Facial Animation Retargeting by Unsupervised Learning of Graph Convolutional Networks

Yuhao Dou
Graduate School of Systems Design
Tokyo Metropolitan University
Tokyo, Japan
0000-0002-3139-0932

Tomohiko Mukai
Graduate School of Systems Design
Tokyo Metropolitan University
Tokyo, Japan
0000-0002-7965-5426

***Abstract*—This paper proposes an unsupervised framework for retargeting human facial animations to different characters. Our method uses a branching structure of two parallel autoencoders and a variant of generative adversarial networks. The two autoencoder branches, composed of graph convolutional networks, share a common latent space through which the retargeting between different mesh structures can be performed. The shared latent codes are obtained by graph pooling operators, and the character face is reconstructed from the latent codes by the unpooling operators. The graph pooling and unpooling operators are designed based on multiple landmarks in optical-based facial motion capture systems. The GAN-based unsupervised learning method requires no paired training animation data between source and target characters. Our experimental results demonstrated that the proposed framework provides a reasonable estimation of a target facial expression that mimics a source character.**

***Index Terms*—facial animation, retargeting, unsupervised learning, graph convolutional network**

I. Introduction

Facial expressions of virtual avatars play an essential role in the narrative of digital content, such as movies and games, by conveying emotional and non-verbal information. Because people are familiar with human faces, they are susceptible to unreal and unnatural expressions in the graphics contents. Hence, producing believable facial animation requires an expensive workload to manually edit the keyframed or captured facial animations by experienced 3D animators, even though a high-fidelity facial capture system is available.

Many approaches have been proposed to improve the efficiency of facial animation production. Animation retargeting is a typical technique to eliminate the need for facial capture and manual editing by transferring existing animations from a source character to target characters. Traditional retargeting approaches establish the correspondence between facial models using multiple anchor points and transfer the anchor movements to preserve the characteristics of the target shape and the source animation. However, the retargeting problem is still challenging when the source and target face models show significant differences in shape and mesh structure, even though several deep learning methods have been proposed to overcome the problem. Moreover, conventional machine learning methods often require paired animation data that cannot be prepared in practice.

In this study, we propose an unsupervised framework to retarget facial animation to other characters using unpaired animation data. The proposed framework is inspired by a skeleton-aware retargeting method for character's full-body movements [1]. This method uses a branching structure of two parallel autoencoders and a variant of generative adversarial networks (GAN). The two autoencoder branches, which consist of graph convolutional networks (GCN), share a common latent space through which the retargeting between different skeletal structures can be performed. Based on this network architecture, our method learns shared latent space from both source and target facial animations through facial shape-aware graph pooling. The target facial expression is generated from the latent code by graph unpooling. These pooling and unpooling operators are designed based on multiple facial landmarks, commonly used in optical-based facial motion capture systems. Thanks to GAN-based learning, our method does not require any paired animations between source and target characters. The main contributions of this study are summarized as follows:

- The facial shape-aware graph pooling and unpooling operators with a facial landmark set for transferring facial deformation to different facial mesh structures
- GAN-based architecture to enable facial animation retargeting without paired training data

II. Related Work

Blendshape-based facial retargeting has been widely used for estimating optimal blendshape parameters to best approximate a source face expression [2], [3]. An artist-friendly method [4] optimizes the limited number of blendshape parameters to facilitate the animator's workflow. A spacetime method [5] formulates the blendshape-based facial retargeting problem as a Poisson equation to consider animation similarities in the velocity domain. These methods successfully reproduce the source animation, but the expression capability is restricted within a pose space defined by the set of blendshape targets of the target character. Compared to those blendshape-based

This work was supported by PlatinumGames Inc. and JSPS KAKENHI Grant Numbers 21K19822.

methods, our framework does not require the creation of the paired blendshapes.

Deformation transfer is used to generate a target facial expression by transferring the deformation gradient of the source animation [6]. This method allows direct transferring of vertex-wise facial movement and has been widely used to generate a set of blendshape targets by retargeting from the source rig [7], [8]. The quality of this method largely depends on the accuracy of surface correspondence between the source and target models. Although several methods have been developed to automatically find the shape correspondence [9], the purely geometric approaches do not consider the dynamics of facial movements.

There are many studies based on deep neural networks. A variational autoencoder has been devised for the facial expression transfer [10]. This method uses nonlinear expression embedding and expression domain translation. Neutral face rigging method [11] enables an end-to-end facial retargeting between different face models by learning latent expression space. The limitation of these methods is that the paired data must be prepared by manual annotations. An unsupervised learning method retargets facial expression to a target model via the rendered images of the source animation [12]. This network architecture is tailored to estimate the blendshape weights. In contrast, our method is designed to work directly on mesh data.

