LoRD: Local 4D Implicit Representation for High-Fidelity Dynamic Human Modeling

Boyan Jiang[1] Xinlin Ren[1] Mingsong Dou[2] Xiangyang Xue[1†]
Yanwei Fu[1†] Yinda Zhang[2†]

[1]Fudan University [2]Google

Abstract. Recent progress in 4D implicit representation focuses on globally controlling the shape and motion with low dimensional latent vectors, which is prone to missing surface details and accumulating tracking error. While many deep local representations have shown promising results for 3D shape modeling, their 4D counterpart does not exist yet. In this paper, we fill this blank by proposing a novel **Lo**cal 4D implicit **R**epresentation for **D**ynamic clothed human, named **LoRD**, which has the merits of both 4D human modeling and local representation, and enables high-fidelity reconstruction with detailed surface deformations, such as clothing wrinkles. Particularly, our key insight is to encourage the network to learn the latent codes of local part-level representation, capable of explaining the local geometry and temporal deformations. To make the inference at test-time, we first estimate the inner body skeleton motion to track local parts at each time step, and then optimize the latent codes for each part via auto-decoding based on different types of observed data. Extensive experiments demonstrate that the proposed method has strong capability for representing 4D human, and outperforms state-of-the-art methods on practical applications, including 4D reconstruction from sparse points, non-rigid depth fusion, both qualitatively and quantitatively. Please check out the project page for video and code: https://boyanjiang.github.io/LoRD/.

1 Introduction

Dynamic 3D human modeling has been a long-standing challenge to 3D vision and graphics communities, as it is critical to various applications, such as VR/AR, animation and robot simulation. Traditional methods leverage well-designed parametric model [2] and physics-based simulation [61,67,22,20] to model the inner human body and deformable outer cloth separately, but they typically demand huge engineering efforts and expensive computational cost. Recently, many learning based methods have been proposed [36,25,44,32,35,11,4,60]; unfortunately, some of these methods can not model fine-grained geometry details beyond inner body, while the others only support frame-wise reconstruction to produce dynamic sequence.

Boyan Jiang, Xinlin Ren and Xiangyang Xue are with School of Computer Science, Fudan University. Yanwei Fu is with School of Data Science, Fudan University.
† Corresponding authors. E-mail: Yanwei Fu (yanweifu@fudan.edu.cn).

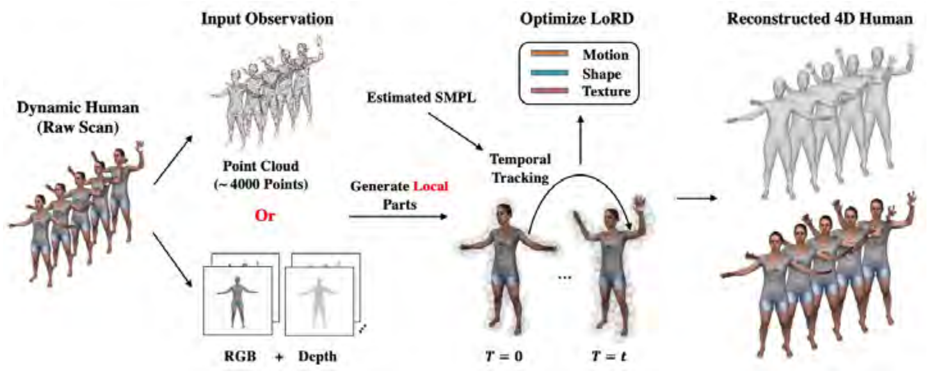


Fig. 1. LoRD represents dynamic human with a set of overlapping local parts. Each part is temporally tracked with the estimated SMPL meshes, and contains low-dimensional latent codes of motion, canonical shape and texture (optional), which can be decoded to recover the detailed temporal changing of local surface patches by a 4D implicit network. During the test-time, these latent codes are optimized based on the different types of input observations, such as sparse point clouds and monocular RGB-D video to produce high-fidelity 4D human reconstruction.

The key challenge of dynamic human modeling is to find a way to model 4D representations for both surface geometry and temporal motion. Typically, existing 4D human representation methods infer the single *holistic* latent code/vector to control global motion and shape, which unfortunately are prone to over-smoothing shapes and missing fine-grained surface details. Recent efforts are made on inferring local representations for 3D modeling [19,15,30,6,54]. Typically, these methods utilize a set of local parts to model the geometry of local surface regions for reconstructing complete 3D shapes. Such local formulation improves the model capacity in recovering the detailed geometry with a stronger generalization ability than global free-form modeling [49,45,52]. However, it is nontrivial to directly enable these local methods to support the 4D scenario of modeling a dynamic 3D human with temporal motions, as their naïve extension to do per-frame reconstruction can not maintain the desirable properties of 4D modeling, such as temporal inter-/extrapolation, 4D spatial completion.

To this end, this paper proposes a **Lo**cal 4D implicit **R**epresentation for **D**ynamic human, named **LoRD**, which combines the merits of 4D human modeling and local representation. The LoRD is capable to produce high-fidelity human mesh sequence. Given a dynamic clothed human sequence over a time span $T \in [0, 1]$, we decouple its temporal evolution into two factors: inner body skeleton motion and outer surface deformation. We handle the skeleton motion with the widely-used SMPL parametric model [40], which uses a shape parameter and a series of pose parameters to represent the temporal changing of inner body. On the other hand, for outer surface deformation, we resort to a local implicit framework. Specifically, we sample a bunch of local parts on the inner body mesh of the canonical frame ($T = 0$), each part is represented by a 3D sphere with the intrinsic parameters (not camera intrinsics) of radius and transformation with respect to the world coordinate frame, and latent codes encoding local deformation and canonical shape information. Since SMPL models have the unified mesh topology, we can find the correspondence in subsequent frames and temporally align the local coordinate systems for each part. Then we use a 4D local implicit network to model the surface deformation within each part conditioned on their latent codes. Such representation utilizes inner body model to handle the global skeleton motion, and leaves the detailed surface dynamics to the powerful local implicit network. This facilitates the dynamic human modeling with high-quality geometry.

Technically, our local representation is learned on 100 human sequences with ground truth mesh and its corresponding inner body mesh, each sequence contains $L = 17$ frames. For each training sequence, we first sample the local parts on the surface of inner body mesh and randomly initialize the latent codes. Then we use objective function introduced by IGR [23] to optimize the local implicit network and latent codes. During the test-time, we fix the local implicit network to support a particular application (e.g., 4D reconstruction from sparse points, non-rigid depth fusion) via the auto-decoding method [52]. To obtain the inner body mesh, we use the existing work H4D [29] to provide plausible body estimation. Moreover, our representation can combine with the H4D motion model to conduct body reference optimization introduced by PaMIR [80], and support inner body refining to handle the imperfect body estimation (detailed in Sec. 3.4). This improves the robustness of LoRD against inaccurate inner body tracking.

To summarize, the main contributions of our work are: 1) We propose a novel local 4D implicit representation, which divides surface of a dynamic human into a collection of local parts and supports high-fidelity dynamic human modeling; 2) To temporally align each part for training and test-time optimization, we leverage inner SMPL body mesh for local part tracking; 3) We design an inner body refining strategy based on our local representation to optimize imperfect initial body estimation; 4) Our representation only requires a small set of data for training, and outperforms the state-of-the-art methods on practical applications, e.g. 4D reconstruction from sparse points, non-rigid depth fusion.

2 Related Work

4D representation Deep learning methods have shown impressive results on 3D-related tasks based on various representations, such as voxels [12,21,71], point clouds [57,18,56,1], meshes [24,31,69,37,7] and neural implicit surfaces [45,52,9,30,10,17,6,19]. While great success has achieved for static 3D object, recent works [49,58,28] attempt to investigate elegant 4D representation of modeling dynamic 3D object with an additional temporal dimension. When targeting the dynamic human, recent methods [49,28] always suffer from missing surface details and inaccurate motion due to the global shape modeling and lack of human motion prior. In contrast, the proposed local 4D representation leverages inner body tracking to handle the global skeleton motion and leaves the detailed dynamics to a set of local parts, which is effective to recover high-fidelity surface deformation, and generalize well to the novel sequences.

Local shape representation The implicit representations conditioned on a global latent vector [52,45] often produce over-smooth results and have failed to recover detailed geometry such as human hands and clothing wrinkles. To tackle this problem, some recent works utilize local implicit representation for shape modeling [19,14,30,54] and neural rendering [53,39], but none of them has used it to build 4D representation that represents how 3D geometry deforms *continuously* over time. Similar to us, there is a family of work [68,8] building human avatar which supports shape generation under arbitrary body poses. However, they process different timestamps independently and do not explicitly estimate temporal correspondences, which are shown to be important for recovering geometry details from multiple input frames or applications like motion completion/prediction. In contrast, our method extends the local representation to 4D scenario by combining the human prior model and 4D implicit network, which can directly produce 4D results with one-shot optimization process.

Dynamic human modeling When it comes to capturing the dynamic human, some methods [75,26,27] require a pre-scanned template as a good initialization to obtain results from monocular color information. Recent methods [47,77,79,62] utilize depth sensors to achieve real-time speed based on the classical deformation graph [63] and volumetric fusion [48], which get rid of subject-specific template. Since these methods are conducted in a frame-by-frame manner without intermediate motion representation, they are prone to error accumulation and hard to recover from tracking failures. Most recently, NDG [5] learns a globally-consistent deformation graph to facilitate non-rigid reconstruction, but requires per-sequence retraining and relies on multi-view depth sensors, which is inconvenient in the actual usage. As a popular line of works, NeRF-based [46] human modeling methods [53,55] typically do not satisfy both local and temporal modeling. Most similar to us, Zheng et al. [78] propose a structured temporal NeRF for dynamic human rendering. We note that these methods mainly focus on rendering quality but usually produce unsatisfactory geometry. In contrast, LoRD models motion and shape jointly with local representation, so that information from two domains can be exchanged through the 4D model and benefit each other, which produces high-fidelity geometry results.

3 Method

Our framework is overviewed in Fig. 2: given a 3D clothed human mesh sequence of length $L = 17$ frames that performs some motions in a normalized time span $[0, 1]$, we first define a set of local parts (Sec. 3.1) around inner body surface of the canonical frame ($T = 0$ in our setup). Then we temporally track these parts which are controlled by the skeleton motion of the inner body model (SMPL). Note that we use the ground truth SMPL mesh during training, whereas the SMPL parameters are estimated with the off-the-shelf method [29] at test-time. Each part contains a motion code c_m, a shape code c_s and a texture code c_t

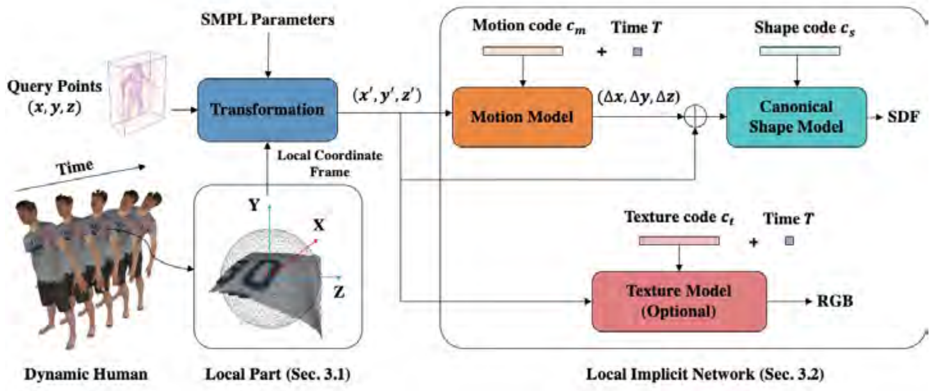


Fig. 2. Overview of our framework. We use a set of spherical parts to model the local surface deformation of dynamic human. Given a 3D point (x, y, z) under the world coordinate frame, we determine which part it falls into and transform it into the local coordinate frame, i.e. (x', y', z'), according to the estimated SMPL parameters. The transformed point is queried into a local implicit network, which is conditioned on the latent codes of local part, to obtain signed distance and RGB (optional) value. Note that our local implicit network is shared by all parts. Meshes are extracted with Marching Cubes [41].

(optional), which can be decoded by our local implicit network (Sec. 3.2) to obtain the reconstructed surface. Overall, we utilize the inner body model to track global skeleton motion and leave the detailed temporal deformation, geometry and texture of the local surface patch to the local implicit network. Training and test-time optimization are discussed in Sec. 3.3 and Sec. 3.4, respectively.

3.1 Local Part Formulation

Inner body model There are many ways to track the global skeleton motion of a dynamic human, e.g. optical/scene flow [66,38], dense human correspondence [73,65], and deformation graph [63]. In our formulation, we choose the widely-used SMPL model [40] as it naturally provides surface correspondence between frames and its low-dimensional representations are easily to be optimized.

LoRD represents a 4D human with a set of local parts (defined as 3D spheres) $\mathcal{P} = \{\mathcal{P}_k\}_{k=1}^{K}$, where $\mathcal{P}_k = \{\mathbf{r}, \mathbf{R}_k, \mathbf{c}_k\}$ is the intrinsic parameters of part k (do not confuse them with camera intrinsics); $\mathbf{r} \in \mathbb{R}$ is the radius of the sphere shared by all parts (we use $r = 5cm$ in our experiments); $\mathbf{R}_k \in \mathbb{R}^9$ and $\mathbf{c}_k \in \mathbb{R}^3$ are the rotation matrix relative to the world coordinate frame and the center of sphere for each part respectively. Given the inner body mesh of the canonical frame, a sampling algorithm (detailed in Supp.) is conducted on its surface to obtain the part centers. Inspired by [30,6], to make the result smooth over the parts border, we use the overlapping strategy during the part sampling process, where each part overlaps with its neighboring parts by maximum 1.5× the part radius

$\mathbf{r}$, and finally produce 2127 parts. The transformation of each part is based on the local coordinate frame as shown in Fig. 2. Details are in Supp. Mat.

