NPMs: Neural Parametric Models for 3D Deformable Shapes

Pablo Palafox[1] Aljaž Božič[1] Justus Thies[1,2] Matthias Nießner[1] Angela Dai[1]
[1]Technical University of Munich [2]Max Planck Institute for Intelligent Systems, Tübingen

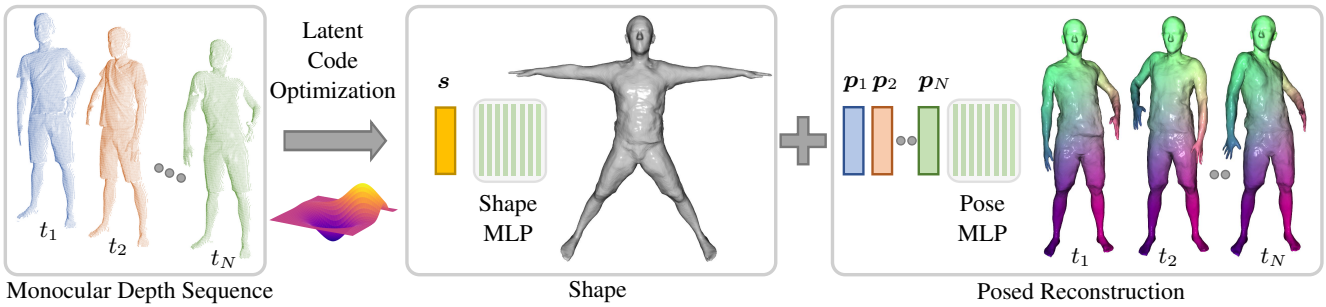

Figure 1: Given an input monocular depth sequence, our Neural Parametric Models (NPMs), composed of learned latent shape and pose spaces, enable optimizing over the spaces to fit to the observations at test time, similar to traditional parametric model fitting (e.g., SMPL [26]). NPMs can be constructed from a dataset of deforming shapes without strong requirements on surface correspondence annotations or category-specific knowledge. Our implicit shape and pose spaces enable expression of finer-scale details while providing a well-regularized space to fit to new observations of deforming shapes.

Abstract

Parametric 3D models have enabled a wide variety of tasks in computer graphics and vision, such as modeling human bodies, faces, and hands. However, the construction of these parametric models is often tedious, as it requires heavy manual tweaking, and they struggle to represent additional complexity and details such as wrinkles or clothing. To this end, we propose Neural Parametric Models (NPMs), a novel, learned alternative to traditional, parametric 3D models, which does not require hand-crafted, object-specific constraints. In particular, we learn to disentangle 4D dynamics into latent-space representations of shape and pose, leveraging the flexibility of recent developments in learned implicit functions. Crucially, once learned, our neural parametric models of shape and pose enable optimization over the learned spaces to fit to new observations, similar to the fitting of a traditional parametric model, e.g., SMPL. This enables NPMs to achieve a significantly more accurate and detailed representation of observed deformable sequences. We show that NPMs improve notably over both parametric and non-parametric state of the art in reconstruction and tracking of monocular depth sequences of clothed humans and hands. Latent-space interpolation as well as shape / pose transfer experiments further demonstrate the usefulness of NPMs. Code is publicly available at https://pablopalafox.github.io/npms.

1. Introduction

Modeling deformable surfaces is fundamental towards understanding the 4D world that we live in, as well as creating or manipulating dynamic content. While significant progress has been made in understanding the reconstruction of 3D shapes [12, 15, 16, 46, 37, 31], representing dynamic, deforming surfaces remains challenging.

Over the past years, parametric 3D models have seen remarkable success for domain-specific representations, such as for human bodies (e.g., SCAPE [2], SMPL [26], Adam [20]), hands (MANO [43]), animals (SMAL [50]) and faces ([39], FLAME [23], [40]). These models have enabled a wide range of exciting applications and are instrumental in modeling deformable 3D objects. However, the construction of such a parametric model is a rather complex and tedious task, requiring notable manual intervention and incorporation of object-specific constraints in order for the parametric model to well-represent the space of possible shapes and deformations. Moreover, such parametric models often struggle to represent additional complexity and details of deforming shapes, e.g., clothing, hair, etc.

We propose *Neural Parametric Models* (NPMs), an alternative formulation to traditional parametric 3D models where we learn a disentangled shape and pose representation that can be used like a traditional parametric model to fit to new observations. We leverage the representation

power of implicit functions to learn disentangled shape and pose spaces from a dataset that does not require surface registration among all samples; this flexibility enables training on a wider variety of data. We also do not make object-specific assumptions about the kinematic chain, the number of parts or the skeleton. For training, our approach only requires that the same identity or shape can be seen in different poses, including a canonical pose.

Once trained, we can leverage our learned shape and pose representations as regularized spaces to be smoothly optimized over to fit to new observations at test time. Additionally, our disentangled implicit representations of shape and pose enable modeling arbitrary connectivity and topology, as well as finer-scale levels of detail. Thus, optimizing over our shape and pose spaces during inference enables representation of a globally consistent shape and temporally consistent poses while maintaining geometric fidelity.

Given a dataset of various shape identities and possibly different topologies, as well as various deformations of each shape identity (but without requiring registration of any identity to the others), we then train both shape and pose spaces in auto-decoder fashion. We learn a shape code for each identity, with shape codes representing an SDF of the shape geometry. Pose codes represent the flow field from the canonical shape of an identity to a given posed shape of the same identity. Flow predictions are conditional on both a shape and a pose latent code, in order to represent shape-dependent deformations as well as to help learning disentangled shape and pose spaces.

We demonstrate our Neural Parametric Models on the task of reconstruction and tracking of monocular depth sequences, as well as their capability in shape and pose transfer and interpolation. In comparison with state-of-the-art parametric 3D models and implicit 4D representations, our NPMs capture higher-quality reconstructions with finer detail and more accurate non-rigid tracking.

In summary, we present the following key contributions:

- We propose an alternative formulation to traditional parametric 3D deformable models, where shape and pose are disentangled into separate latent spaces via two feed-forward networks that are learned from data alone, i.e., without requiring domain-specific knowledge such as the kinematic chain or number of parts.
- Importantly, our approach shows regularization capabilities that enable test-time optimization over the latent spaces of shape and pose to tackle the challenging task of fitting a model to monocular depth sequences, while retaining the detail present in the data.