III. Proposed Framework

Our goal is to retarget animation between two face models which have different mesh structures and sets of expressions. This problem can be formulated as an unpaired cross-domain translation task. Let $\mathbf{S}_\mathrm{A}$ and $\mathbf{M}_\mathrm{A}^{f_a}, f_a \in \mathcal{F}_\mathrm{A}$ be the rest shape and the facial movement of a source character at f_a-th frame, respectively, where $\mathcal{F}_\mathrm{A}$ denotes the set of facial expressions and the f_a-th expression is computed by $\mathbf{S}_\mathrm{A} + \mathbf{M}_\mathrm{A}^{f_a}$. For the target character, the rest shape $\mathbf{S}_\mathrm{B}$ and movements $\mathbf{M}_\mathrm{B}^{f_b}, f_b \in \mathcal{F}_\mathrm{B}$ are also defined similarly. Our facial retargeting problem is defined by a data-driven mapping as $G_{A\to B} : ((\mathbf{S}_\mathrm{A}, \mathbf{M}_\mathrm{A}^{f}), \mathbf{S}_\mathrm{B}) \to (\mathbf{S}_\mathrm{B}, \tilde{\mathbf{M}}_\mathrm{B}^{f})$, where $\tilde{\mathbf{M}}_\mathrm{B}^{f}$ represents the estimated facial expression of the target character mimicking $\mathbf{M}_\mathrm{A}^{f}$, as shown in Fig.1. To solve this problem, the proposed framework is designed based on the skeleton-aware network [1], where the graph convolutional layers, pooling layers, and the loss function are customized for our facial retargeting task.

A. Network Design

Our framework uses a branching structure of two parallel autoencoders consisting of two encoders E_A and E_B, two decoders D_A and D_B, and two discriminators C_A and C_B trained for two characters A and B, as illustrated in Fig.2. Each autoencoder network comprises the graph convolutional layers, the graph pooling layer, and the graph unpooling layer, as shown in Fig.3, where V and $K(K \ll V)$ denote the number of mesh vertices and the number of landmarks, respectively, as detailed in §III-B. The two autoencoders share a common

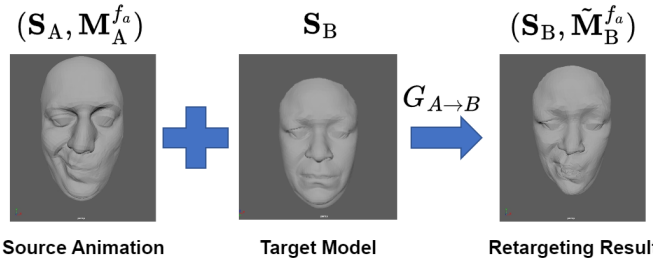

Fig. 1. The retargeting result is obtained by transferring the source animation to the target model

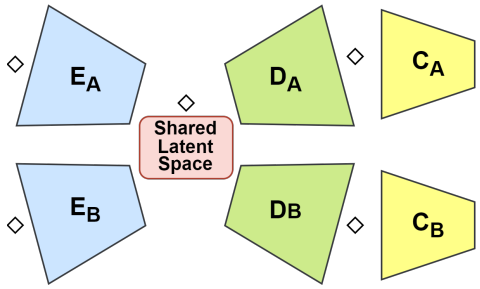

Fig. 2. The overview of the network architecture

latent space learned from the facial movements of different character models. The graph convolutional layer has a batch normalization layer to avoid the tendency to approach zero for the latent code, as shown in Fig.4.

Our network also contains two discriminators to determine whether the generated result is natural. The structure of the discriminator is similar to the encoder, as shown in Fig.5. Note that the last layer is the sigmoid function, such that the output of discriminators is bounded in the unit interval.

B. Landmark-based Graph Pooling and Unpooling

To project the facial deformation of two characters with different topologies into an identical latent space, we assume that the facial deformation of human-like characters is represented by the spatial movements of multiple facial landmarks. In fact, a detailed facial motion can be captured by detecting the translation of a limited number of reflective markers in the optimal facial capture system. We therefore define 68 landmark locations mainly according to the facial capture method [13]. We also define 20 additional landmarks to improve the retargeting performance, as shown in Fig.6.

Our method assigns each vertex to the nearest landmark. Let $\mathbf{s}_{v\in\{1,\cdots,V\}}$ and $\mathbf{p}_{k\in\mathcal{K}}$ denote the vertex positions and landmark position at the rest shape, respectively, where $\mathcal{K} = \{1, \cdots, K\}$ represents a set of landmarks and $K = 88$. Our system assigns v-th vertex to the closest landmark k_v as $k_v = \mathrm{argmin}_{\ k\in\mathcal{K}} \|\mathbf{p}_k - \mathbf{s}_v\|$ where $\|\cdot\|$ represents Euclidean distance. As a result, the set of facial vertices is divided into K subsets $\mathcal{V}_{k\in\mathcal{K}}$ where $\forall(k_1, k_2), \mathcal{V}_{k_1} \cap \mathcal{V}_{k_2} = \emptyset$ holds.

