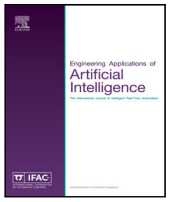
Engineering Applications of Artificial Intelligence 127 (2024) 107404

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Double reverse diffusion for realistic garment reconstruction from images

Jeonghaeng Lee [a], Duc Nguyen [a], Jongyoo Kim [b], Jiwoo Kang [c], Sanghoon Lee [a,*]

[a] Department of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, South Korea
[b] Microsoft Research Asia in Beijing, 100080, China
[c] Sookmyung Women's University, Seoul 04310, South Korea

ARTICLE INFO

Keywords:
3D garment reconstruction
Deep learning
Mesh processing
Graph diffusion
Graph long short term memory
Double reverse diffusion

ABSTRACT

Creating realistic digital 3D avatars has been getting more attention thanks to the introduction of new multimedia formats such as augmented and virtual reality. An important factor making avatars realistic is clothes. In this paper, we investigate a new method to reconstruct realistic garments from a set of images and body information. Early methods working on realistic images struggle to faithfully reconstruct the garment details. As deep learning is increasingly applied to geometric data which can conveniently represent garments, we devise a novel deep learning-based solution to the garment reconstruction problem. We offer a new perspective on the reconstruction problem and treat it as a reversion of the smoothing diffusion process. To achieve this goal, we propose to deform the smoothed human mesh into a clothed human via a Double Reverse Diffusion (DReD) process. For the first reverse diffusion, we introduce a novel operator called Graph Long Short-Term Memory (GraphLSTM) which recursively diffuses features to produce a deformed mesh by modeling the relationships between vertices. Then, the output mesh can be repeatedly upsampled and deformed by the above pipeline to obtain finer garment details, which can be seen as another reverse diffusion process. To obtain features for the reverse diffusion, we extract pixel-aligned features transferred from images and explore to incorporate the visibility of garments from the image viewpoints. Through detailed experiments on two public datasets, we demonstrate that DReD synthesizes more realistic wrinkled garments with lower errors and offers faster inference than previous methods.

1. Introduction

In recent years, the world has been witnessing an increase in demands for new media contents such as virtual, augmented, and mixed reality. Much research has been dedicated to realize more realistic, customized and personalized contents. Accordingly, there have been intensive research for comprehensive 3D human modeling, encompassing advancements in 3D facial reconstruction (Kang et al., 2021), pose estimation (Lee et al., 2022), and avatar creation (Alldieck et al., 2019; Bhatnagar et al., 2019; Peng et al., 2021). However, most early studies focused on predicting joint positions or reconstructing only the naked body model. An important factor to create a realistic avatar is garments. For 3D human reconstruction, complex body deformations and figures are widely modeled by parametric models. Garments introduce even more complicated and diverse shapes and unpredictable deformations, leading to a very challenging problem in avatar creation.

Garment reconstruction has been intensively studied recently. A dominant approach is a physics-based simulation (PBS) (Patel et al., 2020; Guan et al., 2012) in which a deep learning predictive model is trained on simulated data to predict garment wrinkles based on body shape and pose. However, physics-based engines can only approximate a limited range of materials and environments, and usually fail to model the complex interactions between human body and garments. Needless to say, these methods do not provide satisfaction in terms of realism. As a workaround to avoid the need for physics simulators, other research attempted to involve images. Typically, there are two lines of research. On the one hand, many studies start with a well-engineered 3D human model, for *e.g.*, the SMPL model (Loper et al., 2015), which is driven by shape and pose information, and then directly deform the human model into the garment shape (Huang et al., 2020; Alldieck et al., 2019). Although this approach is simple and yields realistic results, it is difficult to model multiple clothing layers. On the other hand, garment templates can be pre-constructed and depending on the human pose and shape, they can be skinned and deformed to match the descriptions from images, enabling multi-layer clothing modeling (Bhatnagar et al., 2019; Jiang et al., 2020). However, templates should be created for each specific garment type, and cannot generalize to unseen garments. Moreover, human body and garment models can intertwine with each other during draping and generate artifacts due to the difference in

* Corresponding author.
E-mail address: slee@yonsei.ac.kr (S. Lee).

https://doi.org/10.1016/j.engappai.2023.107404
Received 21 February 2023; Received in revised form 18 September 2023; Accepted 28 October 2023
Available online 7 November 2023

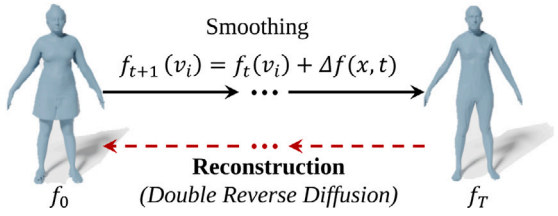


Fig. 1. The key concept that governs our method: Reconstruction can be seen as the reversion of a smoothing diffusion process.

sizes and forms between templated and real garments. In this paper, we aim to reconstruct garments with realistic wrinkles by leveraging natural images to deform an under-cloth human model directly, eliminating the need for PBS data and garment templates.

Triangular meshes have been a de facto standard in the computer graphics industry. It is the most common representation of humans and garments, and yet very few methods natively processed the mesh as a graph. To handle the uneven mesh topology, the works in Alldieck et al. (2019) and Gundogdu et al. (2019) applied graph neural networks (GNNs) (Kipf and Welling, 2017; Defferrard et al., 2016a). However, training GNNs requires a careful strategy. Also, it is difficult to deepen the network architecture as deep GNNs tend to map adjacent nodes to nearly identical embeddings (Rong et al., 2020; Chen et al., 2020). Paradoxically, shallow GNNs cannot model the interaction between nodes with long geodesic distances. Therefore, we need to explore a way to enable deeper networks and extract better representative geometric features for garment reconstruction.

In this paper, we propose a novel 3D garment reconstruction framework that can realistically produce 3D clothes depicted in a set of input images. Our solution deforms mesh vertices into garments directly to avoid using templates and for better generalization. To deform the initial mesh into garments, we completely rethink the problem of reconstruction from the graph diffusion point of view. Our concept is illustrated in Fig. 1. We hypothesize that a fully-clothed human mesh can be transformed into a smooth SMPL model (Loper et al., 2015) via a smoothing process described by a heat diffusion equation. In this way, the 3D garment reconstruction problem can be seen as a reversion of this diffusion process. This reverse process fully considers the geometry of human body, and the vertex deformation is conditioned on a large neighborhood, which better ensures the feasibility of the solution compared to previous works. Conceptually, our work is similar to the popular Denoising Diffusion Probabilistic Model (Sohl-Dickstein et al., 2015; Ho et al., 2020; Dhariwal and Nichol, 2021) (DDPM), but our work is deterministic and the training of our model is radically different from that of DDPMs. Reversing the smoothing, however, is impossible because this diffusion process is irreversible per the second law of thermodynamics, not to mention that the smoothing process might be more complicated than the heat diffusion. Nevertheless, it is possible to leverage the sparse-view images to estimate the reverse process. An overview of the proposed system is shown in Fig. 2. As can be seen, we first extract pixel-aligned features from the images and then reverse the smoothing process gradually. Technically, any existing graph convolutional (GC) operator can be used for the reversion. To better learn the deformation, it is necessary to condition on a wide surface area, which requires many diffusion steps to model the relationship between geodesically distant vertices. Because it is difficult to do so with simple GCNs (Rong et al., 2020; Chen et al., 2020), we propose a new GC operator termed Graph Long Short-term Memory (GraphLSTM), which is a novel adaptation of the standard LSTM (Hochreiter and Schmidhuber, 1997) for graph signals. GraphLSTM can resolve the oversmoothing problem that most deep graph convolutional networks (GCNs) suffer by integrating past information into the current diffusion step. To prolong the diffusion chain without incurring much memory, we propose to use GraphLSTM in a recursive manner by applying it T times to the outputs as depicted in Fig. 2.

To provide features for the reverse diffusion process, we use a convolutional neural network (CNN) to obtain features from input images at multiple scales. To this end, we extract vertex-specific and pixel-aligned features, which can capture the silhouettes of garments, as well as distilling edge information formed by wrinkles. In addition, we also explicitly model the visibility of the 3D model in the input images to indicate the reliability of the features extracted from each view.

It comes to our realization that the feature diffusion with GraphLSTM is agnostic about mesh resolution, so it is possible to apply it recursively to N mesh resolutions as can be seen in Figs. 2 and 5. Concretely, after obtaining a low-resolution output, we upsample the mesh by adding vertices and interpolating the mesh signals, and then reuse it again in the same reverse diffusion process as before, which can be seen as another reverse diffusion. Thus, we term our method Double Reverse Diffusion (DReD). Notably, GraphLSTM is shared across scales, which is radically different from cascaded regression and progressive learning (Karras et al., 2018). To the best of our knowledge, this work is the first to formulate image reconstruction as a reverse diffusion process, as well as in terms of adapting LSTM as a graph processing operator.

Our contributions in this work can be summarized below:

- We introduce a novel framework termed "DReD" (Double Reverse Diffusion) to reconstruct garments from multi-view images in the subspace of the SMPL model. We regard an SMPL mesh as an oversmoothed clothed human body and as a result of a diffusion process, thereby proposing to reverse this process for detailed 3D garment reconstruction. Viewing the reconstruction as the reversion of the smoothing process can potentially unlock a wider application of many other diffusion models into 3D reconstruction.
- We design a novel graph operator called GraphLSTM which is an LSTM adapted to graph signals.
- We present a comprehensive benchmark to show the effectiveness of our solution to 3D garment reconstruction.

2. Related work

2.1. Garment modeling

There has been some research considering reconstruction of fully-clothed human. In Alldieck et al. (2018b), given a realistic monocular video, Alldieck et al. reconstructed 3D body models including full clothing effects, facial expressions, and hairstyles using the SMPL model. After this pioneering work, many subsequent studies (Alldieck et al., 2019; Huang et al., 2020; Habermann et al., 2020) followed the same practice. These methods deformed the vertices of SMPL but did not separate the clothes from the body. Also, they did not solely focus on garments, so as a result, wrinkles are not reconstructed in detail.

In another line of research, garments are modeled separately using templates, and these templates are deformed for every detected clothes layer. In Lähner et al. (2018), Laehner et al. proposed a method that combined a statistical model, which basically is a Principal Component Analysis model similar to SMPL, and a generative network that takes a temporal 3D scan sequence and deforms pre-defined templates into clothes. Because of learning from real data, the output is realistic, but the 4D sequence input is hard to come by in practice. Bhatnagar et al. (2019) introduced a network that predicts SMPL-based garments from multi-view images. BCNet (Jiang et al., 2020) predicted garment and body shapes only from a single-view image, and the garment template was not SMPL. The method was trained and tested on a synthetic dataset, which makes the output unrealistic. The works in Pons-Moll et al. (2017) and Casado-Elvira et al. (2022) fitted SMPL-based garments to monocular videos. As these methods are based on

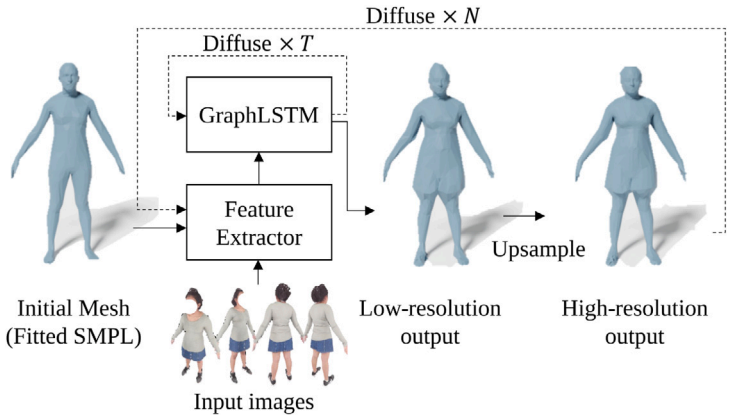


Fig. 2. A bird-eye view of our framework. Given several multi-view images of garments and a fitted SMPL model, we extract features for the SMPL mesh and send it to a GraphLSTM module in which the mesh features are repeatedly diffuse to reverse the smoothing process. The deformed SMPL can be upsampled and recursively subjected to this process to obtain the final high-resolution garments.

optimization, they are quite slow compared to learning methods. Inspired by the neural radiance field (NeRF), in Saito et al. (2020), Huang et al. (2020), Peng et al. (2021) and Xiu et al. (2022), the authors reconstructed an implicit neural volume for humans conditioning on SMPL priors. Qiu et al. (2023) optimized a signed distance function field to model garments from monocular videos. These models can easily produce incorrect shapes and induce artifacts in void space. Hong et al. (2021) proposed to reconstruct garments from a point cloud sequence, which highly restricts the wide applicability of the method.

