EUROGRAPHICS 2021 / N. Mitra and I. Viola
(Guest Editors)

Volume 40 (2021), Number 2

MultiResGNet: Approximating Nonlinear Deformation via Multi-Resolution Graphs

Tianxing Li, Rui Shi and Takashi Kanai

The University of Tokyo, Tokyo, Japan

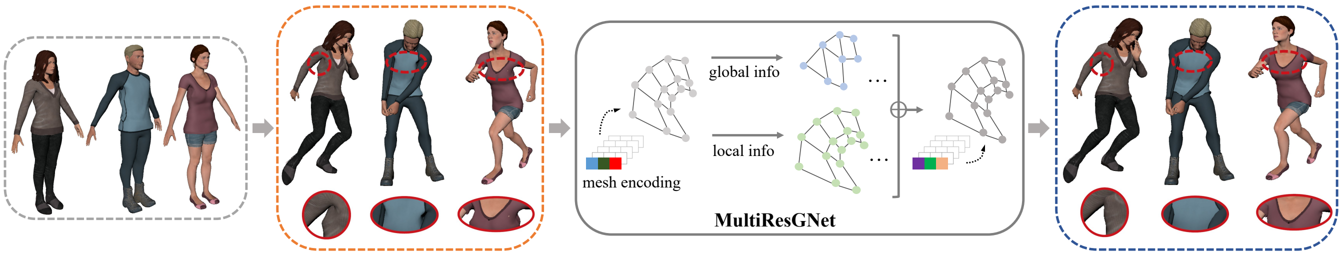

Figure 1: *MultiResGNet is a graph-learning-based nonlinear deformation approximation network. Given a set of characters in the rest pose, they are first roughly deformed by linear-based method. Then, these meshes are encoded with three descriptors and input them into our MultiResGNet. Finally, MultiResGNet can yield realistic nonlinear results by processing both global and local information, thereby greatly reducing the manual work of animators.*

Abstract

This paper presents a graph-learning-based, powerfully generalized method for automatically generating nonlinear deformation for characters with an arbitrary number of vertices. Large-scale character datasets with a significant number of poses are normally required for training to learn such automatic generalization tasks. There are two key contributions that enable us to address this challenge while making our network generalized to achieve realistic deformation approximation. First, after the automatic linear-based deformation step, we encode the roughly deformed meshes by constructing graphs where we propose a novel graph feature representation method with three descriptors to represent meshes of arbitrary characters in varying poses. Second, we design a multi-resolution graph network (MultiResGNet) that takes the constructed graphs as input, and end-to-end outputs the offset adjustments of each vertex. By processing multi-resolution graphs, general features can be better extracted, and the network training no longer heavily relies on large amounts of training data. Experimental results show that the proposed method achieves better performance than prior studies in deformation approximation for unseen characters and poses.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Neural networks; Animation;***

1. Introduction

In animation production, character rigging is an important task which requires animators to spend considerable time and efforts to compute deformations for given poses. For rigs adopted in interactive applications which have less demanding on deformation quality, linear blend skinning (LBS) is one of the most common skinning methods used widely in real-time animation production due to its computational efficiency. However, the drawbacks of this method include "candy wrapper" and "volume loss" artefacts and lack of nonlinear deformation effects. In contrast, to ensure more realistic and attractive deformations, heavily manual interventions, such as manually adjusting weights carefully, implementing additional skinning algorithms, etc., are required by highly-skilled animators. To eliminate this tedious work, studies on automatically generating plausible deformations are being carried out continuously [LMR*15, BODO18, LZT*19, LSK20]. While these methods have achieved success to a certain extent with plausible results, the generalization ability of networks is still a challenge, given that the trained networks are only applicable to characters with the same mesh topologies [LMR*15, BODO18], or prediction accuracy is substantially dependent on large training sets [LZT*19, LSK20].

More recently, following the success of graph convolutional networks, regarding arbitrary meshes as graphs (involving vertices, edges, and spatial connectivity) and learning graph data provide a new perspective on mesh deformation. However, in practice, this is still a challenging task due to the difficulty of encoding of pose-based individual features to graph, and the limitation of network generalization ability. More specifically, first, deformations are personalized that depend on both character subjects and poses. When expressing deformations through graph node features, using the vertices position [LZT*19, LSK20, XZK*20] as features cannot provide the spatial invariance. For example, this representation will result in graphs with different spatial positions (but with the same pose) corresponding to the same output deformations, which will greatly increase the requirements for the number of poses in training. Therefore, our first goal is to design representative graph features to concisely express character deformations in different poses independent of spatial position. Once the expressive features are obtained, next, a graph network needs to be designed for building the relationship between encoded graph features and final deformation predictions. Some vanilla graph-based networks tend to overfit this relationship based on large training set since they are unable to essentially simplify different mesh graphs and extract more general features. To this end, our second goal is to propose a novel network structure with better generalization ability that can summarize features from existing character samples.

In this work, we propose a novel multi-resolution graph network, named MultiResGNet, that can automatically generate complex nonlinear deformations for new characters in arbitrary poses. The outline of the method is shown in Fig. 2. After embedding the standard skeleton, a mesh of the animated character is first roughly deformed by linear-based method. The remaining nonlinear deformation part is then approximated by our trained MultiResGNet, which takes linear skinning graphs as inputs and deviation displacements of each vertex as outputs. The proposed deformation approach provides two key contributions that enhance network generalization ability with the help of informative graph features and multi-resolution graph processes. Specifically, they are:

- An effective graph construction solution introducing three descriptors for each node, which encode the skinning features along with poses, geometry attributes of meshes, and the relationships between meshes and skeletons. We have found that using these graph features, our network is plausible and robust for deformation approximation for unseen poses and characters.
- A novel graph-learning-based network MultiResGNet enabling the reuse of existing artist-created skinning features and their easy application to new character meshes. The training parameters of the network are independent of the number of mesh vertices, but are related to the fixed feature dimensions of the input mesh graphs. To improve the ability of expressing arbitrary graphs, we designed the MultiResGNet with two branches, i.e., a global branch that deals with lower-resolution graphs for integrating structural information and a local branch that handles original-resolution graphs for enhancing and propagating detail features. To address the limitations of insufficient training samples, for the global branch, we designed graph pooling and corresponding unpooling operations to extract the holistic features of various characters and poses, thereby realizing better generalization and performance.

We implemented the proposed strategies on datasets involving a set of high-quality rigs with random poses. After training, the network we designed was found to effectively predict nonlinear deformations for both new character models and new motions, which demonstrates the strong generalization abilities of our approach.

2. Related work

2.1. Skinning Methods

Standard rigging pipelines include the steps of defining the control structure inside the character model, which is usually a skeleton structure, and then binding the skeleton to the mesh with skinning algorithms. There are mainly three types of skinning algorithms: geometry-based, example-based, and physics-based methods.

Linear blend skinning (LBS) as described in [MTLT88] is one of the fastest and most efficient geometry-based methods widely used in real-time animation. This method computes deformed vertex positions from rest pose positions as a weighted combination of joint transformations. Due to the nature of LBS, while joints are rotated in a large range, the undesired deformation effect "volume-loss" will be produced. To eliminate this artefact, studies in [Kav05, KCvO07a, ÖBP*13] replaced linear blending with nonlinear blending without sacrificing large computation time. However, other artefacts, like joint-bulging could happen that still need to be manually fixed by animators. By embedding the mesh into a implicit volume, Vaillant et al. [VBG*13] avoided the self-collision and volume loss during deformation. To compensate the unnatural deformations of naive LBS and generate more natrual and plausible behaviours, helper bone rigs that can create the muscle bulging and soft tissue jiggling effects have been adopted in [Muk15, MK16]. Such helper bone rig techniques have been successfully used in practice for game productions. Despite of the high flexibility of secondary dynamics, it is difficult to control deformations stably in some cases.

Example-based methods provide another perspective on enhancing deformation quality. To remove artefacts from geometry-based skinning, Mohr et al. [MG03] proposed a method by adding additional joints to achieve the convincing deformations. By interpolating a set of examples, the pose-space deformation (PSD) [LCF00] method formulates deformations as a function of articulated poses. It can achieve more attractive skinning effects like skin slide, muscle bulges, and cloth wrinkles while at the cost of additional computation. More recent studies [HTRS10, LD14] also compute weights based on a large set of poses. The limitation of these approaches is that they are only optimized for specific characters, and are hard to generalize for other character models.