The shared latent space is learned using graph pooling and unpooling operators [14] based on the landmark set. In

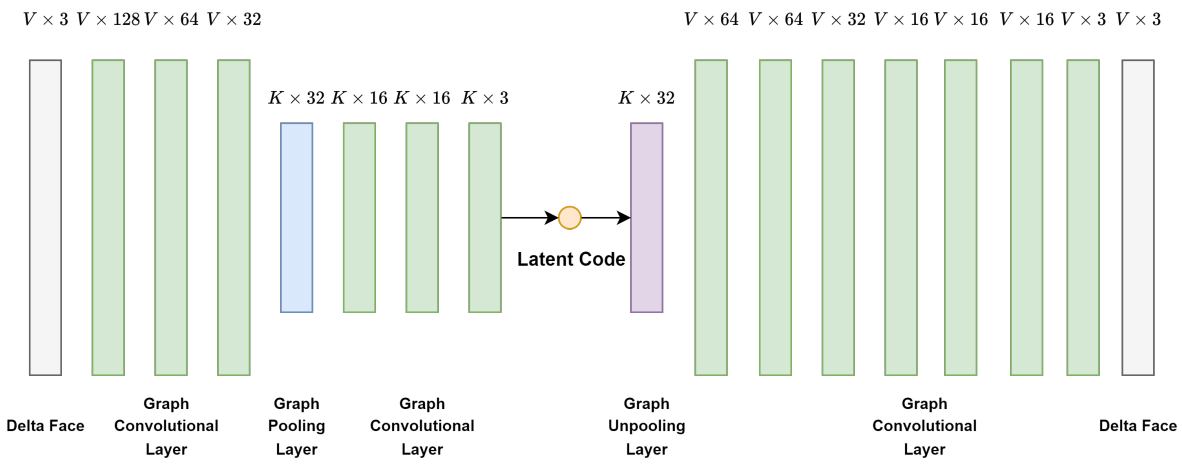

Fig. 3. The structure of graph autoencoder where V and K denote the number of facial mesh vertices and the landmarks, respectively.

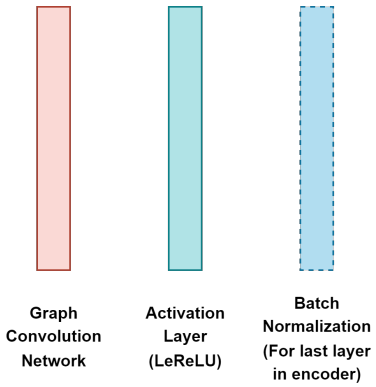

Fig. 4. The structure of graph convolutional layer

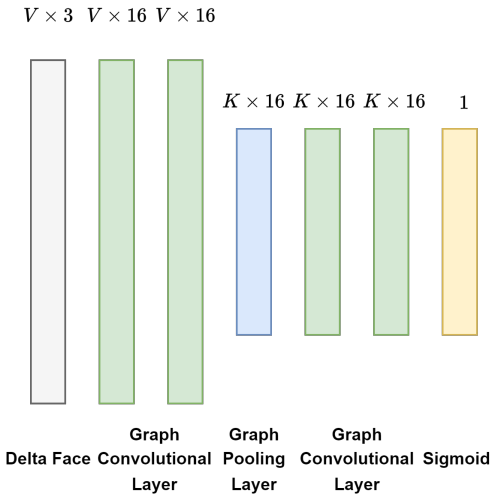

Fig. 5. The structure of discriminator

the encoder stage, the graph pooling operator calculates the average motion features of adjacent vertices assigned to the corresponding landmark. Specifically, we calculate the assignment matrix $\mathbf{P} \in \Re^{K \times V}$ according to the vertex-landmark correspondence. The assignment matrix is first initialized to be zero matrix $\mathbf{P} = \mathbf{0}$, and the matrix element is set to one as $P_{k,n} = 1$ if n-th vertex belongs to k-th landmark. The pooling layer transforms the input variable $\mathbf{X} \in \Re^{V \times H}$ and the adjacency matrix of the graph $\mathbf{A} \in \Re^{V \times V}$ into the output variable $\mathbf{Z} \in \Re^{K \times H}$ and the transformed adjacency matrix $\mathbf{A}_{\text{next}}$ using the assignment matrix $\mathbf{P}$ as follows.

$$\tilde{\mathbf{Z}} = \mathbf{PX} \in \Re^{K \times H} , \tag{1}$$

$$\mathbf{Z} = \text{Degree\_Norm}(\tilde{\mathbf{Z}}) \in \Re^{K \times H} , \tag{2}$$

$$\mathbf{A}_{\text{next}} = \mathbf{PAP}^T \in \Re^{K \times K} , \tag{3}$$

where Degree_Norm$(\cdot)$ divides the input variable by the number of assigned vertices.

In the decoder stage, the vertex movements are restored from the latent code using the unpooling operator. The feature at the landmark node is distributed to the related nodes set $\mathcal{V}_k$ by transforming the latent code $\mathbf{Z}$ using the transposed assignment matrix as $\mathbf{P}^T\mathbf{Z}$. The adjacent matrix $\mathbf{A}$ is restored from the original facial mesh.

