The Journal of Supercomputing (2023) 79:6347–6372
https://doi.org/10.1007/s11227-022-04885-7

Customizing blendshapes to capture facial details

Ju Hee Han[1] · Jee In Kim[1] · Jang Won Suh[2] · Hyungseok Kim[1]

Accepted: 5 October 2022 / Published online: 3 November 2022

Abstract

Blendshape technique is an effective tool in the computer facial animation. Every character requires its own unique blendshapes to cover numerous facial expressions in the Visual Effects industry. Despite outstanding advances in this area, existing techniques still need a professional artist's intuition and complex hardware. In this paper, we propose a framework for customizing blendshapes to capture facial details. The suggested method primarily consists of two stages: Blendshape generation and Blendshape augmentation. In the first stage, localized blendshapes are automatically generated from real-time captured faces with two methods: linear regression and an autoencoder Han (in: IEEE International Conference on Big Data and Smart Computing (BigComp) 2021) (2021). In our experiment, face construction with the former outperforms that of the later method. However, generated blendshapes are slightly missing the source features, especially mouth movements. To overcome this, in the last stage, we extend Han (in: IEEE International Conference on Big Data and Smart Computing (BigComp) 2021), (2021) by adding a blendshape incrementally to minimize erroneous expression transfer.

Keywords Blendshapes · Facial retargeting · Facial animation · Autoencoder · Linear regression · Deep learning

✉ Hyungseok Kim
hyuskim@konkuk.ac.kr

Ju Hee Han
juheee37@gmail.com

Jee In Kim
jnkim@konkuk.ac.kr

Jang Won Suh
John.suh@ellxi.com

[1] Graduate School of Computer Science, Konkuk University, Seoul, South Korea

[2] Embedded Intelligence Lab., Ellexi, Seoul, South Korea

1 Introduction

Computer facial animation has been in the spotlight for a long time in the various entertainment industries, such as movie production, video games and VR/AR [10]. Traditionally, facial animation is in the realm of artists' manual skills [27]. Even though their creation has brought out great qualities of facial animation, intensive labor time and an overwhelming workload remain to be solved. To alleviate such discomfort, ceaseless efforts to produce facial animation with the state-of-the-art technologies have led to significant accomplishments in the computer animation [11]. However, it is still challenging to model the human face because of its realism and computational complexity [29].

Blendshape facial animation is widely used to create the realistic human face [23, 29]. Blendshapes are defined as a set of key poses including a neutral pose, a mouth press right pose, or a jaw open pose in the facial animation [23]. When it comes to facial expressions, feature extraction technique plays a key role in face recognition [22]. But, in the computer animation, blendshape based approach enables animating a character's face in real-time by capturing an actor's face [39]. Fundamentally, this technique deforms a base shape to describe facial expressions with combined weighted parameters. It allows animators to craft various expressions by combining parameters with significant efforts.

Adjusting parameters of blendshapes is a much-studied topic as it is involved in delivering realistic facial motions. Extracting the parameters from a blendshape model is designed to address the difficulties of controlling blendshapes [17]. Ten blendshapes are derived from the face capture, which is low compared to modern-day production standards to generate realistic animation. They compute blending weights in the least squares sense based on their assumption that the facial motion can be expressed as a linear system. Furthermore, various neural networks such as ANN, RBFN [13] or autoencoders [2, 44] have been used to address blendshape and its weights as well as face reconstruction [40, 42]. In this work, detailed facial reconstruction serves as a tool to measure the performance of customized blendshapes, so an autoencoder to define blendshapes as in [40] is adopted. Overall, this research evaluates the performance by comparing the template method that yields template blendshapes. This method gives a technically adept performance on face tracking and recreating faces by using its computed blendshapes. But its blendshapes are unable to cover individual unique facial features, so we propose a novel method to generate personalized blendshapes to capture facial details.

Earlier works use a casual device to process face capture quickly and easily [4, 35]. While Kim et al. [20] adds blendshapes based on PCA to perform facial retargeting, a regression-based method is instead adopted to minimize alignment errors of face reconstruction [4]. By synthesizing the ideas from previous works, this work takes benefits of effectiveness and convenience in that blendshapes are initiated by capture process with an ordinary device and optimized by an application of neural networks and supplementation, leading to detail preserved facial expressions.

The main challenge in performing natural facial animation is to generate proper blendshapes which cover individual unique details. For example, blendshapes

of a person who is unable to move his or her mouth freely should not be same with those of the normal face. In spite of advanced technologies, sculpting adequate blendshapes still depends on skillful artists, which is very expensive and time-consuming.

The goal of this paper is to generate customized blendshapes to capture facial features without reliance on laborious handiwork by taking a simple device for capture. In this work, we propose a two-staged framework for personalized blendshapes. In the first stage, we create individually optimized blendshapes which can cover more facial features than the template model by using two methods: linear regression and an autoencoder. One of blendshape generation methods is determined depending on root mean squared errors. The next stage is blendshape augmentation that updates prior blendshapes by adding a blendshape incrementally to recover original expressions with accuracy. The most error-occurring frame is chosen based on mean squared errors between the original face and the creation. We have assumed that unsatisfactory results with created blendshapes come from the lack of blendshapes, so blendshapes are increased iteratively until mean squared errors are reduced by 80%. Finally, weights are recalculated based on a transpose-based inverse solution.

2 Related work

2.1 Blendshape generation

There have been continuing efforts to employ blendshape models in the computer animation. Blendshape interpolation has been researched actively for decades [17, 30–32]. Facial expressions are represented by controlling parameters in the parameterized models, which is uneasy to define proper parameter sets and combine those values for desirable results [32]. Optimal blendshape modeling for each character plays a significant role in high-fidelity facial animation [3, 8, 43].

Blendshapes can be generated from captured images, video, or RGBD data. Pighin et al. [33] builds photorealistic 3D facial models from multiple images. Basic expressions are captured and then used to define appropriate blendshapes with manual marking on feature points on the face images. This process is repeated to produce personalized blendshapes for each subject. Expressions are automatically transferred by fitting a rendered model onto captured data of the same identity. This method generates convincing facial animation, but the burden of extreme workload is inevitable and requires both source blendshapes and corresponding target models. Our method aims to reduce a great deal of manual labor and time. Further, our model only requires target blendshapes for facial animation.

PCA of motion capture data is one strategy to choose individual target blendshapes [3, 23, 34]. PCA is desirable in that we can obtain the most adequate blendshape with a certain number of blendshapes automatically and the acquired blendshape has orthogonality which is adept for fitting. PCA-based models have been suggested for facial motion capture data [26] and retargeting [13, 38]. PCA and blendshapes are similar in that facial expressions can be represented from either a linear combination of PCA eigenvectors or a linear combination of blendshapes.

Rather than generating blendshapes directly, PCA is substituted for blendshapes [13]. While this approach helps to avoid computational complexity, it is unsuitable for human manipulation as other PCA models are [34, 36]. To be specific, it tends to ignore semantics, so it is difficult to recognize each key pose intuitively [37] and control facial parameters [23]. In addition, face variations tends to be uncovered due to the limited capacity of PCA dimensions.

To overcome PCA issues, 3D regression algorithm is devised to model a user-specific blendshape from 2D video frames [4]. Pre-defined facial expressions are captured by an ordinary web camera instead of facial markers. The captured data is used to find the best fit regressor that adapts 2D images to 3D shape in the training stage. This process includes a two-part blendshape generation procedure. Facial expressions can be approximated in association with three components—identity weights, expression weights, and the transformation—that are involved to establish correspondences between the projected 3D landmark and labeled positions on the image. The next step is to adjust the identity weights to be built for the same individual. Finally, customized blendshapes are modeled after iterations until the fitting converges. Their system is somewhat similar in that our method requires a simple device instead of specialized hardware which needs careful operation and does not use PCA. However, our work differs in that it employs 3D face geometry as input to train blendshapes. Also, the identity weights are unnecessary in the model as our blendshapes are optimally personalized to an individual face.

Recent research has studied the method of creating personalized blendshapes within a short amount of time and reducing the intervention of skillful artists. Casas et al. similarly takes an image-based approach where blendshapes are modeled rapidly from RGB-D data with a Kinect [5]. Their methods require both source and target scans for facial retargeting by aligning two blendshapes. Follow-up work proposes an end-to-end system to generate individually optimized blendshapes automatically using a self-supervised neural network [25]. Based on pre-defined template blendshapes, they construct target blendshapes by estimating blending weights and subsequently tuning trained blendshapes to preserve fine details. Both works have shown successful results in terms of generating user-specific blendshapes within a short amount of computation time. However, additional blendshapes besides the target blendshapes are demanded for this type of implementation. Any extra blendshapes are unnecessary as our method only requires a single set of blendshapes for a target individual for facial animation.