Many methods resort to physics-based models (Guan et al., 2012; Patel et al., 2020; Bertiche et al., 2021). DRAPE (Guan et al., 2012) constructed a simulated dataset to animate clothing effects on the human body of various shapes and poses. In Patel et al. (2020), garments are predicted based on human pose and shape and garment geometry by a neural network (NN). Bertiche et al. (2021) proposed a solution to garment animation according to human body data. Zou et al. (2023) introduced a new large-scale PBS dataset containing diverse poses, garments and motions. However, as mentioned earlier, the physics-based simulated data do not look realistic. In addition, a predictive model as a function of body information and garment type cannot fully capture the sophisticated underlying physics that forms garment wrinkles. Meanwhile, our method predicts garment deformations from images, which is a workaround for the explicit modeling of the underlying body-garment interaction; hence, the problem is simpler and the outputs are more realistic.

2.2. Generic Multi-view 3D reconstruction

Multi-view 3D reconstruction has been intensively studied in the past. Structure-from-Motion (SfM) (Hartley and Zisserman, 2004) has been the most well-known method in this category and since then, it has been perfected in Schonberger and Frahm (2016). This algorithm is critical for different problems such as dependence on matching accuracy, computational complexity, and it takes much time for the optimization. Deep learning has been shown to be very effective in a wide range of tasks and thus, it has been constantly integrated into 3D reconstruction (Fan et al., 2017; Nguyen et al., 2019; Wang et al., 2020; Park et al., 2019). These methods usually focus on static objects like tables and cars, which do not have a wide variety of shapes, poses, and deformations like garments. In Bogo et al. (2016), Bogo et al. reconstructed humans only considered up to shape and pose information without fine deformation of skins and clothes. In Zheng et al. (2019), Zheng et al. reconstructed the whole human mesh, but did not consider garments specifically. Other works have tried to recover different human-related properties (Kemelmacher-Shlizerman and Basri, 2010; Feng et al., 2018).

Recently, there has been much noise created by NeRF, which started by the work of Mildenhall et al. (2020) and followed by a considerable amount of research (Zhang et al., 2020; Wang et al., 2021). From multiple images of a scene, NeRF learns an implicit function that maps 3D locations to density and color, and a 3D mesh can be recovered using marching cubes (Lorensen and Cline, 1987). These methods require many views in order to succeed and they usually cannot generalize to new scenes.

2.3. Graph diffusion

There have been several works investigating diffusion on graph. The most similar research that considers diffusion as a graph operator is perhaps Klicpera et al. (2019). In this study, the authors introduced a new operator that computes a linear combination of different diffusion steps. Our operator has the same spirit because an LSTM keeps track of all the past hidden activations in a cell gate, but the aggregation is non-linear. The work in Jiang et al. (2019) introduced a feature diffusion module by using Laplacian regularization models. In a remote study, Wang et al. (2017) used LSTMs to predict diffusion sequences. Sharp et al. (2022) proposed a diffusion layer in spectral domain for general graph processing purposes. We note that these works are orthogonal to our study, and thus they can possibly be combined into a better framework. Elsewhere, Liang et al. (2016) also employed LSTM for graph signals, however, their operator predominantly uses a dot product approach, neglecting the diffusion between graph nodes.

2.4. Denoising diffusion probabilistic models

In image generation area, DDPMs have set a new milestone in terms of image quality recently. Since its inception (Sohl-Dickstein et al., 2015), numerous studies have been introduced to make these models work with high-resolution and complex images (Ho et al., 2020; Song et al., 2021). These models work on the assumption that noise is added to an image until no information is retained in the image. Thus, the reverse process can be seen as generating image from noise. The main difference between our work and DDPMs is that for DDPMs, an NN is usually trained to reverse only one step of the diffusion process and then runs recursively in the generation step. On the other hand, we use LSTMs to reverse the whole process during training because of the lack of intermediate supervision, and thus the entire reversion does not take as much time as DDPMs.

2.5. Recursive neural network

The idea of a modern recursive neural network can be dated back to Socher et al. (2011) in which the authors used a single NN to predict recursive structures. More recent methods have been proposed (Kim et al., 2016; Guo et al., 2019) in order to increase networks' depths without using adding more trainable parameters. Different from ours, these works were not placed in the context of diffusion, and they did not employ recurrent networks. We employ recursive architecture in order to simulate a diffusion chain and use an LSTM to control the recursion.

3. Proposed method

3.1. Overview

Our goal is to reconstruct the detailed 3D mesh of garments given only sparse multi-view RGB images accompanied with human body information in the form of a registered SMPL model. The overall framework is presented in Fig. 2. First, we describe the mesh deformation pipeline governed by the graph diffusion principle in Section 3.2. Next, we explain how features can be extracted from images (Section 3.3) and transferred to the SMPL mesh (Section 3.4) for the deformation process. Finally, we demonstrate the recursion of the deformation process in Section 3.5, which constitutes our full framework, DReD.

3.2. Mesh deformation via feature reverse diffusion

The motivation of our work stems from the smoothing process of a surface. In this case, the change of the surface over time can be modeled using a heat diffusion equation (Cannon, 1984)

$$\frac{\partial}{\partial t} f(x,t) = \Delta f(x,t), \tag{1}$$

where $f : \mathcal{M}\times\mathcal{R} \rightarrow \mathcal{R}$ is a function defined over garment surface, Δ is a Laplace–Beltrami operator, and t represents time. As triangle mesh can be thought of as a discretization of a continuous surface, the diffusion equation for a mesh at a vertex v can be written as

$$f_{t+1}(v) = f_t(v) + \Delta f_t(v). \tag{2}$$

When f is the coordinate of the vertex in 3D space, (2) is an iterative Laplacian smoothing process (Taubin, 1995). After a number of time steps, we obtain a smoothed version of the original mesh. In this work, we hypothesize that after this forward process, a fully clothed human mesh turns into a smooth SMPL body. Thus, given images of the mesh before being smoothed and a naked SMPL model fitted to the images, we seek to reverse this smoothing diffusion to deform the SMPL mesh into the original fully clothed human that matches the description in the images. Due to the difficulty in reversing this process in a closed-form solution, we try to leverage the sparse input images for the reversion. In this paper, we explicitly model this reverse process step by step using a *GraphLSTM*, which is essentially an LSTM (Hochreiter and Schmidhuber, 1997) but works on graph signals. Instead of reversing each step in 3D directly, we propose to map the mesh into a high-dimensional space and perform the reversion in this feature space. It is expected that the reverse step may be easier here because non-linear relationships can be flattened in higher dimension.

A schematic view of our proposed GraphLSTM is demonstrated in Fig. 3. Similar to a typical LSTM, at every time step t, GraphLSTM controls the memory using four gates: input i_t, forget r_t, cell c_t, and output o_t. To adapt an LSTM to graph signals, we propose to replace several dot products with geometrical GCs (Kipf and Welling, 2017) as can be seen in Fig. 3. Specifically, we keep the dot product for the input features but use GC for the hidden states. Instead of being a typical

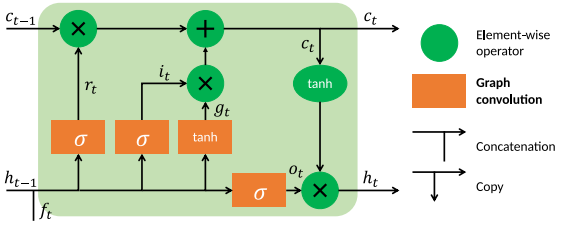


Fig. 3. An overview of the internal structure of our proposed GraphLSTM. σ is a sigmoid activation function.

sequence-to-sequence mapping, our GraphLSTM recursively processes the features as

$$\begin{aligned}
i_t &= \sigma(\mathrm{FC}(f_t) + \mathrm{GC}(h_{t-1})) \\
r_t &= \sigma(\mathrm{FC}(f_t) + \mathrm{GC}(h_{t-1})) \\
o_t &= \sigma(\mathrm{FC}(f_t) + \mathrm{GC}(h_{t-1})) \\
g_t &= \tanh(\mathrm{FC}(f_t) + \mathrm{GC}(h_{t-1})) \\
c_t &= r_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t),
\end{aligned} \tag{3}$$

where FC is a fully connected (FC) layer, and GC is simply an FC layer followed by a multiplication with the adjacency matrix, σ is the sigmoid function, and $\odot$ is the Hadamard product. f is the extracted feature of vertex v and the detail is presented in Section 3.4, and h_{-1} is zero. For each recursion step, we supply a shortcut from the input via $\mathrm{FC}(f)$. After one time step, h_t and c_t are passed to the next time step for the recursion. Thanks to the control flow of LSTM, our model can attend to different time steps of the diffusion in a non-linear fashion to capture the long-range dependencies between geodesically far vertices, thus the vertices can get better updates in each diffusion step.

The recursive solution can also be modeled by simply stacking multiple GC layers. However, training deep GCN is hard (Rong et al., 2020; Chen et al., 2020), and may require much memory. By using a GraphLSTM in a recursive manner, we can not only control the information flow at a node but also save up the memory footprint. On the other hand, one might be tempted to use a single GC layer and recurse multiple times, which is similar to Klicpera et al. (2019). This defines a linear combination of different diffusion steps. In our case, the diffusion is controlled by a GraphLSTM, which has a non-linear cell state, so it can better model any non-linear relationship between different time steps.

After the features are diffused over the mesh, we predict per-vertex offsets using an FC layer without any non-linearity. However, GC is a local operator and it has been shown in our previous work (Nguyen et al., 2019) that it is necessary to model the relationship between all vertices. Therefore, before the final layer, we max-pool the features over all vertices and concatenate with the features for offset prediction.

In order to reverse the diffusion process with GraphLSTM and predict vertex offsets, we need to extract features for the vertices of the SMPL model from the input images. In the following sections, we devise a novel strategy for feature extraction.