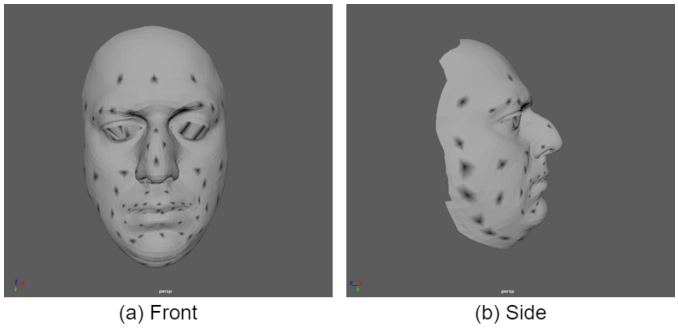

Fig. 6. Landmark locations

C. Loss Function

The loss function of our framework evaluates the weighted sum of three loss terms as follows.

$$L = \lambda_{\text{rec}} L_{\text{rec}} + \lambda_{\text{cyc}} L_{\text{cyc}} + \lambda_{\text{adv}} L_{\text{adv}} \ , \tag{4}$$

where L_{rec}, L_{cyc}, and L_{adv} are the reconstruction loss, the cycle consistency loss, and the adversarial loss, respectively. The weighting coefficients λ_{rec}, λ_{cyc}, and λ_{adv} are set to 1, 1, and 0.25 in our all experiments, respectively.

a) Reconstruction Loss: To train the autoencoder $(E_{\text{A}}, D_{\text{A}})$ for animations of the source character, reconstruction accuracy over all input expressions is evaluated to guarantee the network's generation ability. The reconstruction loss L_{rec} is defined as the expectation of squared reconstruction error.

$$L_{\text{rec}} = \mathbb{E}_{f \in \mathcal{F}_{\text{A}}} \left[\| D_{\text{A}}(E_{\text{A}}(\mathbf{M}_{\text{A}}^f)) - \mathbf{M}_{\text{A}}^f \|^2 \right] \ . \tag{5}$$

b) Cycle Consistency Loss: The cycle consistency loss L_{cyc} [15] serves as a method to impose regularization on the task of unidirectional translation for a facial deformation by making $G_{\text{A}\to\text{B}}$ and $G_{\text{B}\to\text{A}}$ to be inverses of each other such that $G_{\text{B}\to\text{A}}(G_{\text{A}\to\text{B}}(\mathbf{M}_{\text{A}}^f)) \approx \mathbf{M}_{\text{A}}^f$ and $G_{\text{A}\to\text{B}}(G_{\text{B}\to\text{A}}(\mathbf{M}_{\text{B}}^f)) \approx \mathbf{M}_{\text{B}}^f$ as follows.

$$L_{\text{cyc}} = \mathbb{E}_{f \in \mathcal{F}_{\text{A}}} \left[\| D_{\text{A}}(E_{\text{B}}(\tilde{\mathbf{M}}_{\text{B}}^f)) - \mathbf{M}_{\text{A}}^f \|^2 \right] \ . \tag{6}$$

Note that we experimentally confirmed that the cycle consistency loss provides higher quality for our facial retargeting tasks than the latent consistency loss used in the skeleton-aware network [1].

c) Adversarial Loss: Because the facial animation data is unpaired, there is no ground truth for comparing the retargeting result. Therefore, the adversarial loss L_{adv} is introduced into our framework, where a discriminator C_{B} determines whether the result for the face model is real or fake. Like other generative adversarial networks, the discriminator C_{B} is trained using the animations in $\mathcal{F}_{\text{B}}$ as the real examples and the output of $G_{\text{A}\to\text{B}}$ as the fake ones.

$$L_{\text{adv}} = \mathbb{E}_{f \in \mathcal{F}_{\text{A}}} \left[\| C_{\text{B}}(\tilde{\mathbf{M}}_{\text{B}}^f) \|^2 \right] + \mathbb{E}_{f \in \mathcal{F}_{\text{B}}} \left[\| 1 - C_{\text{B}}(\tilde{\mathbf{M}}_{\text{B}}^f) \|^2 \right] . \tag{7}$$

D. Training and Testing

The overview of the training stage of the retargeting from the source character A to the target B is shown in Fig. 7. The facial movement of source character is encoded by E_{A} into the latent code. Then, decoder D_{A} reconstructs the facial movement of the source character via L_{rec}. The retargeting is achieved by feeding the latent code to decoder D_{B} and the L_{cyc} is evaluated between the original latent code generated by E_{A} and the translated latent code by E_{B}. In the test stage, the trained network retargets facial expressions from A to B, as shown in Fig.8. Retargeting is achieved by using D_{B} to the encoded delta face by E_{A}. Note that these describe only the process of A $\rightarrow$ B retargeting, while the training of B $\rightarrow$ A retargeting is symmetric.

IV. Experiments

We implemented the proposed framework using PyTorch Geometric. All experiments were performed on a PC with an NVIDIA GeForce RTX 3080 GPU with 10 GB RAM and an AMD Ryzen 9 5950X/3.4GHz CPU with 128 GB RAM. We used the Adam optimizer and set the batch size to 32. The training stage took about 20 hours for 1000 epochs.

As the GAN architecture network is well-known for its difficulty in training, we used the following two tricks.

- We trained with two time update rules, applying different learning rates for the generator and discriminator. The learning rate for the generator and discriminator were 1e-3 and 1e-4, respectively.
- We trained the generator more in the training stage. The ratio of the number of iterations of the discriminator and the generator was 1:5.