The suggested method of adopting user-specific blendshapes is to find the best match of original facial expressions [19]. These blendshapes are selected by their fitting method for the more precise target face's representation, but because their approach has assumed that the facial shape is changeable regionally, some exaggerated expressions might not be handled properly.

2.2 Blendshape augmentation

A face model can be improved by augmenting blendshapes, resulting in producing extraordinary expressions. Blendshapes combined with facial rigging is used to

drive convincing facial animation [28]. Rather than finetuning constructed blendshapes as in [25], a similar approach as [20] is taken by adding a blendshape incrementally after finding problematic frames in the stage of blendshape reconstruction. While PCA is employed to find error-occurring frames in [20, 12] proves that MSE is effective in comparing mouth expressions between the original face and the recreated face so this method is simply adopted in this work. Their work [20] supplements blendshapes based on landmark correspondences between the source and the retargeted model. Rather, we determine an additional blendshape according to the extent of how the reconstructed target face accurately recovers the captured expressions based on vertex-to-vertex differences.

To build facial animation where fine details are preserved, blendshapes are augmented by an auxiliary texture image that is defined by vertex and normal data [41]. This approach has high benefits in terms of the resolution of the 3D face and head details such as hair, but unexpected blurry artifacts are observed. Texture is also utilized to sculpt facial expressions with blendshapes [9]. Our work differs in the method of updating blendshapes in that blendshapes are increased to rebuild original expressions rather than modifying blended meshes.

Once the blendshape augmentation is completed, blending parameters need to be redesigned to correspond to increased blendshapes in our system. Professionals commonly manipulate blendshape weights for desirable face reconstruction. It is simply a controlled "pin and drag" system in the painting interface brought about by mathematical operations [24]. Further, direct editing method is supplemented by putting local constraints on facial geometry based on geodesic circles [6]. Most approaches were based on inverse problem solutions [7], particularly the pseudo-inverse method [1, 2, 37]. With the rise of deep learning, weights were formed with an autoencoder [2]. The device automatically computes weights, so the model does not consider weight extraction in the phase of blendshape generation. However, there is need to find proper coefficients as additional blendshapes are built. An autoencoder is employed to create blendshapes but not in the case of blending weights. In this work, a transpose-based solution is first used to compute a set of temporal weights for modeling an additional blendshape, and then weights are regained by linear system-based methods.

3 Blendshape customization

3.1 Blendshape generation

It is well known that generating personalized blendshapes for realistic and natural facial expressions is challenging in computer animation [29]. In spite of the state-of-the-art technology, there are still difficulties with modeling human faces because the human face is composed of various fine muscles that are deeply connected, especially the eyes and mouth. To generate a specific character's face animation, sometimes more than 110 blendshapes are utilized specifically for eyes and mouth regions. Our network architecture for blendshape customization has two main sections: (1) Blendshape generation which induces customized

blendshapes automatically from the captured face model and (2) Blendshape augmentation by adding supplementary blendshapes incrementally to minimize errors between the captured face and the new face driven from the previous section. The overall system of blendshape generation is described in Fig. 1.

In our model, the captured face geometry F_{id} is defined as follows where *id* denotes the identity [19].

$$F_{id} = B^0_{id} + \sum_{i=1}^{52} w^i_{id}(B^i_{id}) \tag{1}$$

$$\sum_{i=1}^{52} w^i_{id} = W_{id} \tag{2}$$

$$\sum_{i=1}^{52} B^i_{id} = B_{id} \tag{3}$$

It is a linear combination where w^i_{id} is a vector of *id*'s coefficients and B^i_{id} is a $(1220{\cdot}3) \times 52$ matrix whose column vectors are respective blendshapes. B^0_{id} denotes the neutral face with no facial expression. The notations are expressed simply in Eq. 2 and Eq. 3. The main goal in this stage is to generate individual blendshapes. We expect that there is a linear relationship between the blendshape and the face geometry. It is impossible to derive the blendshape B_{id} for a specific character because there are too many unknown factors in Eq. 1. A neural network-based system is adopted to solve the problem. Previous studies have shown that autoencoders have been applied to many studies regarding facial expressions [14, 18]. Also, the variational autoencoder is used to train the face and it is expected that this tool can be effective for extracting facial basis [21]. In our method, the decoder performs as blendshapes so that the face geometry can be gained from blendshape coefficients.

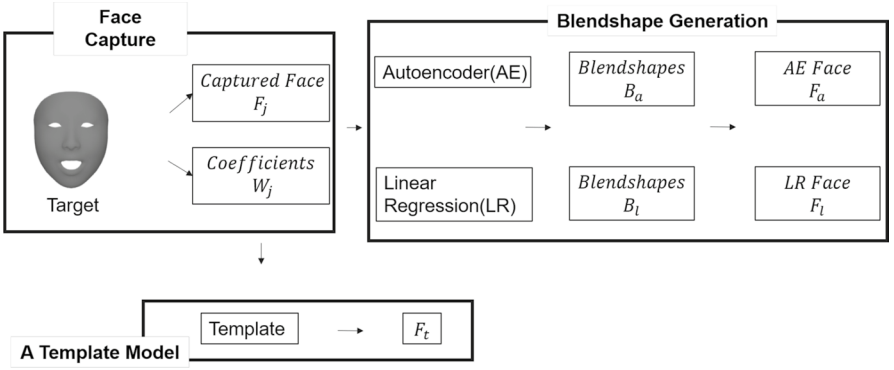

Fig. 1 Blendshape generation

The problem can be solved by implementing simple linear regression and an autoencoder technique.

3.1.1 Linear regression

Blendshapes can be generated by using two methods: linear regression and an autoencoder. We design a simple linear regression as seen in Fig. 2. B_l and F_l represent the blendshapes and face respectively, driven from the linear regression method. We refer to this face as the Linear Regression (LR) face. In the training, the captured face (F_j) including coefficients (W_j) is used as input data. To obtain precise blendshapes (B_l) and the neutral face (B_l^0), root mean square errors (RMSE) of vertices between the input (F_j) and the output face (F_l) are applied as the loss function:

$$\text{loss}_{lr} = \text{MSE}(F_j, F_l) \tag{4}$$

The generated LR face is constructed by employing a large set of coefficients (W_j) combined with the trained pair.

3.1.2 Autoencoder

Fundamentally, our concern is to identify the system which extracts high-dimensional face data from low-dimensional coefficients. Accordingly, an autoencoder is implemented as seen in Fig. 3. Our method is slightly different from existing autoencoder models in that a decoder maps both the code and (W_j) to reconstruct the input. There are already 52 coefficients (W_j) so it is not essential to learn encoding a set of

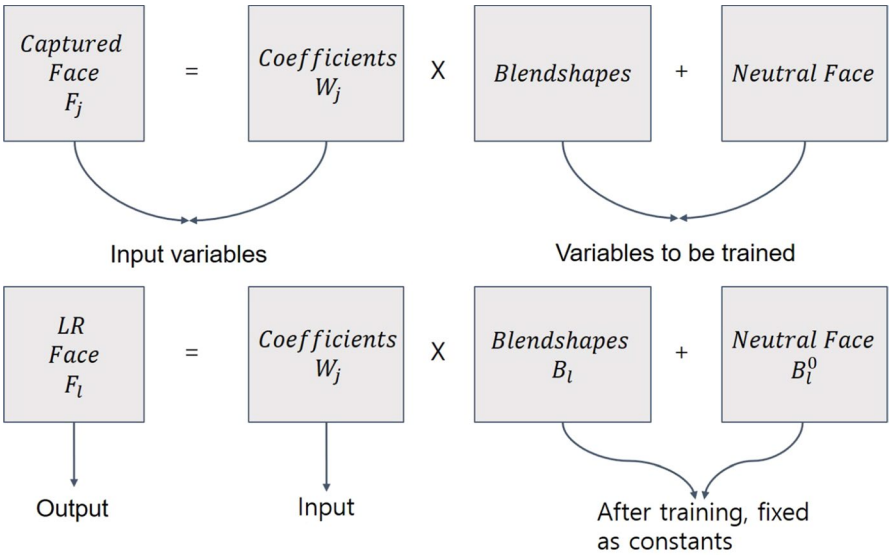

Fig. 2 The proposed workflow for linear regression. (*Upper*) The training phase. (*Lower*) The testing phase