To make the generated deformations closer to real-world human behaviour, physics-based approaches are actively used for animations since they formulate deformations within simulation frameworks. Studies have been explored for skin deformation [MZS*11, KB18], second motion rigs [HTC*13] and cloth simula-

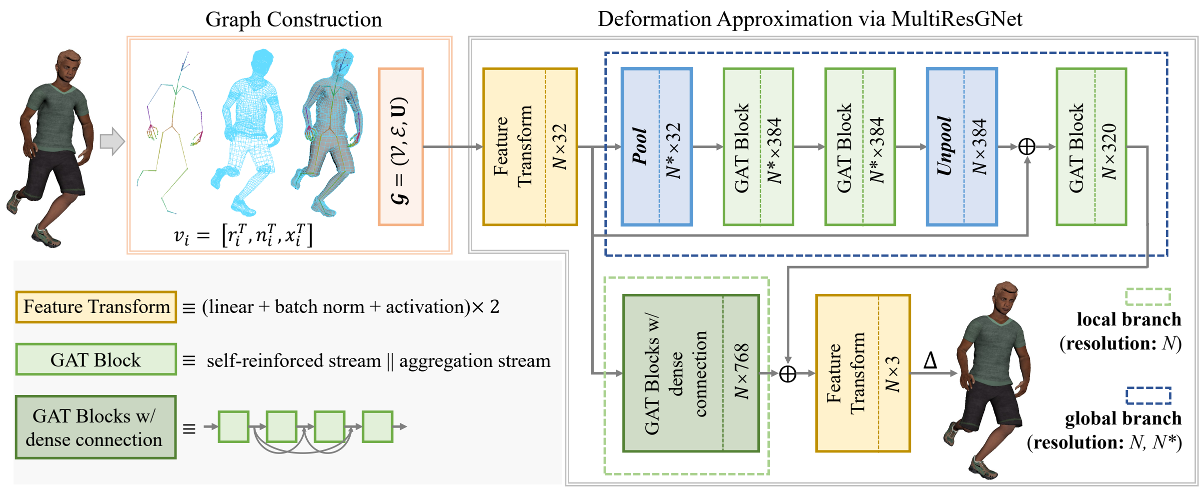
Graph Construction
Deformation Approximation via MultiResGNet
$v_i = [r_i^T, n_i^T, x_i^T]$
$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{U})$
Feature Transform N×32
Pool N*×32
GAT Block N*×384
GAT Block N*×384
Unpool N×384
GAT Block N×320
GAT Blocks w/ dense connection N×768
Feature Transform N×3
Δ
local branch (resolution: N)
global branch (resolution: N, N*)
Feature Transform ≡ (linear + batch norm + activation)× 2
GAT Block ≡ self-reinforced stream || aggregation stream
GAT Blocks w/ dense connection ≡

Figure 2: *Pipeline of our method. It is based on the graph-attention-network (GAT) that is detailed in Sec. 3.3. After the rough linear-based deformation step, We first design a new way of expressing graph features of deformed meshes during the process of graph construction (top left orange wireframe). The constructed graph is then forward into our MultiResGNet (right grey wireframe) to realize the deformation approximation. The designed network consists of a global branch and a local branch. The global branch using our proposed graph pooling and unpooling strategy is used to handle the holistic features for better generalization. The local branch that consists of four densely connected GAT blocks is adopted for detail features extraction. Feature transformation modules (bottom left brown frame) are adopted at the beginning and the end of network, which respectively for high-dimensional feature mapping and feature integration. The internal architecture of the GAT block and the connection are also shown at the bottom left green frames.*

tion [KJM08]. While producing highly detailed results, significant computational cost is required to get satisfactory deformations.

2.2. Learning-Based Animations

Learning-based methods are aimed to learn mappings from parameters to perform deformations using neural networks. For the animation of human bodies, there are methods [PMRMB15, CO18] that approximate soft-tissue deformations of real-humans with the help of network-based regressors. Loper et al. [LMR*15] proposed a skinned vertex-based model that can represent different body shapes with natural poses. Their approach has unique advantages such as being able to create realistic animated human bodies and compatibility with graphics pipelines. Recently, Igor et al. [SGOC20] went a step further and modeled soft-tissue dynamics as a function of shape and motion descriptors. This approach is efficient for generalization to new shapes and motions, but can be used only for meshes with the same number of vertices. A similar strategy in [SOC19] learned the nonlinear deformation of clothing as a function of shape and pose. Both garment fit and wrinkles can be represented with lower computation time. A study for garment deformation has also been conducted in [WSFM19]. One shortcoming of this method is that the network needs to be re-trained for different characters with different garments. For production characters with non-manifold meshes and complicated skeleton structures, based on graph learning, a skin weight prediction method called NeuroSkinning was introduced in [LZT*19]. More recently, Xu et al. [XZK*20] proposed a complete solution for character rigging named RigNet, which includes skeleton and skin weight prediction. Unlike NeuroSkinning and RigNet studies, our method does not generate fixed skinning weights but realizes more versatile nonlinear deformations which include more convincing and elaborate effects under different poses by directly yielding an incremental displacement per mesh node in each pose step.

2.3. Graph Convolution Networks and Applications

Convolutional Neural Networks (CNNs) have achieved a great success in various challenging tasks, especially for dealing with 2D images and regular 3D grids. Alternatively, there are many irregular data structures that can be represented as graphs such as point clouds, social networks and meshes. These complicated graph data cannot be directly applied with traditional convolution. Recently, many attempts have been made to explore the extension of the CNNs to Graph Convolutional Networks (GCNs) by defining the convolution in the spectral domain [BZSL13, LWZH18] and the spatial domain [VCC*17, NAK16], and successfully adopt the extension in many applications. To extract intuitive localized deformations of 3D meshes, Tan et al. [TGL*18] introduced an autoencoder architecture based on spatial construction, which enables reuse of extracted components to reconstruct meshes. Training the graph convolutional variational autoencoder in [LBBM18] solved the problem of 3D object construction from missing data and achieved the completion of partial shapes. In a recent study, Jiang et al. [JZH*20] proposed a method for the automatic reconstruction of the body and clothing shape from a single RGB image. Following the self-attention strategy, the graph-attention-network (GAT) [VCC*17] computes the hidden states of each node by attending over the neighboring nodes. Since the GAT operation is efficient without information on graph structure front, it can be used in inductive learning easily. Mesh deformation studies in [LZT*19] and [LSK20] utilized GAT and its extended structure for skinning weight prediction and nonlinear deformation prediction respectively. In this work, we also extend the original GAT by applying with pooling, unpooling, and dense connection pattern for producing nonlinear deformation effects.

Graph pooling plays an important rolem, as they can avoid overfitting and improve generalization ability of networks. Because of the complicated and irregular characteristics of graph nodes, few graph pooling studies have been done. Ying et al. [YYM*18] proposed DiffPool which learns the assignment matrix of clusters based on node features. The number of clusters needs to be decided beforehand. Gao and Ji [GJ19] introduced TopKPool in Graph U-Net. The strategy of their pooling is to simply keep nodes with top-k scores and drop all other nodes. One drawback of this approach is poor robustness where small local changes will affect the whole pooling results. Recently, Lee et al. [LLK19] presented SAGPool that uses self-attention mechanism to calculate node scores and then mask all but the top nodes. Different from their masking nodes, inspired by the method of computing attention coefficients mentioned in [VCC*17], we conduct merging neighbor nodes according to maximum attention coefficients, which has the effect for contracting edges, so that both nodes features and topologies will be considered for graph processing and conveyed to deeper layers.

3. Method

As illustrated in Fig. 2, our goal is to explore a framework that is able to generate nonlinear deformations by generating offset adjustments based on automatic and rough linear-based deformations. Therefore, the first challenge is to represent linearly deformed meshes with corresponding poses concisely. Once graph representations are defined, the second challenge is to develop a network architecture which allows the large volumes of complicated mesh representations to achieve linear-nonlinear mappings.

Graph neural networks find many applications on processing 3D mesh data. The main idea is to first construct meshes into graphs, and then to use the defined convolution algorithms to achieve tasks. Graph-attention-networks (GAT) allow for dealing with variable-size input by following attention mechanisms. Inspired by them, we introduce an attention-based architecture MultiResGNet to perform nonlinear deformation approximation.

3.1. Deformation Approximation

We assume that character deformation can be regarded as a combination of two parts: the linear deformation computed directly according the bone transformations in the skeleton with the linear blend skinning method, and the nonlinear deformation for correcting visible errors caused by linear deformation.

