A. Overview

In this supplementary document, we first provide details about the proposed neural network modules (Sec. B) and their training procedure (Sec. C). Then, we present more qualitative and quantitative results that were not included in the paper due to the page limit (Sec. D). Lastly, we discuss the limitations of the proposed method (Sec. E). Lastly, we summarize the notations used in the paper to improve paper readability (Sec. F).

B. Network architecture

We first present details about the proposed neural encoders that are used to create the global feature vector \( z \in \mathbb{R}^{596} \) (128, 312, and 156 dimensions for structure, shape, and pose features respectively) and then detail the proposed neural network architectures.

B.1. Structure encoder

The structure encoder consists of 52 small multi-layer perceptrons that are organized in a tree structure, where each node of the tree corresponds to one joint in the human skeleton and outputs a compact feature vector \( b_k \in \mathbb{R}^6 \).

Each MLP node (Table B.1) takes as input a 19-dimensional feature vector – 6 dimensions for the parent feature, 9 for the rotation matrix, 1 for the bone length, and 3 for the joint location – and outputs a small bone code. These bone codes are concatenated to form one structure feature vector as explained in the main paper.

Since the root node does not have a parent node to be conditioned on its feature vector \( b_0 \in \mathbb{R}^6 \), we create \( b_0 \) with a single linear layer that takes as input vectorized \( \theta \) pose parameters and joint locations \( J \).

B.2. PointNet encoder

We implement a PointNet encoder (Figure B.1) to encode a point cloud into a fix-size feature vector. This network is used in Sec. 4.1.1 to create a 128-dimensional shape feature vector \( (P = 128 \text{ in Figure B.1}) \) and two 100-dimensional feature vectors for the inverse and the forward LBS networks \( (P = 100) \).

B.3. Bone projection layers

The bone projection layers \( \Pi_{\omega_k}: \mathbb{R}^{596} \mapsto \mathbb{R}^{12} \) create small per-bone features \( z_k \) and are implemented as efficient grouped 1D convolutions [26].

B.4. ONet

The architecture of the occupancy network is similar to the one proposed in [42] and is illustrated in Figure B.2.

B.5. Linear blend skinning networks

The inverse and the forward LBS networks are similar and implemented as MLPs conditioned on a latent feature vector (Figure B.3).

Forward LBS network. The latent feature vector for the forward LBS network \( c_{\text{fwd}} \in \mathbb{R}^{200} \) is created as a concatenate-
nation of two 100-dimensional feature vectors created by the PointNet encoder. One is created by encoding the estimated canonical vertices $\tilde{V}$, and the other one is created by encoding the estimated posed vertices $\tilde{V}$.

**Inverse LBS network.** The latent feature vector for the inverse LBS network $c_{\text{inv}} \in \mathbb{R}^{280}$ is created as a concatenation of the conditional features produced for the forward LBS network $c_{\text{fwd}}$ and an additional 80-dimensional feature vector created by a single linear layer that takes as input concatenated vectorized pose parameters and joint locations.

**C. Training**

We provide additional details for three independent training procedures. First, we train the inverse and the forward LBS networks and then use these two modules as deterministic differentiable functions for the occupancy training.

**Occupancy training.** All modules except the LBS networks are trained together and have a total of about 1.5M trainable parameters. For each batch, 1536 points are sampled uniformly and 1536 near the surface (1024 in the posed and 512 in the canonical space). Points that are sampled directly in the canonical space are not propagated through the forward LBS network and are associated with pseudo ground truth skinning weights $w_z$ to calculate local codes $z_x$.

The **inverse LBS network** has about 1.5M parameters. Each training batch consists of 1024 uniformly sampled points and 1024 points sampled around the mesh surface.

The **forward LBS network** has about 1.2M parameters. Each training batch consists of 1024 points sampled in the canonical space (512 uniformly sampled and 512 sampled near the surface) and points that are sampled for the training of the inverse LBS network. The latter set of points is mapped to the canonical space via the proposed pseudo ground truth weights.

**D. Additional experiments and results**

We supplement experiments for generalization (Figure D.4), for learning LBS (Sec. D.1), and placing people in scenes (Sec. D.2).

**D.1. Evaluation of linear blend skinning networks**

Our forward LBS network operates in the canonical space and does not need to deal with challenging human poses as the inverse LBS network. Here, we quantify the performance gap between these two networks on the unseen portion of query points for three experimental setups presented in the main paper.

As an evaluation metric, we report the $l_1$ distance between pseudo ground truth weights and predicted weights by the inverse $l_1^{\text{inv}}$ and the forward $l_1^{\text{fwd}}$ LBS networks.

Quantitative results (Table D.2) show that the forward LBS network consistently outperforms the inverse LBS network across all settings. Furthermore, the inverse LBS network performs worse when subjects are not seen during the training.

**D.2. Generating people in scenes**

We provide additional qualitative results (Figure D.5) for the experiment presented in Figure 5 and visualize SDF (Figure D.7) that is used to compute the human-scene collision score. Although the SDF is very noisy, we still use it to compute this score for a fair comparison with PLACE [72].

We further provide an experiment on a larger Replica room [61]. Similar to Sec. 6.4, we sample 50 people from PLACE [72] and select 60 human body pairs that interpenetrate. These pairs are then optimized with our method by minimizing the proposed point-based loss (20).