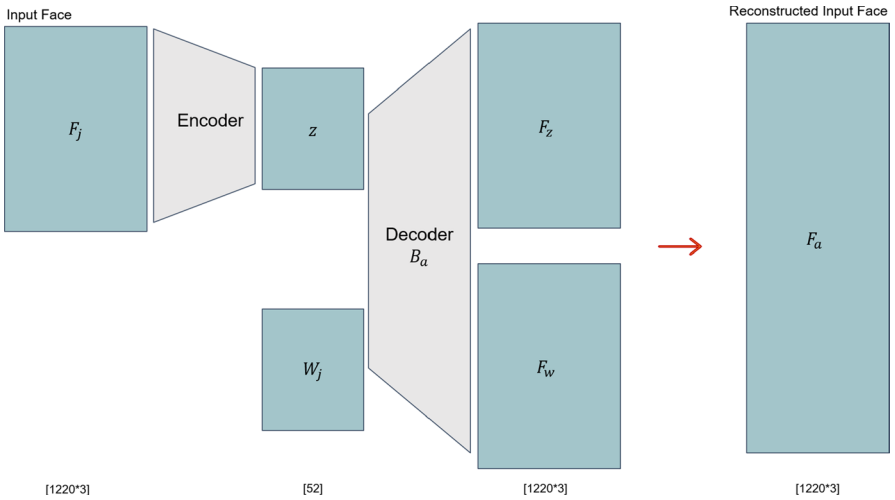

Fig. 3 Our autoencoder system for blendshape construction

data (F_j) for dimensionality reduction. This is not a significant issue because a large dataset construction from a small number of datasets is a key point in our model, as mentioned earlier. In our autoencoder, the trained decoder performs as blendshapes (B_a). The encoder compresses the input (F_j) and produces z code. Our loss function consists of two terms: weight loss that applies the mean square error (MSE) between W_j and z, and MSE between the reconstructed face F_z and F_w from z and W_j, respectively. F_z and F_w are respectively computed as Eq. 5 and Eq. 6. We employ the Adam optimizer for training with a batch size of 52 and an initial learning rate of 0.01 with a conditional decay every epoch during 5000 epochs.

$$F_z = zB_a + B_j^0 \tag{5}$$

$$F_w = W_j B_a + B_j^0 \tag{6}$$

After training, we have the face formed by the autoencoder, Autoencoder (AE) face. A comparison of the two generated faces—the LR face (F_l) and the Autoencoder face (F_a)—is performed with RMSE in each frame. In our experiment, RMSE(F_j, F_a) is larger than RMSE(F_j, F_l). Therefore, the LR blendshape is finally adopted for blendshape creation.

3.2 Blendshape augmentation

In the previous process, the target's personalized blendshapes are generated. Still, the created blendshapes fail to reproduce the original face perfectly. It is supposed that the earlier method is insufficient to cover more facial details and this failure might come from the lack of some particular blendshapes. As in Fig. 4, inaccurate

Fig. 4 Inaccurate face construction in terms of the mouth

face construction is mainly observed in mouth movements. From left is the captured target face, middle is the template face, and right is the LR face. The middle and right faces show incorrect mouth expressions compared to the original face. Moreover, the template is a standard face that does not reflect the original facial shape. In other words, it only displays the same identity, neglecting personal unique characteristics included a facial shape or eyes expressions. Our blendshapes from Sect. 3.1 are updated to minimize errors in the mouth movements and preserve individual facial features.

3.2.1 Problematic frames

There are some problematic frames where our method slightly misses facial expression details. The frame(Fm_k) where the largest MSE(F_j, F_l) is measured as follows.

$$\text{Frame } Fm_k = \underset{k}{\text{argmax}}\, \text{MSE}(F_j, F_l) \text{ for every frame} \tag{7}$$

3.2.2 Blendshape reconstruction

After finding the most problematic frame k, the kth frame is displayed on the screen as in Fig. 5. It turns out that errors have occurred largely in the case of mouth opening. [17] minimizes the sum of differences between the recorded motion captures and corresponding blendshapes by supplementing the blendshape basis by using the radial basis function. Rather than using radial basis function, the blendshapes are complemented by using the differences of F_j and F_l. It can be solved by adding one blendshape and setting its weight as 1 because our assumption is that MSE(F_j, F_l) comes from the lack of a blendshape which correlates to the mouth movement. So, let a temporal 53rd blendshape ($tmpB_{53}$) be the differences of F_j and F_l in the kth frame.

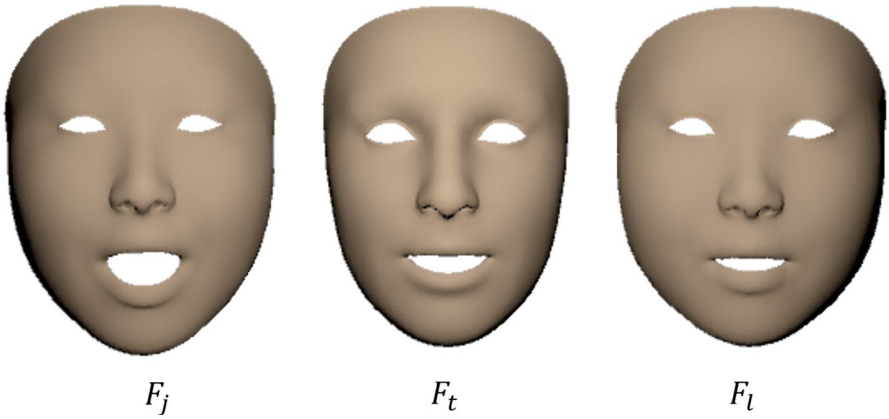

Fig. 5 Face display in the most problematic frame. (From left) The original face, the template face, and the LR face

Another user is involved in testing the trained model. This new user serves as the source(F_b) and the target face is controlled by source coefficients(W_b) and the LR blendshapes(B_l). The source face is derived by taking the first phase of the process mentioned above, where the model makes 52 instructed expressions. For desirable facial retargeting, we aim to have the target face with more structural similarities to the source face than the template model. All three face can be defined by using the same coefficients(W_b):

$$\text{tmp}B_{53} = F_j - F_l \text{ in the } k\text{th frame} \tag{8}$$

$$F_j = F_l + \text{tmp}B_{53}\text{tmp}_{w53} \text{ for every frame} \tag{9}$$

Accordingly, its weight(tmp_{w53}) is 1 in the kth frame and additional weights are extracted in other frames by solving a simple linear equation. With updated data, linear regression is employed to discover augmented blendshapes. Its implementation is the same as the process in Sect. 3.1.1. It is described in Fig. 6 and Fig. 7. $tmpW_{53}$ is a set of the 53rd temporal coefficients. Through this system, there is a renewed

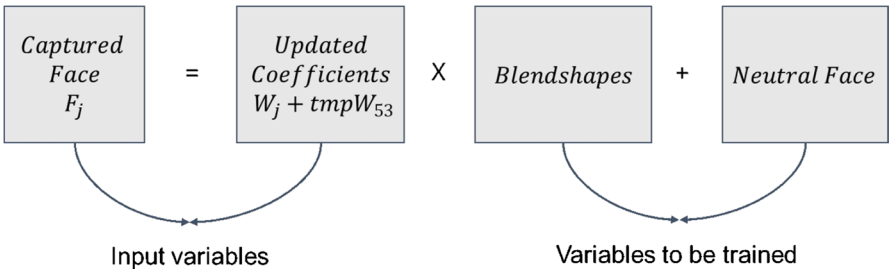

Fig. 6 Linear regression training phase for blendshape reconstruction

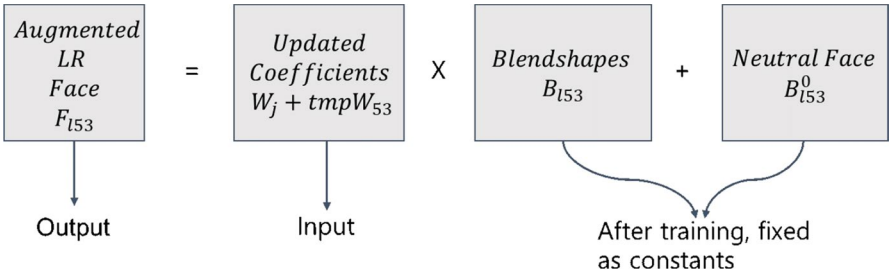

Fig. 7 Linear regression testing phase for blendshape reconstruction

Fig. 8 A newly added 53rd blendshape

face (F_{l53}) with revised blendshapes (B_{l53}) and temporal weights. Figure 8 depicts a newly added 53rd blendshape.

3.2.3 Weight manipulation

Weights need to be updated to reconstruct the face. Adding an extra weight to the prior sets is not sufficient to generate an individually optimized face as blendshapes are newly generated. Many studies have applied a pseudo-inverse to extract adequate weights [1, 2, 37]. A similar approach is taken for updated weights W_{l53}.

$$F_j \approx B^0_{l53} + \sum_{i=1}^{52} B^i_{l53} w^i_l + B^{53}_{l53} w^{53}_{l53} \tag{10}$$

$$F' = F_j - (B^0_{l53} + \sum_{i=1}^{52} B^i_{l53} w^i_l) \tag{11}$$

$$F' = B^{53}_{l53} w^{53}_{l53} = B' w' \tag{12}$$

Simply, B^{53}_{l53} is considered as B' and w^{53}_{l53} as w'. Then, $w' = B'^{+}F'$ by a pseudo-inverse solution. The above process is iteratively repeated until the errors between the captured face and the generated face are optimally minimized. The result of adding 53rd blendshape is shown in Fig. 9. The generated face (F_{l53}) with augmented blendshapes keeps the captured facial details more precisely than the previous version.