Based on a linear blend skinned mesh $M_{linear} \in \mathbb{R}^{3\times N}$ with N vertices, we first extract its mesh graph $\mathcal{G}$, and then learn the function to produce corresponding nonlinear mesh shape corrections. The final deformed mesh $M \in \mathbb{R}^{3\times N}$ can be expressed as:

$$M = M_{linear} + W(\mathcal{G};\alpha), \tag{1}$$

where $W(\cdot)$ is MultiResGNet, a graph-learning-based nonlinear regressor, that takes graph $\mathcal{G}$ as input and calculates shape offsets for each vertex by learning a set of parameters α. It should be noted that, unlike the previous work that also predicts the nodal displacements for character nonlinear deformations [LSK20], our key improvement lies in the better generalization of the trained models by efficiently encoding mesh features via graph $\mathcal{G}$ and then refine it through the multi-resolution graph network $W(\cdot)$.

3.2. Encoding of rough deformed mesh

Having the linear-deformed mesh M_{linear} directly computed, we need to construct a parametric space that is capable of representing different mesh topologies and varying poses. Here, we consider the input of our framework to be a mesh graph $\mathcal{G} = (\mathcal{V},\mathcal{E},\mathbf{U})$ which stores features of vertices and edges. Here, $\mathcal{V} = \{1,...,N\}$ and $\mathcal{E} \subseteq \mathcal{V}\times\mathcal{V}$ denote the set of vertices (nodes) and edges respectively. $\mathbf{U} \in [0,1]^{N\times N}$ is the adjacency matrix where $\mathbf{u}(i,j) \in [0,1]$ indicates whether there is an edge between nodes i and j, $(i,j) \in \mathcal{E}$. For the node $i \in \mathcal{V}$, the set of neighboring nodes is represented by $\mathcal{N}(i)$.

To make graph nodes informative and discriminative among diverse meshes with different geometries and different transformations, it requires to assign attributes to each node. While the feature design method mentioned in [LZT*19, LSK20] can be followed, we found that their approximated results are heavily affected by the amount of character translations due to the global position of the node attribute of their methods. To strengthen the feature expression, we assign each node $i \in \mathcal{V}$ with a feature vector which consists of three descriptors: $v_i = [r_i^T(\vec{\theta}), n_i^T, x_i^T]$. Concretely, to represent skinned mesh features with various poses, given the pose vector $\vec{\theta} = [\vec{\omega}_0^T,...,\vec{\omega}_s^T,...,\vec{\omega}_S^T]$, where S is the total number of joints of a rig, and $\vec{\omega}_s \in \mathbb{R}^3$ denotes the axis-angle of joint s. We define a novel relative skinning feature descriptor as $r_i(\vec{\theta}) \in \mathbb{R}^3$. When the joint s is rotated by an angle, the corresponding rotation matrix can be expressed as:

$$G_s(\vec{\theta}) = \prod_{p\in\mathcal{A}(s)} Z(\vec{\omega}_p), \tag{2}$$

$$Z(\vec{\omega}_p) = I + \sin(\|\vec{\omega}_p\|)\hat{\vec{\omega}}_p + (1-\cos(\|\vec{\omega}_p\|))\hat{\vec{\omega}}_p^2, \tag{3}$$

where $p \in \mathcal{A}(s)$ is the ordered set of joint ancestors of joint s. $Z(\vec{\omega}_p)$ is a local rotation matrix computed by Rodrigues formula through axis angle $\vec{\omega}_p$, skew symmetric matrix $\hat{\vec{\omega}}_p$ corresponding to vector $\vec{\omega}_p$, and identity matrix I.

Then, to reflect the influence of bone movement on each mesh node, we combine the rotation matrix, linear skinning weight and rest pose position. Thus, our relative skinning feature descriptor can be defined as:

$$r_i(\vec{\theta}) = \sum_{k=s}^{S} w_{s,i} G_s(\vec{\theta}) G_s(\vec{\theta}^*)^{-1} \bar{r}_i, \tag{4}$$

where $G_s(\vec{\theta}^*)$ is the rotation matrix of joint s in the rest pose $\vec{\theta}^*$. $w_{s,i}$ and $\bar{r}_i$ respectively denote the skinning weight of the vertex v_i influenced by the joint s, and the rest pose position of the vertex i. Note that $r_i(\vec{\theta})$ is translation-invariant instead of the simple vertex position, because $G_s(\vec{\theta}^*)$ is independent of joint position.

For the descriptor of $n_i \in \mathbb{R}^3$, it is the normal of vertex v_i that is able to represent the attribute of mesh appearance. To capture the relationship between each vertex and the skeleton, we define the distance vector as $x_i = [x_{i,1}, \ldots, x_{i,s}, \ldots, x_{i,S}] \in \mathbb{R}^S$, where $x_{i,s}$ refers to the volumetric geodesic distance [DdL13], i.e., the shortest distance from vertex v_i to joint s passing though the interior voxels.

The total dimension of the node feature v_i is $6+S$. Such graph feature representation has the expressive power of basic linear-based deformations that can capture the skinning features across different joint transformations ($r_i(\vec{\theta})$), the whole range of mesh surface features of different shapes (n_i), and the binding relations between joints and meshes (x_i). The designed feature descriptors are translation-invariant and enable the network to handle data more effectively.

3.3. Refining deformation through MultiResGNet

As constructed descriptive and discriminative graphs are insufficient alone, they have to be directly mapped to per-vertex displacements. As this is a challenging task, for high-dimensional inputs and highly nonlinear outputs, we propose a two-branch strategy that enables the network to learn the features of mesh graphs in a multi-resolution way and then to predict nonlinear offsets.

To acquire the latent representations of irregular mesh graph data, the work of GAT [VCC*17] defines graph convolutions by aggregating the node representations from its neighborhoods, following the self-attention strategy. Due to its directness and without need of knowing graph structure upfront, graph-learning-based character deformation studies [LZT*19, LSK20] both extended original GAT structure for effectively learning complicated graph features. While the designed operations for networks (eg., max pooling, dense connection) are workable to a certain extent that make the prediction results satisfactory, their training relies on a large set of training samples and parameters with redundancies, which makes the learning task prone to overfitting. Meanwhile, the generalization ability of the trained networks are limited that output predictions sometimes are sensitive to tiny changes of the features in the input.

We therefore propose a novel and effective approach to address limitations of network generalization capabilities. Our proposed model, named Multi-Resolution Graph Network (MultiResGNet), integrates multi-resolution graph features for training and inferencing. It contains a local branch and a global branch, dealing with original-resolution graphs and down-sampled lower-resolution graphs respectively. At last, processed features from two branches will be concatenated, and then pass through a feature transformation module to generate the final predictions.

Local branch utilizes several densely connected graph-attention-network (GAT) blocks to deal with local features. Specifically, after the feature transformation module, the node features are transformed into $v^{[l]} = \left\{\vec{v}_1^{[l]}, \ldots, \vec{v}_i^{[l]}, \ldots, \vec{v}_N^{[l]}\right\}$, $\vec{v}_i^{[l]} \in \mathbb{R}^{d^{[l]}}$, where $d^{[l]}$ is the feature dimension after previous l layers' feature transformation. In each GAT block, we adopt the same structure as mentioned in [LSK20], which consists of an aggregation stream and a self-reinforced stream for dealing with complicated detailed features. The aggregation stream performs the self-attention mechanism to compute the attention coefficient of a node with its neighbors $\mathcal{N}(i)$:

$$\alpha_{ij}^{[l]} = \frac{\exp(\mathrm{LeakyReLU}(\vec{a}^{[l]^T}(W\vec{v}_i^{[l]} \| W\vec{v}_j^{[l]})))}{\sum_{q\in\mathcal{N}(i)} \exp(\mathrm{LeakyReLU}(\vec{a}^{[l]^T}(W\vec{v}_i^{[l]} \| W\vec{v}_q^{[l]})))}, \tag{5}$$

where $\|$ is the concatenation operation. $W \in \mathbb{R}^{d^{[l]'} \times d^{[l]}}$, $\vec{a}^{[l]} \in \mathbb{R}^{2d^{[l]'}}$ respectively indicate the weight matrix and the weight vector of a single-layer feed forward network. The aggregation stream and the self-reinforced stream are concatenated to form the GAT block:

$$\vec{v}_i^{[l+1]} = f_{\mathrm{GATB}}(\vec{v}_i^{[l]}) = \sigma\Big(\underbrace{\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{[l]} W\vec{v}_i^{[l]}}_{\text{aggregation}} \| \underbrace{H\vec{v}_i^{[l]}}_{\text{self-reinforced}}\Big), \tag{6}$$

where σ denotes a nonlinear transformation. $H \in \mathbb{R}^{\beta d^{[l]'} \times d^{[l]}}$ is a trainable weight for self-reinforcement.