2. Related Work

Traditional parametric models. Parametric 3D models have become a predominant approach to disentangle 3D deformable shapes into several factors, e.g., shape and pose, for domains such as human bodies [2, 26, 20, 48], hands [43], animals [50] and faces [39, 23, 40]. SMPL [26] is a very popular parametric model of the human body based on blend shapes in combination with a skeleton, and constructed from a dataset of 3D body scans. Extensions exist to also model soft tissue [42] and clothing [4, 45, 38, 28, 41, 1]. To construct such parametric models, various domain-specific annotations are often required, such as the number of parts or the kinematic chain. In contrast, our NPMs can be learned from data belonging to a domain without requiring any expert knowledge or manual intervention. Additionally, approaches such as SMPL [26] or GHUM [48] are skinned vertex-based models which can struggle to represent complex surface features (e.g., wrinkles, clothing). By leveraging recently proposed implicit functions, our approach can naturally capture more intricate surface detail.

Implicit representations for 3D shapes. Implicit representations such as Signed Distance Fields (SDFs) have been widely used to represent surfaces for 3D reconstruction, both static [19, 34, 36, 13] and dynamic [33, 44, 8]. Various approaches to learn 3D shape generation have thus also leveraged such implicit definitions of a surface represented in a volumetric grid, where the regular structure is well-suited for convolutions but also induces cubic memory growth for high resolutions [47, 14].

Recent work in learning continuous implicit functions to represent shapes removes the explicit grid structure limitation, and shows strong promise in generating 3D shapes [10, 17, 31, 32, 37, 11]. In particular, DeepSDF [37] proposes a feed-forward network to predict an SDF value given a query location conditional on a latent code that represents the shape, trained in auto-decoder fashion. However, these approaches produce static surfaces that are not controllable, since shape and pose are entangled within a latent code. Our approach leverages the representation power of implicit functions to learn disentangled implicit spaces –one for shape and one for pose–, enabling controllable 3D models which can be used to fit dynamic data or generate new posed shapes through interpolation in the spaces.

Learned representations for deformable shapes. Recently, various learned approaches for representing deformable objects have been proposed [18, 49, 35, 3, 7, 6, 24]. Groueix et al. [18] proposes a learned, template matching approach. Zhou et al. [49] learns disentangled shape and pose representations from datasets of registered meshes via a combination of self-consistency and cross-consistency constraints, without requiring expert knowledge in the dataset. Our NPMs also do not require any manual annotations, but, unlike [49], we do not require a

template nor registration of identities in the dataset to each other. This enables our NPMs to represent complex detail and broader shape variety, such as clothed bodies; moreover, we further explore how our learned latent spaces can be optimized over at test time to fit to sparse observations.

The recently proposed OFlow [35] builds on the implicit 3D OccNet [31] to learn 4D reconstruction from images or sparse point clouds. OFlow learns a temporally and spatially continuous vector field which assigns a motion vector to every point in space and time, opening a promising avenue for spatio-temporal reconstruction, but remaining limited to very short sequences. IP-Net [3] presents an approach to combine learned implicit functions and traditional parametric models to produce controllable models of humans. An implicit network [11] predicts inner body surface and outer detailed surface, to which SMPL+D [1, 22] is fit for controllability. NPMs also aim to provide a controllable model, but rather than using a SMPL basis, we learn a parametric model of disentangled shape and pose latent spaces which enable fitting by optimizing over the spaces jointly.

3. Method

We introduce Neural Parametric Models (NPMs), a learned approach to construct parametric 3D models from a dataset of different posed identities; unlike traditional parametric 3D models, we do not require the dataset to have annotations for domain-specific properties such as the kinematic chain, skeleton or the surface-to-part mappings. To construct our NPMs, we learn a latent space of (canonically-posed) shapes, along with a latent space of poses conditional on the shape. We can then optimize jointly over the learned shape and pose spaces to fit to a new observation.

Figure 2 shows an overview of our approach. We employ an implicit representation for the shape space, encoding the SDF value for an input point, as well as for the pose space, encoding flow from the canonical to the deformed pose for an input point. These implicit representations, coupled with the joint optimization over the learned spaces, enable capturing details present in the input data while effectively regularizing the shape and pose latent spaces.

3.1. Neural Parametric Models

Given a dataset of meshes featuring a set of shape identities from the same class category in different poses, our goal is to learn a parametric model that not only regularizes the shape and pose latent spaces of the object class, but remains expressive enough to capture local details when fitting the learned model to new observations. To learn NPMs from a dataset, the latter should follow two simple constraints: (1) each shape identity is posed canonically (e.g., T-pose), and (2) each shape identity has several posed or deformed instances for which dense surface correspondences to the canonical shape are available. Various existing datasets (e.g., AMASS [30], DeformingThings4D [25], CAPE [29], MANO [43], etc.) easily fulfill these requirements.

We construct our NPMs by learning disentangled shape and pose spaces, leveraging implicit representations characterized by separate Multi-Layer Perceptrons (MLPs), one for shape and one for pose. The shape encoding learns to implicitly represent the different identities in their canonical pose. The pose space is conditional on both pose and shape codes, and learns a continuous deformation field around a canonical shape mapping points from this canonical shape to a deformed shape.

3.2. Learned Shape Space

Our shape space is learned by an MLP which predicts the implicit SDF for shape identities in their canonical pose; the shape is then defined as the zero iso-surface decision boundary and can be extracted with Marching Cubes [27].

Our shape MLP is trained in auto-decoder fashion as proposed by DeepSDF [37], where no encoder is used during training, directly optimizing over the latent code space. Each canonically-posed shape identity i in the training set is encoded in a D_s-dimensional latent shape code $\boldsymbol{s}_i$. The shape MLP learns to map an input point $\boldsymbol{x} \in \mathbb{R}^3$ in the canonical space, conditioned on $\boldsymbol{s}_i$, to an SDF value prediction $\tilde{d}$:

$$f_{\theta_\text{s}} : \mathbb{R}^3 \times \mathbb{R}^{D_\text{s}} \to \mathbb{R}, \quad (\boldsymbol{s}_i, \boldsymbol{x}) \mapsto f_{\theta_\text{s}}(\boldsymbol{s}_i, \boldsymbol{x}) = \tilde{d}. \quad (1)$$

We train our shape MLP on the S shape identities of the dataset, in their canonical poses (see Fig. 2). To this end, we first normalize our training shapes (both canonically-posed and randomly posed) to reside within a unit bounding box by dividing all shapes by the extent of the largest bounding box in the dataset. We then make our S canonical shapes watertight. Note that the arbitrarily posed shapes used for training the pose MLP do not need to be watertight.