3.2 Local Implicit Network

Besides the intrinsic parameters, each local part also has the latent parameters as low-dimensional codes c_m, c_s and c_t, which encode respectively the information of the local surface deformation, canonical geometry and texture. The goal of the local parts is to represent the detailed temporal deformation and geometry of the local surface patches. To this end, we follow D-NeRF [55] and use a 4D implicit network, which consists of a motion model and a canonical shape model. Moreover, if the observed data contain texture information, the additional texture model would be triggered to predict colors for the vertices of reconstructed mesh. Note that the implicit network is shared by all local parts. Next, We briefly introduce each model and the detailed architecture can be found in Supp. Mat.
Motion model As shown in Fig. 2, we formulate the motion model $f^m(\mathbf{x}, T \mid c_m)$ as a 4D function conditioned by the motion code $c_m \in \mathbb{R}^{128}$, which takes a 3D point $\mathbf{x} = (x, y, z)$ in the local coordinate frame and a time value T (normalized to $[0, 1]$) as input, and predicts a deformation vector $\Delta\mathbf{x}$ that transforms this point to the canonical frame, i.e. $T = 0$, by $\mathbf{x}^* = \mathbf{x} + \Delta\mathbf{x}$. We adopt the network architecture of IM-Net [9], and reduce the feature dimension of each hidden layer by 4 fold [30] to obtain an efficient motion model.
Canonical shape model The canonical shape model $f^s(\mathbf{x} \mid c_s)$ is a neural signed distance function, which only holds a static implicit geometry of the canonical frame as the temporal deformation is handled by the motion model. Specifically, given a 3D query point at time T, we first obtain its position in the space of the canonical frame with the motion model, and then use the canonical shape model that is conditioned on a canonical shape latent code $c_s \in \mathbb{R}^{128}$ to predict the signed distance of the given point towards the surface. The same network architecture as DeepSDF [52] is adopted for canonical shape model. For training and testing efficiency, we reduce the number of layers and the feature channels for each layer to 6 and 256 respectively. During inference, we compute the bounding box of human based on the inner body mesh for each frame, and utilize the Marching Cubes algorithm [41] to extract the iso-surface.
Texture model If the input data contains texture information, e.g. colored point clouds, our representation can be extended to support surface texture inference. We achieve this by learning a function $f^t(\mathbf{x}, T \mid c_t)$ to predict the 4D texture field [50,59,60] of the dynamic local surface conditioned on a texture code $c_t \in \mathbb{R}^{128}$. It takes a 3D point $\mathbf{x}$ in the local coordinate frame and a time value T, and outputs the RGB value of this point. We use the architecture of TextureField [50] decoder for our texture model. Please refer Supp. Mat. for the detailed network architecture. Note that we use our texture model in a per-sequence fashion during the test-time without pre-training, i.e. fit the input sequence with updating the network parameters, for better visualization results.

3.3 Training

Thank to our local formulation, the training of our model is very data efficient. We only use 100 sequences of length $L = 17$ frames from CAPE dataset [42] to learn our representation. During training, we adopt the auto-decoding method [52] and optimize our motion model, canonical shape model, and the latent codes for training parts. Specifically, given a training sequence that contains ground truth clothed meshes and the corresponding inner body meshes, we first sample a bunch of local parts on the surface of the inner body mesh of the first frame. Since the SMPL mesh has the unified surface topology, we can obtain the rotations and locations of each part in the following time steps, thus align their local coordinate frames. Next, we initialize the motion code and canonical shape code for each part with the vectors randomly sampled from $N(0, 0.01)$, these codes are optimized with the network parameters during training. To train our implicit networks, the query points are sampled from three sources, i.e. surface, near surface space and free space in the bounding box.
Loss functions The point sets sampled on-surface and off-surface are denoted as $\mathcal{X}$ and $\bar{\mathcal{X}}$ respectively. We optimize our 4D implicit function $f(\cdot)$ base on the loss functions introduced by IGR [23]:

$$\mathcal{L}_{\mathrm{s}} = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}) + \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) - \boldsymbol{n}(\boldsymbol{x})\|), \quad \mathcal{L}_{\mathrm{e}} = \frac{1}{|\bar{\mathcal{X}}|} \sum_{\boldsymbol{x} \in \bar{\mathcal{X}}} (\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\| - 1)^2$$

where $\mathcal{L}_{\mathrm{s}}$ ensures the zero signed distance values for on-surface points and their normals aligned with the ground truth. $\mathcal{L}_{\mathrm{e}}$ is the regularization term encouraging the learned function to satisfy the Eikonal equation [13]. In addition, we also add a latent regularization term $\mathcal{L}_{\mathrm{c}} = \|c_m\|_2 + \|c_s\|_2$ to constrain the learning of latent spaces. The final objective function for training is $\mathcal{L} = \lambda_1 \mathcal{L}_{\mathrm{s}} + \lambda_2 \mathcal{L}_{\mathrm{e}} + \lambda_3 \mathcal{L}_{\mathrm{c}}$. We use $\lambda_1 = 1.0$, $\lambda_2 = 1e^{-1}$, $\lambda_3 = 1e^{-3}$ in our experiment.
Evaluate SDF for query points During the training process, the sampled points are only evaluated by the local parts that cover them. In our case, "point $\mathbf{x}$ is covered by part k" means the Euclidean distance between $\mathbf{x}$ and the center of part c_k is less than or equal to the pre-defined part radius $\mathbf{r}$, i.e. $d(\mathbf{x}, c_k) \leq \mathbf{r}$. The sampled parts are highly overlapping, thus for one query point, we randomly choose n parts that covered this point to evaluate its SDF, and then average n SDF values ($n = 4$ in our experiments) as the final output. This could encourage the network to produce the smooth results in the overlapping regions. If some points are not covered by any parts, e.g. points sampled in the free space far from surface, then it will choose n-nearest parts to obtain the SDF prediction. Note that this is important for reconstructing complete results, since we cannot ensure the local parts sampled from inner body mesh would completely cover the surface of the clothed human.

3.4 Test-Time Optimization

After learning our local representation, we can then conduct the test-time optimization to reconstruct the dynamic human based on the given observations. In our experiments, we mainly focus on recovering 4D humans from complete point clouds or partial depth sequences. Generally speaking, the test-time optimization is similar to the training process, which performs backward optimization with the auto-decoding fashion, except that we fix the network parameters and only update the latent codes for each local part. Since we leverage the loss functions from IGR [23], and directly perform optimization based on the point clouds with local-based representation, the geometry covered by each part is a non-watertight surface, which causes the extracted surface contains artificial interior back-faces. We borrow the post-processing algorithm from LIG [30] to remove such artifacts. The details about the post-processing algorithm and the choices of hyper-parameters can be found in Supp. Mat. In addition, there are some technical details that we want to clarify below.

Inner body estimation Given a testing sequence, we first need to estimate inner body meshes to sample local parts. As the temporal consistency could facilitate our reconstruction, we use the recent motion based human body estimation method H4D [29] to fit the SMPL parameters via backward optimization.

Inner body refining The fitting results of H4D [29] are accurate enough in most cases, but still imperfect on some sequences, which may cause the observations of some local parts vary too much over time. Inspired by PaMIR [80], we propose a strategy to refine the initial inner body fitting from H4D. Specifically, we first sample and track the local parts on the initial body mesh sequence produced by H4D, and optimize the latent codes for each part. Then we fix the latent codes and local parts, query the SMPL vertices into our local implicit network, and optimize the SMPL parameters for shape and initial pose, and latent vector for motion of H4D. We follow the body reference optimization proposed in PaMIR to build the loss functions of our refining process:

$$\mathcal{L}_{\text{SMPL}} = \begin{cases} |f(x)| & f(x) \geq 0 \\ \frac{1}{\eta}|f(x)| & f(x) < 0 \end{cases}, \quad \mathcal{L}_{reg} = \left\| V - V^{init} \right\|_2,$$

where $\eta = 5$, $f(\cdot)$ is our local implicit signed distance function; $V = (\beta, \theta_0, c_m)$ contains the shape parameter, initial pose parameter and latent motion code of H4D, and the superscript "init" means initial estimations. This reflects the fact that, if the body estimation is accurate, then the vertices of the body mesh will get the negative SDF predictions (inside surface). Moreover, we also use an additional observation loss $\mathcal{L}_{\text{obs}}$, which denotes Chamfer loss for the complete point cloud and point-to-surface loss for partial point cloud from the depth image. The final objective function is $\mathcal{L} = \lambda_1 \mathcal{L}_{\text{SMPL}} + \lambda_2 \mathcal{L}_{\text{obs}} + \lambda_3 \mathcal{L}_{reg}$, where $\lambda_1 = 1.0$, $\lambda_2 = 1e^2$ and $\lambda_3 = 1e^{-3}$ in our experiments. We verify the effectiveness of our inner body refining strategy in Sec. 4.4.

Texture model optimization As mentioned in Sec. 3.2, we optimize the texture model for each testing sequence. Given a colored point cloud sequence, we can obtain the ground truth color $C_T(\mathbf{x})$ of a surface point $\mathbf{x}$ in time T. Then we query $\mathbf{x}$ into the texture model conditioned on the texture code c_t^k of part k to get the color prediction. We also use the average of n predicted colors as the final output (Sec. 3.3). To optimize the network parameters and texture codes, we add the L_1-loss $\mathcal{L}_{\text{color}} = |f^c(\mathbf{x}, T \mid c_t) - C_T(\mathbf{x})|$ into the objective function.

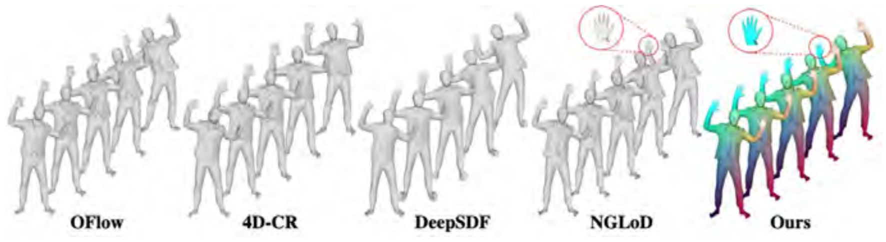


Fig. 3. 4D human fitting. We choose SoTA implicit 3D/4D representations to overfit a given mesh sequence and compare the results with us. The colors on our results indicate the correspondences across different frames, which cannot be obtained by the framewise baselines, i.e. NGLoD, DeepSDF. The zoomed-in part shows we reconstruct better finger details than NGLoD.

4 Experiments

In this section, we evaluate the representation capability of LoRD and its value in practical applications, i.e. 4D reconstruction and non-rigid depth fusion.

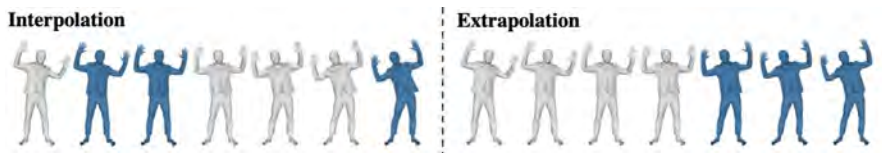


Fig. 4. Temporal inter-/extrapolation. Colored meshes are inter-/extrapolated frames.

Dataset and metric For training and evaluation, we use the CAPE [42] dataset which contains more than 600 motion sequences of 15 persons wearing different types of outfits, and the SMPL registrations are provided. Additionally, some raw scanned sequences with texture information are also available. We choose 100 sub-sequences of length $L = 17$ for training, and use the sub-sequences of novel subjects for testing. To compare with the baseline methods, we use Chamfer Distance-$L2$[45], normal consistency [60] (the average $L2$ distance between the normal of given point on the source mesh and the normal of its nearest neighbor on the target mesh), and F-Score [69] as evaluation metrics.
Implementation details We use PyTorch with Adam optimizer [34] of learning rate $1e^{-3}$ and batch size 1 for both training and test-time optimization. The experiments are conducted on a single Nvidia 2080Ti GPU. The test-time optimization takes around $15min$ for each 17 frames sequence.

4.1 Representation Capability

4D human fitting We first evaluate the efficacy of LoRD in representing dynamic human by overfitting a given mesh sequence. We select one sequence from

Table 1. Comparisons on 4D human fitting. Left: framewise methods, Right: temporal methods. "Ch.-L_2" and "Normal" mean Chamfer Distance ($\times 10^{-4} m^2$) and Surface Normal Consistency respectively. The threshold for computing F-Score is $\tau = 5mm$.

Framewise	Ch.-L_2 ↓	Normal ↓	F-Score ↑
DeepSDF [52]	0.846	0.291	0.669
NGLoD [64]	**0.074**	0.135	**0.969**

Temporal	Ch.-L_2 ↓	Normal ↓	F-Score ↑
OFlow [49]	0.317	0.312	0.675
4D-CR [28]	5.249	0.359	0.425
Ours	0.075	**0.131**	**0.969**

the CAPE dataset for this task. For comparison, we choose 3D neural SDF methods DeepSDF [52] and NGLoD [64], DeepSDF is a global representation which represents the complete shape with a single latent code, while NGLoD is a SoTA local neural SDF representation based on the Octree, both of them are 3D representations that need to work with frame-wise manner to produce a temporal sequence. In addition, we choose the SoTA 4D representation methods OFlow [49] and 4D-CR [28] as our baseline.

The quantitative results are shown in Tab. 1. Our LoRD representation clearly outperforms DeepSDF and all the SoTA 4D representation methods, and performs comparable with framewise method NGLoD. We show the visual results in Fig. 3, the colors of our results indicate the dense correspondences w.r.t the first frame. Specifically, for each vertex on the reconstructed mesh of time T, we use the optimized motion codes to transform it to the first frame, and obtain color value of the nearest vertex. We note that this cannot be achieved by DeepSDF or NGLoD, since they do not model temporal information.

Temporal inter-/extrapolation To further show the superiority of LoRD over the framewise representations, we show the temporal inter-/extrapolation results achieved by our method in Fig. 4. Given a sequence of length $L = 17$ frames, for interpolation, we randomly choose 9 frames as the observations to perform SDF fitting, the goal is to complete the missing frames to obtain a temporally complete sequence. And for extrapolation, we only use the first 9 frames and need predict the future motion of the last 8 frames. Fig. 4 shows that LoRD produces the plausible results on both inter- or extra-polation modes. Again, these temporal completion tasks also cannot be achieved by the framewise 3D representations, e.g. DeepSDF, NGLoD. We also provide the results about interpolation of the latent codes in Supp. Mat. (Sec. 2.2) as a sanity check.