4 Experimental results

4.1 Data acquisition

In this work, we use a real-time capture system, ARKit face tracking API, to acquire face geometry by using iOS 14.5 and Swift 5. With the advent of cutting-edge hardware, Hui proposed the motion tracking algorithm based on Mean-Shift to capture the target motion accurately [16]. The camera is sufficient to track the face geometry, so the aforementioned algorithm is not applied to our capture process. 1220 vertices—each vertex is expressed as x,y,z—and 52 blendshape coefficients in each frame were obtained using an AR application running on iPad Pro 11. As seen in Fig. 10, the camera recognizes a user's face, and the face mesh filter is overlaid on the user. The device then sends the face geometry information to the connected computer by capturing 60 frames per second. The face data of both the source and the target are collected with this capture system.

First, the target face for the blendshape optimization was captured where three phases were built to capture various expressive faces. Initially, a user performs 52 template key poses under instructions such as eyeBlinkLeft, jawOpen, etc. The 52

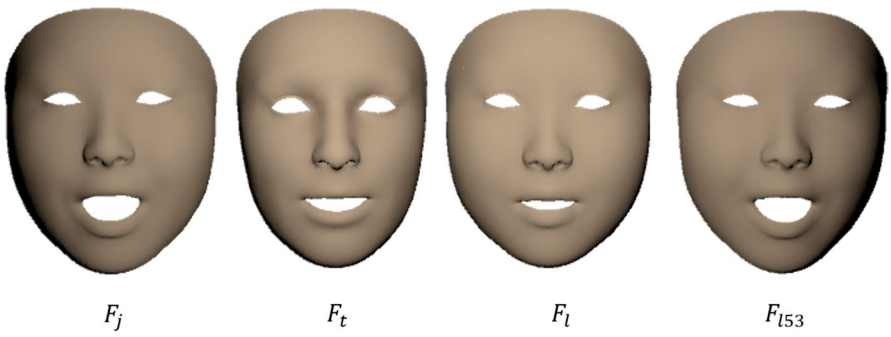

Fig. 9 Face reconstruction after blendshape augmentation. (From left) The captured face, template face, LR face, and F_{l53}

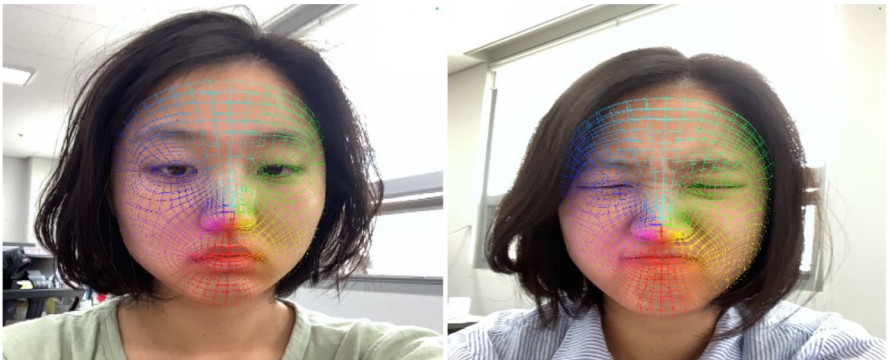

Fig. 10 Face capture running on the app

Table 1 Blendshape coefficients

Expressions	The number of blend-shapes
Left eye	7
Right eye	7
Jaw	4
Mouth	23
Eyebrows	5
Cheek	3
Nose	2
Tongue	1
Total	52

blendshape coefficients consist of 7 expressions for each eye, 4 expressions for the jaw, 23 expression parameters for a mouth, 5 for eyebrows, 3 for the cheek, 2 for the nose, and 1 for the tongue as listed in Table 1. Then, emotional guide pictures are given to replicate expressive faces as similar as possible. Lastly, the camera tracks the user who reads play scripts in which emotion guidelines are clearly written. It enables the capturing of natural expressions represented while speaking. In this way, 4500 frames were obtained from the aforementioned approach.

The real-time capture system offers 52 template blendshapes, but it does not reflect individual facial features and particular facial poses. Every human face is unique, so it is not sufficient for high-quality facial animation. With this work, a method to form a proximal approximation of new blendshapes for natural and realistic facial animation is created. Meanwhile, template models to prove our method's superiority are still needed. With regard to the blendshape customization, the template is compared to a new face created by our method to check if our model is more similar to the captured one. The template face is driven from the linear combination of template blendshapes and captured blendshape coefficients while simultaneously

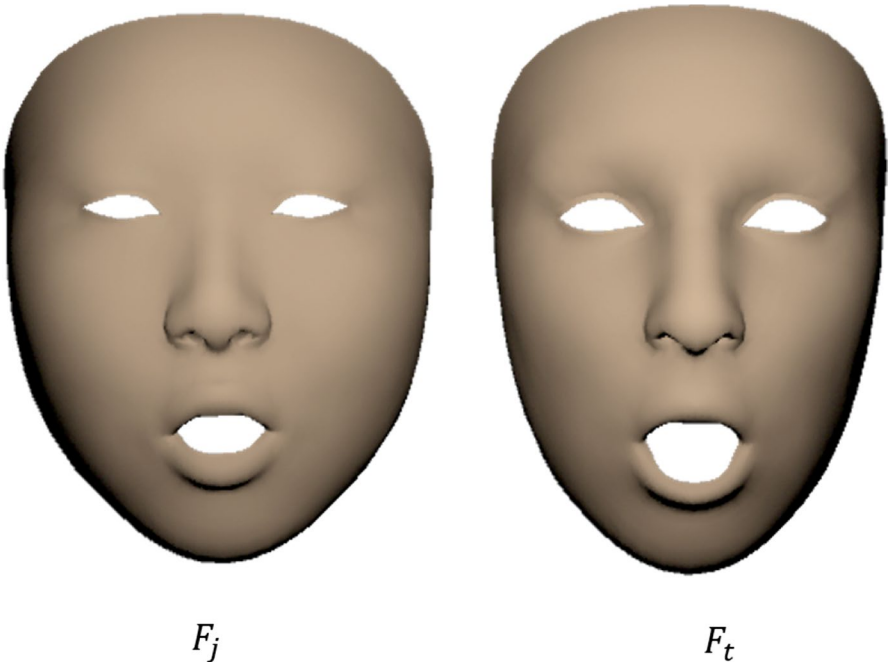

Fig. 11 Comparison of the captured face (*left*) and the template face (*right*)

Table 2 Comparison of blendshape generation methods

Method	Total train time(sec)	RMSE
Linear regression	575.2768	0.00061968
Autoencoder	888.3696	0.00089043

the camera tracks the user's face. We have gained the template face geometry including 1220 vertices and 52 coefficients. The number of each template model's frames is the same as that of corresponding captured frames. Facial vertices are slightly different between the captured model and the template model while they share the same coefficients. The template is set up for a uniform model that does not have any personal details such as a facial shape. Moreover, some facial mismatches happened in the template model as in Fig. 11.

4.2 Blendshape customization

4.2.1 Blendshape generation

Linear regression and autoencoder techniques are implemented to construct individually optimized blendshapes automatically. The training is performed on Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz CPU, NVIDIA Quadro RTX 5000 GPU,

and 24GByte of memory. Both methods generate respective faces by using the same inputs: the target's coefficients (W_j). The two methods are compared in two aspects: total train time and RMSE during 5000 epochs, as shown in Table 2. Linear regression took 575.2768 seconds to train, shorter than the autoencoder, which took 888.3696 seconds. Moreover, RMSE(F_j, F_l) were 0.00061968, smaller than the autoencoder RMSE(F_j, F_a) of 0.00089043. Accordingly, blendshapes (B_l) obtained from linear regression are adopted. Numerically, MSE(F_j, F_l) is smaller than MSE(F_j, F_t) in whole frames as in Fig. 12. Figure 13 shows that the LR face (F_l) is animated like the captured one (F_j) more identically than the template face (F_t).