Next, for each i-th shape identity in the train set, we sample N_s points $\{\boldsymbol{x}_i^k\}_{k=1}^{N_\text{s}} \in \mathbb{R}^3$ along with their corresponding SDF values $\{d_i^k\}_{k=1}^{N_\text{s}} \in \mathbb{R}$. These train samples come from two sources: (1) N_s^ns near-surface points sampled randomly within a distance of 0.05 from the surface of the shape and (2) N_s^u points uniformly sampled within the unit bounding box, such that $N_\text{s} = N_\text{s}^\text{ns} + N_\text{s}^\text{u}$. We refer readers to the appendix for further details.

Finally, to learn the latent shape space we minimize the following reconstruction energy over all shape identities in their canonical pose with respect to the individual shape codes $\{\boldsymbol{s}_i\}_{i=1}^S$ and the shape MLP weights θ_s:

$$\underset{\theta_\text{s}, \{\boldsymbol{s}_i\}_{i=1}^S}{\arg\min} \sum_{i=1}^{S} \Big(\sum_{k=1}^{N_\text{s}} \mathcal{L}_\text{s}(f_{\theta_\text{s}}(\boldsymbol{s}_i, \boldsymbol{x}_i^k), d_i^k) + \frac{\|\boldsymbol{s}_i\|_2^2}{\sigma_\text{s}^2} \Big), \quad (2)$$

where $\mathcal{L}_s$ is a truncated ℓ_1-loss on the predicted SDF $\tilde{d}_i^k$:

$$\mathcal{L}_\text{s}(\tilde{d}_i^k, d_i^k) = \big| \operatorname{clamp}(\tilde{d}_i^k, \delta) - \operatorname{clamp}(d_i^k, \delta) \big| \quad (3)$$

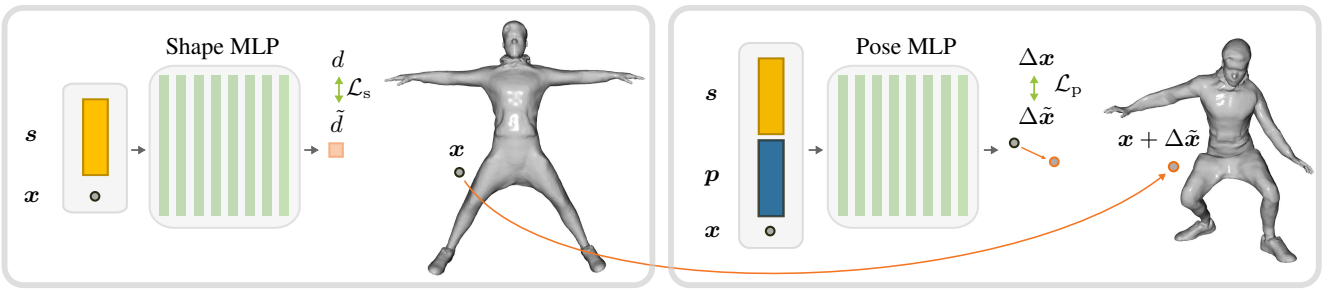

Figure 2: **Architecture Overview**. To train our NPMs, we first learn a latent space of shape identities in their canonical poses (e.g., T-pose) by conditioning our shape MLP on the shape code $\boldsymbol{s}_i$ assigned to each i-th identity. Given this learned shape space, we learn a deformation field around the canonically-posed shape which maps points from this shape's canonical space to a j-th posed version of the shape. We thus train a pose MLP conditioned on both the identity's latent shape code $\boldsymbol{s}_i$ and the corresponding latent pose code $\boldsymbol{p}_j$ to predict a flow vector $\Delta\boldsymbol{x}$ for a query point $\boldsymbol{x}$ sampled in the canonical pose.

and $\text{clamp}(d, \delta) := \min(\max(-\delta, d), \delta)$ defines the truncation region over which we maintain a metric SDF. The ℓ_2-regularization on the latent codes, controlled by the parameter σ_s, is required to enforce a compact shape manifold, as was found in [37].

Implementation details. We use eight fully-connected, F_s-dimensional layers in our shape MLP, with ReLUs, and a final fully-connected layer followed by tanh which regresses the scalar SDF value. In our experiments, $F_\text{s} = 512$ and $D_\text{s} = 256$. We use the Adam optimizer [21] and learning rates of 5×10^{-4} and 1×10^{-3} for the shape MLP f_{θ_s} and the shape codes $\{\boldsymbol{s}_i\}_{i=1}^{S}$, respectively. Additionally, we apply a learning rate decay factor of 0.5 every 500 epochs. We employ a regularization of $\sigma_s = 10^2$ on the shape codes, and set the SDF truncation to $\delta = 0.1$. The latent shape codes $\{\boldsymbol{s}_i\}_{i=1}^{S}$ are initialized randomly from $\mathcal{N}(0,\, 0.01^2)$.

3.3. Learned Pose Space

Our pose space is learned by an MLP which predicts a deformation field f_{θ_p} that maps points around identities in their canonical pose to the corresponding point locations in the space of the deformed pose. In particular, for a query point $\boldsymbol{x}$ in the canonical space of identity i, the pose MLP predicts a flow vector $\Delta\tilde{\boldsymbol{x}}$ that deforms the point from the canonical to the deformed space j, conditional on a D_p-dimensional latent pose code $\boldsymbol{p}_j$ as well as on the latent shape code $\boldsymbol{s}_i$. This flow prediction is conditional on both $\boldsymbol{s}_i$ and $\boldsymbol{p}_j$, since the flow for a deformed pose j of a given identity i will depend on the shape itself (e.g., the flow to the same semantic pose for a large person vs. a small person will look different). Formally, we have:

$$f_{\theta_p} : \mathbb{R}^3 \times \mathbb{R}^{D_\text{s}} \times \mathbb{R}^{D_\text{p}} \rightarrow \mathbb{R}^3,$$
$$(\boldsymbol{s}_i, \boldsymbol{p}_j, \boldsymbol{x}) \mapsto f_{\theta_p}(\boldsymbol{s}_i, \boldsymbol{p}_j, \boldsymbol{x}) = \Delta\tilde{\boldsymbol{x}}.$$

The pose MLP is trained on a set of P deformation fields from the identities' canonical pose to the arbitrary poses available for each train identity. Note that we do not require seeing every identity in the same pose, nor do we require an equal number of posed shapes for each identity.