A. Data Processing

We used the Multiface dataset [16] to train and test our framework. This dataset contained high-quality 4D scans of 13 identities, which covered a variety of facial expressions like smiling, frowning, and reading long sentences, as shown in Fig.9.

In the data preprocessing, we first clipped the face region of the character model, as shown in Fig.10, because the other head region did not deform. Next, the facial expressions with the most significant deformation from the rest shape were extracted to improve the training performance. Specifically, our system selected the manually-specified number of animation frames that showed large deformations from the rest shape $\|\mathbf{M}^f - \mathbf{S}\|$. This process improved the data distribution tendency by eliminating many neutral expressions that appear in the multiface dataset. Finally, we standardized the dataset to reshape the input data distribution into a Gaussian distribution.

As a result, the number of vertices of characters A and B were $V_{\text{A}} = 3264$ and $V_{\text{B}} = 3165$, respectively. The number of expressions was reduced to $|\mathcal{F}_{\text{A}}| = |\mathcal{F}_{\text{B}}| = 3840$ in our experiments. The ratio of the train test split was 8:2, and we did not use the validation set.

B. Qualitative Evaluation

This section shows qualitative evaluations of the retargeting results. Each figure shows the original shape, the retargeting result, and the ground truth from left to right.

Fig.11 shows the retargeting results from character A to character B. Fig.11(a) shows the "mouth open and tongue out" results. In this example, the deformation around the mouth was successfully transferred from the source model to the target. However, the lip shape was unclear and the generated mesh around the jaw was not smooth compared to the ground truth. In the "stretched smile" result (b), the mouth shape with raised corners was well transferred to the target, whereas the retarget result significantly differed from the ground truth. This result indicated that the expression specific to character A was successfully retargeted to a different character B. In the "raising cheeks" result (c), although the overall deformation of

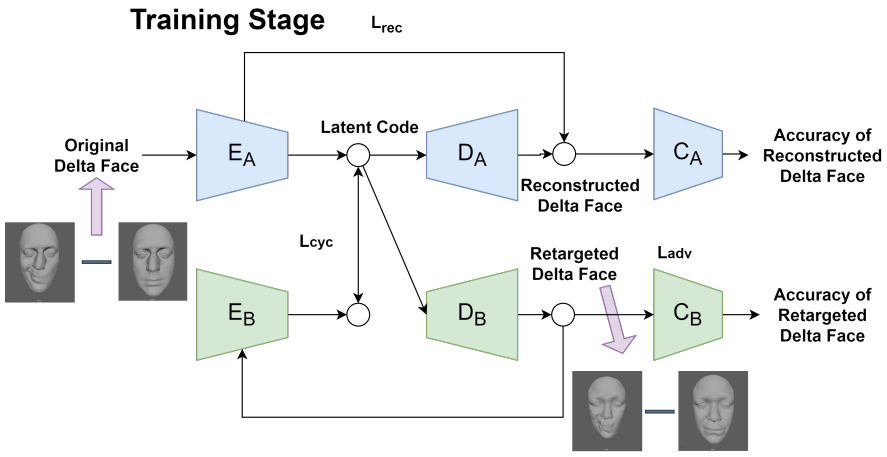

Fig. 7. Network in training time from character A to B.

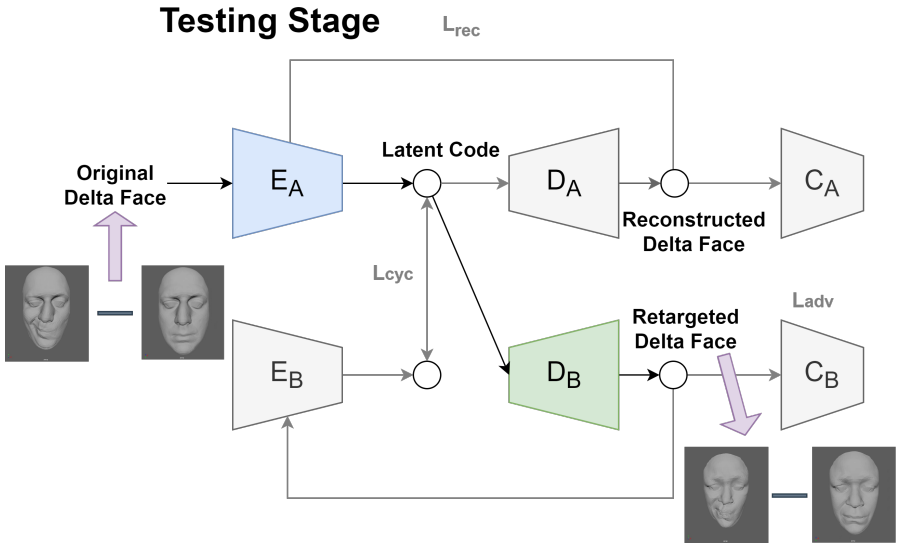

Fig. 8. Network in testing time from character A to B.