4.2 4D Reconstruction from Sparse Points

We then show that LoRD can support various applications. First, we demonstrate that LoRD can achieve high quality 4D reconstruction from sparse point clouds. In this case, we assume the point normal directions are available (oriented point cloud, the same for Poisson Reconstruction [33] and LIG [30]).

Compare to instance-level methods We first compare LoRD with the instance level methods, the "instance-level" in here means we only overfit one sequence at a time and do not consider generalization to other instances. We choose the traditional Poisson Surface Reconstruction with octree depth value

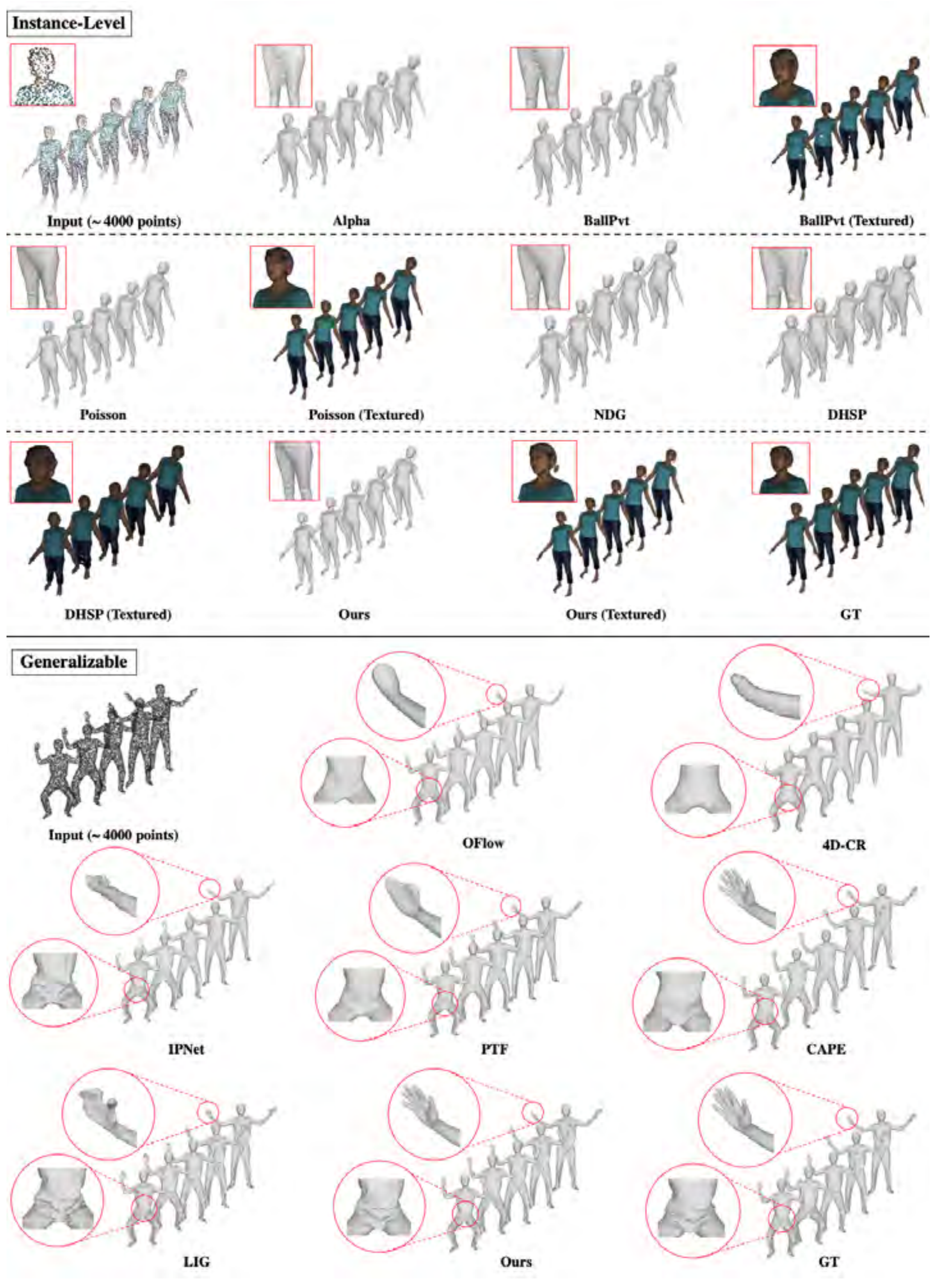


Fig. 5. 4D reconstruction from sparse points. Each input point cloud contains around 4000 points. Note the detailed geometry in the zoomed-in parts and the surface deformation recovered by our method. We provide more qualitative results in Supp. Mat.

$d = 10$ (PSR10) [33], Alpha Shape [16] and Ball Pivoting [3] as the baseline. Moreover, we also compare with the SoTA network-based surface reconstruction method Deep Hybrid Self-Prior (DHSP), and the non-rigid reconstruction method Neural Deformation Graph (NDG). The quantitative results are show in Fig. 6 (a, I), the leftmost column represents the sampled point cloud density (number of points per square meter of surface), the smaller number corresponds to the sparser point cloud, the surface area of SMPL mesh used for point sampling is around $2m^2$. As can be seen, our method outperforms all the baselines by a large margin. More importantly, the sparser point cloud hardly affects our performance while the baseline methods have been significantly affected, this is because LoRD is a 4D representation, sparse observation from each frame can compensate each other through the motion model. The qualitative comparisons are shown in Fig. 5 (above the solid line), our method can recover geometry details on the face and cloth with high resolution texture, while the baselines only produce over-smooth results due to the limited information from sparse inputs.

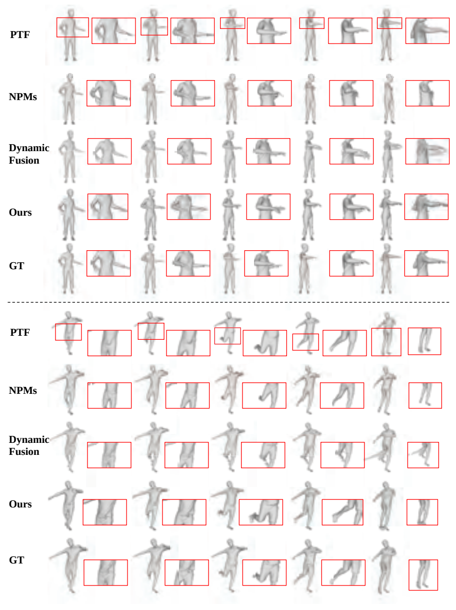
I. Comparisons to instance-level methods

P./m^2	Method	Ch.-L_2 ↓	Normal ↓	F-Score ↑
500	Alpha[16]	1.665	1.205	0.422
	BallPvt[3]	0.740	0.433	0.590
	PSR10 [33]	0.664	0.310	0.714
	DHSP [74]	1.383	0.864	0.520
	NDG [5]	0.706	0.2901	0.712
	Ours	**0.105**	**0.176**	**0.938**
1000	Alpha [16]	0.966	1.191	0.546
	BallPvt [3]	0.337	0.545	0.746
	PSR10 [33]	0.301	0.271	0.822
	DHSP [74]	0.352	1.131	0.686
	NDG [5]	0.316	0.254	0.819
	Ours	**0.105**	**0.160**	**0.946**
2000	Alpha [16]	0.343	1.160	0.726
	BallPvt [3]	0.187	0.546	0.860
	PSR10 [33]	0.175	0.223	0.905
	DHSP [74]	0.181	0.607	0.808
	NDG [5]	0.177	0.217	0.901
	Ours	**0.102**	**0.154**	**0.952**

II. Comparisons to generalizable methods

Type	Method	Ch.-L_2 ↓	Normal ↓	F-Score ↑
Framewise	IPNet [4]	0.752	0.298	0.572
	PTF [72]	0.582	0.278	0.485
	CAPE [42]	0.749	0.332	0.411
	LIG [30]	0.623	0.289	0.875
Temporal	OFlow [49]	5.767	0.344	0.350
	4D-CR [28]	5.162	0.398	0.390
	Ours	**0.306**	**0.204**	**0.908**

(a) 4D reconstruction



(b) Non-rigid depth fusion

Fig. 6. (a) Comparisons on 4D reconstruction from sparse points. The leftmost column in Block I represents the sampled point cloud density, the smaller number corresponds to the sparser point cloud. The results in Block II are obtained from the point cloud of density 2000 points/m^2. (b) Qualitative comparisons on non-rigid depth fusion.

Compare to generalizable methods To show the generalization ability of our method, we train LoRD on the training set of 100 sequences, then fix the network parameters and optimize the latent codes of local parts to fit the input point cloud via back-propagation. In this experiment, we use the point density of 2000 points/m^2 (same as the results in the last group of Fig. 6 (a, I)), and choose 10 testing sequences of novel subjects for evaluation. As framewise baselines, we choose: IPNet [4] and PTF [72], which takes point cloud as input and output reconstructed mesh via feed forward fashion; CAPE [42] and LIG [30], which obtain reconstructions via the backward optimization similar to us. The OFlow and 4D-CR are still considered as the baseline of temporal methods, we remove their encoders, fix the decoder parameters, and perform backward optimization. For OFlow and 4D-CR, we use the ground truth occupancy instead of oriented point cloud as supervision for more stable results. The results are shown in Fig. 5 (below the solid line) and Fig. 6 (a, II), our method beats all the baselines both qualitatively and quantitatively. We can observe the fine-grained geometry recovered by LoRD in the zoomed-in parts of Fig. 5, as well as detailed clothing deformation, which show that our model trained on small set of data can generalize well to the novel motion sequences. More results are in Supp. Mat.

4.3 Non-Rigid Depth Fusion

We further test LoRD with the application of non-rigid depth fusion. Given a static RGB-D camera, with a person standing in front of it performing different actions, the goal is to accurately track the human motion and merge all depth observations in a time span, and finally produce a dynamic mesh sequence. In this experiment, we use the mesh sequences of length $L = 17$ from CAPE dataset [42], and render each frame to get depth image of resolution 512×512. We compute the normal map based on the depth image, and back-project each pixel into 3D space with the known camera intrinsics to obtain the partial oriented point cloud as the observations. Then we run H4D [29] to get the inner body estimation, and use our pretrained LoRD model to perform auto-decoding. Our approach formulates non-rigid fusion as a temporal completion problem within local parts. We choose DynamicFusion [47], NPMs [51] and PTF [72] as our baseline and show the qualitative comparisons in Fig. 6 (b). We observe that PTF produces overly smooth results, NPMs cannot model the detailed surface geometry for different subjects, and DynamicFusion fails to track the human motion that is very fast or contains self-occlusion and leads to unsatisfactory fusions. In contrast, our model is capable to produce more complete fusion results than DynamicFusion, e.g. back of the first example, and more detailed geometry than PTF and NPMs. Additional results including non-rigid fusion on real-world data and the comparison to more recent human specific fusion work DoubleFusion [77] are provided in Supp. Mat. for the sake of space. Our method shows robustness to the SMPL fitting error and provides more complete results than DoubleFusion.

4.4 Ablation Study

Imperfect body tracking We first provide an ablation study to demonstrate the effectiveness of the proposed inner body refining method. We use the 4D

reconstruction task with the point density 2000 point/m^2 for evaluation. Given the initially estimated SMPL inner body, we manually add the random Gaussian noise to it and compare the reconstruction performances before and after refining. Specifically, we perturb the SMPL shape (β) and pose (θ) parameters by $\beta + = \lambda_\beta \cdot \sigma \cdot \mu$ and $\theta + = \lambda_\theta \cdot \sigma \cdot \mu$, where $\mu \in N(0,1)$, $\lambda_\beta = 0.05$, $\lambda_\theta = 0.01$, and $\sigma \in [3,5]$ represents the level of noise. The quantitative results are show in Tab. 2 (left). Without inner body refining, the reconstruction performance drops fast as the noise level up. And by using our refining method, the performance improves and in general stable on different noise levels.
Local part size We then study the effect of different radii for local part. To this end, we use our pretrained model, and test on the task of 4D reconstruction as previous. The comparisons are shown in Tab. 2 (right). As can be seen, the reconstruction performance is affected by the choice of part radius r. We choose $r = 5cm$ in our experiment for slightly better results. We find that the over-small part is inclined to produce artifacts, possibly due to the limited receptive field within part. And the larger part could lead to overly smooth results.

Table 2. Ablation study. Left: the effectiveness of the inner body refining on different noise levels; Right: the effect of the part radius. We choose part radius $r = 5cm$ in our experiments. The visualization examples are in Supp. Mat.

Noise σ	Refining	Ch.-L_2 ↓	Normal ↓	F-Score ↑
3	Before	1.980	0.297	0.730
	After	0.628	0.245	0.776
4	Before	5.469	0.382	0.605
	After	0.896	0.256	0.758
5	Before	6.815	0.435	0.528
	After	0.753	0.260	0.733

Radius **r**	Ch.-L_2 ↓	Normal ↓	F-Score ↑
$3cm$	0.406	0.278	0.858
$5cm$	**0.306**	**0.204**	**0.908**
$8cm$	0.346	0.205	0.905
$10cm$	0.373	0.210	0.896

5 Conclusion

This work introduces LoRD, a local 4D implicit representation for dynamic human, which aims to optimize a part-level temporal network for modeling detailed human surface deformation, e.g. clothing wrinkles. LoRD is learned on a very small set of training data (less than 100 sequences). Once trained, it can be used to fit different types of observed data including sparse point clouds, monocular depth images via auto-decoding. LoRD is capable to reconstruct high-fidelity 4D human and outperforms the state-of-the-art methods.