For training, we sample N_p surface points $\{\boldsymbol{x}_i^k\}_{k=1}^{N_\text{p}}$ on the previously normalized canonical shapes (see Sec. 3.2) for each i-th identity in the dataset; we also store the barycentric weights for each sampled point. Each point is then randomly displaced a small distance $\boldsymbol{\delta n}$ along the normal direction of the corresponding triangle in the mesh. Then, for each j-th posed shape available for the identity, we compute corresponding points $\{\boldsymbol{x}_j^k\}_{k=1}^{N_\text{p}}$ in the posed shape by using the same barycentric weights and $\boldsymbol{\delta n}$ to sample the posed mesh. This approach gives us a deformation field (defined near the surface) between the canonical pose of a given identity i and a deformed pose j of the same identity. For further sampling details, we refer to the appendix.

We use the ground truth flow vector $\Delta\boldsymbol{x}_{ij}^k = \boldsymbol{x}_j^k - \boldsymbol{x}_i^k$ and define an ℓ_2-loss $\mathcal{L}_\text{p}$ on the flow prediction $\Delta\tilde{\boldsymbol{x}}_{ij}^k$. To learn the pose space, we minimize the following energy over all P deformation fields with respect to the individual pose codes $\{\boldsymbol{p}_j\}_{j=1}^{P}$ and pose MLP weights θ_p:

$$\underset{\theta_\text{p}, \{\boldsymbol{p}_j\}_{j=1}^{P}}{\arg\min} \sum_{\substack{j=1 \\ i=m[j]}}^{P} \left(\sum_{k=1}^{N_\text{p}} \mathcal{L}_\text{p}(f_{\theta_\text{p}}(\boldsymbol{s}_i, \boldsymbol{p}_j, \boldsymbol{x}_i^k), \Delta\boldsymbol{x}_{ij}^k) + \frac{\|\boldsymbol{p}_j\|_2^2}{\sigma_\text{p}^2} \right), \tag{4}$$

where $m[\cdot]$ is a dictionary mapping the index j of a posed shape to the corresponding index i of its canonical shape, and σ_p a regularization parameter for the pose codes. Note that we do not optimize over latent shape codes $\boldsymbol{s}_i$ when learning the pose space. However, we found that conditioning the pose MLP prediction on the latent shape codes was required to disentangle pose from shape.

Implementation details. Similar to the shape MLP, we use eight fully-connected, F_p-dimensional layers in our

pose MLP, with ReLUs, followed by a final layer regressing the 3-dimensional flow vector $\Delta\tilde{\boldsymbol{x}}$. In our experiments, we use $F_{\mathrm{p}} = 1024$ and $D_{\mathrm{p}} = 256$. We use the same training scheme as with the shape space training.

3.4. Inference-time Optimization

Once our latent representations of shape and pose have been constructed, we can leverage these spaces at test time by traversing them to solve for the latent codes that best explain an input sequence of L depth maps. We thus fit NPMs to the input data by solving for the unique latent shape code and the L per-frame latent pose codes that best explain the whole sequence of observations.

For each depth map in the input sequence, we project the depth values into a 256^3-SDF grid. We also compute a volumetric mask M_{o} for occluded regions that are further than 0.01 (in normalized units) from the input observed surface, i.e., we do not consider grid points $\boldsymbol{g}$ where $\mathrm{SDF}(\boldsymbol{g}) < -0.01$.

We then obtain initial estimates of the shape code and pose codes with the initialization process described in Sec. 3.4.1. Given the initial shape code, we can extract the canonical shape surface and then sample N_t surface points $\{\boldsymbol{x}_k\}_{k=1}^{N_t}$ ($N_t = 500\,000$ in our experiments) and add random displacements sampled from $\mathcal{N}(0,\, 0.015^2)$.

To fit an NPM to the monocular depth sequence, we minimize the following energy:

$$\tilde{\boldsymbol{s}}, \{\tilde{\boldsymbol{p}}_j\}_{j=1}^{L} = \underset{\boldsymbol{s}, \{\boldsymbol{p}_j\}_{j=1}^{L}}{\arg\min} \sum_{j=1}^{L} \sum_{\forall \boldsymbol{x}_k} \mathcal{L}_{\mathrm{r}} + \mathcal{L}_{\mathrm{c}} + \mathcal{L}_{\mathrm{t}} + \mathcal{L}_{\mathrm{icp}}. \quad (5)$$

We use the same clamped ℓ_1-loss from Eq. 3 to define the reconstruction loss $\mathcal{L}_{\mathrm{r}}$:

$$\mathcal{L}_{\mathrm{r}} = M_{\mathrm{o}}\, \mathcal{L}_{\mathrm{s}}\Big(f_{\theta_{\mathrm{s}}}(\boldsymbol{s}, \boldsymbol{x}_k), \big[\boldsymbol{x}_k + f_{\theta_{\mathrm{p}}}(\boldsymbol{s}, \boldsymbol{p}_j, \boldsymbol{x}_k)\big]_{\mathrm{sdf}}\Big), \quad (6)$$

where $[\cdot]_{\mathrm{sdf}}$ denotes trilinear interpolation of the SDF grid and M_{o} is the previously defined mask for occluded regions. Similar to train time, we enforce shape and pose code regularization:

$$\mathcal{L}_c = \frac{1}{\sigma_{\mathrm{s}}^2} \|\boldsymbol{s}\|_2^2 + \frac{1}{\sigma_{\mathrm{p}}^2} \|\boldsymbol{p}_j\|_2^2, \quad (7)$$

with $\sigma_{\mathrm{s}} = 10^{-1}$ and $\sigma_{\mathrm{p}} = 10^{-4}$. Additionally, we enforce temporal regularization between the current frame j and its neighboring frames $Q = \{j-1, j+1\}$. This is enforced with an ℓ_2-loss on the pose MLP flow predictions for points $\boldsymbol{x}_k$, and controlled with a weight of $\lambda_t = 200$:

$$\mathcal{L}_t = \lambda_t \sum_{q \in Q} \left\| f_{\theta_{\mathrm{p}}}(\boldsymbol{s}, \boldsymbol{p}_j, \boldsymbol{x}_k) - f_{\theta_{\mathrm{p}}}(\boldsymbol{s}, \boldsymbol{p}_q, \boldsymbol{x}_k) \right\|_2^2. \quad (8)$$