3.2.1. GraphLSTM and heat diffusion

We introduce GraphLSTM as an attempt to reverse the smoothing process, which we assumed to be a simple isotropic heat diffusion (or Laplacian smoothing in the discrete case) over the surface. Here, we consider two aspects of the diffusion process. We first look at how diffusion at a single time step is modeled by the GraphLSTM, and then investigate how the diffusion behaves over a time period. Considering an infinitesimal time interval, or a single time step in the discrete case, because GraphLSTM employs GC (Kipf and Welling, 2017) which diffuses information symmetrically from a node to its neighbors, GraphLSTM is a spatially isotropic diffusion operator. In that

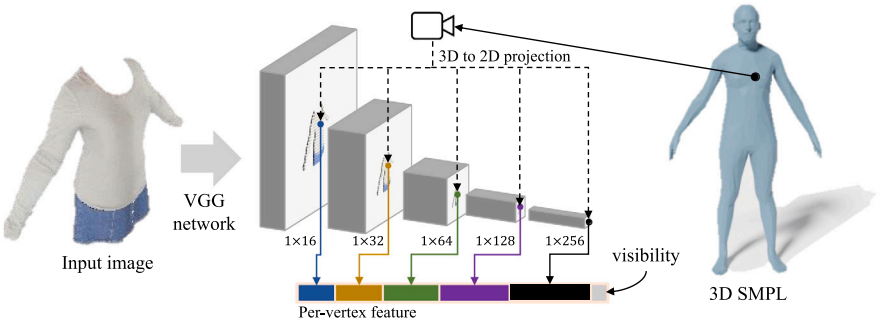


Fig. 4. Our feature extraction pipeline for one view. First, multi-scale 2D feature maps are extracted by using a VGG network (Simonyan and Zisserman, 2015). Then, the SMPL mesh is back-projected onto the feature maps to obtain multi-scale feature vectors for each vertex. The per-vertex feature vectors are concatenated together with an indicator of whether the vertex is visible in the image.

sense, GraphLSTM is able to simulate the heat equation correctly. On the other hand, the diffusion process over a long time period simulated by the recursion of GraphLSTM is not equivalent to the heat equation. While heat diffusion is a memoryless process in which the output of the current state depends only on the current state, GraphLSTM output is heavily affected by previous states of the graph because of the various gates inside an LSTM. In other words, the diffusion at a node can be adjusted to be fast or slow at different times rather than solely depending on the function surface, as can be seen in Fig. 7. From that point of view, GraphLSTM can be regarded as a temporally anisotropic diffusion operator. It is important as the true smoothing process is likely to be more sophisticated than simple heat diffusion and can involve non-linear processes. Therefore, a temporally non-linear weighted average operator like GraphLSTM is necessary to model such a sophisticated process.

3.3. Image features

We use an architecture similar to the popular VGG network (Simonyan and Zisserman, 2015), to extract features from input images as shown in Fig. 4. The network consists of five blocks. The dimension of the first block is 16, and the dimension of each block is twice of that of the previous one. To reduce computation, we downsample the feature maps by two in both height and width dimensions. In the same spirit as VGG, all learnable filters have a size of 3×3.

3.4. SMPL mesh features

3.4.1. Per-vertex features

The extracted 2D features contain rich shape features, so it is desired to transfer them to the fitted SMPL vertices (Loper et al., 2015). The feature transfer pipeline for one view is illustrated in Fig. 4. We project each vertex of the SMPL mesh onto the 2D image plane and calculate the corresponding feature vector. Concretely, we query the nearest corners to a projected point in the 2D grid and use bilinear interpolation to resample the feature. Let $\mathcal{M} = (\mathcal{V}, F)$ be the fitted SMPL mesh with a set of triangular faces F and a set of vertices $\mathcal{V}$. Also, let $F^b(\cdot)$ be the feature vector of a pixel at block b from VGG-lite. The feature for a vertex $v \in \mathcal{V}$ can be computed as

$$\begin{aligned} f^b_{\text{per-vertex}}(v) = {} & w_{00} F^b\left(\lfloor \pi[v; F^b] \rfloor\right) \\ & + w_{10} F^b\left(\lfloor \pi[v; F^b] \rfloor + (1,0)\right) \\ & + w_{01} F^b\left(\lfloor \pi[v; F^b] \rfloor + (0,1)\right) \\ & + w_{11} F^b\left(\lfloor \pi[v; F^b] \rfloor + (1,1)\right), \end{aligned} \tag{4}$$

where $\pi[v; F^b]$ projects v on to the image grid of F^b, $\lfloor \cdot \rfloor$ floors the tuple of projected coordinates to the nearest smaller integer coordinates, and w_{ij}'s are the interpolating coefficients. We compute this projection feature using several scales of the feature maps. In our implementation, the per-vertex feature is defined as

$$f_{\text{per-vertex}}(v) = \left[\{f^b_{\text{per-vertex}}(v)\}_{b=\{1,2,3,4,5\}}\right], \tag{5}$$

where $[\cdot]$ concatenates all the given feature vectors.

The projection feature establishes a direct correspondence between 3D points and 2D input images. This feature is meaningful in several ways. First, it can act as an indicator of whether a projected point is inside the human silhouette. Thus, the network can learn how to move points in order to cover the silhouette. Also, because wrinkles are represented by edges in a 2D image, the network must learn how to deform the surface accordingly. Of course, edges can also be formed by textures, but the disentanglement of these edge types can be trained via supervision.

3.4.2. Vertex visibility

Projecting all points onto an image plane discounts the fact that only certain vertices are visible in the view frustum. Therefore, it is necessary to supply a binary visibility indicator for every point in $\mathcal{M}$ so that the network can determine the reliability of the vertex features. To this end, we first rasterize a depth image $D_{\mathcal{M}}$ of $\mathcal{M}$ using the camera parameters. Next, we bring all vertices into the image space via the same camera parameters while keeping the depths z_v in camera space. We should expect the visible vertices to have similar depth values to those of the nearest pixels in the depth image. Hence, we derive the visibility indicator for a vertex as

$$f_{\text{vis}}(v) = \begin{cases} 1 & \text{if } |D_{\mathcal{M}}(\pi[v; D_{\mathcal{M}}]) - z_v| < \epsilon \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

where ϵ is a small number.

3.4.3. Vertex feature

After calculating the per-vertex and global features, we extract the feature vector for every vertex as

$$f(v) = \text{FC}\left(\left[f_{\text{coo}}(v), f_{\text{per-vertex}}(v), f_{\text{vis}}(v)\right]\right), \tag{7}$$

where $f_{\text{coo}}(v)$ is the 3D coordinate of v. The FC operator mixes these features together in order to prepare for multi-view aggregation.

3.4.4. Multi-view feature aggregation

As the network can accept a number of garment images as input, it is necessary to aggregate the features across all views. At the same time, we would like DReD to be able to handle an arbitrary number of views, so the aggregation method needs to be agnostic to the number of input images. In this work, we simply apply a max pooling operator. In our experiments, we found that max pooling is the most robust way and it achieves decent performance while making the network agnostic to the number of input images. We use the aggregated features as input to the feature diffusion process described in Section 3.2.

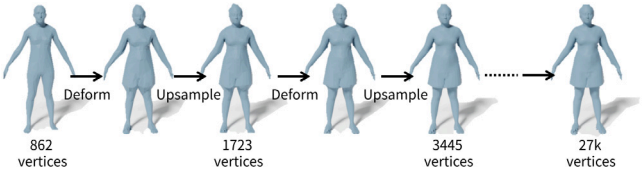


Fig. 5. Mesh deformation can be recursively applied to multiple scales of the SMPL mesh, starting with a mesh of 862 vertices and outputting $27k$ vertices.

3.5. Mesh deformation via Double Reverse Diffusion

3.5.1. Double Reverse Diffusion

It can be noticed that the reverse diffusion via GraphLSTM is agnostic to the resolution of the SMPL mesh, so it can be applied repeatedly to multiple scales of SMPL. This procedure can also be seen as reversing the Laplacian smoothing in the 3D Euclidean space, and hence is another reverse diffusion, which explains the name of our method. Fig. 5 presents an unrolled view of the diffusion in 3D space. To acquire multiple resolutions of the SMPL model, starting from the standard one with 6890 vertices, we sub-divide the triangles to get a high resolution mesh of 27k vertices, which is the highest resolution. To obtain coarser meshes, following Defferrard et al. (2016b), we apply the coarsening phase of the Graclus algorithm (Dhillon et al., 2007) several times. Given the lowest-resolution mesh, we first predict coarse offsets via GraphLSTM and then multiply the deformed mesh with a sparse upsampling matrix. Then, the upsampled deformed mesh is used as input to predict the offsets again. We perform this recursion for five SMPL scales in our implementation. Except for the lowest resolution, we also concatenate the interpolated GCN features of the mesh in the previous scale to the vertex features at the current scale as

$$f_n(v) = \mathrm{FC}\left(\left[f_{\mathrm{coo}}(v), f_{\text{per-vertex}}(v), f_{\mathrm{vis}}(v), f_{n-1}^{\mathrm{up}}(v)\right]\right), \tag{8}$$

where $f_{n-1}^{\mathrm{up}}(v)$ is the upsampled feature of the lower-resolution mesh, n ranges from 1 to 5 and $f_0 = 0$ for all v. We note that the GraphLSTM is shared among all scales, enabling a significant reduction in number of parameters.

The algorithm for the proposed Double Reverse Diffusion can be summarized and represented as shown in the table below.

Algorithm 1: Double Reverse Diffusion for 3D Mesh Generation

Require: Double Reverse Diffusion
Input : Multi-view images, body information (SMPL)
Output : Detailed 3D mesh of garments

for *MeshDiffusionStep* = 1 **to** N **do**
  Extract multi-scale features from multi-view images in Eqs. (4), (5)
  Vertex visibility check in Eq. (6)
  Extract vertex feature $f(v)$ in Eq. (7)
  Feature aggregation across all views with max-pooling
  for *FeatureDiffusionStep* = 1 **to** T **do**
    Vertex feature diffusion using GraphLSTM in Eq. (3)
  Deform 3D mesh with given feature per vertex
  Upsample 3D mesh

3.5.2. Implementation details

In practice, to strike a balance between expressivity and diffusion depth, we use three GraphLSTMs in our DReD. After each recursion in a GraphLSTM, the hidden and cell states are passed to the next GraphLSTM. Thus, if n steps are run for each GraphLSTM, the total number of recursions is $3n$. Also, low-resolution meshes may not require as many recursions as high-resolution ones. Thus, we set the number of recursions different for the five scales of SMPL (see Table 1). After the feature extraction step, before passing to GraphLSTM, we also use a residual graph convolutional block (Wang et al., 2020). This also adds more computing power to the network and improves performance.

4. Training DReD

4.1. Losses

We trained the network using the following loss function

$$\begin{aligned}\mathcal{L} =& \lambda_{\mathrm{sup}}\mathcal{L}_{\mathrm{sup}} + \lambda_{\mathrm{CD}}\mathcal{L}_{\mathrm{CD}} + \lambda_{\mathrm{detail}}\mathcal{L}_{\mathrm{detail}} + \lambda_{\mathrm{silhouette}}\mathcal{L}_{\mathrm{silhouette}} \\ &+ \lambda_{\mathrm{normals}}\mathcal{L}_{\mathrm{normals}} + \lambda_{\mathrm{boundary}}\mathcal{L}_{\mathrm{boundary}}\end{aligned}, \tag{9}$$

where $\{\lambda\}$'s are the loss weights. We provide more details on the losses in the following sections.

4.1.1. Supervised loss

We directly minimize the Lp distance between the predicted deformed mesh vertices $\mathcal{V}_{\mathrm{pred}}$ and the fully clothed ground truth mesh vertices of SMPL $\mathcal{V}_{\mathrm{gt}}$. Formally, the supervised loss is defined as

$$\mathcal{L}_{\mathrm{sup}} = \frac{1}{|\mathcal{V}_{\mathrm{pred}}|} \sum_{x\in\mathcal{V}_{\mathrm{pred}}, y\in\mathcal{V}_{\mathrm{gt}}} \|x-y\|_p^p. \tag{10}$$

Through experiments, we found that $p = 1$ can best recover sharp wrinkles in garments, so it is our default setting.