4.2.2 Blendshape augmentation

As appeared on the face animation above, the generated blendshapes from linear regression do not work well with respect to the captured face reconstruction. Thus, previous blendshapes can be improved by adding a blendshape incrementally. We begin with finding the most problematic frames using MSE between the captured face and the LR face. The (300 multiples $\pm$ 5) frames are ignored because a user's face is captured per 300 frames, resulting in inaccurate captures near those frames. Table 3 shows the ten largest $MSE \cdot 10^8$ and its corresponding frame in regard to the number of blendshapes. On the left of the comparison table, the 3931st frame has the largest error for the whole frame. The faces are also displayed on the screen to

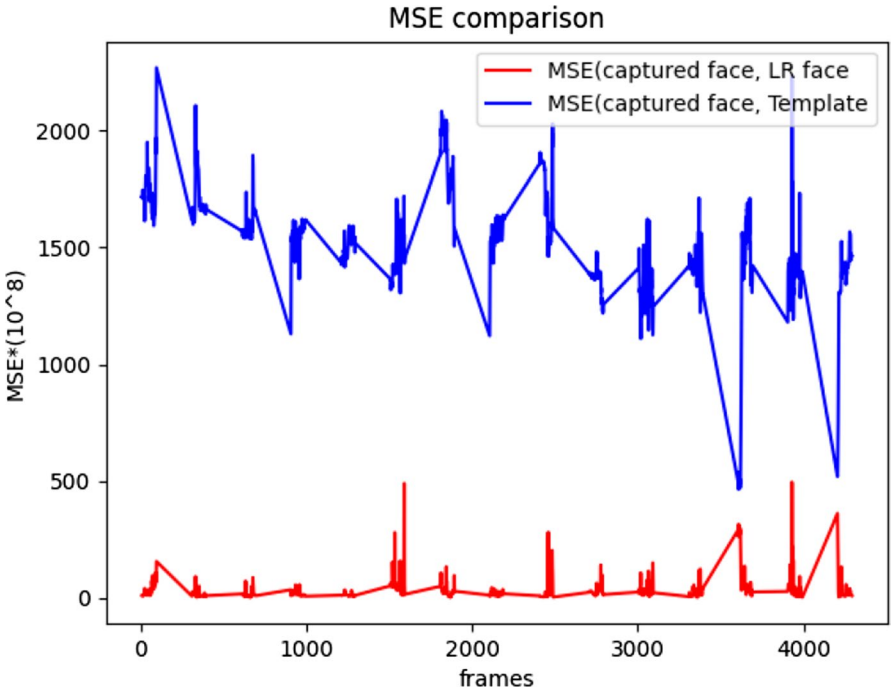

Fig. 12 MSE(F_j, F_l) and MSE(F_j, F_t)

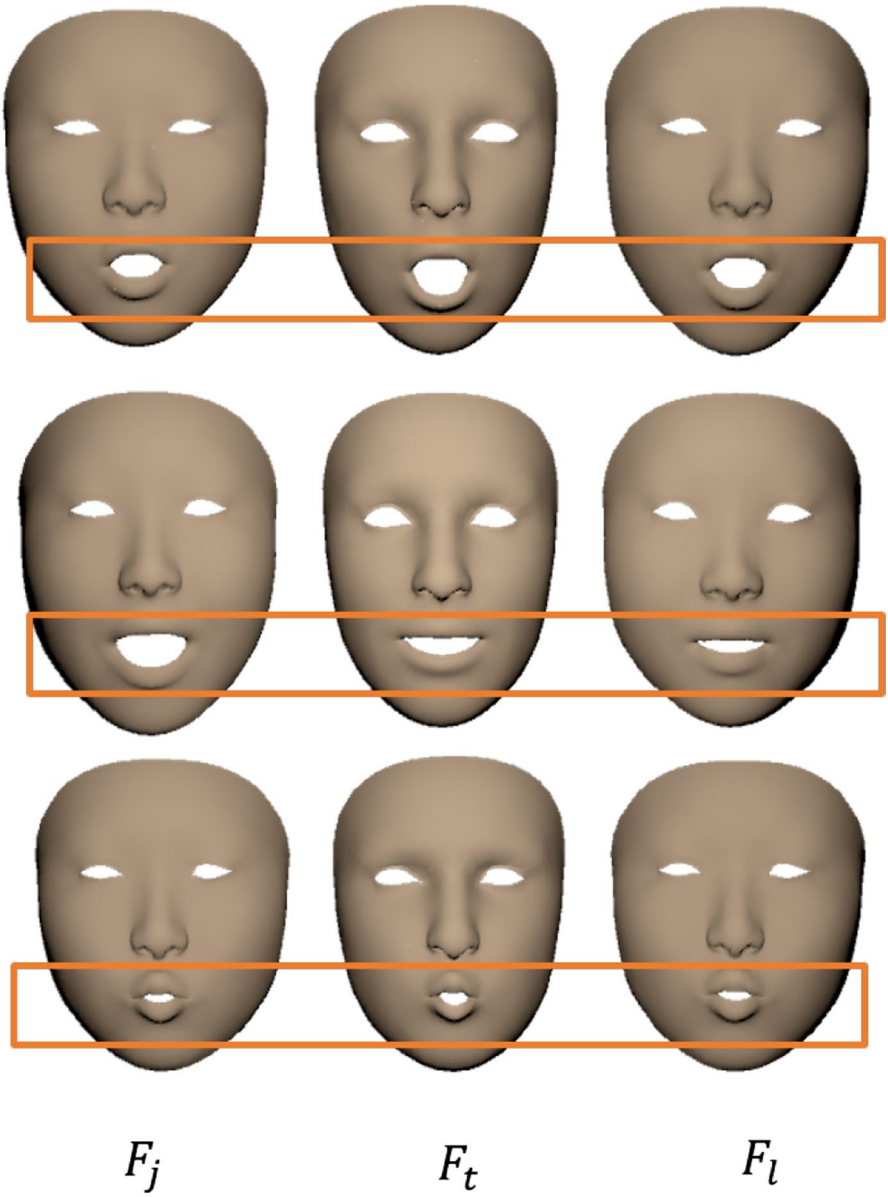

Fig. 13 Face animation with the captured face (*left*), the template face (*middle*), and the LR face (*right*)

check for facial inconsistency visually. As shown in Fig. 14, the LR face's mouth is quite different from the captured face.

Considering that previous blendshapes B_l lack the ability to present mouth-related expressions, an additional blendshape is needed to cover them. The MSE improvement is represented on the graph in Fig. 15 where two values—MSE(F_j, F_l

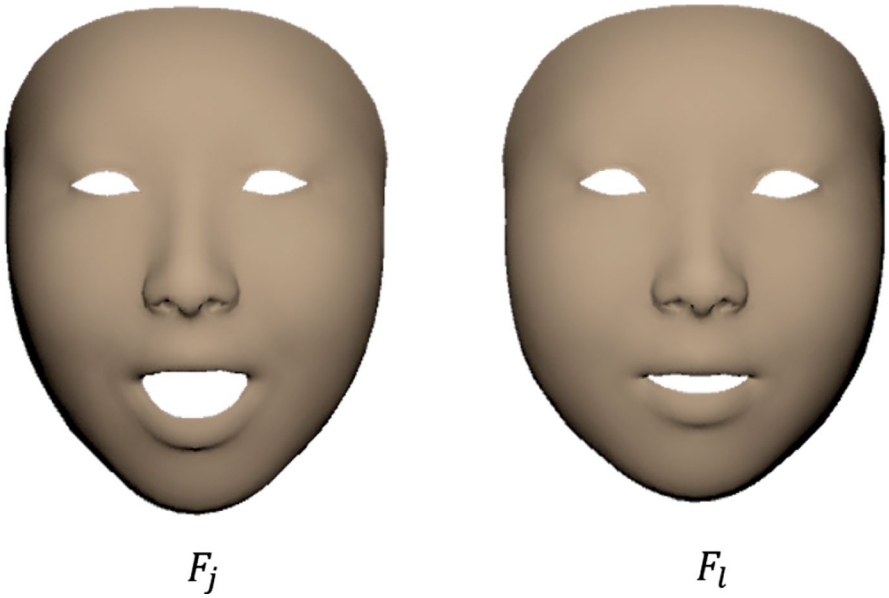

Fig. 14 3931st frame - the captured face (*left*), the LR face (*right*)

Table 3 MSE comparison table regarding to the number of blendshapes

52 blendshapes		53 blendshapes	
$MSE \cdot 10^8$	Frame	$MSE \cdot 10^8$	Frame
494.5087	3931	3.5968	4206
489.0645	1590	3.4971	1812
360.464	4206	3.3162	4207
315.3687	3609	3.2990	2409
314.4987	3608	3.2919	1808
297.6247	3618	3.2887	1810
295.293	3613	3.2855	1809
293.234	4207	3.2845	2425
291.064	3606	3.2822	2410
288.7733	3612	3.2800	2426

) and MSE(F_j, F_{l53})—are compared. Now, F_{l53} is referred to the LR53. It is notable that every error between the captured face and LR53 gets smaller after blendshape addition. Approximately, errors have decreased by 99% in the third largest error-occurring 4206th frame. The outstanding performance is indicated on the screen as in Fig. 16. Advancements stand out in specific frames where errors have occurred largely in the mouth movements.