Furthermore, to ensure the efficient learning of local features, m GAT blocks with the dense connection pattern is adopted, resulting in the output of the local branch:

$$\begin{aligned}\vec{v}_i^{[l+m]} &= f_{\mathrm{GATB}}(\vec{v}_i^{[l+m-1]}, \gamma^{[l+m-1]}) \| \vec{v}_i^{[l+m-1]} \\ &= f_{\mathrm{GATB}}(\vec{v}_i^{[l+m-1]}, \gamma^{[l+m-1]}) \| \ldots \| f_{\mathrm{GATB}}(\vec{v}_i^{[l]}, \gamma^{[l]}) \| \vec{v}_i^{[l]},\end{aligned} \tag{7}$$

where $\gamma^{[\cdot]}$ represents all parameters of the transformation function f_{GATB} in different layers. $\vec{v}_i^{[l+m]}$ is the result of fusing all the intermediate GAT block layer outputs. Thanks to these densely connected GAT blocks, local details can be well captured by combining features from different scales, and the problem of vanishing gradients can then be effectively avoided. In the processing in GAT blocks, the number of graph nodes is constant, but the dimension of node features is changing. Thus, on the path of our local branch, the number of nodes always equals to N.

Despite the outstanding performance in extracting detail features, pure single-resolution graph convolution is not enough for complicated character deformation approximation. It neglects the entire graph structure information and tends to overfit the relationship between input features and output deformations, thus sometimes generalizing poorly to new characters and poses. To address the aforementioned limitations, we additionally propose the global branch with the attention-based pooling operation to globally summarize all the node feature representations.

Global branch utilizes lower-resolution graphs (with sparse vertices) to capture the overall structural information. To achieve the coarse representation of the original graph, there is a need to first define the pooling strategy. In our work, we present an edge contraction operation which follows the attention mechanism to calculate the coefficient $\alpha_{ij}^{[l]}$ of a node with its neighbors in Equation (5). We regard the obtained coefficient as being the edge score of each pair of nodes. By sorting all edges of their scores, the node which is adjacent to i with the highest attention coefficient can be depicted as:

$$j^* = \arg\max_{j\in\mathcal{N}(i)} \alpha_{ij}^{[l]}. \tag{8}$$

Iteratively, we implement edge contraction on the overall mesh.

Figure 3: *Five example characters in our test set.*

Edges with highest score are contracted and the nodes that belong to the contracted edges are ignored for other edge contractions. The defined pooling operation allows both nodal features and topology features to yield hierarchical representations. Through this process, the total number of nodes will become N^* that roughly equals to 50% of the original N.

The pooling operation will create newly merged nodes. Together with the edge score, we combine the features from the previous pair of nodes to obtain the new node features:

$$\vec{g}^{[l+1]}_{ij^*} = \alpha^{[l]}_{ij^*}(\vec{v}^{[l]}_i + \vec{v}^{[l]}_{j^*}). \tag{9}$$

In order to learn the global information, the new node features $\vec{g}^{[l+1]}_{ij^*}$ of the lower-resolution mesh graph are then fed as input for several GAT blocks. After processing with b GAT blocks, the features are transformed into $\vec{g}^{[l+b+1]}_{ij^*}$.

For the inverse of pooling, we also perform the corresponding unpooling operation. To restore the graph to its original structure, nodes are given the mapping of location information from the pooling layer. After this step, the number of nodes will be restored from N^* to N. Moreover, the unpooled features of nodes are computed with the edge score:

$$\vec{g}^{[l+b+2]}_i = \vec{g}^{[l+b+2]}_{j^*} = \vec{g}^{[l+b+1]}_{ij^*}/\alpha^{[l]}_{ij^*}. \tag{10}$$

Lastly, the output features from the local branch and global branch are concatenated together. This strategy enables fusion of the fine detail information and spatial context information. These fused features finally undergo a feature transformation module f_{trans} to enable the final prediction:

$$\Delta_i = f_{trans}(\vec{v}^{[l+m]}_i \| f_{\text{GATB}}(\vec{g}^{[l+b+2]}_i)), \tag{11}$$

where $\Delta_i \in \mathbb{R}^3$ is the corrective displacement of the node i in the graph. While forwarding a linear-deformed mesh graph into our MultiResGNet $W(\cdot)$, it can produce a set of refined displacements Δ of mesh vertices. It should be noted that the trainable weights (W, H, etc.) in the network are related to the dimension of node features but are not affected by the number of mesh vertices N. Therefore, the trained networks has no constraints on the number of vertices and can be applied to meshes with different topologies.

4. Dataset, Network Implementation, and Training

To evaluate our proposed method, we created 75 character models by different shapes and customizations for training, five characters for validation, and 10 characters for testing with Adobe Fuse CC. All characters share the same standard skeleton with 65 joints. As shown in Fig. 3, there are five test examples and their statistics are provided in Tab. 1. Each character contains linear-based deformations as the baseline, and high-quality nonlinear deformations as the learning objective. Specifically, the LBS method is used to create linear-based deformations where skinning weights are directly determined by the geodesic voxel distance without modification and a vertex is set to be influenced by no more than four joints. Here, we have tried to replace this LBS baseline with Dual Quaternion Skinning (DQS) method [KCvO07b], and found that the accuracy of the approximation results has no obvious improvement. For the ground truth data, to reduce the tedious rigging workload, the characters are firstly auto-skinned by Mixamo [mix]. Then, for the inaccurate deformations with obvious artefacts in specific poses (e.g., elbow bending, twisting, etc.), manual refinements [LCF00] were done by animators for more realistic effects. In particular, each character in our training set is animated with 250 poses, of which 150 poses are randomly generated and the remaining 100 poses are from frequent motions of walking, jumping, and running.

Table 1: *Statistics of the example characters*

Index	Joints	Heights	Number of vertices
1	65	165.69cm	17351
2	65	180.28cm	15389
3	65	180.60cm	16623
4	65	189.96cm	16208
5	65	171.58cm	16438

To further evaluate our method on standard human models, we also leveraged the SMPL dataset [LMR*15] which contains detailed deformations of different bodies performing animation sequences. For our training set, it includes five female subjects and three male subjects. These characters are deformed with sample poses, with the total number of 2955, which are selected from motion sequences of dancing, jumping, and butterfly kicking. For testing, we use one female and one male characters and animate them separately with dancing and front kicking motions.

Next, we provide the implementation detail of our networks. As shown in Fig. 2, after graph construction, the graph features are first normalized using the mean and variance for each dimension and input into a feature transformation module that contains two hidden layers with hidden neurons of (64, 32) and applied with batch normalization and tanh nonlinear activation function. Then, the transformed features are forwarded into two branches for the processing of multi-resolution graphs. To learn the overall structural features, pooling is implemented so that the number of nodes is reduced by roughly half to become N^*. The size of the feature dimension remains unchanged. The lower-resolution graph features are then processed by two GAT blocks. Each GAT block involves a graph-attention-based aggregation stream with hidden feature size of 16, multi-head number is 8, and a self-reinforced stream with hidden feature size of 128. Features will be transformed with tanhshrink activation function in the last layer of the GAT block. Unpooling operation, which is the inverse operation of pooling, is conducted to restore the original graph resolution of N. For the local branch,

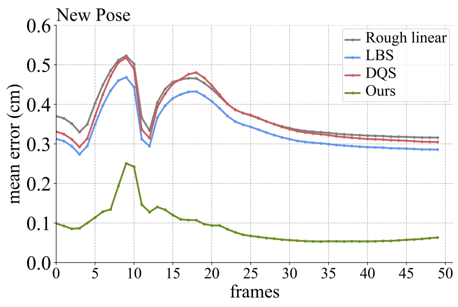

Figure 4: *Quantitative evaluation of generalization to new poses. The test motion "playing golf" with 50 frames is applied with a character in training set. We show the mean distance error, comparing our method with the input rough deformation of linear-based deformation, LBS, and DQS.*

the parameters setting of GAT block is same as that mentioned for the global branch. Finally, features from two branches are concatenated together and fed into the last feature transformation module with two hidden layers, where hidden neurons are (256, 64).