Acknowledgements This work was supported by Shanghai Municipal Science and Technology Major Projects (No.2018SHZDZX01, and 2021SHZDZX0103). The corresponding authors are Xiangyang Xue, Yanwei Fu and Yinda Zhang.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Representation learning and adversarial generation of 3d point clouds. arXiv preprint arXiv:1707.02392 **2**(3), 4 (2017)
2. Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J., Davis, J.: Scape: shape completion and animation of people. In: ACM SIGGRAPH 2005 Papers, pp. 408–416 (2005)
3. Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G.: The ball-pivoting algorithm for surface reconstruction. IEEE transactions on visualization and computer graphics **5**(4), 349–359 (1999)
4. Bhatnagar, B.L., Sminchisescu, C., Theobalt, C., Pons-Moll, G.: Combining implicit function learning and parametric models for 3d human reconstruction. In: European Conference on Computer Vision. pp. 311–329. Springer (2020)
5. Bozic, A., Palafox, P., Zollhofer, M., Thies, J., Dai, A., Nießner, M.: Neural deformation graphs for globally-consistent non-rigid reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1450–1459 (2021)
6. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
7. ChaoWen, Zhang, Y., Li, Z., Fu, Y.: Pixel2mesh++: Multi-view 3d mesh generation via deformation. In: ICCV (2019)
8. Chen, X., Jiang, T., Song, J., Yang, J., Black, M.J., Geiger, A., Hilliges, O.: gdna: Towards generative detailed neural avatars. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20427–20437 (2022)
9. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: CVPR. pp. 5939–5948 (2019)
10. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6970–6981 (2020)
11. Choi, H., Moon, G., Lee, K.M.: Beyond static features for temporally consistent 3d human pose and shape from a video. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
12. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: ECCV (2016)
13. Crandall, M.G., Lions, P.L.: Viscosity solutions of hamilton-jacobi equations. Transactions of the American mathematical society **277**(1), 1–42 (1983)
14. Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., Tagliasacchi, A.: Cvxnet: Learnable convex decomposition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 31–44 (2020)
15. Deng, B., Lewis, J.P., Jeruzalski, T., Pons-Moll, G., Hinton, G., Norouzi, M., Tagliasacchi, A.: Nasa neural articulated shape approximation. In: European Conference on Computer Vision. pp. 612–628. Springer (2020)
16. Edelsbrunner, H., Mücke, E.P.: Three-dimensional alpha shapes. ACM Transactions on Graphics (TOG) **13**(1), 43–72 (1994)
17. Erler, P., Guerrero, P., Ohrhallinger, S., Mitra, N.J., Wimmer, M.: Points2surf: Learning implicit surfaces from point clouds. In: ECCV (2020)

18. Fan, H., Su, H., Guibas, L.J.: 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. pp. 605–613 (2017)
19. Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.: Local deep implicit functions for 3d shape. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4857–4866 (2020)
20. Gillette, R., Peters, C., Vining, N., Edwards, E., Sheffer, A.: Real-time dynamic wrinkling of coarse animated cloth. In: Proceedings of the 14th ACM SIGGRAPH/eurographics symposium on computer animation. pp. 17–26 (2015)
21. Girdhar, R., Fouhey, D.F., Rodriguez, M., Gupta, A.: Learning a predictable and generative vector representation for objects. In: ECCV (2016)
22. Goldenthal, R., Harmon, D., Fattal, R., Bercovier, M., Grinspun, E.: Efficient simulation of inextensible cloth. In: ACM SIGGRAPH 2007 papers, pp. 49–es (2007)
23. Gropp, A., Yariv, L., Haim, N., Atzmon, M., Lipman, Y.: Implicit geometric regularization for learning shapes. arXiv preprint arXiv:2002.10099 (2020)
24. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: Atlasnet: A papier-*mâché* approach to learning 3d surface generation. arXiv preprint arXiv:1802.05384 (2018)
25. Guler, R.A., Kokkinos, I.: Holopose: Holistic 3d human reconstruction in-the-wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10884–10894 (2019)
26. Habermann, M., Xu, W., Zollhoefer, M., Pons-Moll, G., Theobalt, C.: Livecap: Real-time human performance capture from monocular video. ACM Transactions On Graphics (TOG) **38**(2), 1–17 (2019)
27. Habermann, M., Xu, W., Zollhofer, M., Pons-Moll, G., Theobalt, C.: Deepcap: Monocular human performance capture using weak supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5052–5063 (2020)
28. Jiang, B., Zhang, Y., Wei, X., Xue, X., Fu, Y.: Learning compositional representation for 4d captures with neural ode. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5340–5350 (2021)
29. Jiang, B., Zhang, Y., Wei, X., Xue, X., Fu, Y.: H4d: Human 4d modeling by learning neural compositional representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19355–19365 (2022)
30. Jiang, C., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T.: Local implicit grid representations for 3d scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6001–6010 (2020)
31. Kanazawa, A., Tulsiani, S., Efros, A.A., Malik, J.: Learning category-specific mesh reconstruction from image collections. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 371–386 (2018)
32. Kanazawa, A., Zhang, J.Y., Felsen, P., Malik, J.: Learning 3d human dynamics from video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5614–5623 (2019)
33. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. ACM Transactions on Graphics (ToG) **32**(3), 1–13 (2013)
34. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
35. Kocabas, M., Athanasiou, N., Black, M.J.: Vibe: Video inference for human body pose and shape estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5253–5263 (2020)

36. Lassner, C., Romero, J., Kiefel, M., Bogo, F., Black, M.J., Gehler, P.V.: Unite the people: Closing the loop between 3d and 2d human representations. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6050–6059 (2017)
37. Liao, Y., Donne, S., Geiger, A.: Deep marching cubes: Learning explicit surface representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2916–2925 (2018)
38. Liu, X., Qi, C.R., Guibas, L.J.: Flownet3d: Learning scene flow in 3d point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 529–537 (2019)
39. Lombardi, S., Simon, T., Schwartz, G., Zollhoefer, M., Sheikh, Y., Saragih, J.: Mixture of volumetric primitives for efficient neural rendering. ACM Transactions on Graphics (TOG) **40**(4), 1–13 (2021)
40. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: Smpl: A skinned multi-person linear model. ACM transactions on graphics (TOG) **34**(6), 1–16 (2015)
41. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM siggraph computer graphics **21**(4), 163–169 (1987)
42. Ma, Q., Yang, J., Ranjan, A., Pujades, S., Pons-Moll, G., Tang, S., Black, M.J.: Learning to Dress 3D People in Generative Clothing. In: Computer Vision and Pattern Recognition (CVPR) (2020)
43. Maas, A.L., Hannun, A.Y., Ng, A.Y., et al.: Rectifier nonlinearities improve neural network acoustic models. In: Proc. icml. vol. 30, p. 3. Citeseer (2013)
44. Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Sridhar, S., Pons-Moll, G., Theobalt, C.: Single-shot multi-person 3d pose estimation from monocular rgb. In: 2018 International Conference on 3D Vision (3DV). pp. 120–130. IEEE (2018)
45. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019)
46. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision. pp. 405–421. Springer (2020)
47. Newcombe, R.A., Fox, D., Seitz, S.M.: Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 343–352 (2015)
48. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: 2011 10th IEEE international symposium on mixed and augmented reality. pp. 127–136. IEEE (2011)
49. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Occupancy flow: 4d reconstruction by learning particle dynamics. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5379–5389 (2019)
50. Oechsle, M., Mescheder, L., Niemeyer, M., Strauss, T., Geiger, A.: Texture fields: Learning texture representations in function space. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4531–4540 (2019)
51. Palafox, P., Božič, A., Thies, J., Nießner, M., Dai, A.: Npms: Neural parametric models for 3d deformable shapes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12695–12705 (2021)

52. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019)
53. Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., Zhou, X.: Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9054–9063 (2021)
54. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: European Conference on Computer Vision. pp. 523–540. Springer (2020)
55. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-nerf: Neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10318–10327 (2021)
56. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from RGB-D data. In: CVPR (2018)
57. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
58. Rempe, D., Birdal, T., Zhao, Y., Gojcic, Z., Sridhar, S., Guibas, L.J.: Caspr: Learning canonical spatiotemporal point cloud representations. Advances in Neural Information Processing Systems **33** (2020)
59. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2304–2314 (2019)
60. Saito, S., Yang, J., Ma, Q., Black, M.J.: Scanimate: Weakly supervised learning of skinned clothed avatar networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2886–2897 (2021)
61. Selle, A., Su, J., Irving, G., Fedkiw, R.: Robust high-resolution cloth using parallelism, history-based collisions, and accurate friction. IEEE transactions on visualization and computer graphics **15**(2), 339–350 (2008)
62. Su, Z., Xu, L., Zheng, Z., Yu, T., Liu, Y., Fang, L.: Robustfusion: Human volumetric capture with data-driven visual cues using a rgbd camera. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16. pp. 246–264. Springer (2020)
63. Sumner, R.W., Schmid, J., Pauly, M.: Embedded deformation for shape manipulation. In: ACM siggraph 2007 papers, pp. 80–es (2007)
64. Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11358–11367 (2021)
65. Tan, F., Tang, D., Dou, M., Guo, K., Pandey, R., Keskin, C., Du, R., Sun, D., Bouaziz, S., Fanello, S., et al.: Humangps: Geodesic preserving feature for dense human correspondences. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1820–1830 (2021)
66. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: European conference on computer vision. pp. 402–419. Springer (2020)
67. Terzopoulos, D., Platt, J., Barr, A., Fleischer, K.: Elastically deformable models. In: Proceedings of the 14th annual conference on Computer graphics and interactive techniques. pp. 205–214 (1987)

68. Tiwari, G., Sarafianos, N., Tung, T., Pons-Moll, G.: Neural-gif: Neural generalized implicit functions for animating people in clothing. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11708–11718 (2021)
69. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: ECCV (2018)
70. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. arXiv preprint arXiv:2106.10689 (2021)
71. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Transactions on Graphics (TOG) **36**(4), 72 (2017)
72. Wang, S., Geiger, A., Tang, S.: Locally aware piecewise transformation fields for 3d human mesh registration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7639–7648 (2021)
73. Wei, L., Huang, Q., Ceylan, D., Vouga, E., Li, H.: Dense human body correspondences using convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1544–1553 (2016)
74. Wei, X., Chen, Z., Fu, Y., Cui, Z., Zhang, Y.: Deep hybrid self-prior for full 3d mesh generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5805–5814 (2021)
75. Xu, W., Chatterjee, A., Zollhöfer, M., Rhodin, H., Mehta, D., Seidel, H.P., Theobalt, C.: Monoperfcap: Human performance capture from monocular video. ACM Transactions on Graphics (ToG) **37**(2), 1–15 (2018)
76. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. Advances in Neural Information Processing Systems **34** (2021)
77. Yu, T., Zheng, Z., Guo, K., Zhao, J., Dai, Q., Li, H., Pons-Moll, G., Liu, Y.: Doublefusion: Real-time capture of human performances with inner body shapes from a single depth sensor. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7287–7296 (2018)
78. Zheng, Z., Huang, H., Yu, T., Zhang, H., Guo, Y., Liu, Y.: Structured local radiance fields for human avatar modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15893–15903 (2022)
79. Zheng, Z., Yu, T., Li, H., Guo, K., Dai, Q., Fang, L., Liu, Y.: Hybridfusion: Real-time performance capture using a single depth sensor and sparse imus. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 384–400 (2018)
80. Zheng, Z., Yu, T., Liu, Y., Dai, Q.: Pamir: Parametric model-conditioned implicit representation for image-based human reconstruction. IEEE transactions on pattern analysis and machine intelligence (2021)

Supplementary Material

In this supplementary material, we provide implementation details, additional experimental results, visualization of the surface deformation within a local part, additional qualitative results, and discussions about limitations and future work of our approach.

1 Implementation Details

1.1 Network Architecture

Motion model We adapt the architecture of the IMNet [9] for our motion model. As shown in Fig. 7 (a), the input is the concatenation of: the motion code $c_m \in \mathbb{R}^{128}$, 3D query point $\mathbf{x} \in \mathbb{R}^3$ and normalized time value $T \in \mathbb{R}^1$. The network is based on multi-layer perceptrons with the skip connection to copy the input to concatenate with the output feature of the first 4 layers, each layer has the nonlinear activation of LeakyReLU ($\alpha = 0.2$) [43] except the last layer. We follow LIG [30] to reduce the feature dimension of each hidden layer by 4 fold to obtain an efficient motion model. The output of our motion model is a deformation vector $\mathbf{x}^* \in \mathbb{R}^3$ that transforms the given point to its position in the space of the canonical frame, i.e. $T = 0$.
Canonical shape model The canonical shape model uses the auto-decoder network proposed in DeepSDF [52], which is shown in Fig. 7 (b). The input is the concatenation of the canonical shape code $c_s \in \mathbb{R}^{128}$ and 3D query point $\mathbf{x} \in \mathbb{R}^3$, and the network predicts a signed distance value $\mathbf{s} \in \mathbb{R}^1$ for the given point. We reduce the number of hidden layers from 8 to 6 and the feature channels from 512 to 256 for the efficiency. And following IGR [23], the softplus activation ($\beta = 100$) and geometric initialization are also used.
Texture model The texture model is used to produce the colored results in Fig. 1 and 5 of the main paper. We modify the decoder architecture proposed in Texture Fields [50] for our 4D texture inference, and the architecture is shown in Fig. 7 (c). The texture model is fed with the texture code $c_t \in \mathbb{R}^{128}$ and the concatenation of a 3D point $\mathbf{x} \in \mathbb{R}^3$ and a normalized time value $T \in \mathbb{R}^1$, and predicts the RGB values $\mathbf{c} \in \mathbb{R}^3$ for the given point. There are five residual blocks in the texture model network, each of which consists of two fully connected layers with a skip connection from the input to the second layer. The input of each block is summed up with the feature encoded from the texture code $c_t \in \mathbb{R}^{128}$.

