesh} + \mathcal{L}^{ss}_{mesh} + \mathcal{L}_{point}. \tag{30}$$

$t \leftarrow t + 1$.

until $t = t_{max}$.

Algorithm 3 *MAPConNet in Semi-supervised Learning.*

Require: Labelled (pose aligned) dataset

$$\mathcal{X}^L = \{\mathbf{x}^{A,1}, \ldots, \mathbf{x}^{A,n}, \mathbf{x}^{B,1}, \ldots, \mathbf{x}^{B,n}, \ldots\},$$

and unlabelled (pose unaligned) dataset

$$\mathcal{X}^U = \{\mathbf{x}^{\alpha,\alpha_1}, \ldots, \mathbf{x}^{\alpha,\alpha_{n_\alpha}}, \mathbf{x}^{\beta,\beta_1}, \ldots, \mathbf{x}^{\beta,\beta_{n_\beta}}, \ldots\},$$

where $\alpha, \beta, \ldots$ are used to distinguish identities in $\mathcal{X}^U$ from those in $\mathcal{X}^L$, and $\alpha_1, \beta_1, \ldots$ are used to emphasize that poses are unaligned across identities.

1: **Initialise:** Generator G with trainable weights θ_G, feature extractor F (which is a part of G), time step $t = 0$, initial learning rate η_0, maximum time step t_{max}, and current number of unlabelled iterations done $n_U = 0$.
2: **repeat**
3: **if** iteration t is labelled **then**
4: Sample a triplet from $\mathcal{X}^L$, e.g. $(\mathbf{x}^{A1}, \mathbf{x}^{B2}, \mathbf{x}^{B1})$, in the same format as line 3 of Algorithm 1
5: Execute lines 4–9 of Algorithm 1.
6: **else**
7: **if** $n_U \equiv 0 \pmod 3$ **then**
8: Sample a triplet from both $\mathcal{X}^L$ and $\mathcal{X}^U$, e.g. $\mathbf{x}^{\alpha,\alpha_1}, \mathbf{x}^{\alpha,\alpha_2} \in \mathcal{X}^U$ and $\mathbf{x}^{C,3} \in \mathcal{X}^L$.
9: **else if** $n_U \equiv 1 \pmod 3$ **then**
10: Sample a triplet from both $\mathcal{X}^L$ and $\mathcal{X}^U$, e.g. $\mathbf{x}^{D,4}, \mathbf{x}^{D,5} \in \mathcal{X}^L$ and $\mathbf{x}^{\beta,\beta_1} \in \mathcal{X}^U$.
11: **else**
12: Sample a triplet from $\mathcal{X}^U$ only, e.g. $\mathbf{x}^{\gamma,\gamma_1}, \mathbf{x}^{\gamma,\gamma_2}, \mathbf{x}^{\delta,\delta_1} \in \mathcal{X}^U$.
13: **end if**
14: With a triplet in the same format as line 3 of Algorithm 2, execute lines 4–15 of Algorithm 2.
15: $n_U \leftarrow n_U + 1$.
16: **end if**
17: $t \leftarrow t + 1$.
18: **until** $t = t_{max}$.

semi-supervised model (*V*) again produces more realistic outputs than the supervised models, demonstrating its generalisability to complex unseen topologies.

In addition, Figure 14 shows outputs generated using various combinations of our disentangled latent representations. As one can observe, our model can indeed produce latent representations that are disentangled into identity and pose. This disentanglement is achieved in both synthetic and realistic datasets.

Finally, Figure 15 shows outputs generated from noisy inputs. A small but significant random uniform noise is added to each vertex of the pose input. As shown by the figure, out models are still able to generate outputs with the accurate pose and identity across multiple datasets.

C. Limitations and future work

Whilst our method achieves state-of-the-art results in our experiments, there are still limitations to be addressed in future work. For instance, we require both pose and identity inputs in CC during unsupervised learning to have the same vertex order, as the pose input is used as ground truth to supervise the model output which also needs to have the same vertex order as the identity input. There are potential solutions, such as modifying the behaviour of the correspondence module, or designing order-invariant loss functions (alternative to $\mathcal{L}_{rec}$) without impacting correspondence learning. However, this is not a serious issue since it is not impractical to maintain the same ordering within each identity during real world motion capture.