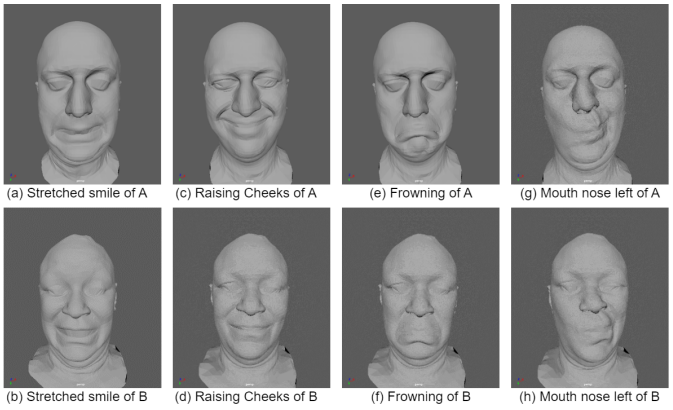

Fig. 9. Facial expressions in Multiface dataset [16].

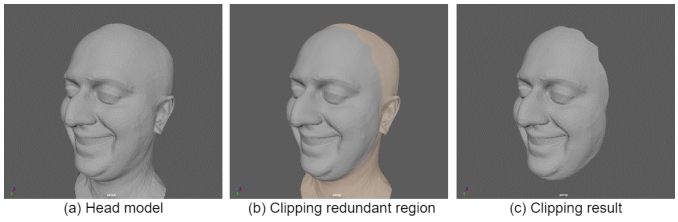

Fig. 10. Clipping of character head model.

the lips was transferred to the target, some artifacts appeared on the upper lip, and the mouth was not so clearly opened. In the ”pronunciation ’Oooo.’” (d), the lips puckering of the original face model was retargeted to approximate the ground truth well. However, the jaggy artifacts appeared around the mouth.

Fig.12 shows the retargeting results from character B to A. In the ”mouth open and tongue out” example (Fig.12(a)), the mouth shape of the source model was successfully transferred