4.1.2. Scan loss

As scan meshes contain finer details than the ground truth SMPL models, it is beneficial to learn the details from them so as to capture high-frequency details. We use Chamfer distance (CD) to measure the discrepancy between a predicted mesh and a scan. For the sake of completeness, we present the CD measure below

$$\begin{aligned}\mathcal{L}_{\mathrm{CD}}(\mathcal{V}_{\mathrm{pred}}, \mathcal{V}_{\mathrm{scan}}) =& \frac{1}{\left|\mathcal{V}_{\mathrm{pred}}\right|} \sum_{x\in\mathcal{V}_{\mathrm{pred}}} \min_{y\in\mathcal{V}_{\mathrm{scan}}} \|x-y\|_2^2 \\ &+ \frac{1}{|\mathcal{V}_{\mathrm{scan}}|} \sum_{y\in\mathcal{V}_{\mathrm{scan}}} \min_{x\in\mathcal{V}_{\mathrm{pred}}} \|y-x\|_2^2\end{aligned}, \tag{11}$$

where $\mathcal{V}_{\mathrm{scan}}$ denotes the scan mesh vertices.

4.1.3. Data loss

We found that despite the supervised and CD losses, the model still hesitates to generate sharp wrinkles. To encourage the network, we propose to use a detail loss that focuses on fine details. Let us denote the normal vector at a vertex v by $n(v)$. For every $x \in \mathcal{V}_{\mathrm{pred}}$ and $y \in \mathcal{V}_{\mathrm{scan}}$, we define the following two error terms

$$\begin{aligned}l_{\mathrm{pred}}(x) &= \left\|x-y^*\right\|_2^2 \cdot \mathbb{1}\left[n(x)\cdot n(y^*) > \delta\right] \\ y^* &= \mathrm{argmin}_{y\in\mathcal{V}_{\mathrm{scan}}} \|x-y\|_2^2\end{aligned}, \tag{12}$$

and

$$\begin{aligned}l_{\mathrm{scan}}(y) &= \left\|y-x^*\right\|_2^2 \cdot \mathbb{1}\left[n(x^*)\cdot n(y) > \delta\right] \\ x^* &= \mathrm{argmin}_{x\in\mathcal{V}_{\mathrm{pred}}} \|y-x\|_2^2\end{aligned}, \tag{13}$$

where $\mathbb{1}[\cdot]$ is an indicator function, $\cdot$ the vector dot product, and δ a threshold. Then, the detail loss is computed based on a Geman-McClure function (Barron, 2019; Pons-Moll et al., 2017) as follows

$$\mathcal{L}_{\mathrm{detail}} = \sum_{x\in\mathcal{V}_{\mathrm{pred}}} \frac{l_{\mathrm{pred}}(x)}{l_{\mathrm{pred}}(x)+\sigma^2} + \sum_{y\in\mathcal{V}_{\mathrm{scan}}} \frac{l_{\mathrm{scan}}(y)}{l_{\mathrm{scan}}(y)+\sigma^2}, \tag{14}$$

where σ defines a threshold for outlier. Intuitively, this loss targets vertices having similar neighborhoods as ground truth.

4.1.4. Silhouette loss

To better establish correspondences between the predicted mesh and the input images, we make use of a loss function between the silhouette of the deformed SMPL S^i and the binary mask M^i of garments in the ith view. To make the silhouette differentiable with respect to the vertex positions, we render the mesh with white color and without shading using a differentiable renderer (Ravi et al., 2020). We render the silhouettes using the same intrinsics and extrinsics as the input views. The silhouette loss is calculated as

$$\mathcal{L}_{\text{silhouette}} = \frac{1}{N}\sum_{i=1}^{N} \|S^i - M^i\|_2^2, \tag{15}$$

where N is the number of input views.

4.1.5. Normal loss

We also enforce the similarity of normals between the predictions and the ground truths via a cosine similarity as

$$\mathcal{L}_{\text{normals}} = \frac{1}{|\mathcal{V}_{\text{pred}}|}\sum_{x\in\mathcal{V}_{\text{pred}}, y\in\mathcal{V}_{\text{gt}}} \left(1 - (n(x)\cdot n(y))^2\right), \tag{16}$$

where "$\cdot$" is the vector dot product.

4.1.6. Boundary loss

As we have segmentation of garments, we define the boundary between different classes to be the vertices that have at least one neighbor, not from the same class. The boundaries can be found by running a breadth-first-search on the labeled meshes. We would like the network to be aware of the borders between different pieces of clothes even without segmentation. Let us denote $\mathcal{B}_i^j$ for boundary i of label j. The boundary loss is defined as

$$\mathcal{L}_{\text{boundary}} = \sum_{i,j}\frac{1}{|\mathcal{B}_i^j|}\sum_{\substack{x\in\mathcal{V}_{\text{pred}}, y\in\mathcal{V}_{\text{gt}} \\ x,y\in\mathcal{B}_i^j}} \|x-y\|_2^2. \tag{17}$$

4.2. Dataset

We used the Sizer dataset (Tiwari et al., 2020) as it contains scans, scan textures, scan segmentations, registered SMPL models, and color images. This dataset contains basic clothing types such as T-shirts, long/short-sleeved shirts, pants, shorts, hoodies, and so on. All people are captured in A-pose. We rendered sixty views per scan into RGB images and segmentation maps. We randomly divided the dataset into training and test sets based on personal identity. We then trained on the training portion and tested on the rest. However, the Sizer dataset contains only A poses and clothes are repetitive. Therefore, to test the generalizability of the method, we recruited the MGN dataset (Bhatnagar et al., 2019) which is similar to Sizer but it contains many diverse poses in addition to the A-pose and both the garment form and textures are unique and more diverse.

4.3. Input data

In our experiments, for both datasets, we used four views if not mentioned otherwise. In an ablation study in which we reconstruct with an arbitrary number of views, each batch randomly includes one to four views. We used an image resolution of 512 × 512. We applied a garment segmentation mask obtained by QANet (Yang et al., 2022) to the input image to remove all identity information, preventing the network from memorizing the faces, which can improve the generalizability. Note that during inference, only background subtraction is needed for the input images. To overcome the repetition of garments, we augmented the data by modulating the brightness and saturation of the input images. This helps the network much less prone to overfitting.

Table 1
Numerical results when using different backbones for DReD. The best two numbers are highlighted in boldface.

Backbone	#params	Recursion pattern	L_{sup} ($\times 1e^{-2}$)	CD ($\times 1e^{-3}$)
GCN	4.25M	1, 1, 1, 1, 1	0.724	**0.217**
GraphResBlock	4.25M	1, 1, 1, 1, 1	0.701	0.232
	3.96M[a]	1, 1, 1, 2, 2	0.751	0.255
	3.96M[a]	1, 1, 2, 2, 3	**0.681**	0.233
GraphLSTM	4.25M	1, 1, 1, 1, 1	0.705	0.251
	4.25M	1, 1, 1, 2, 2	0.707	0.220
	4.25M	1, 1, 2, 2, 3	**0.680**	**0.214**
GraphLSTM (w/o EMA)	4.25M	1, 1, 2, 2, 3	0.693	0.220

[a] Cannot increase due to memory constraint.

4.4. Training details

We trained the network with an AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 1e−4. A weight decay term was set to 1e − 6. The batch size was 2 and the experiments ran for 300 epochs, which takes about 4 days on an NVIDIA 2080 RTX. Additionally, we applied an exponential moving average (EMA) scheme to the model weights (Izmailov et al., 2018). We updated the EMA every iteration with a decay of 0.999. We found that the model with EMA weights is marginally better in terms of quantitative metrics.

5. Experimental results

5.1. Evaluation metrics

To assess the performance of our network, we measure the discrepancy between the predictions and both the scans and ground truth SMPLs. Because the prediction itself is an SMPL, we simply use L_{sup}, which is L1 distance, between the prediction and the ground truth SMPL. Also, we measure CD between the predicted mesh and the ground truth scan as they have different number of vertices.

5.2. Ablation study and analysis

In this section, we perform a thorough assessment of each component in our system to make sure that all choices are optimal. When conducting the ablation study, all training details were maintained with the same settings as described in Section 4.4.

5.2.1. Diffusion backbone

Firstly, we validate the choice of GraphLSTM as a diffusion unit. We compare our GraphLSTM with a simple cascade of GC layers and a series of residual GC blocks (Wang et al., 2020) with or without recursion. To compare numerically, we recruit L_{sup} and CD. While L_{sup} compares the prediction with ground truth SMPL, which covers the low- and mid-frequencies, CD compares with ground truth scan which contains high-frequency components. As can be seen in Table 1, a simple cascade of GC layers does not produce good results, reflected by the high errors, especially for L_{sup}. Employing residual GC blocks (GraphResBlock) (Wang et al., 2020) helps reduce the errors, especially the supervised loss. Out of the three backbones, our GraphLSTM achieves the best supervised loss and CD. In terms of memory, GraphLSTM has better memory utilization as GraphResBlock cannot perform recursion with the same number of parameters as GraphLSTM if we are to keep all the training details the same. Another interesting observation is that the performance gets better as more recursion loops are added for the GraphLSTM backbone. Though not as clear, this observation also applies to GraphResBlock. In short, the choice of GraphLSTM with recursion brings the best performance in our experiments, and hence will be our default choice if not otherwise mentioned.

Furthermore, we investigate the use of EMA. As seen in the last row of Table 1, using only the trained weights marginally decreases the numerical results. Therefore, we keep using EMA in all our experiments.

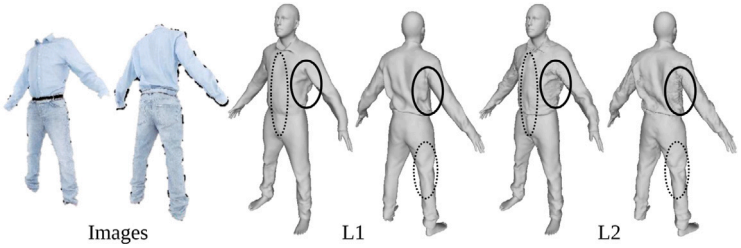


Fig. 6. Qualitative comparison between L1 and L2 supervised losses. L2 loss makes the network produce more details as shown in the dotted oval, but at the same time induces more noise as can be seen in the solid oval regions.

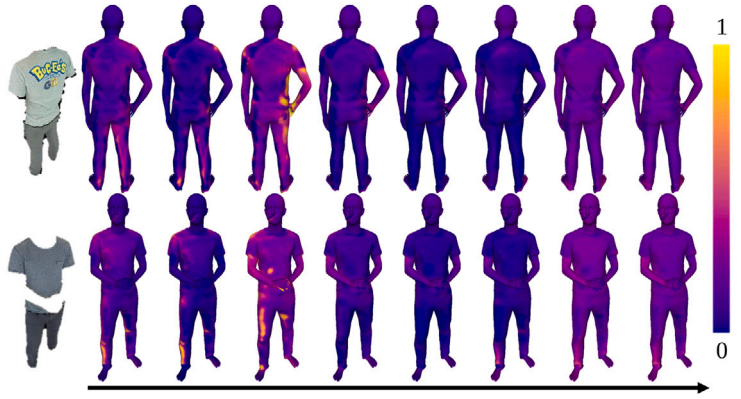


Fig. 7. Visualization of the forget gate activations in the final scale of SMPL. We recurse three times for each of the three GraphLSTM cells, which constitutes eight forget gate activations. We average the channels of each activation and display using the colormap shown on the right. The topological propagation of information is indicated by the black arrow.

Table 2
Ablation study on the visibility features and data augmentation. "-" indicates the ablated component. Best numbers are highlighted in boldface.

	-Vertex visibility	Visibility pooling	-Augmentation	Full
L_{sup}	0.725	0.728	0.716	**0.680**
CD	0.238	0.245	0.236	**0.214**

Table 3
Ablation study on the loss function components. "-" indicates the ablated component. Best numbers are highlighted in boldface.