In semi-supervised learning, as our model can utilise unlabelled samples, it achieves superior performance compared to prior supervised methods which only have access to a limited number of labelled samples. Our model also tolerates a certain degree of domain gap between the labelled and unlabelled datasets as our model (*V*) can handle unseen inputs from both SMPL and DFAUST. However, this might not necessarily hold if the labelled and unlabelled sets are from *drastically* different domains, for instance, humans and animals. This might require fundamentally redesigning the correspondence module for cross-domain pose transfer. This is also a challenging problem to tackle due to the lack of cross-domain ground truths such as output, correspondence, or template pose.

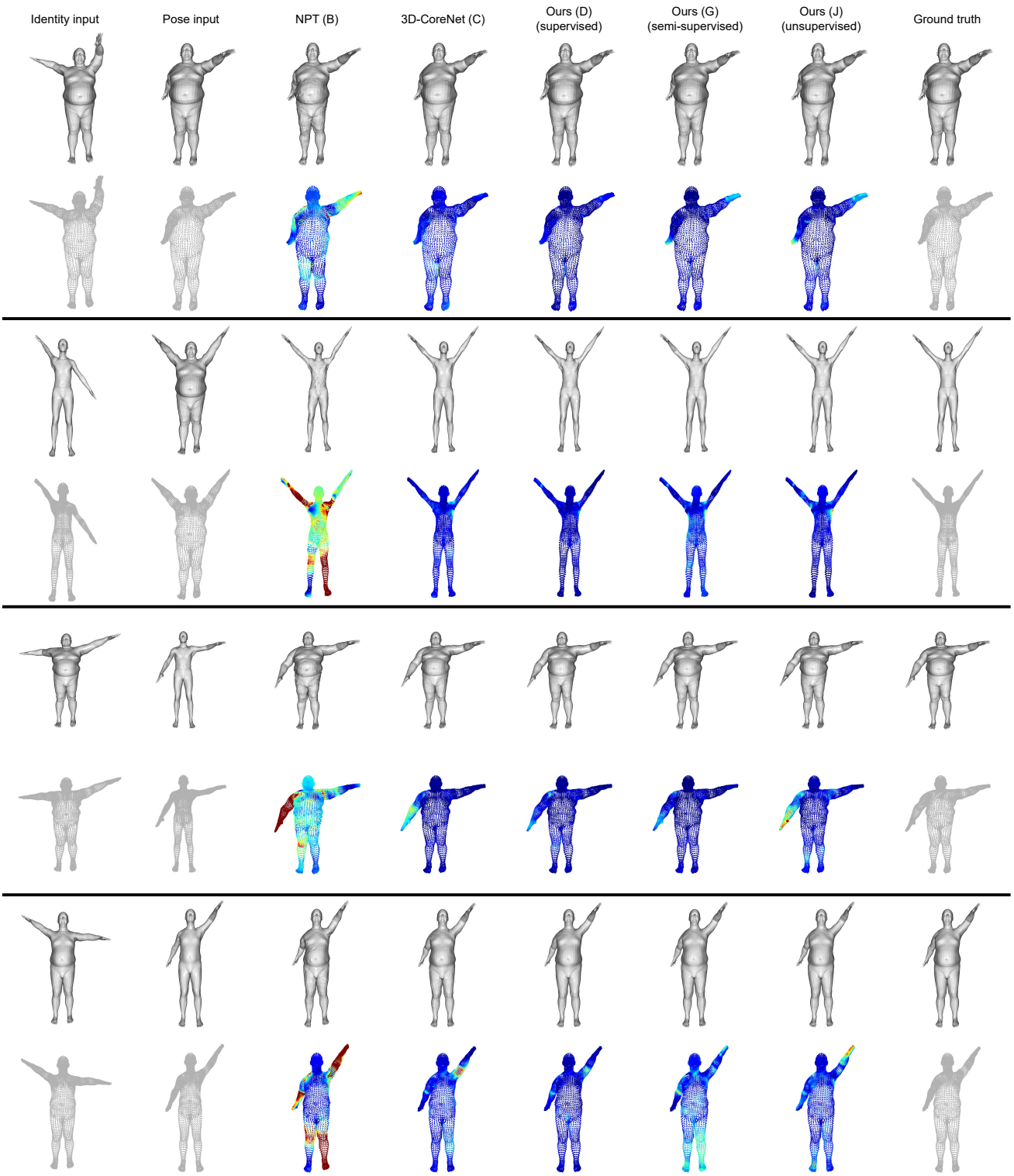

Figure 10. **Additional qualitative comparison on unseen SMPL inputs.** These are additional examples of pose transfer performed using various methods (labels defined in Table 1). Similar to Figure 5, the first rows show the rendered surfaces and the second rows show the corresponding point clouds with PMD heatmaps (dark red: high error; dark blue: low error).