As it is confirmed that the ten most problematic frames are solved after adding a blendshape, the focus is now on the new error-occurring frames in the

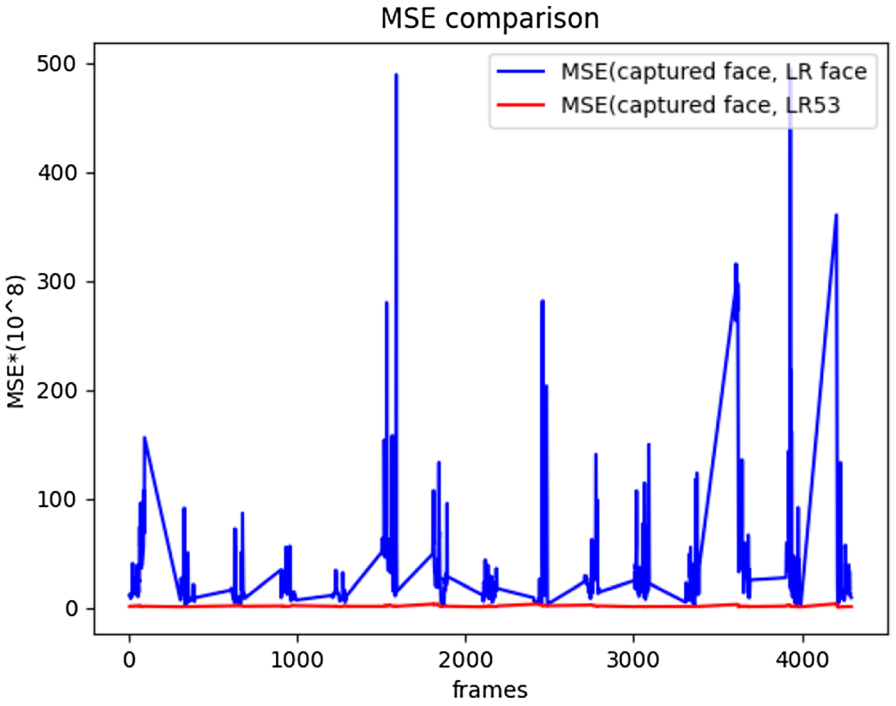

Fig. 15 MSE improvement after blendshape addition

updated face. The original plan was to add blendshapes one by one for precise target face construction. However, as errors are too tiny to supplement existing blendshapes, it is decided not to perform an iterative process. It is hard to find remarkable discrepancies between two faces in the 4206th frame where errors are largest after blendshape augmentation as shown in Fig. 16 and Table 3. We have concluded that 53 blendshapes are enough to cover facial details with minimal error as the performance has improved when blendshapes are increased.

For 4500 frames, the MSE(F_j, F_l) and MSE(F_j, F_t) are visualized on the graph in Fig. 12. Both MSE were multiplied by 10^8 to clarify the results. The blue line represents the difference between the captured face(F_j) and the LR face(F_l). The red line shows the gap between the captured face(F_j) and the template face(F_t). It is apparent that the LR face has more similar structure than the other.

In Fig. 13, it is proved that our method has more similar structure to the captured face by comparing three faces—the captured face (F_j), the LR face(F_l) and the template face(F_t). Noteworthy facial distinction is shown in the eyes, mouth, and facial shape. In the template face, the eyes moved slight differently from those in the other two faces. It has occurred in the mouth, as well. Although the eyes of the LR face are not exactly the same as those of the captured one, the LR face will be improved by updating blendshapes. In general, the facial shape of

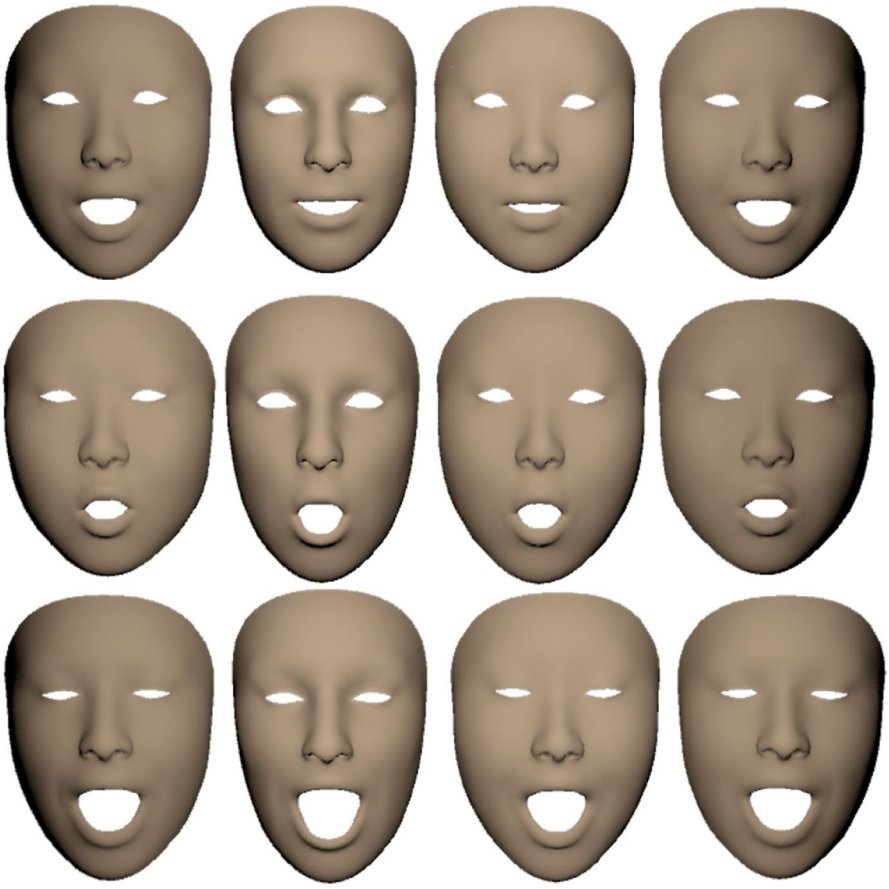

Fig. 16 Face animation after blendshape addition. (From left) The captured face, template face, LR face, and LR53

the template face looks dissimilar to the other faces, which might induce poor correspondences in some facial segments.

5 Conclusions

This paper proposed two stages of customizing blendshapes to capture facial details without depending manual work of skillful artists: blendshape generation and blendshape augmentation. For data collection, a user's face is captured and geometric data are acquired from the target and the template model with a simple and portable device. It is highly practical because the capturing process does not require a specialized high-cost device and professional execution. Also, this mobile device is not affected by spatial limitations, allowing captures in any place.

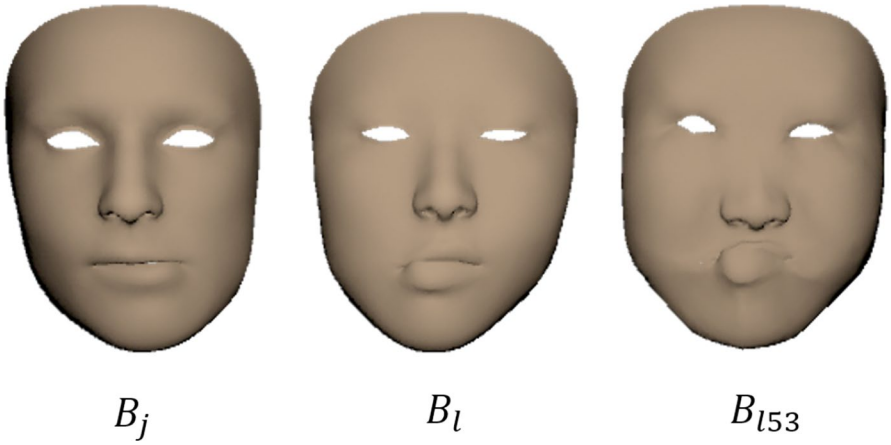

Fig. 17 Augmented blendshapes missing visual significations. (From left) Template blendshape, LR blendshape, augmented blendshape

In the first stage, given target data, the blendshapes which cover individual's unique facial details were built with two methods: LR and AE. LR is proved to be more suitable in that numerical errors between the original face and the face created by the former are smaller than AE. Furthermore, it is more advantageous in the perspective of computation time. The higher performance with LR is perhaps due to our AE where compressed z code does not reflect weights technically. In other words, there is only a low relativity between z code and expression parameters.

Blendshape model is extended to add more blendshapes to minimize error between the source expressions and recreated results based on the assumption that incorrect facial animation comes from the lack of existing blendshapes. Additional blendshape involved with opening the mouth is supplemented because the most problematic frame where errors have occurred largely is related to such motion. Accuracy of facial animation driven from augmented blendshapes increases significantly, reducing errors by 99%. If necessary, additional blendshapes could be augmented to reduce errors further. Weights are updated to correspond to increased blendshapes by taking a pseudo-inverse based approach. PCA is not applied to the aforementioned process to avoid missing facial features and unmeaningful visual representations. The contribution of blendshape customization includes that manual editing is not necessary and blendshapes are added incrementally to generate personalized blendshapes.