Training was implemented on nVIDIA GeForce RTX2080Ti GPU. During the training, we used the Adam optimization algorithm with an initial learning rate of 0.01 and set the decay factor with 0.75 to reduce the learning rate when loss stopped decreasing for eight epochs. To minimize errors between approximated and ground truth displacements, the mean squared error is used as the loss function. To ensure stable training, we randomly located a vertex and selected the surrounding 1024 consecutive vertices each time until traversing the entire meshes. The total training with our created dataset and with SMPL dataset took roughly one week, and 25 hours respectively.

5. Experiments

This section evaluates our proposed method for nonlinear deformation approximation both qualitatively and quantitatively. By comparing with other state-of-the-art methods, we also demonstrate the benefits of our proposed graph construction solution and network structure.

5.1. Quantitative Evaluation

To demonstrate the generalization capabilities of our method to new poses, we evaluated a character appearing in the training set and animated it with an unseen motion of playing golf. Fig. 4 shows the per-vertex mean distance error of rough linear based deformation, LBS method, DQS method and our proposed method in 50 frames. It can be observed that through our MultiResGNet prediction, the generated nonlinear shape corrections are relatively average, with about 0.28cm improvement compared with the input rough linear-based deformation. For the deformation of LBS and DQS, even if the skinning weights are manually adjusted, due to the inherent limitation of the algorithms, the overall mean errors are still relatively large. In contrast, with our graph-learning-based approximation, the mean error of the deformation reaches to 0.09cm. Based

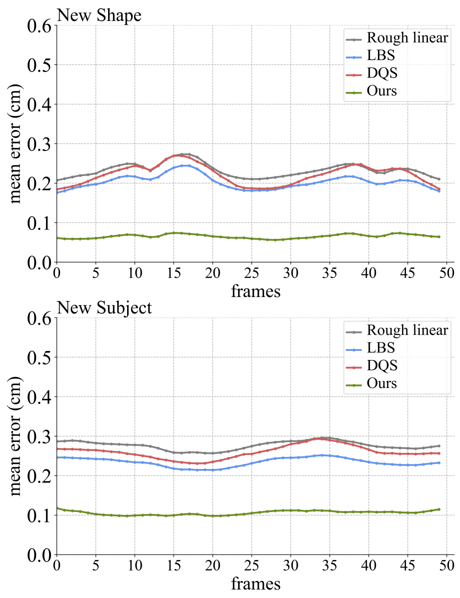

Figure 5: *Quantitative evaluation of generalization to new characters. Top plot: Per-vertex mean error of a new character shape deformation with walking motion in 50 frames. Bottom plot: Per-vertex mean error of a new subject model deformation with running motion in 50 frames.*

on significantly decreased pose training samples, our approximation with lower errors demonstrates that our proposed network with novel graph descriptors has a good generalization ability for predicting nonlinear deformation with new poses.

To evaluate the generalization ability of our method to new characters, we first define the new characters as having different shapes (but the same number of vertices) or different subject models (different mesh geometries) from the characters in training set. For these two cases, we separately conducted experiments with walking and running motions to evaluate our trained networks. As shown in Fig. 5, the per-vertex distance error of the new shape case is about 0.06cm and of the new subject case is about 0.11cm on average. Compared with the input rough linear-based deformations, our approximations can increase the average accuracies of 0.16cm and 0.13cm respectively. We found that our approximation has less errors and higher accuracy improvement for new shape deformations. It illustrates that our networks have better generalization abilities for new shapes where the geometry structure of meshes is unchanged from the character in the training set. When approximating for new subject models, the trained networks also have certain inference although the mesh geometry is completely new.

5.2. Qualitative Evaluation

To illustrate Fig. 4 in a more intuitive way, we show the qualitative deformation results under the animation of playing golf in Fig. 6. We focused on the representative frames which have the largest distance error (frm9) and the average error (frm39) of all frames. In the 9th frame, the largest errors tend to occur in the armpits

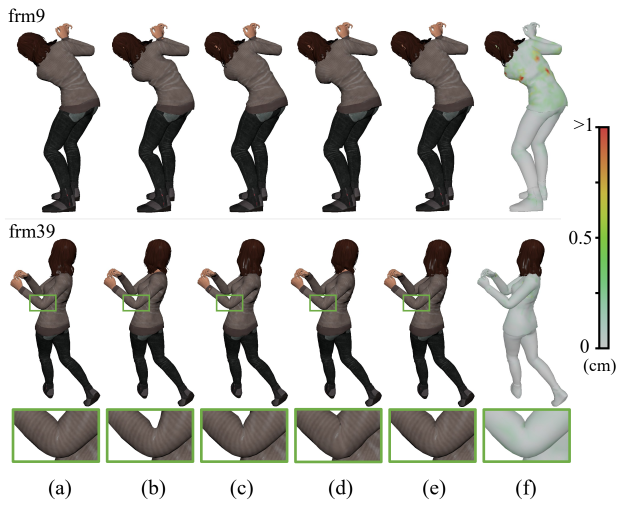

Figure 6: *Evaluation of generalization to new poses. We show deformations of the motions of a character playing golf in the 9th and 39th frames, and side-by-side compare the ground truth (a), rough linear-based deformation (b), LBS with weight refinements (c), DQS with weight refinements (d), our prediction (e) and our approximation colormap (f). The vertices of the approximated mesh are colored to indicate the per-vertex distance error in centimetres.*

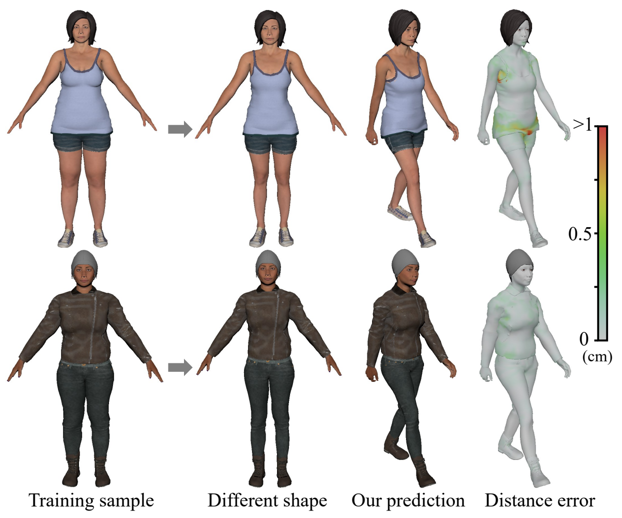

Figure 7: *Evaluation of generalization to new shapes. Test thinner characters in walking poses. Colormaps depict the per-vertex distance error of the predicted deformation.*

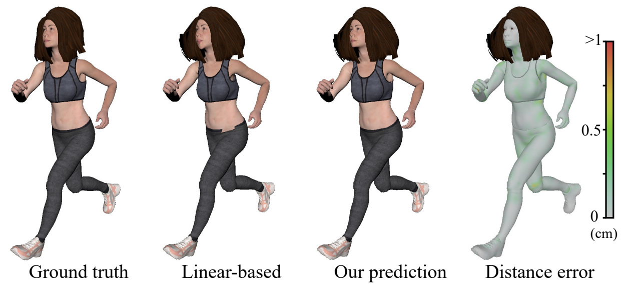

Figure 8: *Evaluation of generalization to new subjects. Test new subject in a running pose. Based on the rough linear-based deformation, our method corrects each vertex with a displacement so that the deformation becomes nonlinear, and no noticeable errors are found in the right colormap.*

and shoulders due to bending and substantial stretches. Since the skinning weight are automatically generated without modification, rough linear-based deformation has the significant artefacts in the right armpit. The deformations of LBS and DQS have improved this problem to a certain extent due to the manual refinements, but the visible volume loss and joint bulging (in the shoulder area) still exist in the results of LBS and DQS respectively which cause the mean errors of the entire body to be still large in Fig. 4. Our deformation tends to produce the result that is most similar to the ground truth. The overall approximation error of upper body is higher than other body parts, this is because the large movements are concentrated on this area. However, these errors are relatively small compared to the whole body and barely noticeable during the animation. In the 39th frame, we provide the detail of deformation results when the elbow is bent. Visually, the approximation with our method can nicely mimic the desired elbow bending behavior around the joint and give the best deformation effect than other methods. Also, the whole character can be deformed well so that is hardly distinguishable from the ground truth.