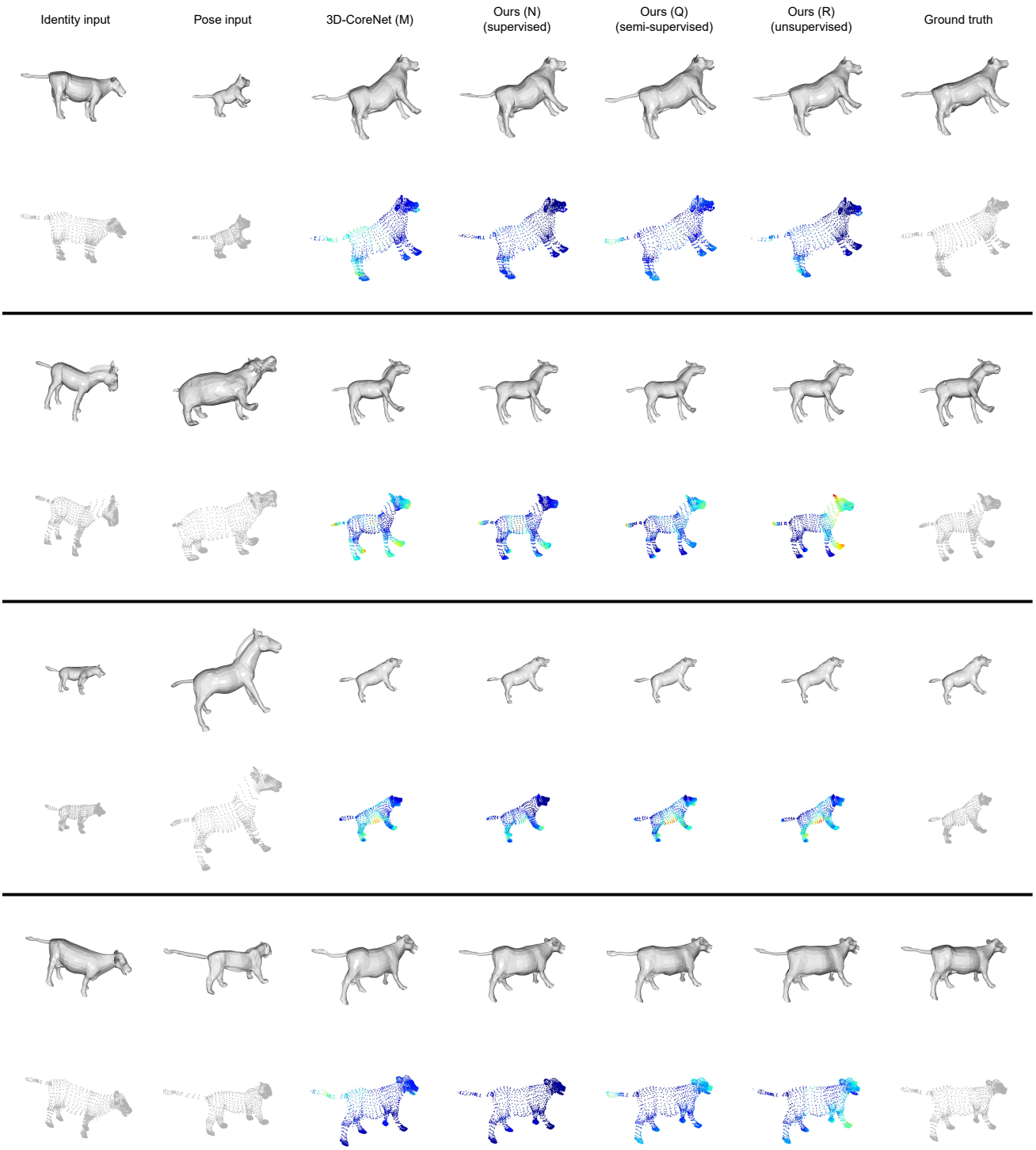

Figure 11. **Additional qualitative comparison on unseen SMAL inputs.** These are additional examples of pose transfer performed using various methods (labels defined in Table 1). Similar to Figure 6, the first rows show the rendered surfaces and the second rows show the corresponding point clouds with PMD heatmaps (dark red: high error; dark blue: low error).