As we added a blendshape for the mouth and decided not to use PCA for visual significations and intuitive control [23], it was meaningful to identify each blendshape visually. However, some bizarre key poses were observed in augmented blendshapes as seen in Fig. 17. Augmented blendshape showed unnatural facial poses. Among many possible reasons, unmeaningful facial expressions are represented because uncorrelated small regions are associated to define a key pose.

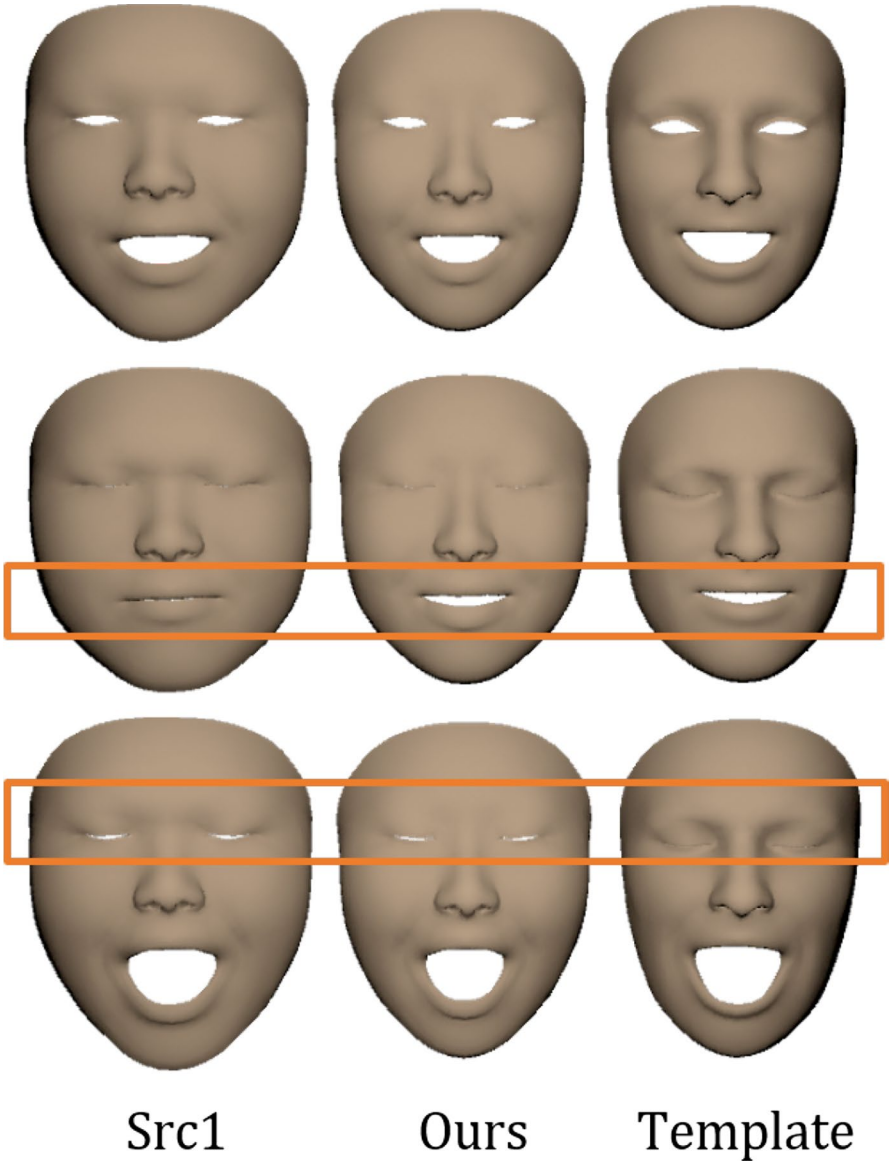

Fig. 18 (From left) The first source face, the target face of our method in Sect. 3.1, the retargeted template face. All face shares the same coefficients

Data quantity can be one reason for inadequate results. A larger amount of data can improve the quality of blendshapes. Also, dividing facial parts semantically or locally can be another solution for natural and realistic blendshapes.

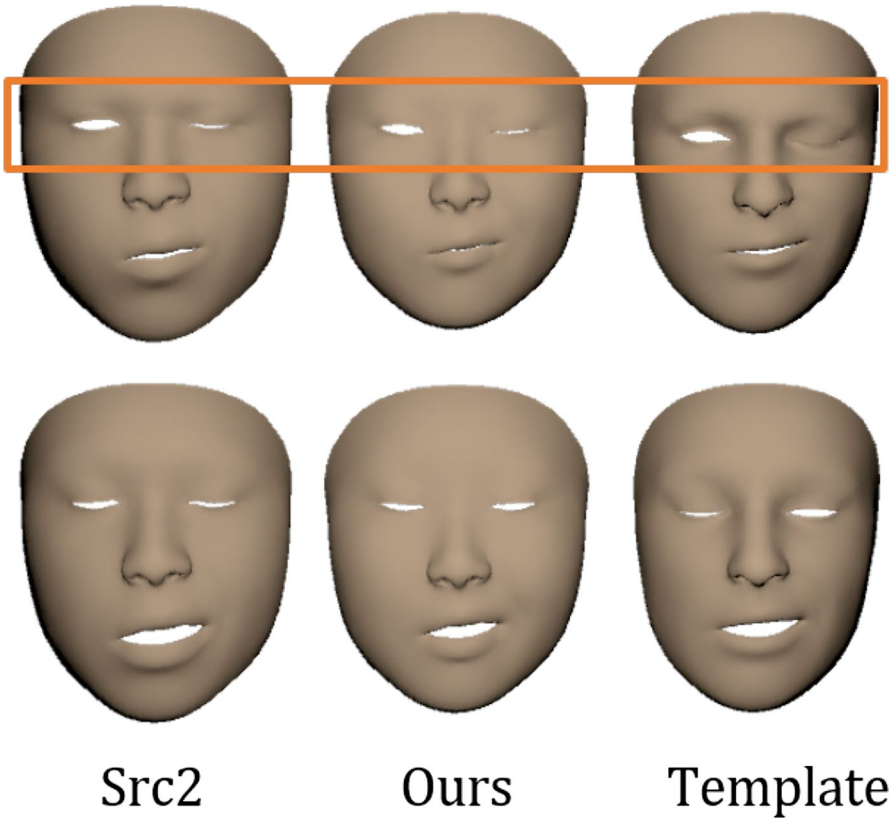

Fig. 19 Facial retargeting. (From left) The second source, the retargeted face of our method in Sect. 3.1, the retargeted template face,

The success of blendshape customization depends on how much facial details are preserved in the animation process. The mouth is considered as one of the most challenging parts to model blendshapes due to active interaction of deeply connected muscles. Even though the eyes are also one of the hardest regions, they are not handled carefully as our focus is on the mouth-related expressions. Based on MSE, blendshapes related to the mouth were enhanced, resulting in more accurate retargeting in mouth movements. Since the MSE were used to determine the need of blendshape addition, errors in eyes were very tiny, so they are ignorable mathematically. Thus, the further study to find adequate methods that can be used to supplement blendshapes for eyes to capture fine scales is still needed.

We further extend our work to generate detail-preserved blendshapes for minimal-error facial retargeting. By using target blendshapes we have created in this paper, facial retargeting is implemented with three sources. 600 frames are captured from the first source (Src1) and retargeted to our target acquired from the method in