1.2 Local Part Coordinates

In this section, we introduce how to select part centers on SMPL meshes and define the local coordinate system for each part.
Part center sampling Start from a template mesh of SMPL topology in the rest-pose, we first uniformly sample a point set P ($|P| = 100K$) on its surface, and then perform the following steps recurrently to maintain a set of the selected part centers C: 1) randomly choose a point $\mathbf{x}$ from P; 2) remove all the points

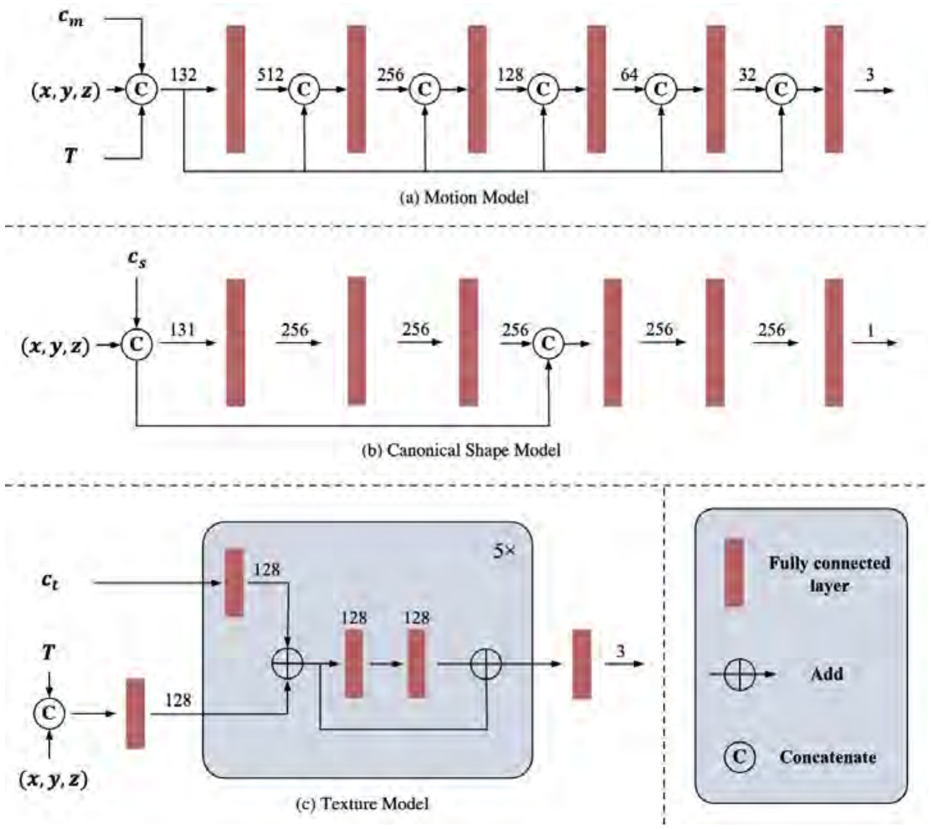


Fig. 7. Detailed network architectures in our framework.

whose Euclidean distance from $\mathbf{x}$ is less than or equal to $0.5\mathbf{r}$ from P, where $\mathbf{r} = 5cm$ is the part radius we pre-defined; 3) add $\mathbf{x}$ to C. The loop terminates when $|P| = 0$, and we finally obtain 2127 part centers.

Local coordinate frame Given a point $\mathbf{c}_k$ on a triangle face as the part center, we use the face normal as the up-axis $\vec{a}_{k_1}$, the direction vector from point c_k to a vertex of the triangle as another axis $\vec{a}_{k_2}$, and finally the last axis $\vec{a}_{k_3} = \vec{a}_{k_1} \times \vec{a}_{k_2}$. The rotation matrix of part k is then define as $\mathbf{R}_k = [\vec{a}_{k_1}, \vec{a}_{k_2}, \vec{a}_{k_3}]$. Now, a 3D point P_{glo} in the world coordinate frame can be transformed to P^k_{loc} in the local coordinate frame of part k with $P^k_{loc} = \mathbf{R}^T_k (P_{glo} - \mathbf{c}_k)$.

1.3 Other Details

Point sampling The query points used for training and test-time optimization come from three sources: 1) surface; 2) near surface space; 3) free space in the bounding box. During training, we sample $M = 10000$ surface points on the ground truth mesh, while at the test time, the points are randomly chosen from the input point cloud. Given the on-surface points, we obtain the near surface points by adding a displacement vector sampled from a Gaussian distribution $N(0, 0.01)$ to each on-surface point. And $M/8$ free space points are uniformly sampled within the human bounding box. We compute the initial bounding box with the inner body mesh, and pad $10cm$ on each axis as the sampling region.

Auto-decoding During the test time, we use the trained model to fit complete or partial point clouds via the auto-decoding manner [52] (main paper Sec. 3.4). Specifically, we fix the parameters of the local implicit network and optimize the latent codes with back-propagation by minimizing the objective function introduced in Sec. 3.3 of the main paper. We initialize the latent codes for each local part with the random vectors sampled from a Gaussian distribution $N(0, 0.01)$ and use the Adam optimizer [34] with learning rate $1e^{-3}$ to perform backward optimization for 3000 iterations. We use the same objective function and loss weights as training (Sec. 3.3 of the main paper). The optimization process takes around $15min$ for each sequence of $L = 17$ frames on a single GeForce RTX 2080Ti GPU card.

Mesh postprocessing As mentioned in the main paper (Sec. 3.4), the original extracted mesh of our method contains interior back-faces and some tiny floating components in the outside. This possibly because that for the off-surface points, the Eikonal term [23] only constrains the $L2$-norm of their gradients rather than specific SDF values, which may confuse the prediction of gradient direction on the points that far from the part centers. We remove these artifacts by using the post-processing algorithm introduced in LIG [30]. Specifically, we first compute the centroid and surface normal for each face of the original mesh reconstructed by the network, and find its k nearest points on the input point cloud to calculate the mean normal consistency as the normal alignment score. Then a Laplacian kernel is used to smooth the normal alignment score, all faces with the score below a certain normal alignment threshold n and disconnected components with area below a are discarded. We used the same postprocessing parameters as LIG, except that we only preserve the most biggest connected component rather than use the area threshold a as we focus on reconstructing single object.

Mesh surface extraction Different from LIG [30], which only evaluates the occupied grid cells and assumes all empty ones to be "exterior" space to extract the isosurface, we construct a SDF volume and evaluate every grid point with our local implicit function, ensuring to reconstruct the outer surface not covered by any local parts defined on the inner body mesh. Similar to the strategy illustrated in the main paper Sec. 3.3, we evaluate a given point with n parts that covered it ($n = 32$ in the surface extraction process), and get the final prediction via average-pooling. And if not covered by any parts, n nearest parts are used. To make the inference more efficient, instead of cubic volume used in previous work [45,52], we resort to a rectangular volume that defined as the bounding box of inner body mesh with $10cm$ padding on each axis. It takes around $15s$ to extract a mesh of resolution 256.

2 Additional Experimental Results

2.1 Non-Rigid Fusion

Real-world performance To further demonstrate the value of our method in the actual application scenario, we perform extended evaluation on the real data. Specifically, we capture a human motion depth sequence with a static Azure Kinect sensor, and use our model pre-trained on 100 sequences to conduct non-rigid reconstruction based on the point cloud from raw depth images (note that the background is filtered out). The qualitative results are shown in Fig. 8, DynamicFusion produces fine-grained surface details on cloth but contains noisy (body edge) and incomplete (face, arms and legs) areas. In contrast, our model recovers smoother and more complete geometry with plausible temporal deformation thanks to the local 4D formulation. We use part radius $r = 10cm$ for more stable results. Note that the experiments in here and Sec. 4.3 of the main paper, we only reconstruct the partial surface observed by the static camera in the time span, which cannot be quantitatively evaluated, thus we only show the qualitative comparisons.

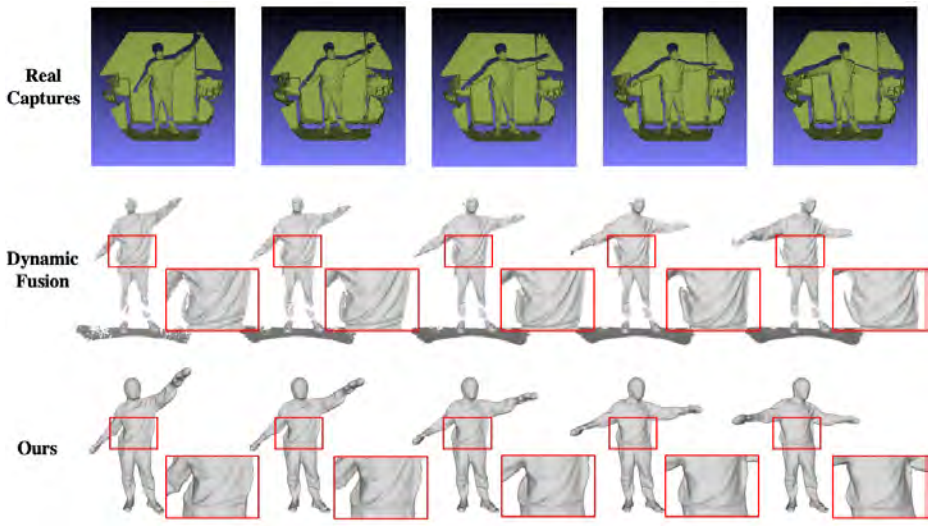


Fig. 8. Monocular depth fusion on the real-world depth captures. The depth images are captured with a static Azure Kinect sensor. The first row shows the point cloud from raw depth captures. Our method is capable to recover the plausible geometry with detailed surface deformation, e.g. clothing wrinkles in the zoomed in part, in such challenging scenario.

Comparisons with DoubleFusion Qualitative results are shown in Fig. 9. We add the same SMPL body mesh used in LoRD into DoubleFusion as body prior, which generally makes the fusion reliable, but it still struggles with aligning finer motion. Also, there are some missing parts, e.g. arm or leg, caused by fitting error. In contrast, LoRD employs a temporal model to provide global geometry consistency between frames, which is more robust to fitting error.

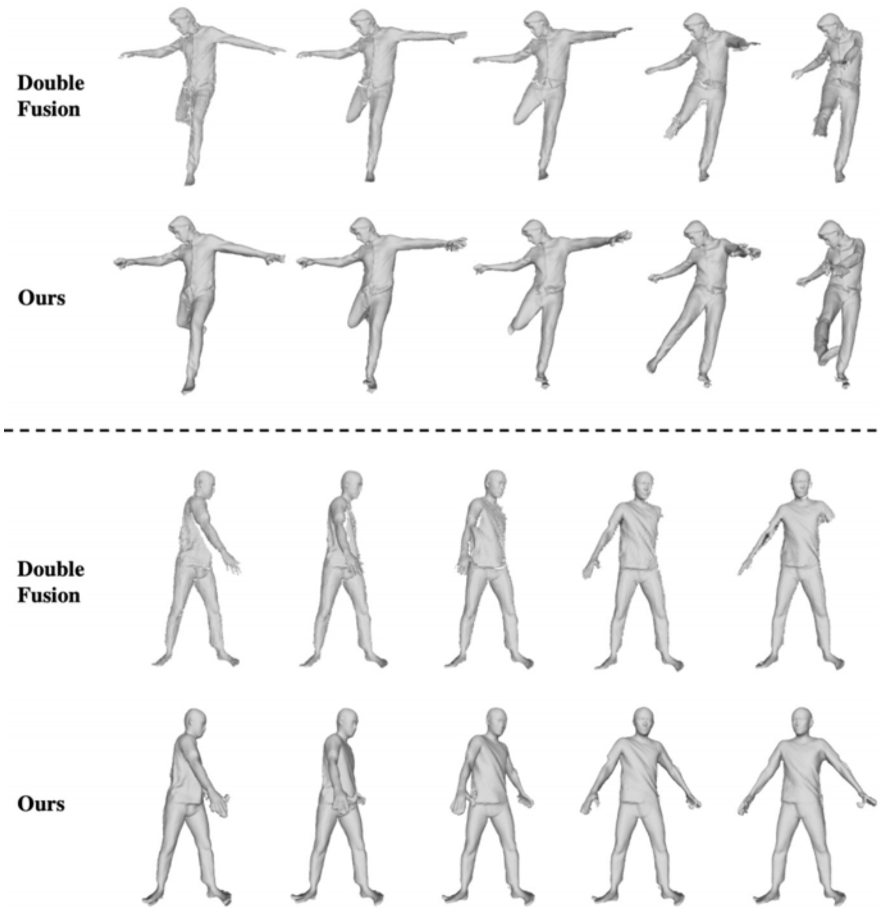


Fig. 9. Qualitative comparisons with DoubleFusion. We choose two examples and uniformly sample 5 out of 17 frames for visualization.

Moving monocular camera In addition to the static camera, we also conduct non-rigid depth fusion in the setting of a moving monocular camera, that is, a person performing some motions during a time span with a depth camera rotating around him concurrently. In this case, the observation of each time step is still partial, but almost every body part can be observed during the time span. Our goal is to compensate each frame based on the geometry information observed in other frames. Specifically, given a dynamic mesh sequence of $L = 17$ frames, we make the camera rotates around the up-axis (Y-axis in our setup) and render a 512×512 depth image every 360/17 degrees. In this experiment, we assume the camera poses are known, and use the same 10 novel sequences chosen in the generalizable 4D reconstruction task for evaluation. We choose PTF [72], NPMs [51] and CAPE [42] as our baseline. For PTF, we input the partial point cloud to the pretrained single-view model, and obtain the reconstructions with feed-forward fashoin. And for NPMs and CAPE, we use the auto-decoding manner to optimize the latent codes in their formulation based on each frame partial observation. Note that we provide the ground truth SMPL pose to CAPE, and only optimize the cloth latent code. All these methods are working in the frame-wise manner to produce the sequence results. The quantitative results in Tab. 3 show that our LoRD representation outperforms all the baseline methods by a large margin. The qualitative comparisons are shown in Fig. 14. Thanks to the local 4D representation, the proposed method achieves high-quality completion results, and produces the detailed geometry and plausible temporal deformation in the invisible areas, while PTF and NPMs fail to hallucinate accurate geometry from partial observations as they do not utilize the temporal information, and CAPE cannot model the high-fidelity surface details.

Table 3. Quantitative comparisons on monocular depth fusion with a moving camera. Our method outperforms all the baseline methods by a large margin.