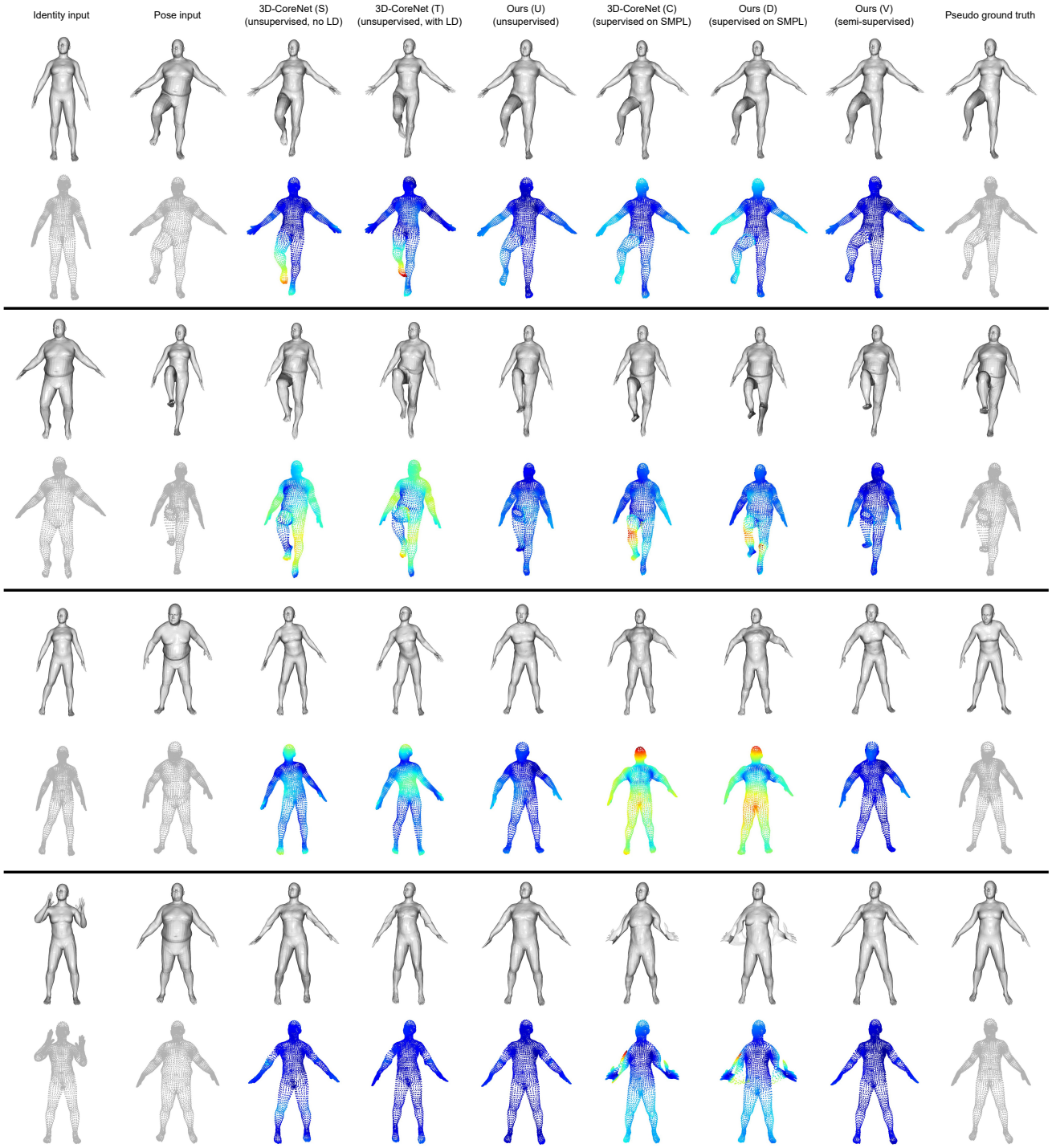

Figure 12. **Additional qualitative comparison on unseen DFAUST inputs.** These are additional examples of pose transfer performed using various methods (labels defined in Table 1). The first rows show the rendered surfaces and the second rows show the corresponding point clouds with PMD heatmaps (dark red: high error; dark blue: low error). The pseudo ground truths are generated using SMPL+H.