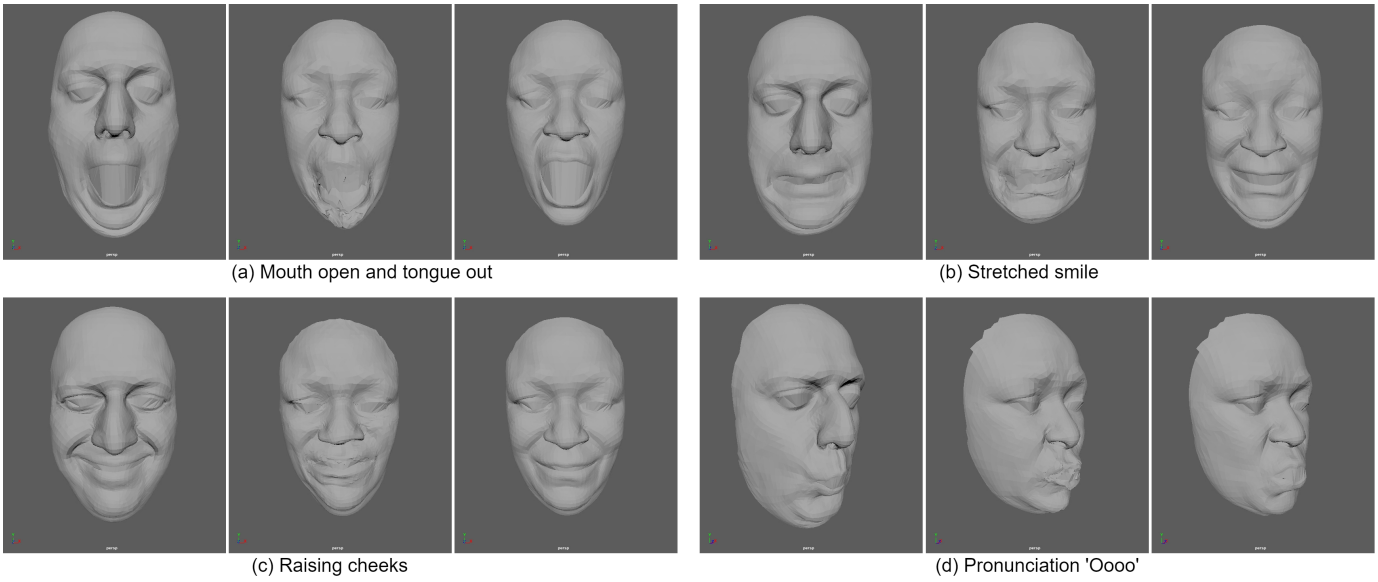

Fig. 11. Retargeting from Character A to B

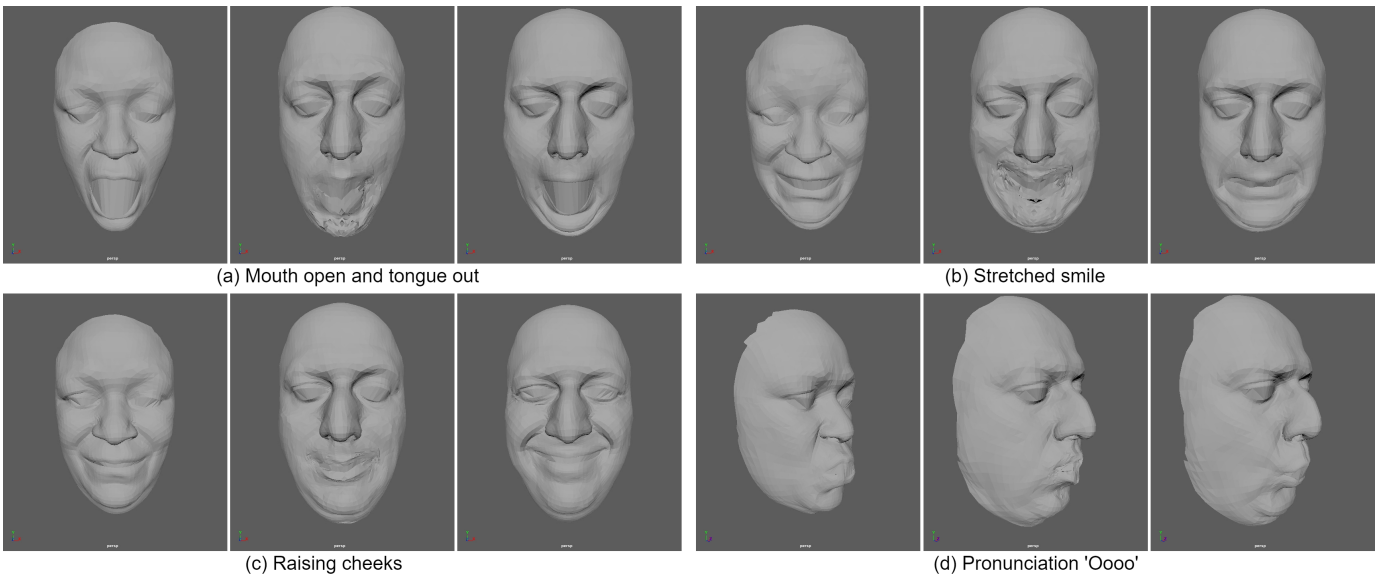

Fig. 12. Retargeting from Character B to A

to the target and produced the facial shape close to the ground truth. However, noise-like artifacts appeared around the jaw. In the results of ”smile stretched” (b), the retarget result showed a considerable difference to character B’s expression, as well as the transfer results in the reverse direction shown in Fig.11(a). These results demonstrated the potential capability of our expression retargeting, but non-negligible mesh collapses were caused in the lower half of the face. Fig.12 (c) and (d) show the ”raising cheeks” and ”pronunciation ’Oooo’” results. In these results, the target model successfully reproduced the mouth deformation of the source face although the detailed shape around the mouth was lost.

These results verified that our method could retarget facial animation between two human characters with distinct topologies when the facial deformation was not complex. For example, the deformations of ”mouth open and tongue out” and ”raising cheeks” were successfully transferred to the target. However, the retargeting did not work for several expressions. For example, the retargeting of ”raising cheeks” worked well from character A to B, but it failed for the reverse direction, as shown in Fig.11(c) and Fig.12(c). The retargeting also failed for several complex facial deformations. For instance, our method caused deformation artifacts in the combination of the motion ”raise upper lip” and ”scrunch nose.” Moreover, several results caused non-smooth artifacts, as shown in Fig.14. The possible reason is that our method did not consider the smoothness of mesh in our loss function.

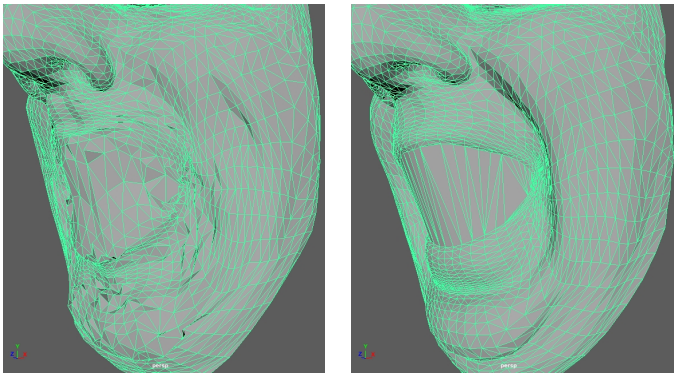

Fig. 13. The topology is different between the result and ground truth

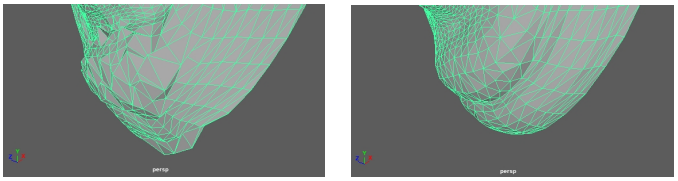

Fig. 14. The comparison of smoothness between the result and ground truth

V. Conclusions

We have proposed an unsupervised facial retargeting method for human characters. The proposed neural network is trained to encode the human facial deformation into a common latent space using graph pooling and unpooling operations based on facial shape-aware landmarks. The latent code is then decoded back to a target mesh structure, which enables transferring facial animations to different character models. Our experimental results verified that the proposed framework can reasonably retarget human facial animations using unpaired facial animation data. We believe that the proposed method potentially enhances the production efficiency in entertainment, human-computer interaction, virtual reality, and related fields.