Fig. 7 shows the sample frames of walking motion for two test characters with new body shapes. For the character in the lower row, the error colormap on the mesh demonstrates the outstanding generalization ability of our trained network where the approximated deformation closely matches the ground truth. The numerical error of each frame corresponds to the Fig. 5 - generalization to new shapes. In the upper row of Fig. 7, because of the change in the body shape and the lower edge of the camisole does not fit the body, the biggest deformation error occurs when the character takes a step. In addition, the inaccurate deformation in the right armpit area is approximated because mesh vertices in this region are affected by multiple joints and the region is also the edge of camisole that is adjacent to the body, thus tending to be problematic.

Additionally, in Fig. 8, we visually evaluated the quality of our proposed approach on generalization to new subject models, where we compared the deformations between ground truth, linear-based method, and our approximations. The test character model is a new subject that its mesh geometries are completely different from characters in training set. As shown in the figure, the overall deformation is successfully predicted using our MultiResGNet, with better performance than the linear-based deformation. Detailed improvements are achieved especially in the area where the pants and belly contact. The prediction error distribution of the whole body is very small and relatively average, for the reason that the clothing is fitted and the running motion is a whole-body movement. Here, the hair part was not approximated by our network, so it remained the same as linear-based results.

Finally, we further demonstrate the generalization ability of our

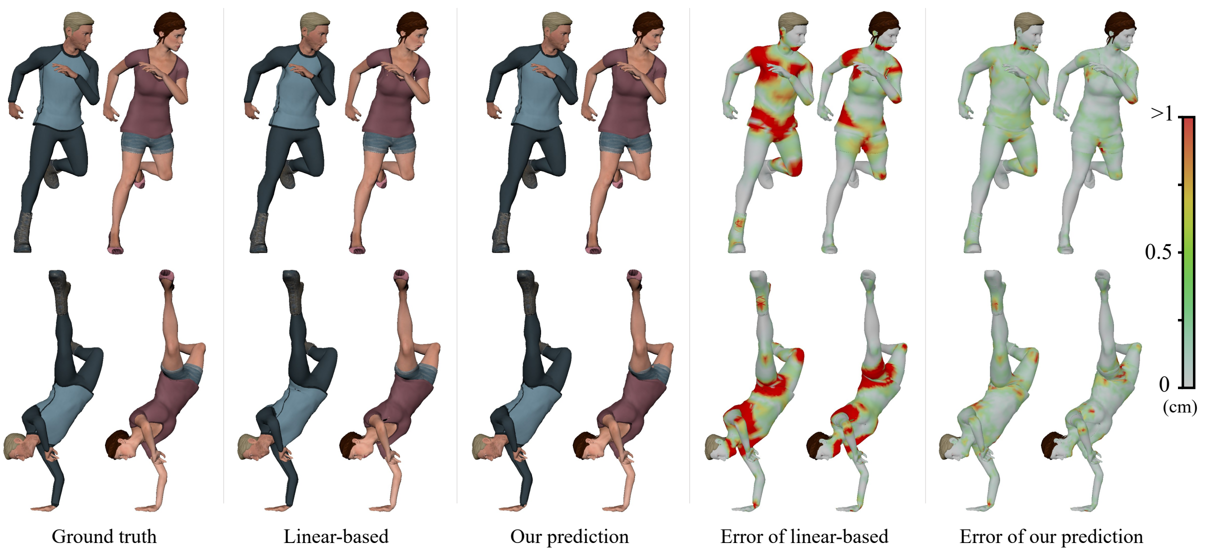

Figure 9: *Evaluation of generalization to new characters with new poses. The dancing motion is used for testing for two new characters. We show ground truth, linear-based deformation(as input), our approximation result and colormaps of per-vertex error.*

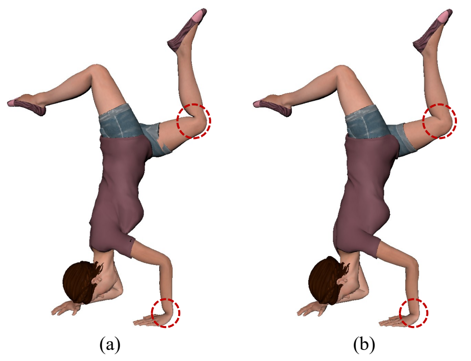

Figure 10: *Deformation results with vertex position features (a) and with our relative skinning features (b).*

network to approximate nonlinear deformations for new characters with new poses. As shown in Fig. 9, we selected two test characters, and animated them with dancing motions. As expected, plausible and realism deformations can be produced without particularly noticeable artefacts. The areas with large errors are concentrated in the armpits and crotch, and near the joints which are controlled by multiple bones. As our network is only trained with a small number of representative character models and poses, they are still able to produce visually plausible deformation effects that can meet the animation production needs.

5.3. Comparisons

As we were interested in the effectiveness of our two key contributions: the representation of graph attributes and the network structure handling multi-resolution graphs. In this subsection, we conducted experiments with two types dataset, i.e., our created humanoid characters with different customizations and the SMPL realistic human bodies, to measure the generalization ability of our proposed deformation method.

Comparison on our created dataset. First, to verify the effectiveness of our proposed graph features, we performed an ablation study to compare deformations between common position features and our newly proposed relative skinning features. The graphs containing these two different features were respectively input into the same MultiResGNet, following the training settings mentioned in Sec. 4. We used the inverted pose for testing, and its position is not at the origin. In Fig. 10, the approximated deformation with our features is more natural than with position features, especially in the areas of knees, wrists, and inner thighs. Specifically, the mean error of (a) and (b) is about 0.26cm and 0.15cm respectively. Because of the translation-invariance of ours, the local predicted offsets remain unchanged even though the global position changes significantly. In this way, the trained network can learn a variety of deformations through a small number of training poses (250 poses per character), and is effective for all spatial positions.

Next, we compared our approach with other learning-based methods. In addition, to further evaluate the generalization ability of our proposed method, we also compared the performance of the different network structure and the different number of training samples, where the global branch used in our method was removed and the amount of training characters was decreased to 25 and 50. As listed in Tab. 2, we respectively compared with MLP (with hidden neurons of (256, 512, 512, 128), batch normalization and tanh activation function), NeuroSkinning [LZT*19], DenseGATs [LSK20], our network without the global branch (with single-resolution graphs), and our network trained with 25 and 50 samples. Since the network of NeuroSkinning also uses GAT, here we compared with its structure and predict our output (instead of the weights in its original paper). For the NeuroSkinning and

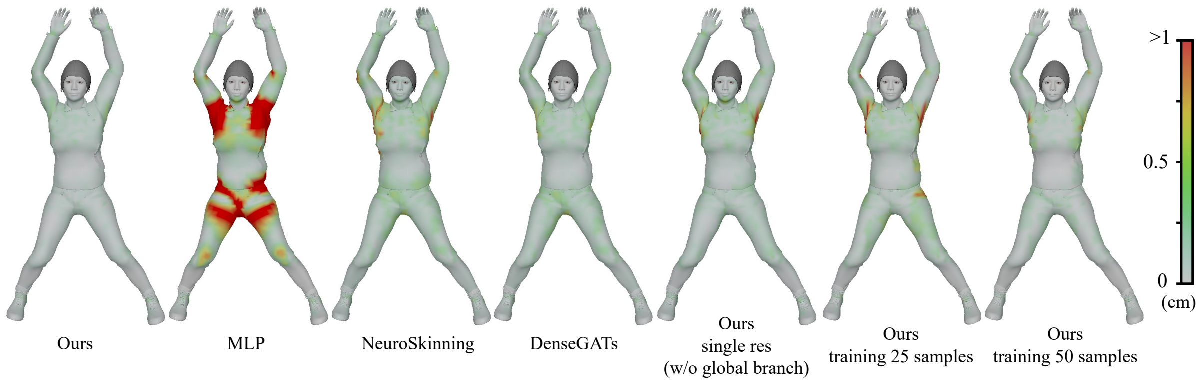

Figure 11: *Comparison among different learning-based methods. Colormap shows the distance error to ground truth.*

DenseGATs, we adopted the same features in their original work and trained the networks with 75 characters and 8500 poses per character. For the training of MLP, we utilized our proposed features and also trained it with 75 characters and 8500 poses per character. To evaluate the generalization ability of our proposed method, we respectively trained our network with 75 characters (original training data setting), 50 characters, and 25 characters, and each of the character was animated with 250 poses. To verify the effectiveness of the multi-resolution strategy, we removed the global branch from the original structure and used the same training amount with 75 characters and 250 poses. The qualitative results are shown in Fig. 11. As observed, the greatest approximation errors are generated by the MLP network, which shows that the pure fully connected network cannot achieve large amounts of complicated approximations of nonlinear offsets and hardly apply the prior skinning knowledge to new character models. As graph-learning-based methods, NeuroSkinning and DenseGATs successfully predict the overall vertex displacements for characters whose translation does not change dramatically. Also, there are still undesirable effects on the areas near the armpits and belly. For our network without the global branch, since the graph resolution throughout the network is constant that equals to N, it can produce convincing effects in most mesh areas but also causes some obvious errors. With less training data, the approximation accuracy of our method could maintain to some degree, but deformation errors near joints are also produced. We found that setting the number of training characters to 75 allows the deformation predicted by the network achieve a visually plausible effects that errors of mesh vertex deviations are within a small range. Thanks to the representative graph features and multi-resolution graph operation, our method enable easier generalization to new characters with new poses based on knowledge which is learned from existing deformations.