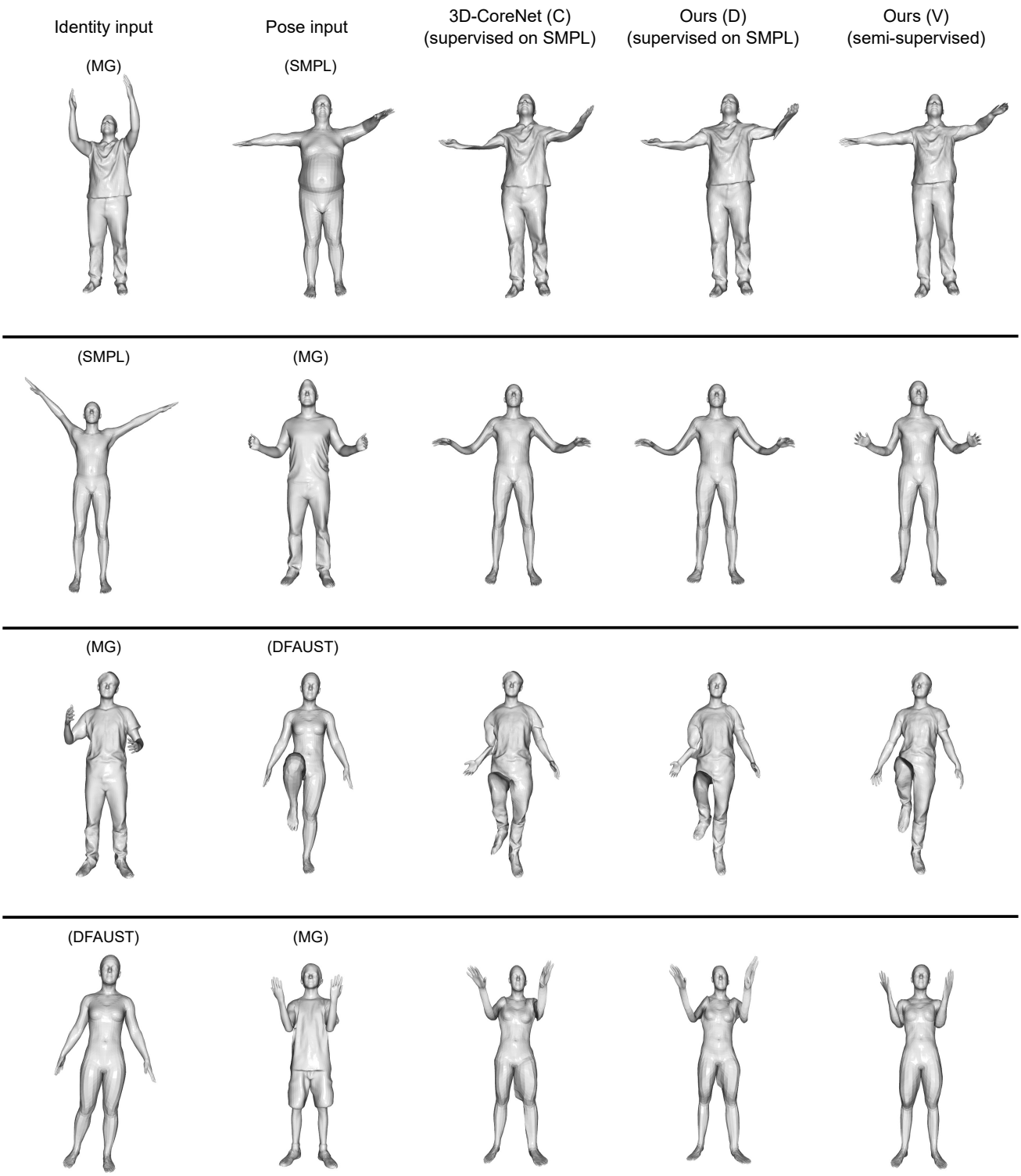

Figure 13. **Additional qualitative comparison on unseen MG, SMPL, and DFAUST inputs.** These are additional examples of pose transfer performed using various methods (labels defined in Table 1) with inputs from a mixture of different datasets (source dataset labelled in parentheses).