	Ch.-L_2 ↓	Normal ↓	F-Score ↑
PTF [72]	37.936	0.810	0.145
NPMs [51]	0.981	1.933	0.429
CAPE [42]	1.010	0.356	0.377
Ours	**0.395**	**0.226**	**0.750**

2.2 Interpolation of Latent Codes

Like many other local implicit representations, our model does not have a compact global latent space to sample from, so the representation ability is often measured by the capability of fitting observations, validated in Sec 4.1 of main paper. Nevertheless, we can still interpolate between two human sequences by interpolating representation between the corresponding local parts (as shown in Fig. 10). We first linearly interpolating c_s and SMPL poses showing smooth change between two subjects. Then we interpolate c_m, c_s and per frame SMPL pose jointly to show the representation ability of motion space, since the local deformation is related to global motion. Note that the texture model is optimized per sequence without continuous latent space.

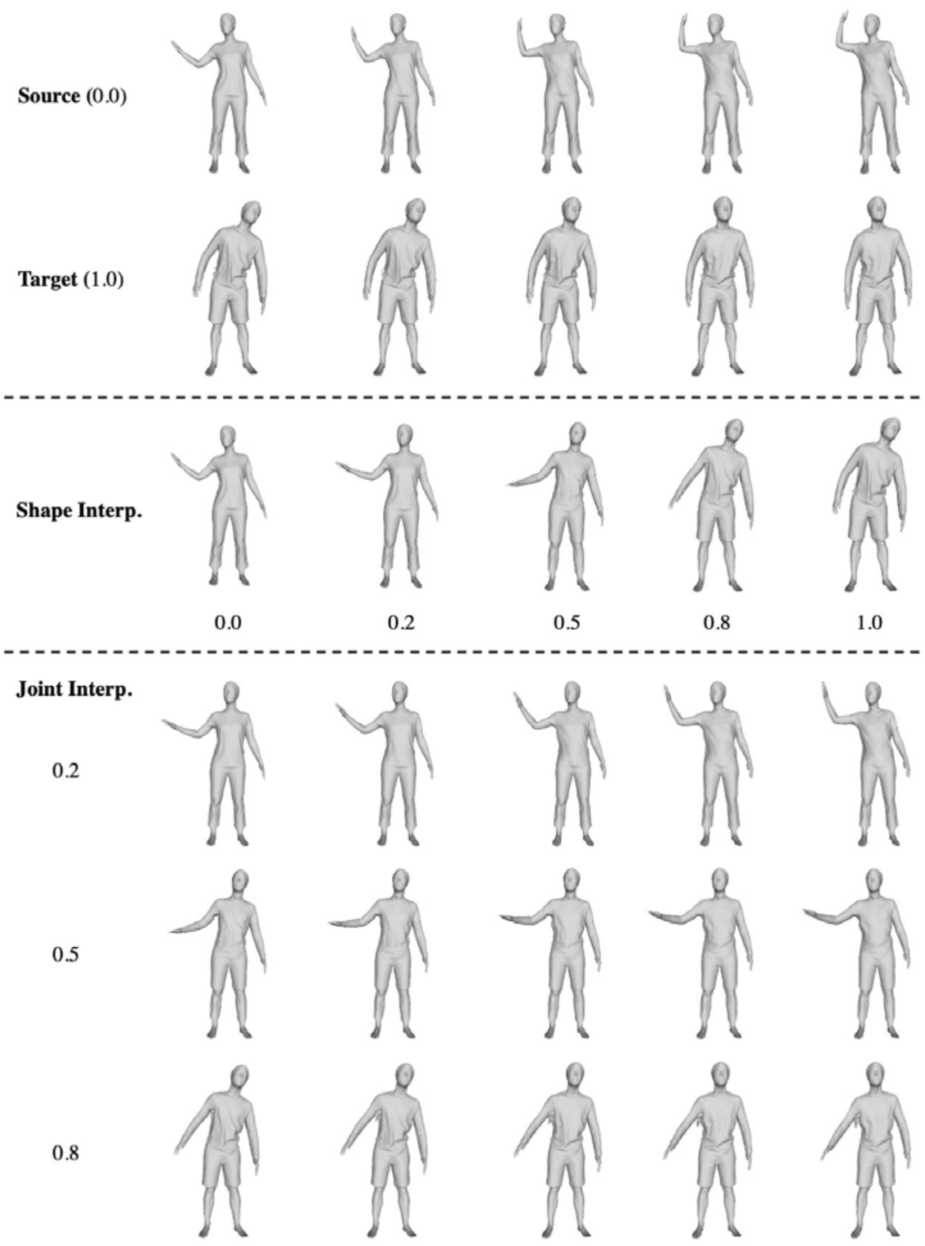


Fig. 10. We choose two sequences (Source and Target) to perform interpolation of latent codes. "Shape Interp." means we interpolate latent shape codes and SMPL poses of the first frames to show the evolution of shape. "Joint Interp." indicates we interpolate shape code, motion code and per frame pose jointly. The decimal values denote the interpolation coefficients

2.3 Ablation Study

Effect of sequence length L The key novel capability of our model is to represent the temporal deformation of 3D shape, so that setting $L = 1$ makes this infeasible, and our model degenerates to framewise method LIG [30] and thus achieves similar performance. Additionally, we test different L on instance-level reconstruction from sparse points task and show the results in Tab. 4. Note that longer sequence demands stronger network capacity while more geometry information can be exchanged between frames through our motion model. The results show that LoRD is able to work with different sequence lengths. We choose $L = 17$ following 4D-CR [28] during training to learn our 4D representation. For longer sequence during test-time, we can utilize the sliding window strategy similar to HMMR [32] to recurrently recover the whole sequence.

Table 4. Evaluations of LoRD on different sequence lengths.

L	Ch.-$L2$ ↓	Normal ↓	F-Score ↑
5	0.175	0.160	0.914
10	0.207	0.176	0.880
17	0.192	0.158	0.945
20	0.159	0.173	0.875
30	0.389	0.221	0.718

Generalization Besides the results shown in Sec. 4.2 of the main paper, we also choose 10 sequences from our training set, and get the performance (Ch.-L_2=0.317, Normal=0.170, F-Score=0.926) on 4D reconstruction task. The results are in general comparable with the performance on testing set (last row of Fig. 6 (a, II) in the main paper), which reflects that thanks to the local part formulation, our model has strong generalization ability with the prior of local surface deformation learned from the training sequences. We further verify this by training our model on even 1 motion sequence of 17 frames, which also can produce the high-quality reconstructions on novel sequences. We show the qualitative results in Sec. 4.2 of Supp. Mat.

Comparison with only test-time training In Fig. 11, we show the loss curves of optimization w/ and w/o pretraining. It can be seen that the pretraining helps optimization converge faster and more stable. By taking about 50 mins per sequence, optimizing from scratch obtains slightly better performance than optimizing with pretraining.

3 Local Surface Visualization

Our LoRD representation aims to use a local part-level network to model the detailed temporal deformation of surface patches. To show this, we visualize the temporal deformation of a local patch in Fig. 12. Specifically, we choose a

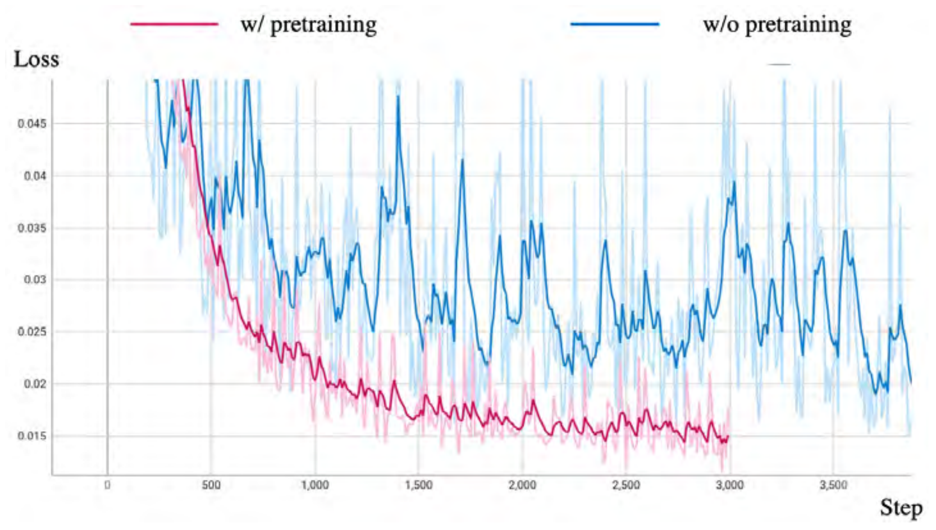


Fig. 11. Loss curves during optimization w/ and w/o pretraining.

point cloud sequence of 17 frames, each of which has $10K$ points, and perform auto-decoding to optimize the latent codes of each local part. After that, we select a part and extract the surface patch with the local implicit network (main paper Sec. 3.2) conditioned on its latent codes. We show the temporal deformation of this local patch under the global (above) and local (below) coordinate frames respectively. It can be seen that the temporal changing within the local part is smooth and coherent, and our method successfully models the detailed deformation, e.g. changing of the clothing wrinkle.

4 Qualitative Results

4.1 4D Reconstruction

Shape quality Fig. 15, 16 and 18 are the extended figures of Fig. 5 in the main paper, which show more qualitative comparisons with the SoTA methods on 4D reconstruction from sparse points. And Fig. 19 shows some additional reconstruction results of our model trained on 100 sequences.
Textured results We show more textured results in Fig. 13. We choose raw scan mesh sequences from CAPE dataset (containing holes and noises) and use our pretrained LoRD model in this experiment. The results above are obtained with colored sparse point clouds sampled from the scan meshes as input. And the results below are obtained from the rendered RGB-D sequences.

4.2 Ablation study

Imperfect body tracking Fig. 17 (before refining) shows some challenging cases that LoRD produces artifacts with inaccurate body tracking. We provide

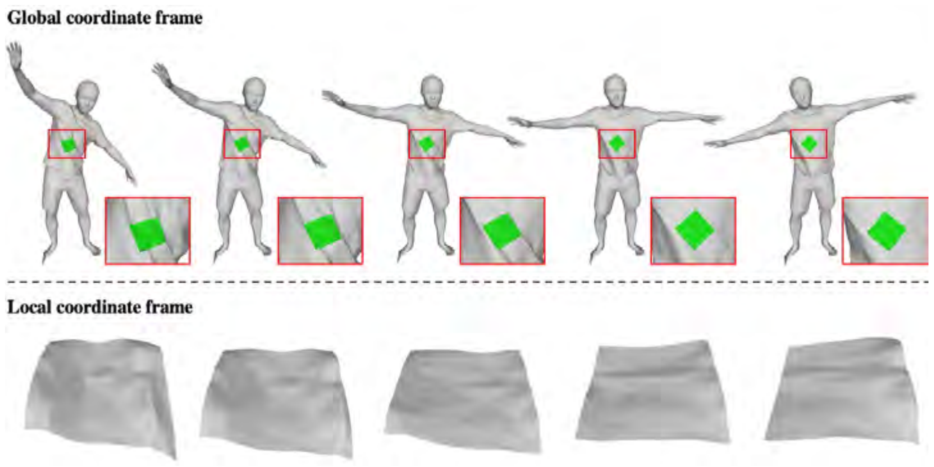


Fig. 12. We select a local part and visualize the surface patch within it. The results above show the temporal deformation of this patch (green) under the global coordinate frame, while below under the local coordinate frame.

the reconstructions after refining that demonstrate the effectiveness of our inner body refining method. For each example, we stack the ground truth clothed mesh (green) together with the initialized inner body mesh (gray) on the left to reflect the inaccurate estimation, and show the reconstructions on the right. The results show our refining process successfully corrects the noisy inner body, which facilitates the reconstruction.

Local part size Fig. 20 shows the reconstruction results of different part radii. We can observe that the part size slightly affects the quality of reconstructions, the over-small ($\mathbf{r} = 3cm$) part produces some artifacts around body and hands and the reconstructed surface is under-smooth, whereas the larger parts ($\mathbf{r} = 8cm \& \mathbf{r} = 10cm$) tend to recover overly smooth results for some frequency details such as clothing wrinkles and fingers.

Generalization As mentioned in the main paper Sec. 3.3, the training of our LoRD representation is very data-efficient. We show the results of our model trained on 100 sequences (Fig. 5, 6 (b) in the main paper and Fig. 18, 19 in the Supp. Mat). Additionally, we train our model on one motion sequence of length $L = 17$ from the training set, then test on the novel sequences, and show the qualitative results in Fig. 21. As shown, our model trained on even one sequence still gains generalization ability and produces the high-fidelity reconstructions, which demonstrates the generalization power of our local 4D representation.

5 Limitations and Future work

The proposed LoRD representation shows the powerful capability and achieves the state-of-the-art performance on various tasks. Now we discuss a few limitations of our method which also points to the future directions.

First, we now rely on the SMPL body model to temporally track local parts, though existing methods can produce accurate body in many cases, it is still challenging to work in the complex real scenario. Extending our representation to cooperate with more general tracking methods such as scene flow, deformation graph, would make our method stronger and capable of modeling non-human objects, e.g. animals.

Second, the current experiments mainly focus on 4D reconstruction from 2.5/3D data, e.g. sparse point clouds, RGB-D. Combining the recent neural rendering techniques [46,70,76] with our LoRD representation to support 4D reconstruction from pure RGB videos would be a promising future direction.

Third, we currently use a unified part radius in our formulation. However, different body parts contain various levels of detail, which may be suitable to model by different sizes of parts. Defining the part radii according to the body part label would be one solution.

We believe that the proposed representation could potentially be a building block for various applications, e.g. Metaverse, Robotics, animation, and provides some insights for future research directions.