Finally, we employ an ICP-like loss $\mathcal{L}_{\mathrm{icp}}$ to further robustify the fitting (please refer to the appendix for details.) We use the Adam optimizer [21] and learning rates of 5×10^{-4} and 1×10^{-3} for the shape and pose codes, respectively.

Given the optimized shape code and L pose codes, to reconstruct the input sequence our approach only requires extracting the canonically-posed implicit surface via Marching Cubes [27] once (see Sec. 3.2). We then deform the reconstructed canonical mesh into every frame in the input sequence by querying the pose MLP $f_{\theta_{\mathrm{p}}}$ for every vertex in the canonical mesh.

3.4.1 Predicting Shape and Pose Initializations

To provide a good initialization for our latent code optimization, we train two 3D convolutional encoders $f_{\Omega_{\mathrm{s}}}$ and $f_{\Omega_{\mathrm{p}}}$ to predict initial estimates of the latent shape and pose codes, respectively. Both encoders take as input the back-projected depth observation in the form of a partial voxel grid. We then employ 3D convolutions and a final fully-connected layer to output a latent code estimate. To train these encoders, we make use of the latent shape and pose vectors learned from the train set, and use them as target codes for training the encoders. We found that this learned initialization provides robust initial code estimates, resulting in accurate reconstruction and tracking results. Additional architecture details can be found in the appendix.

4. Experiments

We evaluate our NPMs on both synthetic and real-world datasets on the task of model fitting to a monocular depth sequence observation (Sec. 4.1). We additionally demonstrate shape and pose transfer in Sec. 4.2, and show that our learned shape and pose spaces exhibit smooth, clear interpolation through the spaces in Sec 4.3.

Datasets. NPMs can be learned for any class of non-rigidly deformable objects. We perform a comprehensive comparison with state-of-the-art methods on clothed human datasets, and show the general applicability of our method by learning an NPM for hands. For clothed humans, we evaluate on the recent CAPE [29] dataset, which provides real-world scans of clothed humans and their corresponding SMPL+D registration. We also demonstrate our approach on synthetic human-like identities from the DeformingThings4D [25] dataset. For training, we use 45k arbitrarily posed shapes from 118 distinct identities: 33 from [25], 35 from [29] (13 in different clothing), and 50 from AMASS [30]. We test our human NPM on 4 identities from [29] and 4 from [25], on a total of over 1600 frames distributed across 8 sequences (4 from each dataset). We also learn a hand NPM from 40k posed shapes from 400 distinct identities from MANO [43], and test it on 500 frames, with 5 identities and sequences of 100 frames each.

Evaluation metrics. We measure both reconstruction and tracking performance. To quantitatively measure reconstruction quality we report two established metrics (following the evaluation protocol of [31]), which are computed on a per-frame basis. *Intersection over union* (IoU) measures the overlap between the predicted mesh and the groundtruth mesh. We randomly sample 10^6 points from the unit bounding box (where our normalized meshes reside) and determine if the points lie inside or outside the ground truth / predicted mesh. *Chamfer-*ℓ_2 (C-ℓ_2) offers a measure combining the accuracy and completeness of the reconstructed surface. Following [31], we use 100k randomly sampled surface points on the ground truth and predicted meshes. Additionally, we evaluate tracking with *End-Point Error* (EPE), measuring the average ℓ_2-distance between estimated keyframe-to-frame deformations and ground truth deformations as proposed in [6]; we sample 100k surface points and select a keyframe every 50 frames.

4.1. Model Fitting to Monocular Depth Sequences

Real human data. We show a comparison to state of the art on monocular depth data rendered from CAPE [29] real scans in Tab. 1, and qualitatively in Fig. 3. We compare with SMPL [26], a state-of-the-art traditional parametric model, as well as the state-of-the-art deep-learning-based approaches OFlow [35] and IP-Net [3]. We fit a SMPL model to the input depth maps by minimizing the reconstruction loss between surface points and the SDF grid extracted from the depth map (see Sec. 3.4), enforcing surface points to lie at the zero-level set of the SDF grid. To guide this SMPL fitting, we use OpenPose [9] to provide sparse keypoint correspondences, minimizing the reprojection error between projected SMPL joints and OpenPose predictions. To increase robustness, we also constrain the 3D error between SMPL joints in 3D and back-projected (using the input depth map) OpenPose predictions. Temporal regularization is applied by minimizing vertex-to-vertex distances between neighboring frames.

IP-Net is trained on the same combination of human data employed to learn our human NPM. Since OFlow was developed for continuous sequences of up to 17 frames (and we found performance to degrade noticeably for longer sequences), we prepare a train dataset of over 200k frames that fulfills this requirement; at test time, we evaluate the average of 17-frame subsequences covering the full test sequence. Our approach to learn shape and pose spaces –enabling latent code optimization for fitting– provides both effective shape and pose regularization over the manifolds, while capturing local details. This results in more accurate reconstruction and tracking performance.

Synthetic human data. We also evaluate on synthetic sequences from the DeformingThings4D [25] dataset, in comparison with SMPL [26] and OFlow [35] in Tab. 2. Our learned shape and pose spaces effectively capture significantly improved reconstruction and tracking in our model fitting experiments. Fig. 4 shows a qualitative comparison to state-of-the-art methods, demonstrating our global reconstruction and tracking along with the captured local detail.