	$-\mathcal{L}_{sup}$	$-\mathcal{L}_{CD}$	$-\mathcal{L}_{data}$	$-\mathcal{L}_{silhouette}$	$-\mathcal{L}_{normals}$	L2	Full
L_{sup}	1.060	0.693	0.717	0.719	0.703	N/A	**0.680**
CD	0.224	0.227	0.227	0.250	0.219	0.201	**0.214**

5.2.2. Features

Secondly, we shed light on the impact of the visibility features on the final results. As can be seen from Table 2, even though the visibility indicator is only a scalar, it has a huge influence on the final performance. Omitting this feature raises both the quantitative errors measured by $\mathcal{L}_{sup}$ and CD. Additionally, we perform multi-view pooling using the vertex visibility to compare with the max pooling used in our work. Concretely, given the visibility of vertices in different images, we calculate the average of the vertex features from only the views that the vertex is visible. For vertices that are not visible in any view, we simply take the mean of the features of visible vertices so that the network can reason about these points based on all visible points. The result of this visibility pooling is shown in Table 2. As can be seen, the errors are higher than those achieved by max pooling, thus max pooling is a better choice for multi-view feature aggregation in our case.

5.2.3. Loss functions

Next, we measure the importance of the losses by omitting each of these terms one by one from the total loss. We detail our findings in Table 3. As can be seen, the performance degrades without any of those losses. Training without either $\mathcal{L}_{sup}$ or CD results in higher error for the other distance. Each of the other losses also causes a huge decline in performance if it is omitted. Thus, each loss term has some certain contribution to the final performance.

We also tried replacing the L1 loss with L2 and as expected, the CD error is much smaller when training with L2. Because CD can capture high frequencies in the scans, it is expected that the predicted meshes would carry more details. Fig. 6 shows an example of the predicted meshes from two models trained with L1 and L2 losses. Even though the L2 results do possess more details as seen in the thin dotted regions, they also suffer from high-frequency spikes shown in the solid ovals. This is due to the fact that L2 loss value has a much smaller range than that of L1, so when training with L2, CD loss dominates in the total loss, which results in too much high frequency in the final outputs. It is also more difficult to tune suitable weights for the losses partly due to the correlation between L2 and CD, and we are satisfied with the current quality, so we leave further exploration for future work.

5.2.4. Importance of anonymity

In Alldieck et al. (2019) and Bhatnagar et al. (2019), the authors emphasized the importance of not leaking person identity because this may cause overfitting. We confirm this in our experiment, as training without covering the person identities leads to serious overfitting. However, the authors in Alldieck et al. (2019) and Bhatnagar et al. (2019) hid the identities by using segmentation maps as inputs, so the generated wrinkles are very limited. Meanwhile, we do so by masking all the skin regions and heavily rely on data augmentations during training, which effectively preserve 2D wrinkle information in the inputs, which results in much better performance. As shown in Table 2, excluding augmentation results in a huge loss in performance.

5.2.5. Visualization of forget gate activations

Forget gate is an important and intuitive gate in LSTM. Its role is to control the amount of past information propagated to the next

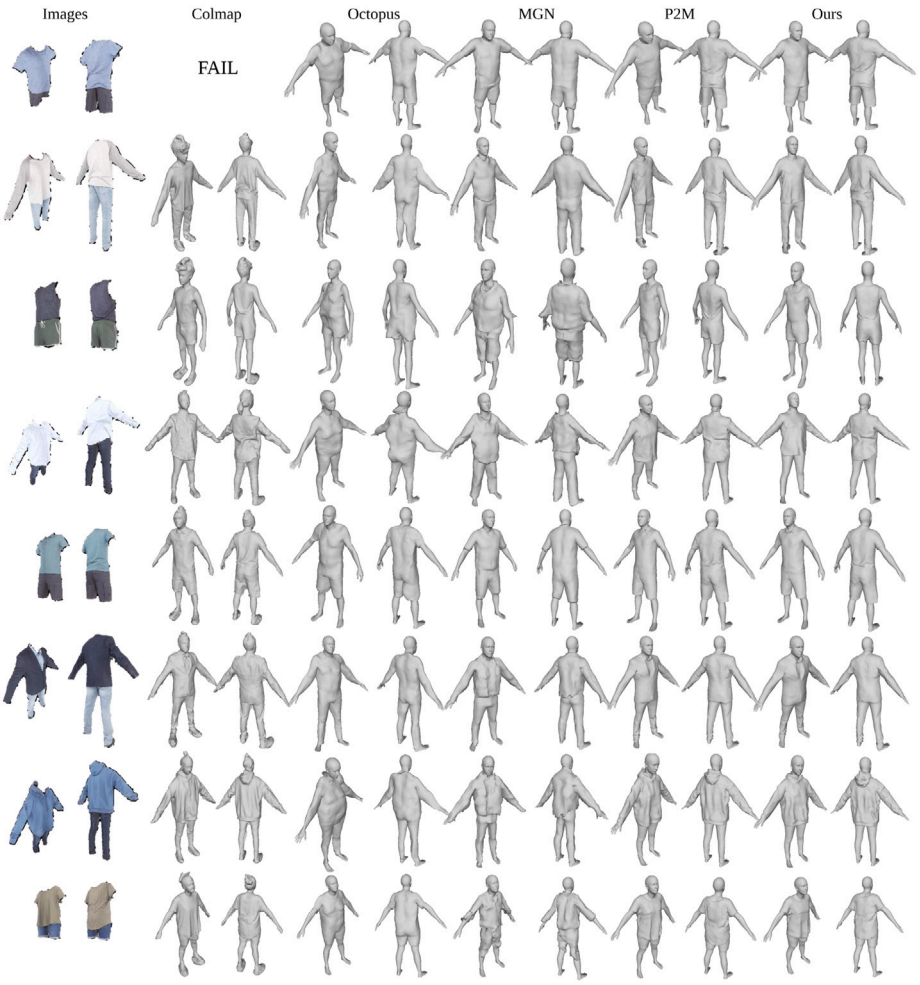


Fig. 8. Qualitative results our model against Colmap (Schonberger and Frahm, 2016), Octopus (Alldieck et al., 2019), MGN (Bhatnagar et al., 2019) and P2M (Wang et al., 2020). We show two of the four input images and the corresponding views of the 3D meshes.

layer. Here, we expect that each vertex diffuses information differently, which can be simulated via the forget gate. In order to verify our hypothesis, we visualize the forget gate activations in Fig. 7. We show eight activations resulting from three recursions of three GraphLSTM cells. These activations are extracted from the processing of the highest resolution of SMPL. We average the channels of each activation and apply a colormap for visualization. As can be seen in the figure, the forget gates are activated very differently among vertices. Also, the forget gates tend to be activated at the beginning and end of the recursion, with the strongest activations occurring in the first loop of the second GraphLSTM (the third activations in Fig. 7). Unfortunately, due to the highly abstract features learned by the network, it is difficult to explain and recognize any intuitive patterns from the figure. We leave this question for our future work.

5.3. Qualitative results

5.3.1. Sizer dataset

To qualitatively quantify our method, we compare DReD against Colmap (Schonberger and Frahm, 2016), Octopus (Alldieck et al., 2019), MGN (Bhatnagar et al., 2019) and P2M (Wang et al., 2020). Colmap (Schonberger and Frahm, 2016) perfected the traditional structure-from-motion pipeline and generally produces high-quality reconstructed dense point clouds from a set of images, which can be turned into meshes via Poisson reconstruction (Kazhdan et al., 2006). We used sixty views for Colmap reconstruction. Octopus (Alldieck et al., 2019) and MGN (Bhatnagar et al., 2019) are two deep learning-based methods that share a similar goal to ours. Octopus predicts poses and vertex offsets while MGN is a template-based method that predicts each garment detected in the input image. Both of them first predict rough garments using segmentation maps, and then optimize the output against silhouettes for more garment details. Finally, P2M (Wang et al., 2020) is a vertex deformation method that was originally proposed for simple object reconstruction such as chairs, tables, cars, and so on. We adapted P2M by simply reducing the number of parameters to be similar to ours and trained P2M from scratch using our same training recipe.

Some examples of the reconstructed meshes are shown in Fig. 8. Among all the methods, Colmap (Schonberger and Frahm, 2016) registers a near-ground-truth quality as the reconstructed garments are very realistic. The major problem with Colmap is that it is susceptible to unexpected failure due to the matching process and it may require to tune hyper-parameters for different scenes. Besides, because of relying on feature matching, it requires a large number of views for a dense reconstruction and also takes much time for the process. Meanwhile, our method needs only four views and its runtime is only a fraction of a second.

Other deep learning methods underperform Colmap and DReD. Octopus (Alldieck et al., 2019) and MGN (Bhatnagar et al., 2019) could not generate wrinkles due to the choice of using segmentation maps as input. Even the post-optimization cannot help because silhouettes cannot convey intricate deformations of garments. We argue that this is not an efficient strategy because to cover all wrinkles, there needs to have a significant number of silhouettes due to a high degree of

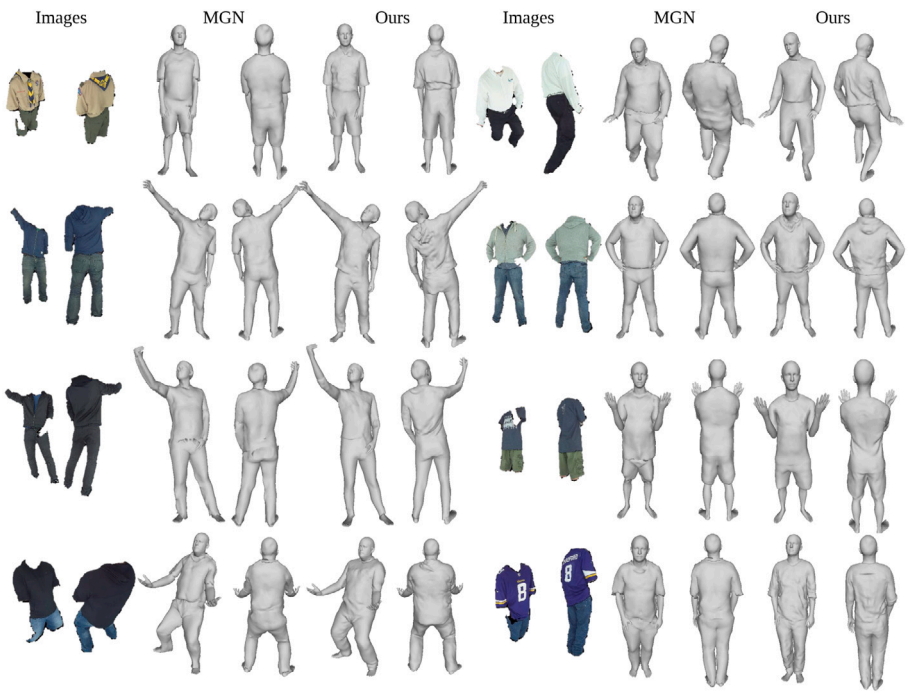


Fig. 9. Qualitative results on the MGN datasets (Bhatnagar et al., 2019) between MGN (Bhatnagar et al., 2019) and our proposed method. We show two of the four input images and the corresponding views of the 3D meshes.