Comparison on the SMPL dataset. Since DenseGATs [LSK20] and our approach both approximate deformations by estimating the nonlinear residuals based on rough deformations, and DenseGATs achieved the good result in the previous experiment, next we will apply the SMPL dataset to conduct further comparison.

In Fig. 12, we demonstrate the generalization capabilities to a new SMPL character body. In particular, we followed the training setting mentioned in Sec. 4 and then test the trained network using one female character (height: 169cm) with new body mass performing the dancing motion sequence. Since this task is not complicated, both two methods have achieved satisfactory results with a very low average error (DenseGATs: 0.15cm; ours: 0.08cm) relative to the ground truth. In contrast, our approach generated more nature deformations in the areas of base of the thigh and the chest near the upper arm that is clamped.

Furthermore, with the SMPL data, we also quantitatively demonstrate the generalization capacities to the new character body and poses. Specifically, we used the trained network to predict deformations for a male subject (height: 177cm) with the front kick-

Table 2: *Evaluation of predicted distance errors (cm) using different learning-based methods.*

Structure	mean error	max error	min error
Ours	**0.1121**	**0.5795**	0.0038
MLP	0.3962	13.0968	**0.0017**
NeuroSkinning	0.2377	1.3316	0.0104
DenseGATs	0.1887	1.0324	0.0099
Ours single res	0.1610	1.2328	0.0101
Ours train 25 samples	0.2594	1.4826	0.0101
Ours train 50 samples	0.1728	1.3570	0.0103

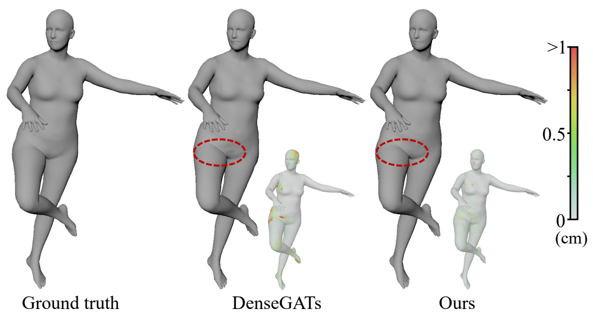

Figure 12: *Generalization to a new SMPL character body: DenseGATs [LSK20] and ours.*

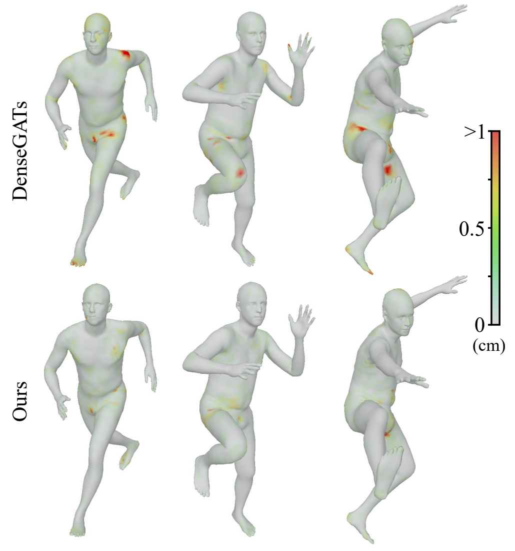

Figure 13: *Generalization to a new SMPL character body with new front kicking poses: DenseGATs [LSK20] and ours.*

ing motion, while both the character and the motion are unseen in the training set. As shown in Fig. 13, the deformation results of DenseGATs tend to generate large approximation errors in regions near joints. Unlike DenseGATs, the relative skinning descriptor of ours includes rich pose knowledge which can be used to approximate new complicated poses. To this end, our proposed method could make full use of prior feature knowledge and outperforms DenseGATs for unseen cases. Numerically, the average and maximum errors of the results using our method are (0.14cm, 1.32cm), which are also better than theirs (0.23cm, 1.80cm).

5.4. Memory and Performance

We have implemented our network on the GPU for evaluation. Once the network is trained, the network weight size is about 8.0 MB. We tested both SMPL characters (6890 vertices) and our created characters (15000∼20000 vertices), the average run-time of deformation approximation for one character is about 26ms. Due to the GPU memory limitation, up to 70 characters can be run at the same time. Since data preprocessing will increase time-consuming, the run-time for 70 characters is about 45ms.

6. Conclusions and Future Work

This paper presents MultiResGNet, a graph-learning-based method for automatically approximating nonlinear deformations. The advantage of our approach is that the proposed network has better generalization ability, which makes use of available artist-designed skinning mesh features for unseen animated characters deformation. The designed feature descriptors can represent graph mesh attributes along with poses in a more efficient way, which make the training process simpler with fewer pose samples. Furthermore, by using our proposed novel expressive graph features, we are able to model nonlinear offsets as a function of mesh graphs, with the help of multi-resolution graph network. We conducted experiments to evaluate our method and the results demonstrate that the generalization abilities to new poses and new character models outperform existing methods. Our approach provides an end-to-end deformation approximation system, which allows application to film production when a large collection of characters need to be accurately deformed. Artists can directly produce complicated deformation effects through the trained network, thus avoiding troublesome manual processing.

Although our presented network has a strong generalization ability and yields high accuracy prediction results, it also has a few drawbacks. First, the learned network is dependent on the setting of linear-based deformation, such as the skinning method, the number of bone influences, the binding distance, etc. Thus, when generating the nonlinear deformation for a new character, its setting of linear-based deformation should be modified to keep it consistent with the training data. Secondly, in some cases where characters wear tight clothing, the produced deformation results may have collision artefacts between the body and clothing. In the future, investigating an end-to-end method that can constrain collision conditions would be an interesting research direction. Thirdly, the training samples we used mostly follow the body closely. For extremely loose clothing and other garment types like skirts, the deformation would largely depend on the diversity of training samples. In the future, we would like to increase the diversity of garments and add situations where characters fit differently with garments in training samples, and then verify the effectiveness of our method. Additionally, we currently focus on the deformation for humanoid characters which have the same standard skeleton, and the dimension of input graph features is dependent on the number of skeletal joints. In the future, if the deformed objects have various number of joints, a union of joints can be defined to represent all the conditions in the dataset, so as to keep the feature dimension of each object the same.

Acknowledgement

Tianxing Li and Rui Shi acknowledge receipt of Japanese government (MEXT) scholarships. This work was partially supported by JSPS KAKENHI, Grant Number JP19K11990.