Method	**IoU** ↑	**C-**ℓ_2 ($\times 10^{-3}$) ↓	**EPE** ($\times 10^{-2}$) ↓
OpenPose+SMPL	0.68	0.243	2.82
OFlow*	0.55	0.755	2.65
IP-Net	0.82	0.034	2.52
Ours (no Shape Enc)	0.83	0.023	0.77
Ours (no Pose Enc)	0.78	0.174	3.61
Ours (no S&P Enc)	0.77	0.185	3.65
Ours	**0.83**	**0.022**	**0.74**

Table 1: Comparison with state-of-the-art methods on real scans of CAPE [29]. *Since OFlow [35] works only on sequences of up to 17 frames, we report the average over sub-sequences of such length.

Method	**IoU** ↑	**C-**ℓ_2 ($\times 10^{-3}$) ↓	**EPE** ($\times 10^{-2}$) ↓
OpenPose+SMPL	0.64	0.251	2.04
OFlow*	0.40	2.688	7.52
Ours	**0.78**	**0.051**	**1.07**

Table 2: Comparison with state of the art on test sequences from DeformingThings4D [25]. *Since OFlow [35] works only on sequences of up to 17 frames, we report the average over sub-sequences of such length.

Method	**IoU** ↑	**C-**ℓ_2 ($\times 10^{-3}$) ↓	**EPE** ($\times 10^{-2}$) ↓
OFlow	0.74	0.105	1.12
Ours	**0.83**	**0.019**	**0.61**

Table 3: Comparison with OFlow [35] on D-FAUST [5].

4D point cloud completion on D-FAUST. We additionally compare with OFlow [35] on their 4D point cloud completion task on D-FAUST [5] in Tab. 3. We use a pretrained OFlow model provided by the authors and test on 20k dense point cloud trajectories sampled from the ground truth meshes. For our method, we only consider a monocular sequence of depth maps as input, resulting in more partial observations. Even with more partial data, our NPM fitting achieves significantly improved performance.

What is the effect of the encoder initialization? In Table 1, we evaluate the effect of our encoder initialization for our NPMs optimization. In place of the encoder predicted initialization, we use the average shape and pose latent codes from the train set. We measure the effect of not using the shape encoder (*no Shape Enc*), not using the pose

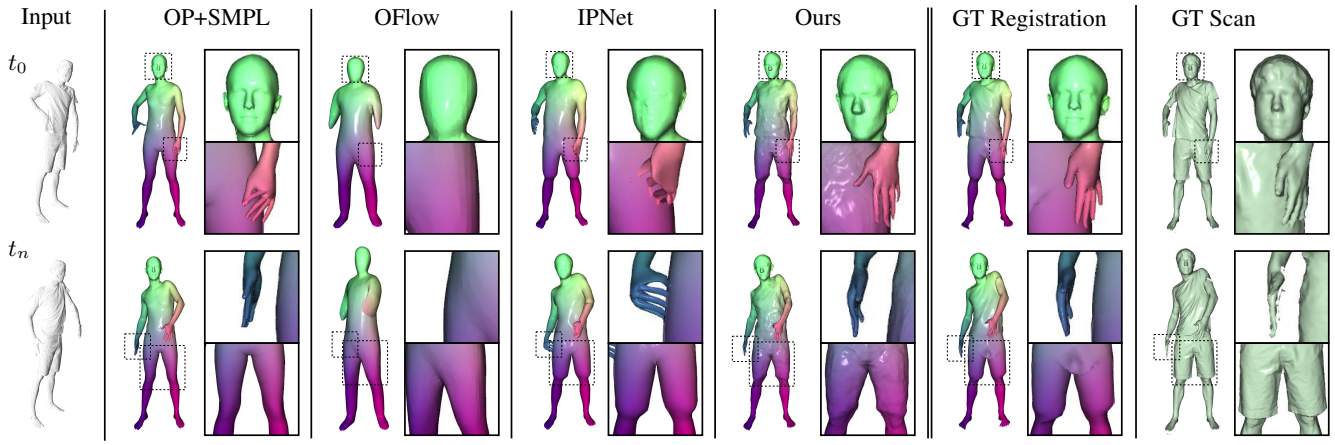

Figure 3: Comparison to state-of-the-art methods on the task of model fitting to a monocular depth sequence input (left column). From left to right, we compare with OpenPose [9] + SMPL [26], OFlow [35] and IP-Net [3]; our NPMs effectively capture local details present in the input views. The last two columns show the ground truth registration provided by CAPE [29] and the original scans, from which the input depth maps are rendered.

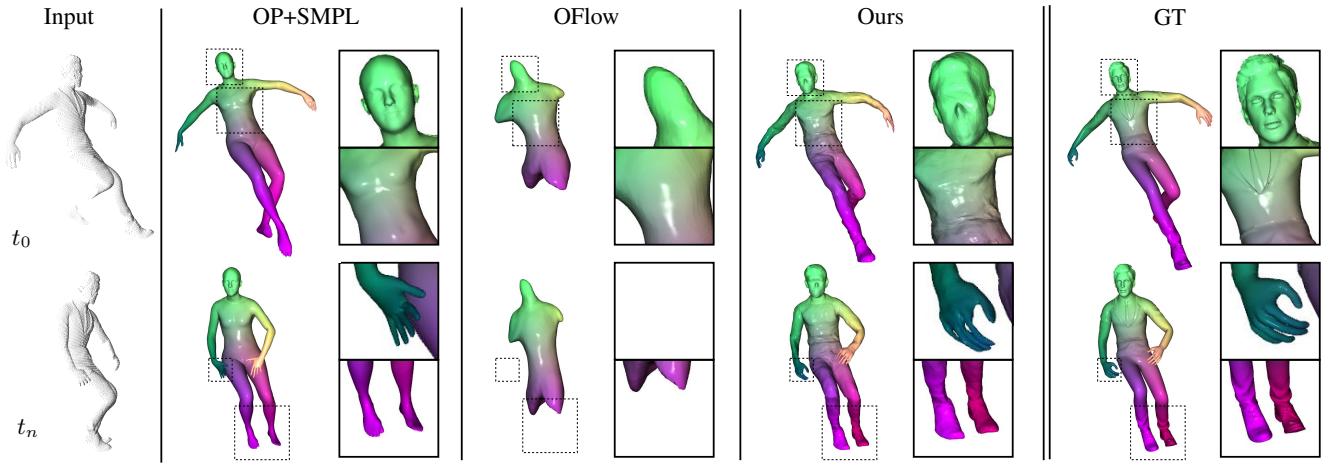

Figure 4: Comparison to state-of-the-art methods on the task of model fitting to a monocular depth sequence input (left column) from a synthetic dataset (DeformingThings4D [25]). From left to right, we compare with OpenPose [9] + SMPL [26] and OFlow [35]; our NPMs effectively capture local details present in the input views.

encoder (*no Pose Enc*), and not using neither the shape nor the pose encoder (*no S&P Enc*) for code initialization. The shape and pose code estimates provided by our encoders result in a closer initialization and an improvement in reconstruction and tracking performance.