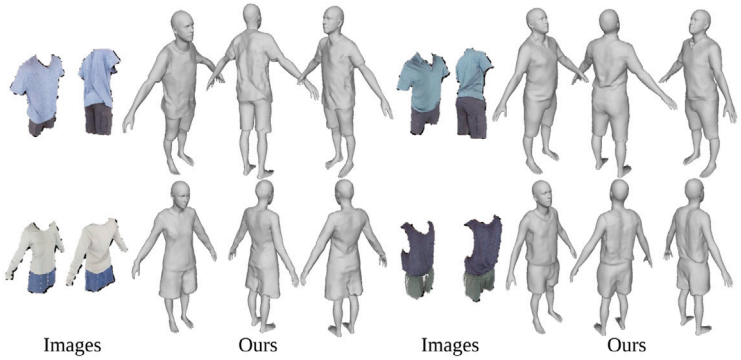


Fig. 10. Qualitative results of our model when testing on novel garment classes. We held out the t-shirt and tank-top classes for testing.

freedom. We resolve this problem by using RGB images but filter out the identities together with data augmentation. As a result, our reconstructed garments possess highly realistic wrinkles that faithfully follow those observed in the input images. Additionally, in the case of hoodie and tank-top, template-based MGN always fails as there is no corresponding template. Even in the case in which a template exists like the short pants in the last row, MGN still fails due to the size difference, while offset-based methods like Octopus and ours can handle it easily. Thus, template-based methods are not versatile enough to cover a wide variety of garments in practice. In addition, while optimization methods like Octopus, MGN or Colmap usually take around 3 min and even more, the inference time of our model is several order of magnitude faster, which is 0.945 s on a single RTX 2080Ti.

P2M (Wang et al., 2020) achieved good results as it can produce highly sharp wrinkles and is very competitive to ours. Nonetheless, the results are still noisy and wrinkles in some places are over-generated. Our method not only can reconstruct more accurate wrinkles but is also more memory-efficient than P2M.

5.3.2. Novel garment class

Sizer provides garment classification classes, for *e.g.*, shirt or t-shirt, so it is possible to conduct an experiment in which the network is trained on certain clothing classes and then tested on the rest. This experiment is challenging as it requires the networks to be able to generalize to unseen garment topologies. In our experiment, we trained DReD on all long top clothes including shirt, vest, hoodie, and so on, and held out the t-shirt and tank-top classes for test. The results of this experiment are shown in Fig. 10. It is clear that even though these clothes are new to the network, our method can still predict accurate clothing forms and reconstruct fine wrinkles on the garments. This experiment clearly shows a strong generalizability of our method.

5.3.3. Novel pose and diverse garment texture

The Sizer dataset contains only human models with A-pose and garments have very simple textures. A natural question is whether the method can generalize to unseen diverse poses and clothes. Hence, we resort to the MGN dataset (Bhatnagar et al., 2019) which has unique poses and clothes. Fig. 9 shows some results of MGN and ours on this method. As can be seen, MGN performed well in this case because it was trained on this dataset. However, the main problem with template-based models is still there as it is not possible to predict out-of-template garments as shown in the top left example. Meanwhile, DReD performed robustly throughout the whole dataset, be it an unusual pose

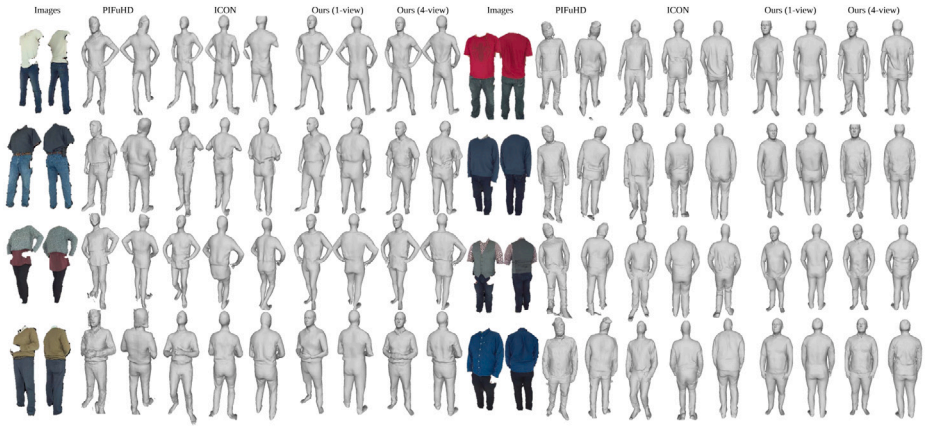


Fig. 11. Single-view garment reconstruction by PIFuHD (Saito et al., 2020), ICON (Xiu et al., 2022) and our proposed method. We used the images in the first column as inputs. For ICON, we additionally show the results when using back-view images. For our method, we additionally show the back view and the 4-view outputs as a reference.

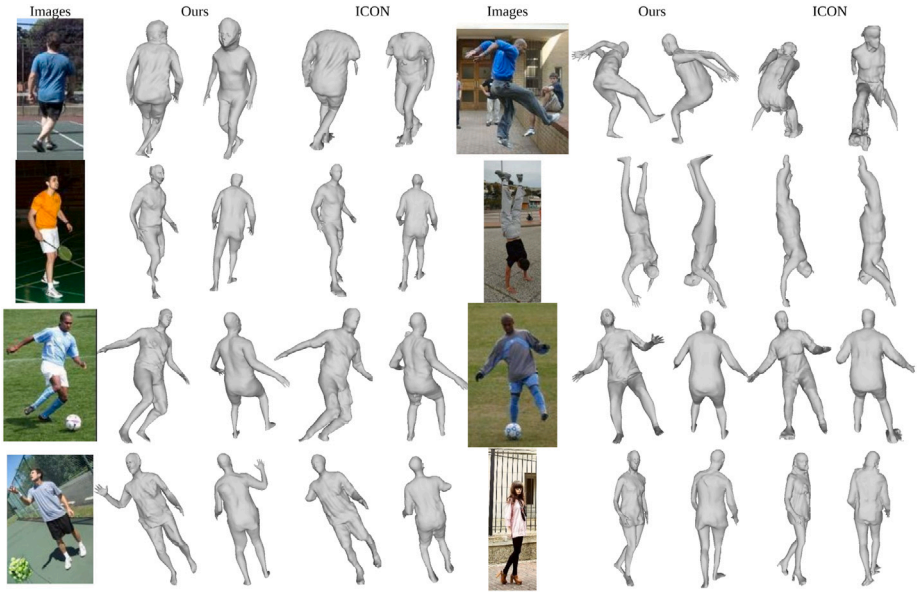


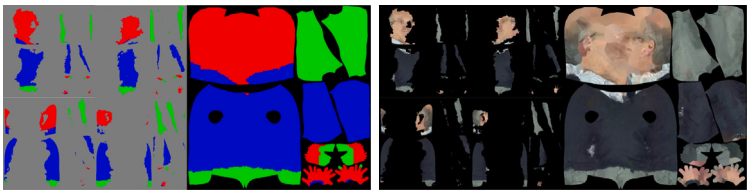
Fig. 12. Single-view garment reconstruction for in-the-wild images by ICON (Xiu et al., 2022) and our proposed method. We show both the visible and occluded sides of the reconstructed meshes.

or colorful textures. Interestingly, in the bottom right corner, it can be observed that the predicted mesh was deformed according to the number "8" on the t-shirt. This is attributed to limited data as the model has never encountered such difficult textures in the Sizer dataset. It is possible to disentangle actual wrinkles and textures by training on a dataset with more diverse clothes via 3D supervision. All in all, the results in this section strongly evidence that our method is capable of learning clothing deformations from multiple 2D images.

5.3.4. Single-view reconstruction

Despite learning to predict from multiple images, our method is flexible enough to be able to reconstruct from a single RGB thanks to the robust max-pooling. Because it is not easy to acquire multi-view images in real life, it is important to be able to reason from a single image. Thus, we test our method using a single image on the MGN dataset and compare our results with state-of-the-art single-view methods: PIFuHD (Saito et al., 2020) and ICON (Xiu et al., 2022). We used the ground truth SMPL pose and shape information for ICON. Some visualizations are compiled in Fig. 11. In this scenario, we used the first column of the images as input for all the methods. Additionally for ICON, we show their results with the back-view images in the third column of their results. Unlike PIFuHD, ICON is able to handle back-view images as it leverages a strong SMPL prior. For PIFuHD and our method, we also show the back-view for reference. In terms of wrinkles, PIFuHD performed well as their meshes contain many details. However, upon a closer look, it can be noticed that there are many artifacts and the meshes are generally not smooth. ICON seems to be more stable as it conditions on SMPL, but the results still lack fine details. Even though our meshes do not possess as many wrinkle details, overall they look less noisy thanks to the strong human prior powered by SMPL. Also, the meshes from PIFuHD have different topologies and cannot be re-purposed easily. On the other hand, our SMPL-based meshes have unified topology and can be readily re-animated thanks to the accompanied skinning weights. All in all, DReD is able to generalize well enough to predict even from a single view, making it more applicable in practice.

In addition, we perform an experiment on in-the-wild images to validate the generalizability of our method. For this purpose, we recruit the Up3D dataset (Lassner et al., 2017), which provides human-centric photos in daily life and fitted SMPL parameters. We show some of these results in Fig. 12. Again, we compare our method with ICON (Xiu et al., 2022). As can be seen from the figure, our results look comparable to those of ICON, but sometimes are not as good in terms of sharp wrinkles



(a) (b)

Fig. 13. (a) Partial and complete texture segmentation maps. (b) Partial and complete color texture maps.

Table 4
Quantitative results of our proposed method and P2M (Wang et al., 2020). Best numbers are highlighted in boldface.

Method	#params	L_{sup} ($\times 1e^{-2}$)	CD ($\times 1e^{-3}$)
P2M (Wang et al., 2020)	4.88M	0.805	0.307
DreD	4.25M	**0.680**	**0.214**

because ICON is trained on a much larger and more diverse dataset, and due to the slightly inaccurate SMPL fitting and the low quality of the image. Note that facial features may look distorted as DReD does not learn to deform skin regions. ICON has a post-optimization step, which can fix the imprecision to deliver better reconstruction quality in some cases, but at times it can back-fire heavily like the first two rows of the second column. Still, this experiment shows that our method has a good potential to generalize well despite not being engineered as a single-view method.

5.4. Quantitative results

Because Octopus (Alldieck et al., 2019) and MGN (Bhatnagar et al., 2019) predict view-dependent meshes, it is not easy to align their results to ours. We tried matching the bounding boxes of their results to the bounding boxes of the corresponding ground truths and then computed the L1 distance between the predictions and ground truths. We obtained the distance of 0.379 for MGN and 0.0068 for our method. We note that the large errors come from the fact that not only MGN predictions are imprecise, but also the alignment is highly inaccurate because the bounding box highly depends on the offsets of the mesh. Thus this is not a fair comparison to MGN. Therefore, we show only the numerical results of ours versus P2M (Wang et al., 2020) as tabulated in Table 4. As can be seen, DReD achieves much better performance than P2M. Note that P2M can also be seen as a form of diffusion, but they employ different networks in different SMPL scales. Meanwhile, our recursive diffusion can provide longer diffusion at lower memory footprint.

5.5. Textured mesh reconstruction

For completeness, we demonstrate the reconstruction of textured mesh. After obtaining deformed meshes, we follow Alldieck et al. (2018a) to stitch texture maps from the multi-view images. Concretely, first of all, partial textures are extracted by unwrapping each 2D view into a predefined SMPL UV space as shown in the left of Fig. 13(b). In addition, we unwrap the corresponding segmentation maps which consist of three labels: upper and lower garments, and everything else as can be seen in Fig. 13(a). Then, we solve a multi-label graph-cut optimization to fuse the partial texture segmentation maps. Using the same graph created above and the complete texture segmentation, we then merge the partial RGB textures and use a simple inpainting method (Telea, 2004) to complete the texture map as shown in the right of Fig. 13(b). We demonstrate some textured meshes obtained this way in Fig. 14. As can be seen, the textures are estimated well thanks to the precise deformation of the vertices. These results serve as proof that DReD is capable of making realistic 3D textured avatars, which enables many interesting applications in immersive contents.

5.6. Discussion and limitation

Through the rigorous experiments, we have shown that our DReD can reconstruct very detailed garments while being much faster compared to previous works that employ optimization. The reverse diffusion process predicts vertex deformation by conditioning on a large neighborhood, which makes the prediction more reasonable. As the diffusion is modeled by the recursion of an our proposed GraphLSTM, the physical memory can be saved up compared to methods that utilize large neural networks. Our method may not be as good as implicit ones at predicting sharp features as those methods enjoy more freedom in predicting output meshes, but they are susceptible to producing unreasonable garment deformation.