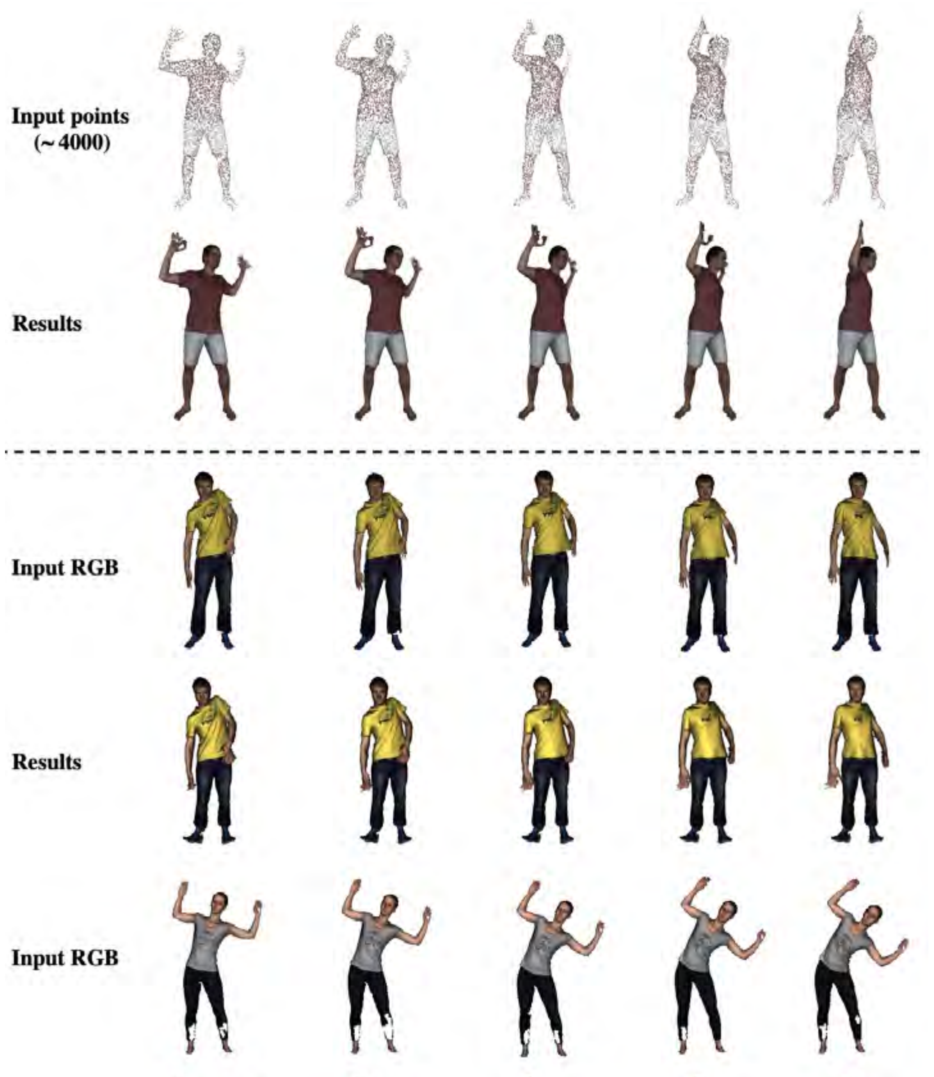


Fig. 13. More textured results achieved by our method. Note that the results below are obtained from RGB-D inputs, we only show color images for visualization.

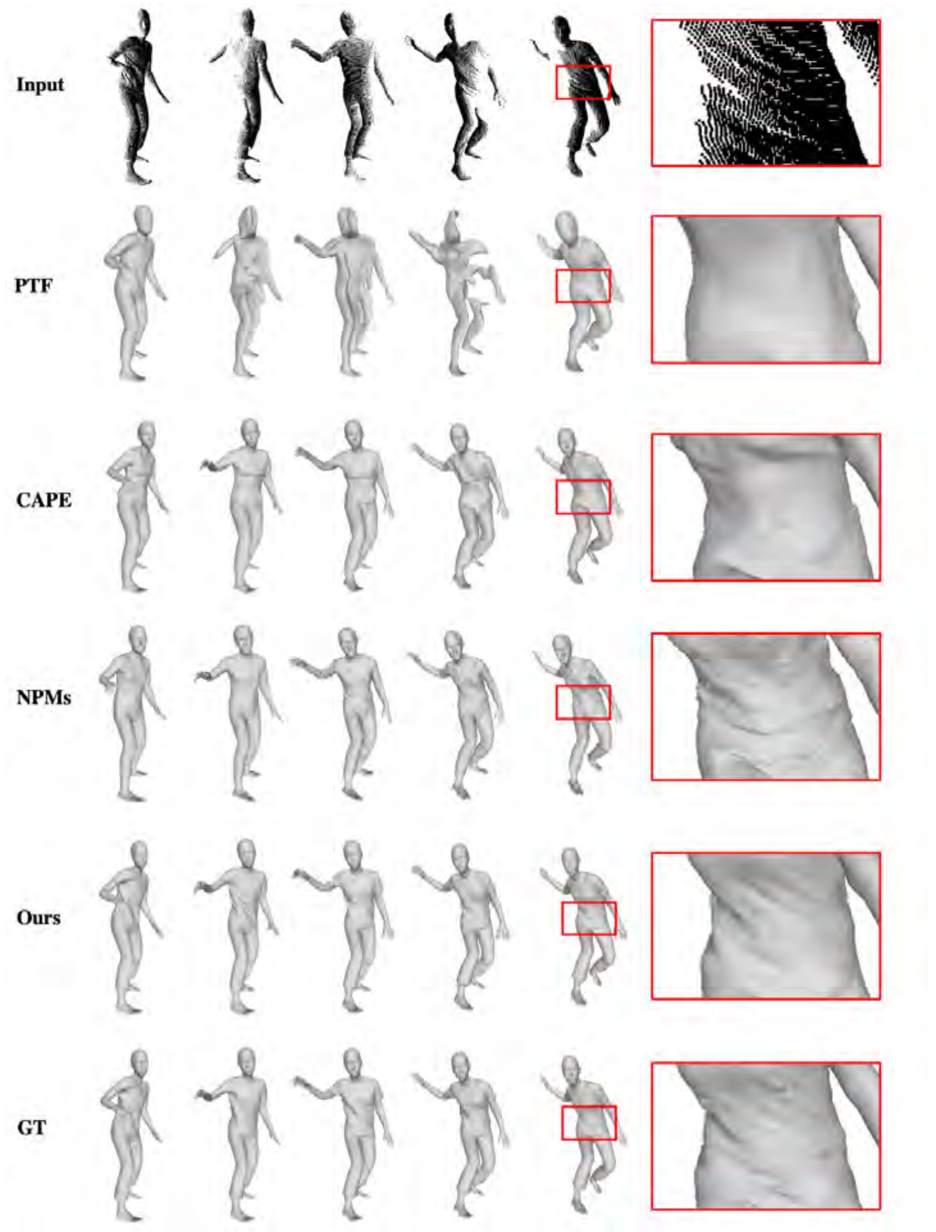


Fig. 14. Qualitative results on monocular depth fusion with a moving camera. The reconstructed sequences have $L = 17$ frames, and we uniformly choose 5 frame for visualization. We assume a moving camera that rotates around the performer, and render a depth image every 360/17 degrees. The partial point clouds are obtained by back-projecting the depth images with the camera intrinsics, and rotate according to the known camera extrinsics.

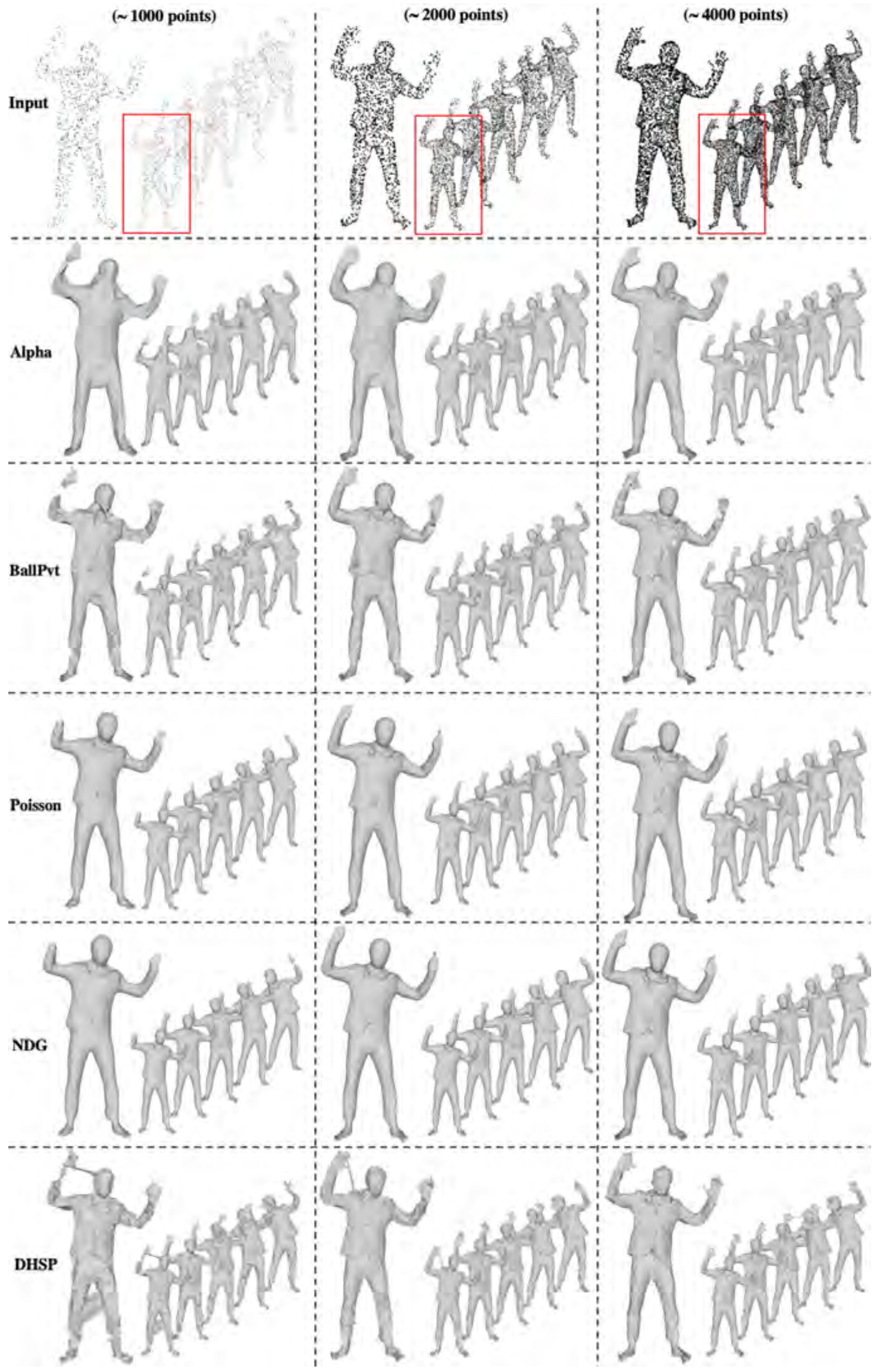


Fig. 15. 4D reconstruction from sparse points (instance-level) (1).

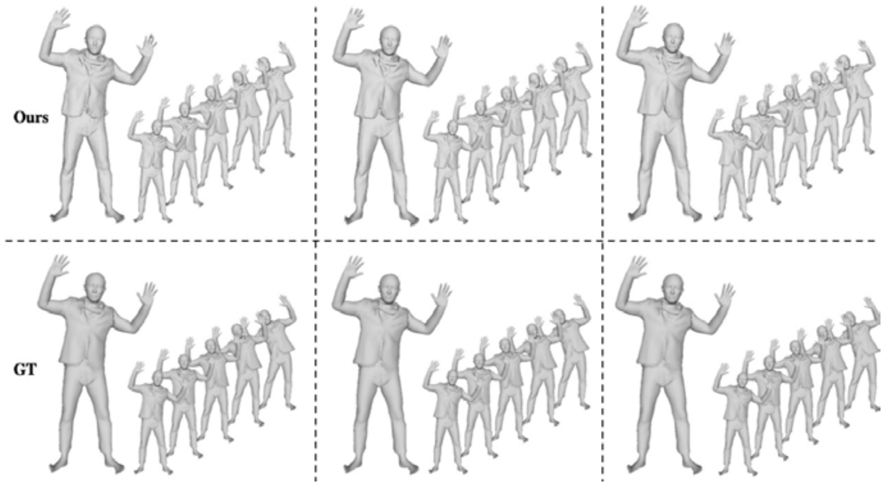


Fig. 16. 4D reconstruction from sparse points (instance-level) (2). The reconstructed sequences have $L = 17$ frames, and we uniformly choose 5 frame for visualization. We zoom in the first frame to show the surface details clearer.

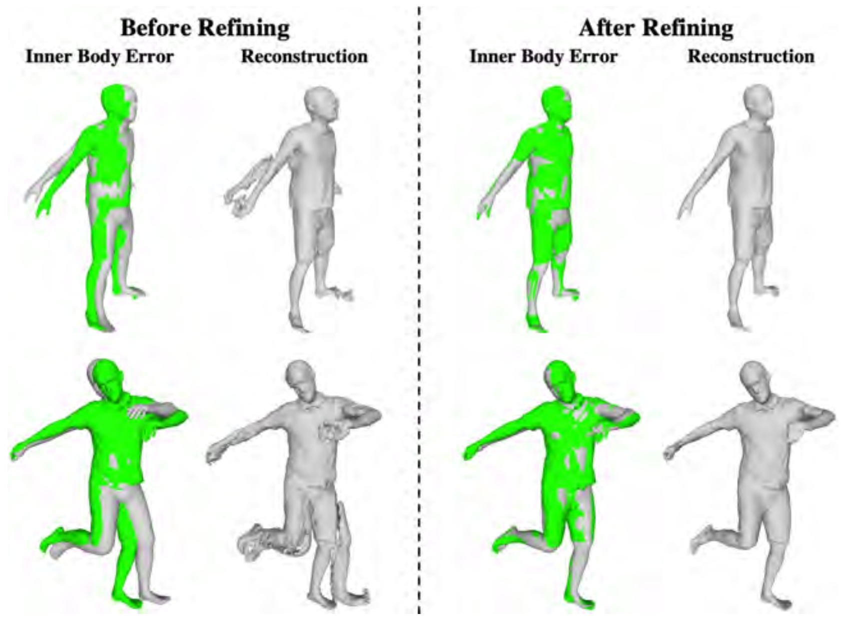


Fig. 17. Effectiveness of the inner body refining. We show the inner body and the reconstructed meshes before (left) and after (right) our inner body refining process. Note that we stack the ground truth clothed mesh (green) with the inner body estimation (gray) to show the inaccurate parts.