Hand registration. NPMs can be constructed on various datasets with posed identities. We demonstrate its applicability to hand data generated with the MANO [43] parametric model. Fig. 5 shows our hand NPM fitting a test monocular depth sequence. We accurately capture both global structure and smaller-scale details (e.g., creasing from bent knuckles), achieving an IoU of 0.86, Chamfer-ℓ_2 of 1.39×10^{-5}, and EPE of 5.89×10^{-3}.

4.2. Shape and Pose Transfer

NPMs enable shape and pose transfer: we can transfer a given identity to a posed shape (shape transfer), and given a source identity in different poses, we can repose a target identity with the poses of the source identity (pose transfer). This is possible due to our disentangled shape and pose embedding spaces, which enables novel combinations of shape and pose latent codes. In Fig. 6 and the supplemental video, we show additional examples of shape and pose transfer.

4.3. Latent-Space Interpolation

Our latent spaces of shape and pose can be traversed to obtain novel shapes and poses. Interpolation through the

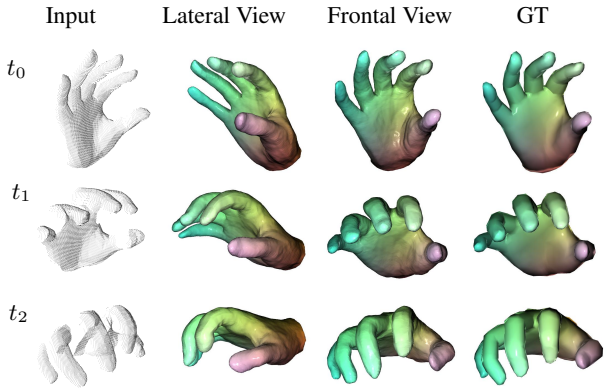

Figure 5: Registration of our hand NPM to a test sequence of monocular depth views generated using MANO [43].

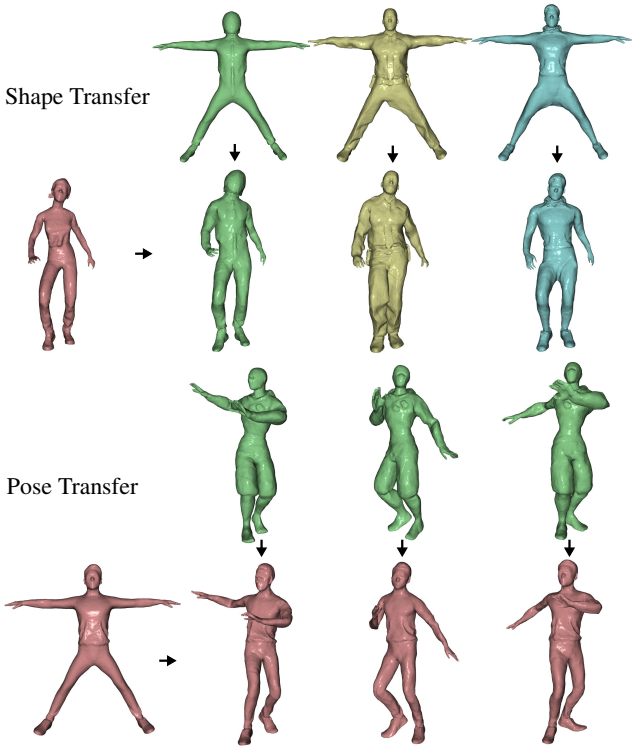

Figure 6: Shape and pose transfer with NPMs. We can transfer a given identity to a posed shape (shape transfer); given a source identity in different poses, we can repose a target identity with the poses of the source (pose transfer).

learned spaces (Fig. 7 and supplemental video) suggests continuity of our shape and pose latent spaces.

Limitations. While NPMs demonstrate potential for constructing and fitting learned parametric models, several limitations remain. For instance, our implicit representation of shape and pose deformation can struggle with very flat surfaces, as they comprise little volume and must have precisely defined inside / outside; incorporating semantic information into NPMs could help in this regard. Although NPMs can capture fine-scale details present in the input data (e.g., clothing boundaries), high-frequency details (e.g., the outline of a tie) remain challenging. Our learned spaces also do not consider the physics of deformation, which could encourage volume preservation and constrain deformations to physically correct movements.

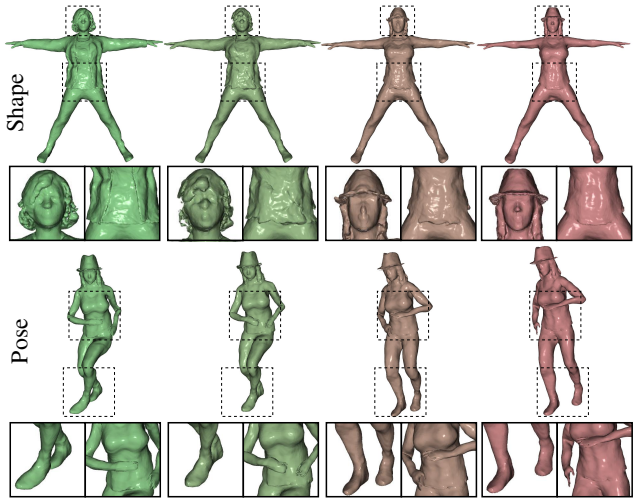

Figure 7: Shape and pose latent-space interpolation.

5. Conclusion

In this paper, we introduced Neural Parametric Models, enabling the construction of learned parametric models with disentangled shape and pose representations which can accurately represent 4D sequences of dynamic objects. Unlike traditional parametric models, our NPMs leverage learned implicit functions to expressively capture local details in shape and pose, and our test-time latent code optimization enables accurate fitting to observed details in input monocular depth sequences, outperforming parametric and learned 4D representations. Our learned NPMs also enable effective shape and pose transfer, and demonstrate smooth interpolations within the spaces across new shapes and poses. We additionally demonstrate more general applicability to a hands dataset, and believe this opens many promising avenues for various other domains in spatio-temporal modeling.