Nevertheless, our method suffers from several limitations. First, our method is not suitable for loose clothes and dresses, partly due to the lack of these types of garments in the training dataset. Second, it is difficult to deal with multiple layers of clothes. To resolve this problem, we can rely on the segmentation labels of different clothes and predict the corresponding offsets like MGN (Bhatnagar et al., 2019). Lastly, our method requires a good SMPL fitting. ICON, MGN and Octopus escape this pitfall by utilizing a post-optimization step. In the future, we can bake the finetuning of SMPL parameters into the model and train in a differentiable manner to correct the SMPL prediction on-the-fly.

6. Conclusion

We introduced a multi-view garment reconstruction solution named DReD. At the center of our solution lies an assumption that reconstruction is the inverse problem of smoothing, which is described by a forward diffusion process. Our framework proposed to reverse this smoothing process by simulating the reverse diffusion process via a recursion. Starting from a naked human body in the form of SMPL, we gradually diffuse the vertex features and predict per-vertex offsets to reconstruct a fully-clothed body whose clothes are described in the input images. In order to control the diffusion for mesh vertices, we introduced GraphLSTM which is a customized LSTM for graph signals. To get the features for the diffusion process, we extract features from the multi-view images and transfer the to the mesh vertices via reprojection. The same deformation pipeline can be repeatedly applied to multiple SMPL resolutions, which can be seen as another reverse diffusion process. Extensive experiments demonstrate that DReD can reconstruct garments with greater realism and detail compared to leading methods. Qualitatively, the results on detailed wrinkles are superior, and quantitatively, they achieve a reduced error rate of only 69% $\sim$ 84%. Moreover, DReD is able to generalize to unseen human poses and complex textured garments with satisfactory visual results. In the future, the success of our diffusion approach opens up new interesting application directions with other diffusion models, for e.g., DDPMs, including animating results for motion or ensuring temporal significance. Also, it is interesting to see how we can use a high-quality static scan model of the same human to improve the temporal prediction quality.

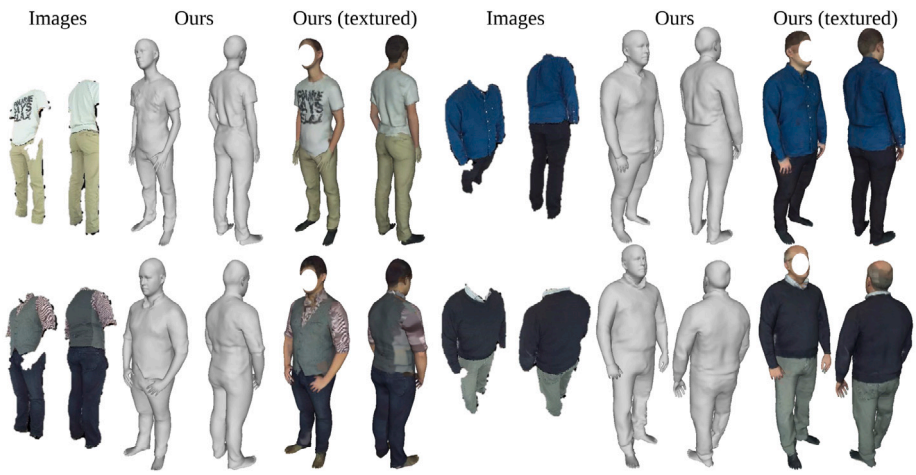


Fig. 14. Textured mesh reconstruction. First, clothed human meshes are reconstructed by DReD and then, texture maps are unwrapped via Alldieck et al. (2018a).

CRediT authorship contribution statement

Jeonghaeng Lee: Involved in the development of the method, Performed some experiments in this paper, Wrote the majority of the experimental section, Contributed to the majority of the reply letter as well as the revised version. **Duc Nguyen:** Proposed the idea in the paper, Designed and performed some experiments, Wrote the main body of the paper, Helped revise the reply letter. **Jongyoo Kim:** Provided high-level advices on the project, Briefly involved in the writing of the paper. **Jiwoo Kang:** Provided high-level advices on the project, Briefly involved in the writing of the paper. **Sanghoon Lee:** Supervised the whole project.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Sanghoon Lee reports financial support was provided by National Research Foundation of Korea.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1A2C3011697) and the Yonsei Signature Research Cluster Program of 2023 (2023-22-0008)

References

Alldieck, T., Magnor, M., Bhatnagar, B.L., Theobalt, C., Pons-Moll, G., 2019. Learning to reconstruct people in clothing from a single RGB camera. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2019-June. IEEE Computer Society, pp. 1175–1186. http://dx.doi.org/10.1109/CVPR.2019.00127, arXiv:1903.05885. URL https://arxiv.org/abs/1903.05885v2.

Alldieck, T., Magnor, M., Xu, W., Theobalt, C., Pons-Moll, G., 2018a. Detailed human avatars from monocular video. In: Proceedings - 2018 International Conference on 3D Vision, 3DV 2018. Institute of Electrical and Electronics Engineers Inc., pp. 98–109, arXiv:1808.01338. URL http://arxiv.org/abs/1808.01338.

Alldieck, T., Magnor, M., Xu, W., Theobalt, C., Pons-Moll, G., 2018b. Video based reconstruction of 3D people models. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 8387–8397. http://dx.doi.org/10.1109/CVPR.2018.00875, arXiv:1803.04758. URL https://arxiv.org/abs/1803.04758v3.

Barron, J.T., 2019. A general and adaptive robust loss function. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2019-June. IEEE Computer Society, pp. 4326–4334. http://dx.doi.org/10.1109/CVPR.2019.00446, arXiv:1701.03077. URL https://arxiv.org/abs/1701.03077v10.

Bertiche, H., Madadi, M., Tylson, E., Escalera, S., 2021. DeePSD: Automatic deep skinning and pose space deformation for 3D garment animation. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, pp. 5451–5460. http://dx.doi.org/10.1109/ICCV48922.2021.00542, arXiv:2009.02715. URL https://arxiv.org/abs/2009.02715v2https://ieeexplore.ieee.org/document/9710972/.

Bhatnagar, B., Tiwari, G., Theobalt, C., Pons-Moll, G., 2019. Multi-garment net: Learning to dress 3D people from images. In: Proceedings of the IEEE International Conference on Computer Vision, Vol. 2019-Octob. pp. 5419–5429. http://dx.doi.org/10.1109/ICCV.2019.00552, arXiv:1908.06903.

Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., Black, M.J., 2016. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In: European Conference on Computer Vision. Springer, pp. 561–578.

Cannon, J.R., 1984. The One-Dimensional Heat Equation, no. 23. Cambridge University Press.

Casado-Elvira, A., Trinidad, M.C., Casas, D., 2022. PERGAMO: Personalized 3D garments from monocular video. Comput. Graph. Forum 41 (8), 293–304. http://dx.doi.org/10.1111/cgf.14644, arXiv:2210.15040. URL https://onlinelibrary.wiley.com/doi/full/10.1111/cgf.14644https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14644https://onlinelibrary.wiley.com/doi/10.1111/cgf.14644http://arxiv.org/abs/2210.15040.

Chen, M., Wei, Z., Huang, Z., Ding, B., Li, Y., 2020. Simple and deep graph convolutional networks. In: 37th International Conference on Machine Learning, ICML 2020, Vol. PartF16814. International Machine Learning Society (IMLS), pp. 1703–1713. http://dx.doi.org/10.48550/arxiv.2007.02133, arXiv:2007.02133. URL https://arxiv.org/abs/2007.02133v1.

Defferrard, M., Bresson, X., Vandergheynst, P., 2016a. Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems. Neural information processing systems foundation, pp. 3844–3852. http://dx.doi.org/10.48550/arxiv.1606.09375, arXiv:1606.09375. URL https://arxiv.org/abs/1606.09375v3.

Defferrard, M., Bresson, X., Vandergheynst, P., 2016b. Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems. Neural information processing systems foundation, pp. 3844–3852, arXiv:1606.09375. URL http://arxiv.org/abs/1606.09375.

Dhariwal, P., Nichol, A., 2021. Diffusion models beat GANs on image synthesis. In: Advances in Neural Information Processing Systems. arXiv:2105.05233. URL http://arxiv.org/abs/2105.05233https://openreview.net/forum?id=AAWuCvzaVt.

Dhillon, I.S., Guan, Y., Kulis, B., 2007. Weighted graph cuts without eigenvectors a multilevel approach. IEEE Trans. Pattern Anal. Mach. Intell. 29 (11), 1944–1957. http://dx.doi.org/10.1109/TPAMI.2007.1115.

Fan, H., Su, H., Guibas, L., 2017. A point set generation network for 3D object reconstruction from a single image. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Vol. 2017-Janua. pp. 2463–2471. http://dx.doi.org/10.1109/CVPR.2017.264, arXiv:1612.00603. URL http://arxiv.org/abs/1612.00603.

Feng, Y., Wu, F., Shao, X., Wang, Y., Zhou, X., 2018. Joint 3d face reconstruction and dense alignment with position map regression network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 534–551.

Guan, P., Reiss, L., Hirshberg, D.A., Weiss, A., Black, M.J., 2012. DRAPE: Dressing any person. ACM Trans. Graph. 31 (4), http://dx.doi.org/10.1145/2185520.2185531, URL https://dl.acm.org/doi/abs/10.1145/2185520.2185531.

Gundogdu, E., Constantin, V., Seifoddini, A., Dang, M., Salzmann, M., Fua, P., 2019. GarNet: A two-stream network for fast and accurate 3D cloth draping. In: Proceedings of the IEEE International Conference on Computer Vision, Vol. 2019-Octob. Institute of Electrical and Electronics Engineers Inc., pp. 8738–8747. http://dx.doi.org/10.1109/ICCV.2019.00883, arXiv:1811.10983. URL https://arxiv.org/abs/1811.10983v3.

Guo, Q., Yu, Z., Wu, Y., Liang, D., Qin, H., Yan, J., 2019. Dynamic recursive neural network. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 5142–5151. http://dx.doi.org/10.1109/CVPR.2019.00529, URL https://ieeexplore.ieee.org/document/8954249/.

Habermann, M., Xu, W., Zollhofer, M., Pons-Moll, G., Theobalt, C., 2020. DeepCap: Monocular human performance capture using weak supervision. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 5051–5062. http://dx.doi.org/10.1109/CVPR42600.2020.00510, arXiv:2003.08325. URL https://arxiv.org/abs/2003.08325v1.

Hartley, R., Zisserman, A., 2004. Multiple View Geometry in Computer Vision. Cambridge University Press, http://dx.doi.org/10.1017/cbo9780511811685, URL https://www.cambridge.org/core/books/multiple-view-geometry-in-computer-vision/0B6F289C78B2B23F596CAA76D3D43F7A.

Ho, J., Jain, A., Abbeel, P., 2020. Denoising diffusion probabilistic models. Adv. Neural Inf. Process. Syst. 2020-December, arXiv:2006.11239.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735, URL https://direct.mit.edu/neco/article/9/8/1735-1780/6109.

Hong, F., Pan, L., Cai, Z., Liu, Z., 2021. Garment4D: Garment reconstruction from point cloud sequences. Adv. Neural Inf. Process. Syst. 33, 27940–27951, arXiv:2112.04159. URL https://github.com/hongfz16/Garment4D.