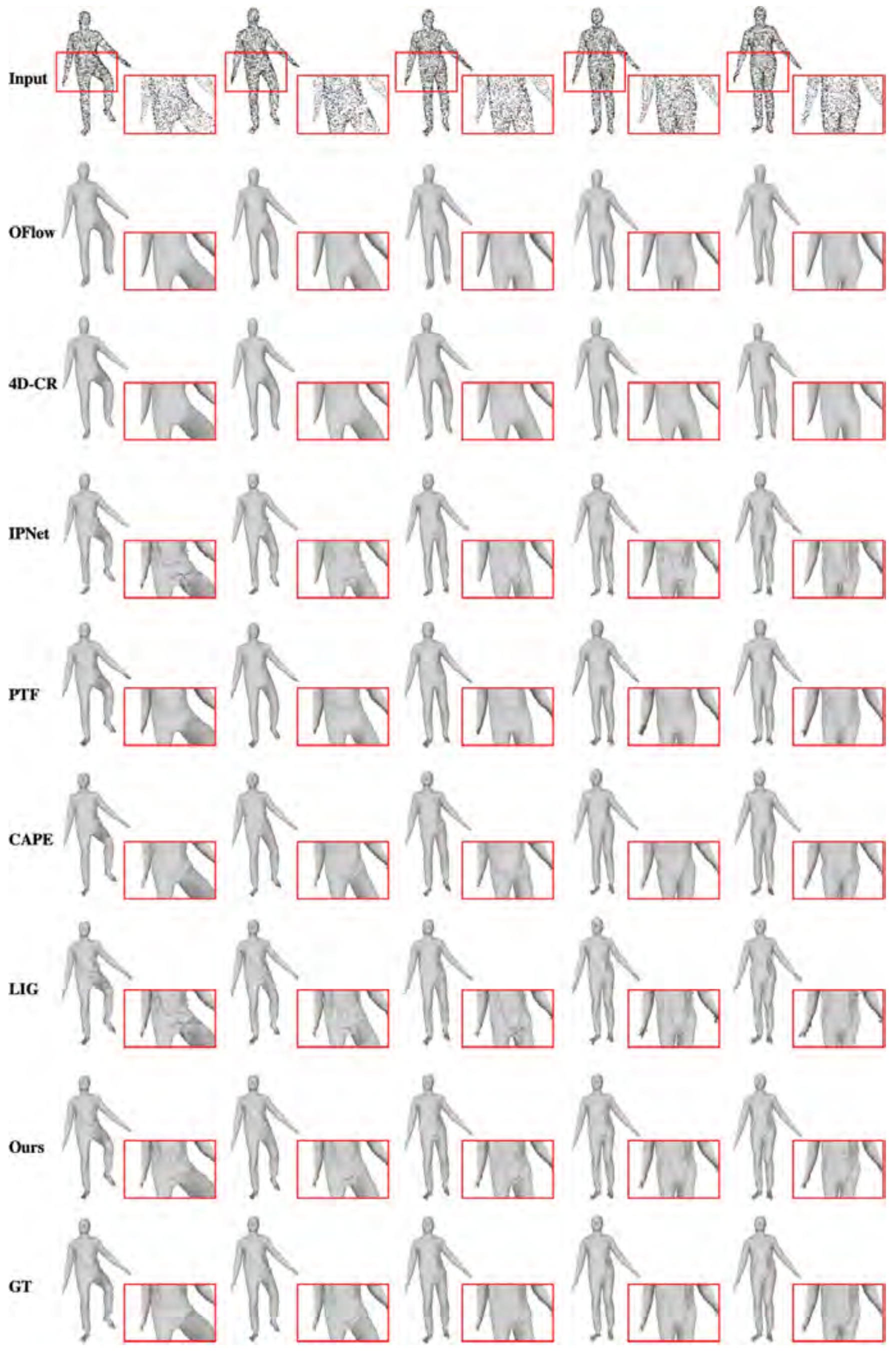


Fig. 18. 4D reconstruction from sparse points (generalization). The reconstructed sequences have $L = 17$ frames, and we uniformly choose 5 frame for visualization.

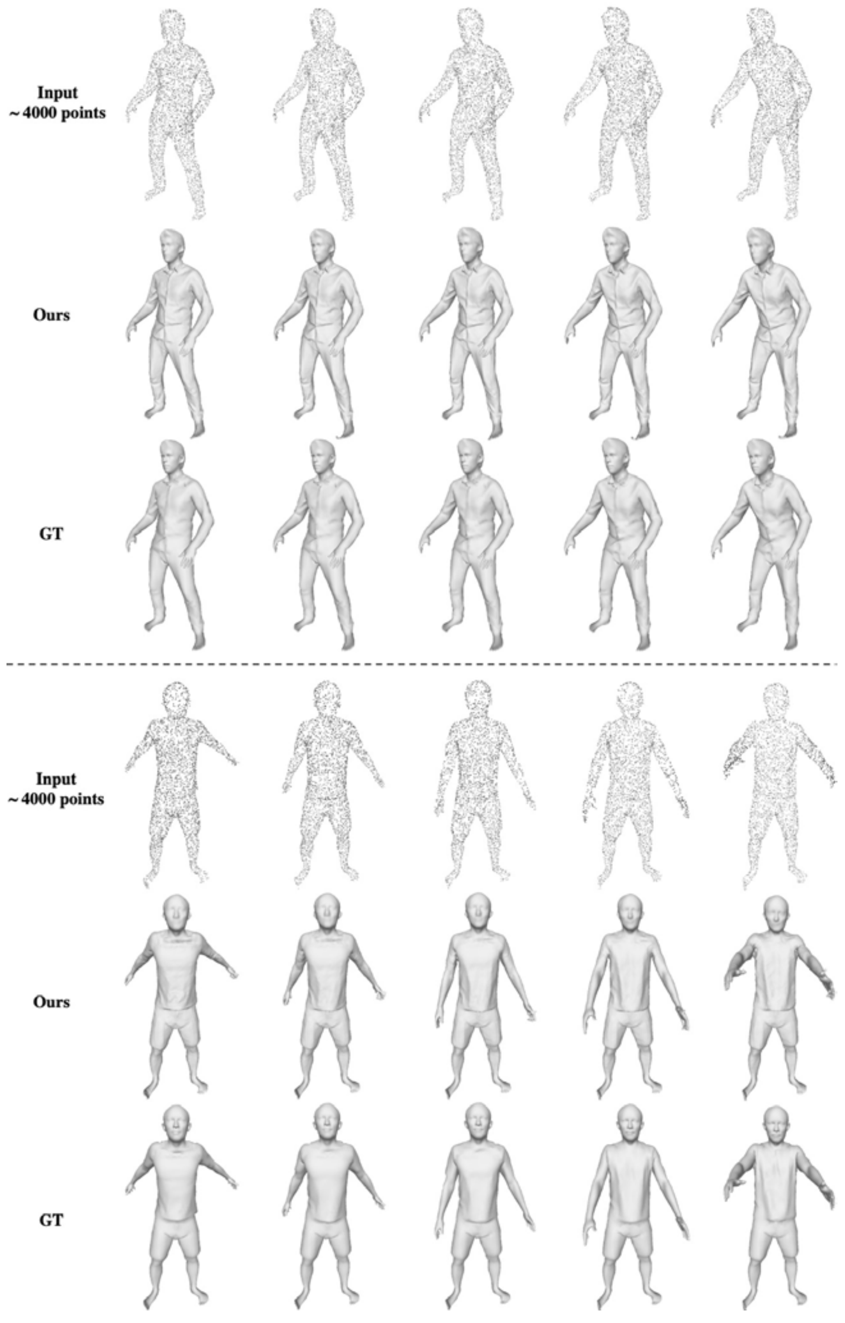


Fig. 19. 4D reconstruction from sparse points (generalization). Here we show more qualitative results achieved by our model trained on 100 sequences. The reconstructed sequences have $L = 17$ frames, and we uniformly choose 5 frame for visualization.

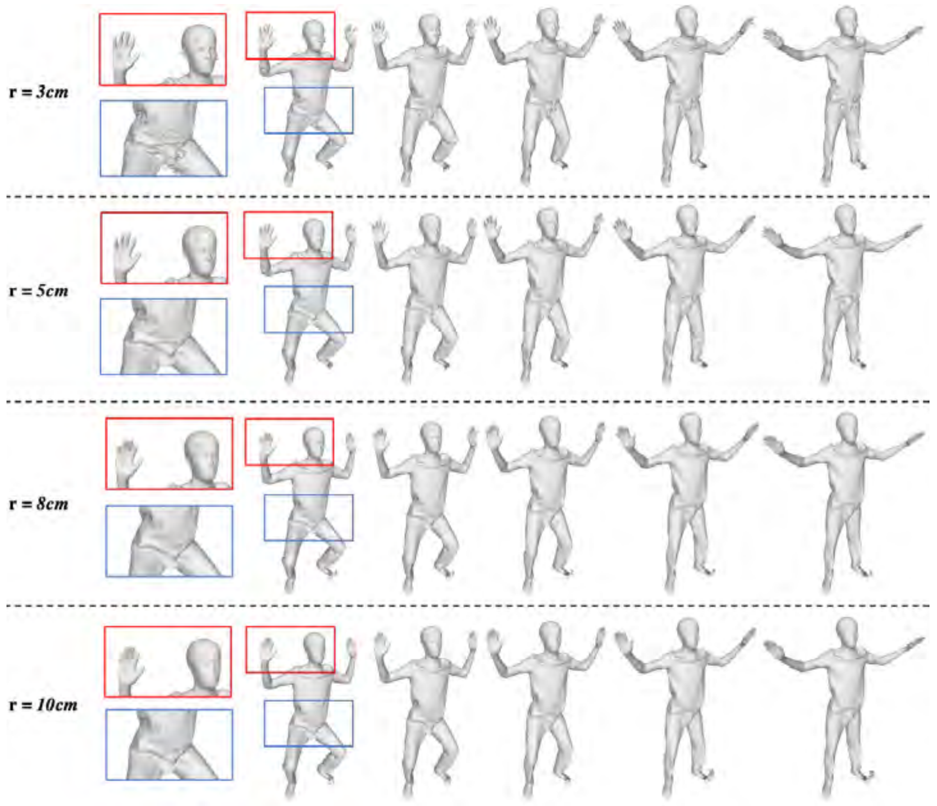


Fig. 20. Effect of the part radius. Different sizes of local parts affect the reconstruction performance but not very heavily. We choose part radius $\mathbf{r} = 5cm$ in our experiments as it in general produces better results.

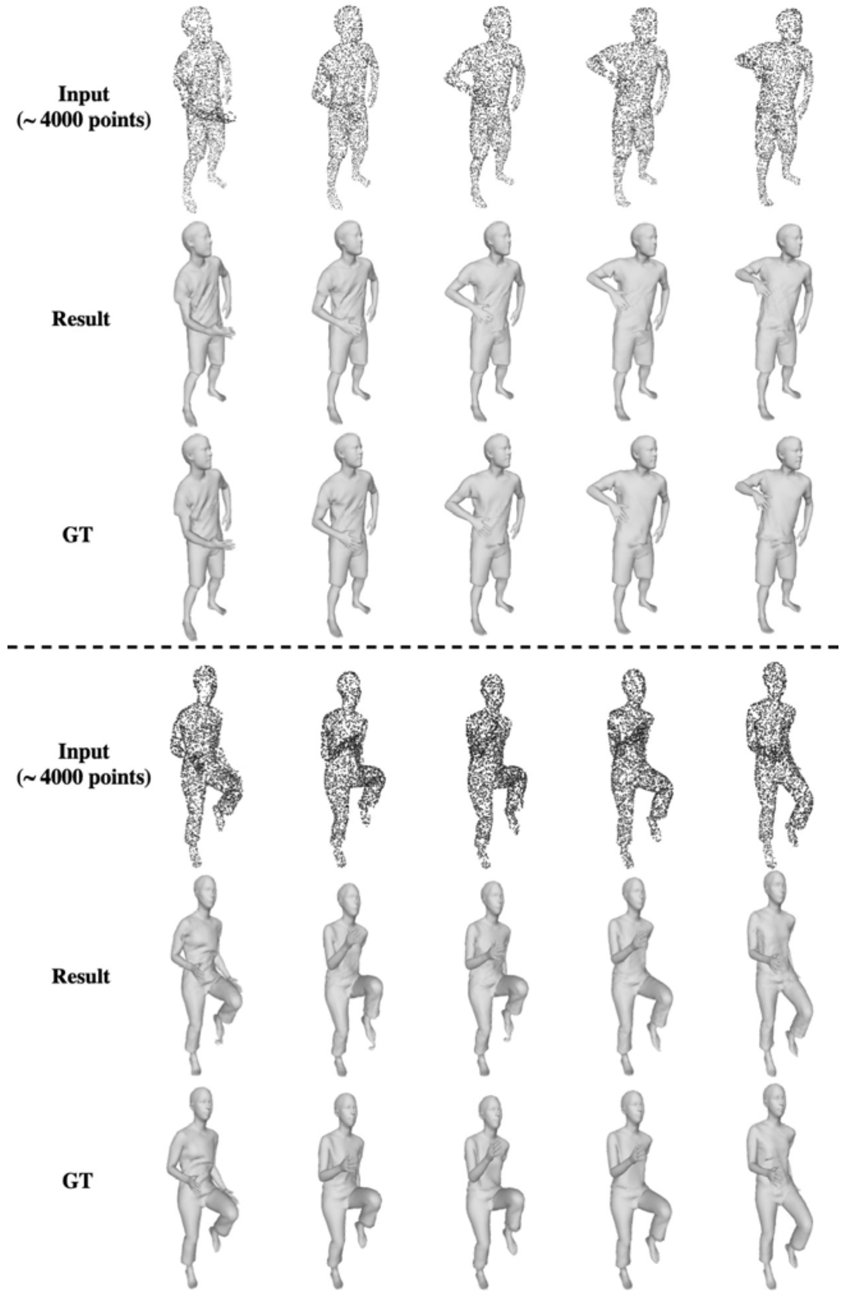


Fig. 21. Qualitative results from our model trained on only 1 motion sequence of length $L = 17$, which show that our LoRD representation can learn local deformation prior from very few data and generalize to novel sequences with high-quality geometry and temporal deformation.