References

[BODO18] Bailey S. W., Otte D., Dilorenzo P., O'Brien J. F.: Fast and deep deformation approximations. *ACM Trans. Graph. 37*, 4 (2018). URL: https://doi.org/10.1145/3197517.3201300, doi:10.1145/3197517.3201300. 1

[BZSL13] Bruna J., Zaremba W., Szlam A., LeCun Y.: Spectral networks and locally connected networks on graphs, 2013. arXiv:1312.6203. 3

[CO18] Casas D., Otaduy M. A.: Learning nonlinear soft-tissue dynamics for interactive avatars. *Proc. ACM Comput. Graph. Interact. Tech. 1*, 1 (July 2018). doi:10.1145/3203187. 3

[DdL13] Dionne O., de Lasa M.: Geodesic voxel binding for production character meshes. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2013), ACM Press, New York, NY, USA, pp. 173–180. doi:10.1145/2485895.2485919. 4

[GJ19] Gao H., Ji S.: Graph U-Nets, 2019. arXiv:1905.05178. 4

[HTC*13] Hahn F., Thomaszewski B., Coros S., Sumner R. W., Gross M.: Efficient simulation of secondary motion in rig-space. In

Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (2013), ACM Press, New York, NY, USA, p. 165–171. doi:10.1145/2485895.2485918. 2

[HTRS10] Hasler N., Thormählen T., Rosenhahn B., Seidel H.-P.: Learning skeletons for shape and pose. In *Proc. Symposium on Interactive 3D Graphics and Games* (2010), ACM Press, New York, NY, USA, p. 23–30. doi:10.1145/1730804.1730809. 2

[JZH*20] Jiang B., Zhang J., Hong Y., Luo J., Liu L., Bao H.: BCNet: Learning body and cloth shape from a single image, 2020. arXiv:2004.00214. 3

[Kav05] Kavan L.: Spherical blend skinning: A real-time deformation of articulated models. In *Proc. Symposium on Interactive 3D Graphics and Games* (2005), ACM Press, New York, NY, USA, pp. 9–16. doi:10.1145/1053427.1053429. 2

[KB18] Komaritzan M., Botsch M.: Projective skinning. *Proc. ACM Comput. Graph. Interact. Tech. 1*, 1 (July 2018). doi:10.1145/3203203. 2

[KCvO07a] Kavan L., Collins S., Žára J., O'Sullivan C.: Skinning with dual quaternions. In *Proc. Symposium on Interactive 3D Graphics and Games* (2007), ACM Press, New York, NY, USA, pp. 39–46. doi:10.1145/1230100.1230107. 2

[KCvO07b] Kavan L., Collins S., Žára J., O'Sullivan C.: Skinning with dual quaternions. I3D '07, Association for Computing Machinery, p. 39–46. URL: https://doi.org/10.1145/1230100.1230107, doi:10.1145/1230100.1230107. 6

[KJM08] Kaldor J. M., James D. L., Marschner S.: Simulating knitted cloth at the yarn level. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 1–9. doi:10.1145/1360612.1360664. 2

[LBBM18] Litany O., Bronstein A., Bronstein M., Makadia A.: Deformable shape completion with graph convolutional autoencoders. In *Proc. Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 1886–1895. doi:10.1109/CVPR.2018.00202. 3

[LCF00] Lewis J. P., Cordner M., Fong N.: Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Proc. SIGGRAPH '00* (2000), ACM Press, New York, NY, USA, pp. 165–172. doi:10.1145/344779.344862. 2, 6

[LD14] Le B. H., Deng Z.: Robust and accurate skeletal rigging from mesh sequences. *ACM Trans. Graph. 33*, 4 (July 2014), 84:1–84:10. doi:10.1145/2601097.2601161. 2

[LLK19] Lee J., Lee I., Kang J.: Self-attention graph pooling, 2019. arXiv:1904.08082. 4

[LMR*15] Loper M., Mahmood N., Romero J., Pons-Moll G., Black M. J.: SMPL: A skinned multi-person linear model. *ACM Trans. Graph. 34*, 6 (Oct. 2015). doi:10.1145/2816795.2818013. 1, 3, 6

[LSK20] Li T., Shi R., Kanai T.: DenseGATs: A graph-attention-based network for nonlinear character deformation. In *Proc. Symposium on Interactive 3D Graphics and Games* (2020), ACM Press, New York, NY, USA. doi:10.1145/3384382.3384525. 1, 2, 3, 4, 5, 9, 10, 11

[LWZH18] Li R., Wang S., Zhu F., Huang J.: Adaptive graph convolutional neural networks. In *Proc. 32nd AAAI Conference on Artificial Intelligence* (2018), AAAI Press, Palo Alto, CA, pp. 3546–3553. URL: https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16642. 3

[LZT*19] Liu L., Zheng Y., Tang D., Yuan Y., Fan C., Zhou K.: NeuroSkinning: Automatic skin binding for production characters with deep graph networks. *ACM Trans. Graph. 38*, 4 (July 2019), 114:1–114:12. doi:10.1145/3306346.3322969. 1, 2, 3, 4, 5, 9

[MG03] Mohr A., Gleicher M.: Building efficient, accurate character skins from examples. *ACM Trans. Graph. 22*, 3 (July 2003), 562–568. URL: https://doi.org/10.1145/882262.882308, doi:10.1145/882262.882308. 2

[mix] Mixamo auto-rigger. https://www.mixamo.com. Accessed: 2020-09-12. 6

[MK16] Mukai T., Kuriyama S.: Efficient dynamic skinning with low-rank helper bone controllers. *ACM Trans. Graph. 35*, 4 (July 2016). doi:10.1145/2897824.2925905. 2

[MTLT88] Magnenat-Thalmann N., Laperrière R., Thalmann D.: Joint-dependent local deformations for hand animation and object grasping. In *Proceedings on Graphics Interface '88* (1988), Canadian Information Processing Society, Toronto, Ont., Canada, pp. 26–33. doi:10.5555/102313.102317. 2

[Muk15] Mukai T.: Building helper bone rigs from examples. In *Proc. Symposium on Interactive 3D Graphics and Games* (2015), ACM Press, New York, NY, USA, pp. 77–84. doi:10.1145/2699276.2699278. 2

[MZS*11] McAdams A., Zhu Y., Selle A., Empey M., Tamstorf R., Teran J., Sifakis E.: Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (July 2011), 37:1–37:12. doi:10.1145/2010324.1964932. 2

[NAK16] Niepert M., Ahmed M., Kutzkov K.: Learning convolutional neural networks for graphs, 2016. arXiv:1605.05273. 3

[ÖBP*13] Öztireli A. C., Baran I., Popa T., Dalstein B., Sumner R. W., Gross M.: Differential blending for expressive sketch-based posing. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2013), pp. 155–164. doi:10.1145/2485895.2485916. 2

[PMRMB15] Pons-Moll G., Romero J., Mahmood N., Black M. J.: Dyna: A model of dynamic human shape in motion. *ACM Trans. Graph. 34*, 4 (July 2015). doi:10.1145/2766993. 3

[SGOC20] Santesteban I., Garces E., Otaduy M. A., Casas D.: SoftSMPL: Data-driven modeling of nonlinear soft-tissue dynamics for parametric humans. *Computer Graphics Forum (Eurographics 2020) 39*, 2 (2020), 65–75. URL: https://doi.org/10.1111/cgf.13912, doi:10.1111/cgf.13912. 3

[SOC19] Santesteban I., Otaduy M. A., Casas D.: Learning-based animation of clothing for virtual try-on. *Computer Graphics Forum (Eurographics 2019) 38*, 2 (2019), 355–366. URL: https://doi.org/10.1111/cgf.13643, doi:10.1111/cgf.13643. 3

[TGL*18] Tan Q., Gao L., Lai Y., Yang J., Xia S.: Mesh-based autoencoders for localized deformation component analysis. In *Proc. 32nd AAAI Conference on Artificial Intelligence* (2018), AAAI Press, Palo Alto, CA, pp. 2452–2459. URL: https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16585. 3

[VBG*13] Vaillant R., Barthe L., Guennebaud G., Cani M.-P., Rohmer D., Wyvill B., Gourmel O., Paulin M.: Implicit skinning: Real-time skin deformation with contact modeling. *ACM Trans. Graph. 32*, 4 (July 2013). URL: https://doi.org/10.1145/2461912.2461960, doi:10.1145/2461912.2461960. 2

[VCC*17] Veličković P., Cucurull G., Casanova A., Romero A., Liò P., Bengio Y.: Graph attention networks, 2017. arXiv:1710.10903. 3, 4, 5

[WSFM19] Wang T. Y., Shao T., Fu K., Mitra N. J.: Learning an intrinsic garment space for interactive authoring of garment animation. *ACM Trans. Graph. 38*, 6 (Nov. 2019). doi:10.1145/3355089.3356512. 3

[XZK*20] Xu Z., Zhou Y., Kalogerakis E., Landreth C., Singh K.: RigNet: Neural rigging for articulated characters. *ACM Trans. Graph. 39*, 4 (2020). URL: https://doi.org/10.1145/3386569.3392379, doi:10.1145/3386569.3392379. 2, 3

[YYM*18] Ying R., You J., Morris C., Ren X., Hamilton W. L., Leskovec J.: Hierarchical graph representation learning with differentiable pooling, 2018. arXiv:1806.08804. 4