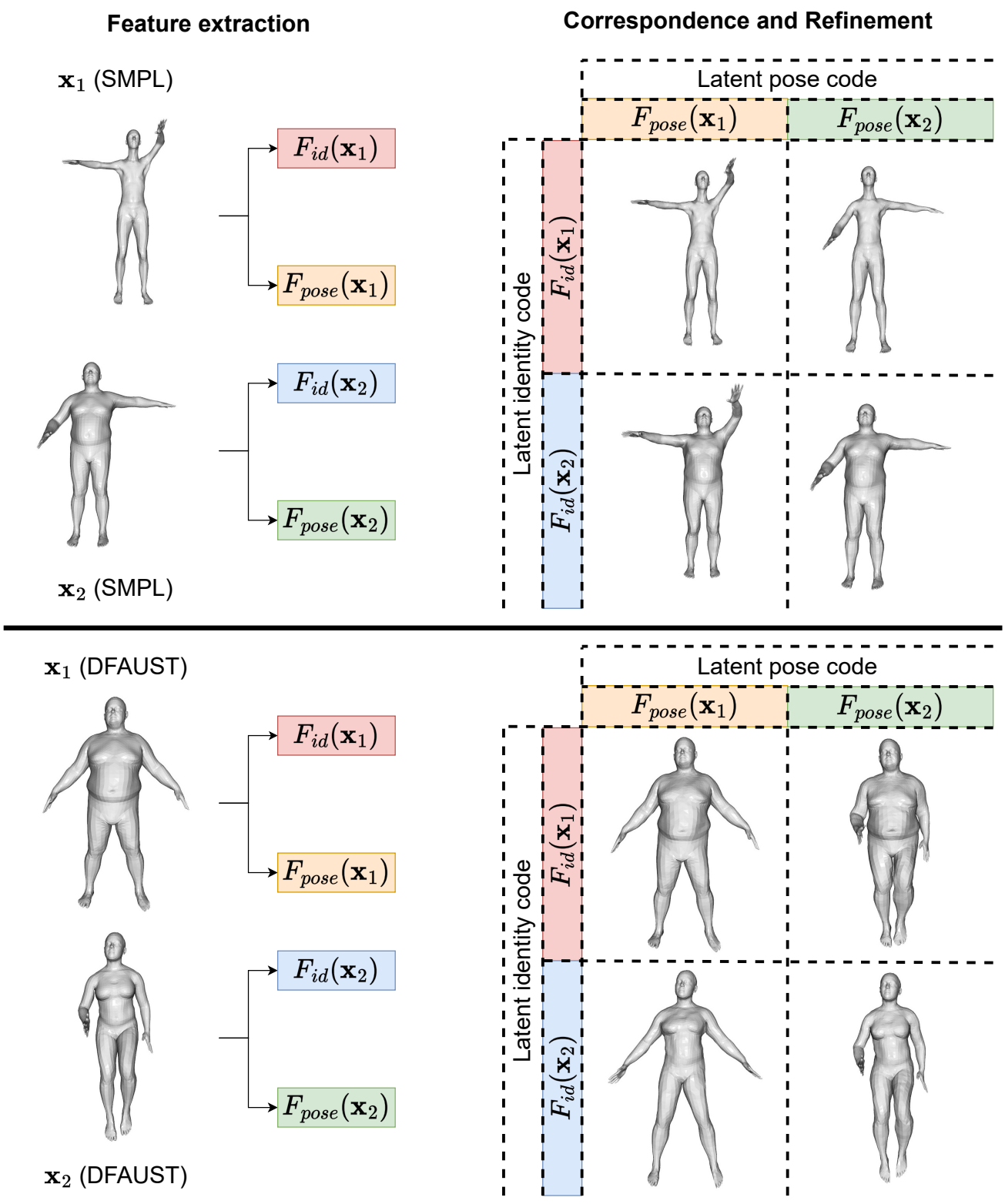

Figure 14. **Outputs generated from disentangled latent codes.** These results are generated by method (*V*) defined in Table 1 using various combinations of disentangled latent representations as described in Section 3.2. The vertex orders of $\mathbf{x}_1$ and $\mathbf{x}_2$ are aligned in order for their latent codes to be combined and processed by the model correctly.