Acknowledgments

This project is funded by the Bavarian State Ministry of Science and the Arts coordinated by the Bavarian Research Institute for Digital Transformation (bidt), a TUM-IAS Rudolf Mößbauer Fellowship, the ERC Starting Grant Scan2CAD (804724), and the German Research Foundation (DFG) Grant Making Machine Learning on Static and Dynamic 3D Data Practical.

References

[1] Thiemo Alldieck, Marcus Magnor, Bharat Lal Bhatnagar, Christian Theobalt, and Gerard Pons-Moll. Learning to reconstruct people in clothing from a single rgb camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1175–1186, 2019. 2, 3

[2] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005. 1, 2

[3] Bharat Lal Bhatnagar, Cristian Sminchisescu, Christian Theobalt, and Gerard Pons-Moll. Combining implicit function learning and parametric models for 3d human reconstruction. *arXiv preprint arXiv:2007.11432*, 2020. 2, 3, 6, 7

[4] Bharat Lal Bhatnagar, Garvita Tiwari, Christian Theobalt, and Gerard Pons-Moll. Multi-garment net: Learning to dress 3d people from images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5420–5430, 2019. 2

[5] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J Black. Dynamic faust: Registering human bodies in motion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6233–6242, 2017. 6

[6] Aljaž Božič, Pablo Palafox, Michael Zollhöfer, Justus Thies, Angela Dai, and Matthias Nießner. Neural deformation graphs for globally-consistent non-rigid reconstruction. *arXiv preprint arXiv:2012.01451*, 2020. 2, 6

[7] Aljaž Božič, Pablo Palafox, Michael Zollöfer, Angela Dai, Justus Thies, and Matthias Nießner. Neural non-rigid tracking. In *NeurIPS*, 2020. 2

[8] Aljaz Bozic, Michael Zollhofer, Christian Theobalt, and Matthias Nießner. Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7002–7012, 2020. 2

[9] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):172–186, 2019. 6, 7

[10] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2

[11] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6970–6981, 2020. 2, 3

[12] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 1

[13] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph.*, 36(3):24:1–24:18, 2017. 2

[14] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6545–6554, 2017. 2

[15] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5868–5877, 2017. 1

[16] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 1

[17] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7154–7164, 2019. 2

[18] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. 2

[19] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard A. Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew J. Davison, and Andrew W. Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, Santa Barbara, CA, USA, October 16-19, 2011*, pages 559–568, 2011. 2

[20] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8320–8329, 2018. 1, 2

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4, 5

[22] Verica Lazova, Eldar Insafutdinov, and Gerard Pons-Moll. 360-degree textures of people in clothing from a single image. In *2019 International Conference on 3D Vision (3DV)*, pages 643–653. IEEE, 2019. 3

[23] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6):194–1, 2017. 1, 2

[24] Yang Li, Aljaz Bozic, Tianwei Zhang, Yanli Ji, Tatsuya Harada, and Matthias Nießner. Learning to optimize non-rigid tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4910–4918, 2020. 2

[25] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. 3, 5, 6, 7

[26] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 1, 2, 6, 7

[27] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 3, 5

[28] Qianli Ma, Jinlong Yang, Anurag Ranjan, Sergi Pujades, Gerard Pons-Moll, Siyu Tang, and Michael J Black. Learning to dress 3d people in generative clothing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6469–6478, 2020. 2

[29] Qianli Ma, Jinlong Yang, Anurag Ranjan, Sergi Pujades, Gerard Pons-Moll, Siyu Tang, and Michael J. Black. Learning to dress 3d people in generative clothing. In *Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3, 5, 6, 7

[30] 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. 3, 5

[31] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 1, 2, 3, 6

[32] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Deep level sets: Implicit surface representations for 3d shape inference. *arXiv preprint arXiv:1901.06802*, 2019. 2

[33] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 343–352, 2015. 2

[34] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2011, Basel, Switzerland, October 26-29, 2011*, pages 127–136, 2011. 2

[35] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5379–5389, 2019. 2, 3, 6, 7

[36] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013. 2

[37] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 1, 2, 3, 4

[38] Chaitanya Patel, Zhouyingcheng Liao, and Gerard Pons-Moll. The virtual tailor: Predicting clothing in 3d as a function of human pose, shape and garment style. *arXiv preprint arXiv:2003.04583*, 2020. 2

[39] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *2009 sixth IEEE international conference on advanced video and signal based surveillance*, pages 296–301. Ieee, 2009. 1, 2

[40] Stylianos Ploumpis, Haoyang Wang, Nick Pears, William AP Smith, and Stefanos Zafeiriou. Combining 3d morphable models: A large scale face-and-head model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10934–10943, 2019. 1, 2

[41] Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J Black. Clothcap: Seamless 4d clothing capture and retargeting. *ACM Transactions on Graphics (TOG)*, 36(4):1–15, 2017. 2

[42] Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J Black. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics (TOG)*, 34(4):1–14, 2015. 2

[43] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), Nov. 2017. 1, 2, 3, 5, 7, 8

[44] Miroslava Slavcheva, Maximilian Baust, Daniel Cremers, and Slobodan Ilic. Killingfusion: Non-rigid 3d reconstruction without correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1386–1395, 2017. 2

[45] Garvita Tiwari, Bharat Lal Bhatnagar, Tony Tung, and Gerard Pons-Moll. Sizer: A dataset and model for parsing 3d clothing and learning size sensitive 3d clothing. *arXiv preprint arXiv:2007.11610*, 2020. 2

[46] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 1

[47] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1912–1920, 2015. 2

[48] Hongyi Xu, Eduard Gabriel Bazavan, Andrei Zanfir, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. Ghum & ghuml: Generative 3d human shape and articulated pose models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6184–6193, 2020. 2

[49] Keyang Zhou, Bharat Lal Bhatnagar, and Gerard Pons-Moll. Unsupervised shape and pose disentanglement for 3d

meshes. In *European Conference on Computer Vision*, pages 341–357. Springer, 2020. 2

[50] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1, 2