Huang, Z., Xu, Y., Lassner, C., Li, H., Tung, T., 2020. ARCH: Animatable reconstruction of clothed humans. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 3090–3099. http://dx.doi.org/10.1109/CVPR42600.2020.00316, arXiv:2004.04572. URL https://arxiv.org/abs/2004.04572v2.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G., 2018. Averaging weights leads to wider optima and better generalization. In: 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018, Vol. 2. Association For Uncertainty in Artificial Intelligence (AUAI), pp. 876–885, arXiv:1803.05407. URL http://arxiv.org/abs/1803.05407.

Jiang, B., Lin, D., Tang, J., Luo, B., 2019. Data representation and learning with graph diffusion-embedding networks. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2019-June. IEEE Computer Society, pp. 10406–10415. http://dx.doi.org/10.1109/CVPR.2019.01066.

Jiang, B., Zhang, J., Hong, Y., Luo, J., Liu, L., Bao, H., 2020. BCNet: Learning body and cloth shape from a single image. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). In: LNCS, vol. 12365, Springer Science and Business Media Deutschland GmbH, pp. 18–35. http://dx.doi.org/10.48550/arxiv.2004.00214, arXiv:2004.00214. URL https://arxiv.org/abs/2004.00214v2.

Kang, J., Lee, S., Lee, S., 2021. Competitive learning of facial fitting and synthesis using uv energy. IEEE Transactions on Systems, Man, and Cybernetics: Systems 52 (5), 2858–2873.

Karras, T., Aila, T., Laine, S., Lehtinen, J., 2018. Progressive growing of GANs for improved quality, stability, and variation. In: 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings. International Conference on Learning Representations, ICLR, http://dx.doi.org/10.48550/arxiv.1710.10196, arXiv:1710.10196. URL https://arxiv.org/abs/1710.10196v3.

Kazhdan, M., Bolitho, M., Hoppe, H., 2006. Poisson surface reconstruction. In: Eurographics Symposium on Geometry Processing (2006). URL http://hhoppe.com/poissonrecon.pdf.

Kemelmacher-Shlizerman, I., Basri, R., 2010. 3D face reconstruction from a single image using a single reference face shape. IEEE Trans. Pattern Anal. Mach. Intell. 33 (2), 394–405.

Kim, J., Lee, J.K., Lee, K.M., 2016. Deeply-recursive convolutional network for image super-resolution. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2016-Decem. pp. 1637–1645. http://dx.doi.org/10.1109/CVPR.2016.181, arXiv:1511.04491.

Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings. International Conference on Learning Representations, ICLR, arXiv:1609.02907. URL http://arxiv.org/abs/1609.02907.

Klicpera, J., Weißenberger, S., Günnemann, S., 2019. Diffusion improves graph learning. In: Advances in Neural Information Processing Systems, Vol. 32. Neural information processing systems foundation, http://dx.doi.org/10.48550/arxiv.1911.05485, arXiv:1911.05485. URL https://arxiv.org/abs/1911.05485v6.

Lähner, Z., Cremers, D., Tung, T., 2018. DeepWrinkles: Accurate and realistic clothing modeling. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). In: LNCS, vol. 11208, Springer Verlag, pp. 698–715. http://dx.doi.org/10.1007/978-3-030-01225-0_41, arXiv:1808.03417. URL https://arxiv.org/abs/1808.03417v1.

Lassner, C., Romero, J., Kiefel, M., Bogo, F., Black, M.J., Gehler, P.V., 2017. Unite the people: Closing the loop between 3D and 2D human representations. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). URL http://up.is.tuebingen.mpg.de.

Lee, K., Kim, W., Lee, S., 2022. From human pose similarity metric to 3d human pose estimator: temporal propagating lstm networks. IEEE transactions on pattern analysis and machine intelligence 45 (2), 1781–1797.

Liang, X., Shen, X., Feng, J., Lin, L., Yan, S., 2016. Semantic object parsing with graph LSTM. Lecture Notes in Comput. Sci. 125–143. http://dx.doi.org/10.1007/978-3-319-46448-0_8.

Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J., 2015. SMPL: a skinned multi-person linear model. ACM Trans. Graph. 34 (6), 1–16. http://dx.doi.org/10.1145/2816795.2818013, URL https://dl.acm.org/doi/10.1145/2816795.2818013.

Lorensen, W.E., Cline, H.E., 1987. Marching cubes: A high resolution 3D surface construction algorithm. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987, Vol. 21. Association for Computing Machinery, Inc, pp. 163–169. http://dx.doi.org/10.1145/37401.37422.

Loshchilov, I., Hutter, F., 2019. Decoupled weight decay regularization. In: 7th International Conference on Learning Representations.

Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R., 2020. Nerf: Representing scenes as neural radiance fields for view synthesis. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). In: LNCS, vol. 12346, Springer Science and Business Media Deutschland GmbH, pp. 405–421. http://dx.doi.org/10.1007/978-3-030-58452-8_24, arXiv:2003.08934.

Nguyen, A.D., Choi, S., Kim, W., Lee, S., 2019. GraphX-Convolution for point cloud deformation in 2D-to-3D conversion. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, pp. 8627–8636. http://dx.doi.org/10.1109/ICCV.2019.00872, arXiv:1911.06600. URL https://ieeexplore.ieee.org/document/9008115/.

Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S., 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2019-June. pp. 165–174. http://dx.doi.org/10.1109/CVPR.2019.00025, arXiv:1901.05103. URL http://arxiv.org/abs/1901.05103.

Patel, C., Liao, Z., Pons-Moll, G., 2020. TailorNet: Predicting clothing in 3D as a function of human pose, shape and garment style. pp. 7363–7373. http://dx.doi.org/10.1109/cvpr42600.2020.00739, arXiv:2003.04583. URL http://arxiv.org/abs/2003.04583.

Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., Zhou, X., 2021. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, pp. 9050–9059. http://dx.doi.org/10.1109/CVPR46437.2021.00894, arXiv:2012.15838. URL https://arxiv.org/abs/2012.15838v2.

Pons-Moll, G., Pujades, S., Hu, S., Black, M.J., 2017. ClothCap: Seamless 4D clothing capture and retargeting. In: ACM Transactions on Graphics, Vol. 36. ACM PUB27 New York, NY, USA, http://dx.doi.org/10.1145/3072959.3073711, URL https://dl.acm.org/doi/abs/10.1145/3072959.3073711.

Qiu, L., Chen, G., Zhou, J., Xu, M., Wang, J., Han, X., 2023. REC-MV: Reconstructing 3D dynamic cloth from monocular videos. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 4637–4646, URL https://github.com/GAP-LAB-CUHK-SZ/REC-MV.

Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.-Y., Johnson, J., Gkioxari, G., 2020. Accelerating 3D deep learning with PyTorch3D. arXiv:2007.08501.

Rong, Y., Huang, W., Xu, T., Huang, J., 2020. DropEdge: Towards deep graph convolutional networks on node classification. In: International Conference on Learning Representations. http://dx.doi.org/10.48550/arxiv.1907.10903, arXiv:1907.10903. URL http://arxiv.org/abs/1907.10903https://openreview.net/forum?id=Hkx1qkrKPr.

Saito, S., Simon, T., Saragih, J., Joo, H., 2020. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3D human digitization. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 81–90. http://dx.doi.org/10.1109/CVPR42600.2020.00016, arXiv:2004.00452. URL https://arxiv.org/abs/2004.00452v1https://ieeexplore.ieee.org/document/9157468/.

Schonberger, J.L., Frahm, J.-M.M., 2016. Structure-from-motion revisited. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2016-Decem. IEEE, pp. 4104–4113. http://dx.doi.org/10.1109/CVPR.2016.445, URL https://github.com/colmap/colmap.http://ieeexplore.ieee.org/document/7780814/.

Sharp, N., Attaiki, S., Crane, K., Ovsjanikov, M., 2022. DiffusionNet: Discretization agnostic learning on surfaces. ACM Trans. Graph. 41 (3), 1–16. http://dx.doi.org/10.1145/3507905, arXiv:2012.00888. URL http://arxiv.org/abs/2012.00888.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. arXiv:1409.1556.

Socher, R., Lin, C.C.Y., Ng, A.Y., Manning, C.D., 2011. Parsing natural scenes and natural language with recursive neural networks. In: Proceedings of the 28th International Conference on Machine Learning, ICML 2011. pp. 129–136, URL www.socher.org.

Sohl-Dickstein, J., Weiss, E.A., Maheswaranathan, N., Ganguli, S., 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In: 32nd International Conference on Machine Learning, ICML 2015, Vol. 3. International Machine Learning Society (IMLS), pp. 2246–2255, arXiv:1503.03585.

Song, J., Meng, C., Ermon, S., 2021. Denoising diffusion implicit models. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=St1giarCHLP.

Taubin, G., 1995. Estimating the tensor of curvature of a surface from a polyhedral approximation. In: IEEE International Conference on Computer Vision. pp. 902–907. http://dx.doi.org/10.1109/iccv.1995.466840.

Telea, A., 2004. An image inpainting technique based on the fast marching method. J. Graph. Tools 9 (1), 23–34. http://dx.doi.org/10.1080/10867651.2004.10487596, URL http://www.tandfonline.com/doi/abs/10.1080/10867651.2004.10487596.

Tiwari, G., Bhatnagar, B.L., Tung, T., Pons-Moll, G., 2020. SIZER: A dataset and model for parsing 3D clothing and learning size sensitive 3D clothing. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). In: LNCS, vol. 12348, Springer Science and Business Media Deutschland GmbH, pp. 1–18. http://dx.doi.org/10.1007/978-3-030-58580-8_1, arXiv:2007.11610. URL https://arxiv.org/abs/2007.11610v1.

Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W., 2021. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S., Vaughan, J.W. (Eds.), Advances in Neural Information Processing Systems, Vol. 34. Curran Associates, Inc., pp. 27171–27183. http://dx.doi.org/10.48550/arxiv.2106.10689, arXiv:2106.10689. URL http://arxiv.org/abs/2106.10689https://proceedings.neurips.cc/paper/2021/file/e41e164f7485ec4a28741a2d0ea41c74-Paper.pdf.

Wang, N., Zhang, Y., Li, Z., Fu, Y., Yu, H., Liu, W., Xue, X., Jiang, Y.-G., 2020. Pixel2Mesh: 3D mesh model generation via image guided deformation. IEEE Trans. Pattern Anal. Mach. Intell. 8828 (c), 1. http://dx.doi.org/10.1109/tpami.2020.2984232.

Wang, J., Zheng, V.W., Liu, Z., Chang, K.C.C., 2017. Topological recurrent neural network for diffusion prediction. In: Proceedings - IEEE International Conference on Data Mining, ICDM, Vol. 2017-Novem. Institute of Electrical and Electronics Engineers Inc., pp. 475–484. http://dx.doi.org/10.1109/ICDM.2017.57, arXiv:1711.10162. URL https://arxiv.org/abs/1711.10162v2.

Xiu, Y., Yang, J., Tzionas, D., Black, M.J., 2022. ICON: Implicit clothed humans obtained from normals. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 13286–13296. http://dx.doi.org/10.1109/CVPR52688.2022.01294, arXiv:2112.09127. URL https://arxiv.org/abs/2112.09127v2https://ieeexplore.ieee.org/document/9878790/.

Yang, L., Song, Q., Wang, Z., Liu, Z., Xu, S., Li, Z., 2022. Quality-aware network for human parsing. IEEE Trans. Multimed..

Zhang, K., Riegler, G., Snavely, N., Koltun, V., 2020. Nerf++: Analyzing and improving neural radiance fields. arXiv preprint arXiv:2010.07492.

Zheng, Z., Yu, T., Wei, Y., Dai, Q., Liu, Y., 2019. Deephuman: 3d human reconstruction from a single image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7739–7749.

Zou, X., Han, X., Wong, W., 2023. CLOTH4d: A dataset for clothed human reconstruction. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 12847–12857, URL www.github.com/AemikaChow/AiDLab-fAshIon-Data.