However, the results of the retargeted animation still need to be improved since many artifacts cause the generated facial meshes. For example, the jaggy artifacts are generated in the retargeting results because the spatial smoothness is not considered in the loss function. Moreover, the feature in the landmark node is simply distributed to the adjacent nodes, so the generated deformation is averaged in the corresponding region. This simple approach results in undesirable deformation. We will explore potential solutions to overcome these issues in our future work. For example, we will add a domain-adversarial loss using two classifiers for the latent code [17]. Moreover, automatic tagging techniques like [18] should be introduced to reduce the manual labor of landmark tagging. We will also investigate a more practical approach to enhancing retargeting quality by considering temporal coherence.

References

[1] K. Aberman, P. Li, D. Lischinski, O. Sorkine-Hornung, D. Cohen-Or, and B. Chen, "Skeleton-aware networks for deep motion retargeting," ACM Transactions on Graphics (TOG), vol. 39, no. 4, pp. 62–1, 2020.

[2] P. Joshi, W. C. Tien, M. Desbrun, and F. Pighin, "Learning controls for blend shape based realistic facial animation," in Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2003, p. 187–192.

[3] E. Chuang and C. Bregler, "Performance driven facial animation using blendshape interpolation," Computer Science Technical Report, Stanford University, vol. 2, no. 2, p. 3, 2002.

[4] Y. Seol, J. Seo, P. H. Kim, J. P. Lewis, and J. Noh, "Artist friendly facial animation retargeting," ACM Transactions on Graphics (TOG), vol. 30, no. 6, pp. 1–10, 2011.

[5] Y. Seol, J. P. Lewis, J. Seo, B. Choi, K. Anjyo, and J. Noh, "Spacetime expression cloning for blendshapes," ACM Transactions on Graphics (TOG), vol. 31, no. 2, pp. 1–12, 2012.

[6] R. W. Sumner and J. Popović, "Deformation transfer for triangle meshes," ACM Transactions on graphics (TOG), vol. 23, no. 3, pp. 399–405, 2004.

[7] J. Saito, "Smooth contact-aware facial blendshapes transfer," in Proceedings of the Symposium on Digital Production, 2013, pp. 7–12.

[8] C. Mousas and C.-N. Anagnostopoulos, "Structure-aware transfer of facial blendshapes," in Proceedings of the 31st Spring Conference on Computer Graphics, 2015, pp. 55–62.

[9] S. Bian, A. Zheng, L. Gao, G. Maguire, W. Kokke, J. Macey, L. You, and J. J. Zhang, "Fully automatic facial deformation transfer," Symmetry, vol. 12, no. 1, p. 27, 2019.

[10] J. Zhang, K. Chen, and J. Zheng, "Facial expression retargeting from human to avatar made easy," IEEE Transactions on Visualization and Computer Graphics, vol. 28, no. 2, pp. 1274–1287, 2020.

[11] D. Qin, J. Saito, N. Aigerman, T. Groueix, and T. Komura, "Neural face rigging for animating and retargeting facial meshes in the wild," arXiv preprint arXiv:2305.08296, 2023.

[12] S. Kim, S. Jung, K. Seo, R. B. i Ribera, and J. Noh, "Deep learning-based unsupervised human facial retargeting," in Computer Graphics Forum, vol. 40, no. 7. Wiley Online Library, 2021, pp. 45–55.

[13] J. Shen, S. Zafeiriou, G. G. Chrysos, J. Kossaifi, G. Tzimiropoulos, and M. Pantic, "The first facial landmark tracking in-the-wild challenge: Benchmark and results," in Proceedings of the IEEE international conference on computer vision workshops, 2015, pp. 50–58.

[14] H. Gao and S. Ji, "Graph u-nets," in international conference on machine learning. PMLR, 2019, pp. 2083–2092.

[15] A. Anoosheh, E. Agustsson, R. Timofte, and L. Van Gool, "Combogan: Unrestrained scalability for image domain translation," in Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2018, pp. 783–790.

[16] C.-h. Wuu, N. Zheng, S. Ardisson, R. Bali, D. Belko, E. Brockmeyer, L. Evans, T. Godisart, H. Ha, X. Huang et al., "Multiface: A dataset for neural face rendering," arXiv preprint arXiv:2207.11243, 2022.

[17] A. Royer, K. Bousmalis, S. Gouws, F. Bertsch, I. Mosseri, F. Cole, and K. Murphy, "Xgan: Unsupervised image-to-image translation for many-to-many mappings," in Domain Adaptation for Visual Understanding. Springer, 2020, pp. 33–49.

[18] D. Jiang, Y. Jin, F.-L. Zhang, Z. Zhu, Y. Zhang, R. Tong, and M. Tang, "Sphere face model: A 3d morphable model with hypersphere manifold latent space using joint 2d/3d training," Computational Visual Media, vol. 9, no. 2, pp. 279–296, 2023.