Quantitative results (Table D.3) demonstrate that our approach improved collision scores over the baseline [72], except for the human-scene score which is unreliable due to the aforementioned noisy SDF (Figure D.7). The qualitative results displayed in Figure D.6 show that our method successfully resolves deep interpenetrations with scene geometry which could not be straightforwardly achieved with differentiable mesh-based collision methods – for example, a modified version of the approach presented in [62]. This indicates that our volumetric error signal is more effective than the surface error signal imposed by mesh-based methods.

**E. Limitations**

We observed some challenging scenarios in which our learned inverse linear blend skinning network may fail to correctly map a query point to the canonical space and consequently distort occupancy in the posed space. This problem occurs when the network is not well trained and two body parts are close to each other or even self-intersect. An example of a failure case of an unseen subject is displayed in Figure E.8. Therefore, a promising future direction is to explicitly model self-contact for learning the occupancy representation.
Figure D.4. Generalization experiment. Qualitative results for the generalization experiment (Sec. 6.3, Table 2) for DFaust [5] unseen poses (top row) and MoVi [20] unseen subjects (bottom row).

<table>
<thead>
<tr>
<th>Experiment type</th>
<th>$l_1^{inv}$</th>
<th>$l_1^{fwd}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-person occupancy (Sec. 6.2)</td>
<td>0.1894</td>
<td>0.1252</td>
</tr>
<tr>
<td>Generalization: unseen poses (Sec. 6.3)</td>
<td>0.1997</td>
<td>0.0818</td>
</tr>
<tr>
<td>Generalization: unseen subjects (Sec. 6.3)</td>
<td>0.2138</td>
<td>0.1098</td>
</tr>
</tbody>
</table>

Table D.2. Evaluation of the inverse and the forward linear blend skinning networks. Reported $l_1$ distance shows that the forward LBS network consistently outperforms the inverse LBS network across all experiment settings. This is expected because the inverse LBS network reasons about different body shapes and poses, while the fwd-LBS network reasons only about body shapes since the pose in the canonical space is constant. "Multi-person" and "unseen poses" experiments are performed on the DFaust [5] dataset, while the "unseen subjects" experiment is performed on the MoVi [20] dataset. More details on the experimental setups are available in the paper (Sec. 6.2, Sec. 6.3).

F. Notation

Lastly, we summarize the key notation terms in Table F.4 for improved readability.

Table D.3. Improved PLACE [72]. Results on two Replica [61] rooms. Our proposed optimization method successfully mitigates interpenetrations between scene geometry and other humans. Note that the human-scene score is unreliable metric due to noisy scene SDF.
Figure D.5. **Improved PLACE** [72]. Additional viewpoints of a Replica room [61] for results presented in the paper (Figure 5). Our point-based loss effectively resolves collisions of the human pairs. Quantitative results are reported in Table D.3.
Figure D.6. **Improved PLACE** [72]. Results demonstrate that our method can resolve challenging interpenetrations with scene geometry. Note that this complex penetrations with the thin mesh geometry cannot be straightforwardly fixed with mesh-based intersection methods [62] that impose a surface-based error signal. This demonstrates that our flexible volumetric point-based loss is more efficient, which is quantified by the improved collisions scores displayed in Table D.3.
Figure D.7. **Noisy SDF** that is used to compute the human-scene score for results presented in the paper (Sec. 6.4, Table 3, Figure 5).

Figure E.8. **Failure case.** An example of an unseen MoVi [20] subject when two hands self-intersect. The inverse LBS network may incorrectly map a given query point to the canonical space for self-intersected regions which consequently distorts occupancy representation in the posed space.

<table>
<thead>
<tr>
<th>Input parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K \in \mathbb{R}$</td>
</tr>
<tr>
<td>$x \in \mathbb{R}^3$</td>
</tr>
<tr>
<td>$G_k \in \mathbb{R}^{4,4}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMPL parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
</tr>
<tr>
<td>$\mathcal{S}$</td>
</tr>
<tr>
<td>$\mathcal{P}$</td>
</tr>
<tr>
<td>$\mathcal{W}$</td>
</tr>
<tr>
<td>$\mathcal{J}$</td>
</tr>
<tr>
<td>$\bar{T} \in \mathbb{R}^{N,3}$</td>
</tr>
<tr>
<td>$V \in \mathbb{R}^{N,3}$</td>
</tr>
<tr>
<td>$\bar{V} \in \mathbb{R}^{N,3}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMPL functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B_P$</td>
</tr>
<tr>
<td>$B_S$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Estimated parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{x} \in \mathbb{R}^3$</td>
</tr>
<tr>
<td>$\hat{V} \in \mathbb{R}^{N,3}$</td>
</tr>
<tr>
<td>$\hat{V} \in \mathbb{R}^{N,3}$</td>
</tr>
<tr>
<td>$w_\hat{x} \in \mathbb{R}^K$</td>
</tr>
<tr>
<td>$w_\hat{x} \in \mathbb{R}^K$</td>
</tr>
</tbody>
</table>

Table F.4. **Notation summary.**
References


[41] Bruce Merry, Patrick Marais, and James Gain. Animation space: A truly linear framework for character animation. ACM Transactions on Graphics, 2006. 2


[57] Hanan Samet. The design and analysis of spatial data structures. Addison-Wesley Reading, MA, 1990. 2


[59] Vincent Sitzmann, Julien Martel, Alexander Bergman, David