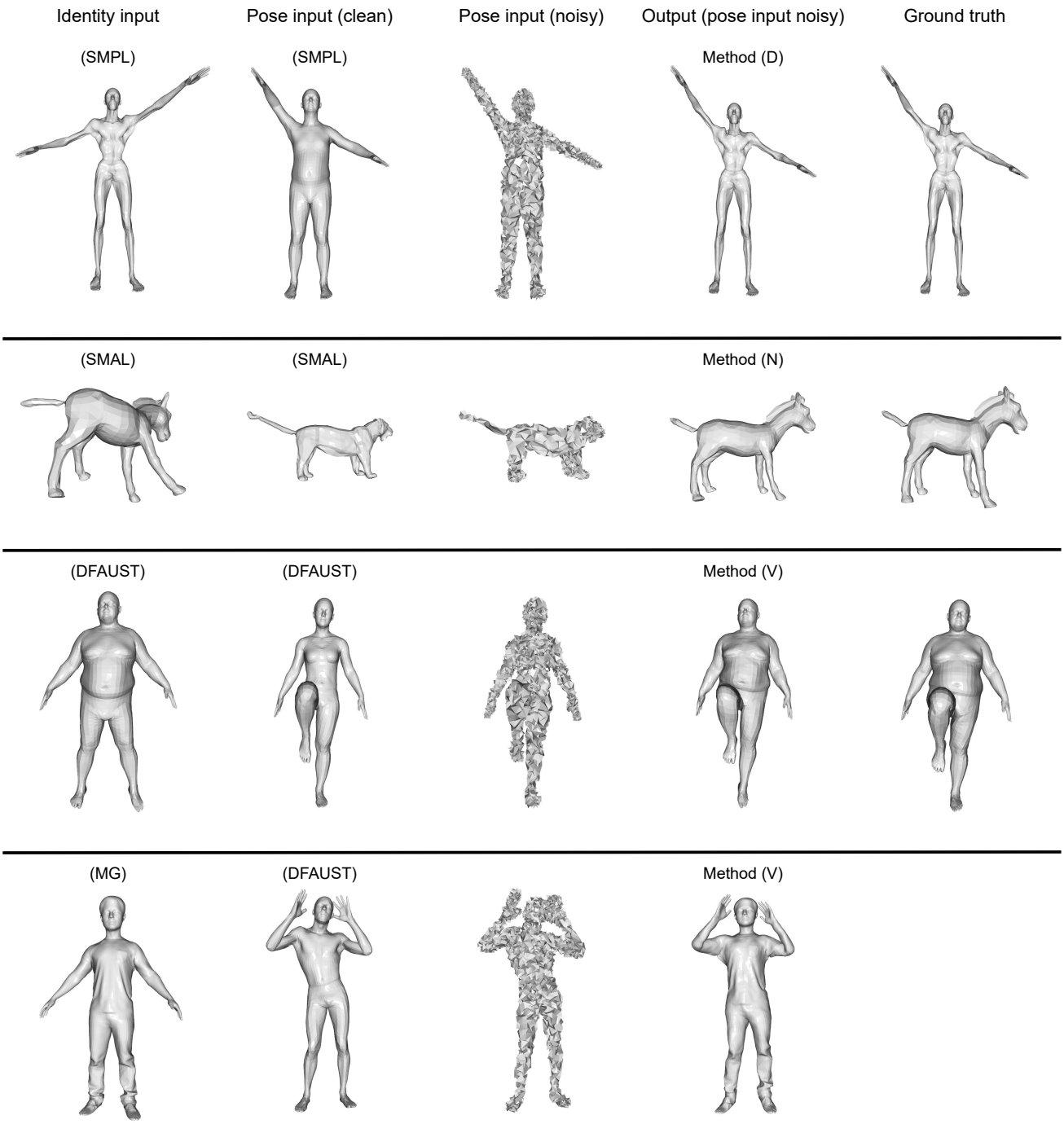

Figure 15. **Outputs generated from noisy inputs.** These results are generated from a noisy pose input using various methods (labels defined in Table 1). The vertices of the pose input are perturbed using random uniform noises. The clean pose inputs are shown here for reference but are not used as inputs for the model.