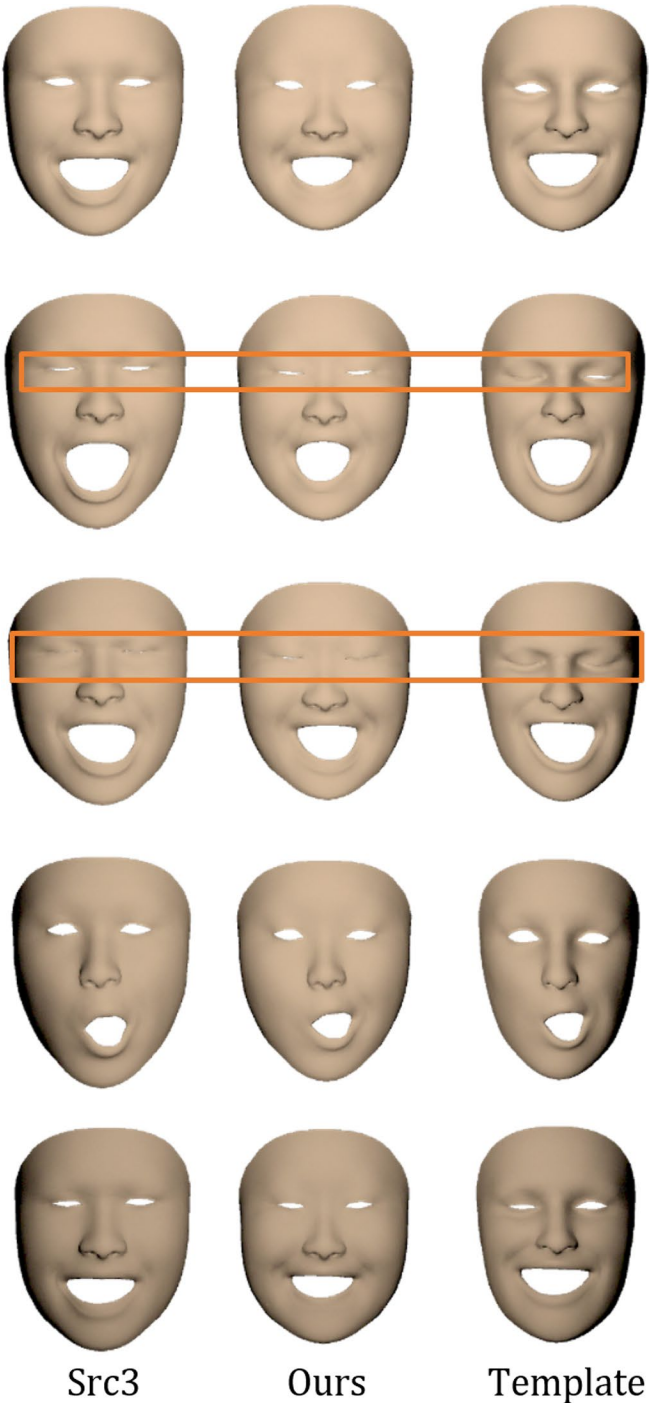

Fig. 20 Facial retargeting. (From left) The third source, the retargeted face with our method, the retargeted template face

Sect. 3.1. This is represented in Fig. 18. When the first source squints his eyes, the target's eyes are also slightly opened while the retargeted template face just squeezes eyes. The following notable part is the mouth. Although our method moves like the source more similarly than the template model, mismatching of retargeting expressions happen around the mouth since target blendshapes in this implementation have not been augmented yet. We plan to perform facial animation with our suggested method of blendshape addition to improve the quality of facial retargeting.

Also, we have gained 900 frames and 1000 frames from other two sources (Src2 and Src3), respectively. Figure 19 indicates facial retargeting with the second source. The retargeted template face sometimes has incorrect retargeting with the eyes. For example, the first row shows that the template tends to squeeze the right eye while the original face just winks slightly. Our work takes a similar approach of Bouaziz et al. [3] in that expressions are transferred by customizing blendshapes. Their work adopts PCA models and a template blendshape to create user-specific blendshapes, but it might be insufficient in some cases where facial nerves are damaged. However, our method can solve this issue by defining an additional blendshape for each character as it is effective enough at transferring expressions of the second source who is unable to move left facial muscles onto the target as in Fig. 19. Figure 20 depicts mapping from the third source to the target faces with our method. Inaccurate facial mapping is observed in the template model's eyes while others faithfully reproduce the source expressions. Specifically, the retargeted template of the bottom row winks unlike other faces. It is significant that our methods are successful to cover source expressions. Future work will be a demonstration of facial retargeting with enhanced target blendshapes.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11227-022-04885-7.

Acknowledgements This work was extended from the paper presented in 2021 IEEE International Conference on Big Data and Smart Computing [15]. It was supported by Institute of Information and communications Technology Planning and Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2020-0-00872, SaaS Technology for Development of Veterinary Medical Image Interpretation based on AI) and the Bio-Synergy Research Project (2013M3A9C4078140) of the Ministry of Science, ICT and Future Planning through the National Research Foundation.

References

1. Anjyo K, Todo H, Lewis JP (2012) A practical approach to direct manipulation blendshapes. J Graph Tools 16(3):160–176
2. Berson E, Soladie C, Barrielle V, Stoiber N (2019) A robust interactive facial animation editing system. Motion Interaction Games, pp 1–10
3. Bouaziz S, Wang Y, Pauly M (2013) Online modeling for realtime facial animation. ACM Trans Graph (ToG) 32(4):1–10
4. Cao C, Weng Y, Lin S, Zhou K (2013) 3d shape regression for real-time facial animation. ACM Trans Graphics (TOG) 32(4):1–10
5. Casas D, Feng A, Alexander O, Fyffe G, Debevec P, Ichikari R, Li H, Olszewski K, Suma E, Shapiro A (2016) Rapid photorealistic blendshape modeling from rgb-d sensors. In: Proceedings of the 29th International Conference on Computer Animation and Social Agents, pp 121–129

6. Cetinaslan O, Orvalho V (2020) Sketching manipulators for localized blendshape editing. Graph Mod 108:101059
7. Cetinaslan O, Orvalho V (2020) Stabilized blendshape editing using localized Jacobian transpose descent. Graph Mod 112:101091
8. Chaudhuri B, Vesdapunt N, Shapiro L, Wang B (2020) Personalized face modeling for improved face reconstruction and motion retargeting. In: European Conference on Computer Vision, pp 142–160. Springer
9. Chen K, Zheng J, Cai J, Zhang J (2020) Modeling caricature expressions by 3d blendshape and dynamic texture. arXiv preprint arXiv:2008.05714
10. Chheang V, Jeong S, Lee G, Ha JS, Yoo KH (2020) Natural embedding of live actors and entities into 360 virtual reality scenes. J Supercomput 76(7):5655–5677
11. Cong M, Fedkiw R (2019) Muscle-based facial retargeting with anatomical constraints. In: ACM SIGGRAPH 2019 Talks, pp 1–2
12. Costigan T, Gerdelan A, Carrigan E, McDonnell R (2016) Improving blendshape performance for crowds with gpu and gpgpu techniques. In: Proceedings of the 9th International Conference on Motion in Games, pp 73–78
13. Costigan T, Prasad M, McDonnell R (2014) Facial retargeting using neural networks. In: Proceedings of the Seventh International Conference on Motion in Games, pp 31–38
14. Ding C, Tao D (2015) Robust face recognition via multimodal deep face representation. IEEE Trans Multimedia 17(11):2049–2058
15. Han JH, Kim JI, Kim H, Suh JW (2021) Generate individually optimized blendshapes. In: 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), pp 114–120. IEEE
16. Hui Q (2019) Motion video tracking technology in sports training based on mean-shift algorithm. J Supercomput 75(9):6021–6037
17. Joshi P, Tien WC, Desbrun M, Pighin F (2006) Learning controls for blend shape based realistic facial animation. In: ACM Siggraph 2006 Courses, pp 17–es
18. Kan M, Shan S, Chang H, Chen X (2014) Stacked progressive auto-encoders (spae) for face recognition across poses. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1883–1890
19. Kang J, Lee S (2020) A greedy pursuit approach for fitting 3d facial expression models. IEEE Access 8:192682–192692
20. Kim PH, Seol Y, Song J, Noh J (2011) Facial retargeting by adding supplemental blendshapes. In: PG (Short Papers)
21. Kingma DP, Welling M (2013) Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114
22. Kommineni J, Mandala S, Sunar MS, Chakravarthy PM (2021) Accurate computing of facial expression recognition using a hybrid feature extraction technique. J Supercomput 77(5):5019–5044
23. Lewis JP, Anjyo K, Rhee T, Zhang M, Pighin FH, Deng Z (2014) Practice and theory of blendshape facial models. Eurographics (State of the Art Reports) 1(8):2
24. Lewis JP, Anjyo KI (2010) Direct manipulation blendshapes. IEEE Comput Graph Appl 30(4):42–50
25. Li J, Kuang Z, Zhao Y, He M, Bladin K, Li H (2020) Dynamic facial asset and rig generation from a single scan. ACM Trans. Graph. 39(6):215–1
26. Li Q, Deng Z (2008) Orthogonal-blendshape-based editing system for facial motion capture data. IEEE Comput Graph Appl 28(6):76–82
27. Lombardi S, Saragih J, Simon T, Sheikh Y (2018) Deep appearance models for face rendering. ACM Transactions on Graphics (TOG) 37(4):1–13
28. Orvalho V, Bastos P, Parke FI, Oliveira B, Alvarez X (2012) A facial rigging survey. Eurographics (State of the Art Reports) pp 183–204
29. Parent R (2012) Computer animation: algorithms and techniques. Newnes
30. Parke FI (1972) Computer generated animation of faces. In: Proceedings of the ACM annual conference-Volume 1, pp 451–457
31. Parke FI (1974) A parametric model for human faces. The University of Utah
32. Parke FI, Waters K (2008) Computer facial animation. CRC press
33. Pighin F, Hecker J, Lischinski D, Szeliski R, Salesin DH (2006) Synthesizing realistic facial expressions from photographs. In: ACM SIGGRAPH 2006 Courses, pp 19–es
34. Pighin F, Lewis JP (2006) Facial motion retargeting. In: ACM SIGGRAPH 2006 Courses, pp 2–es

35. Rao S, Ortiz-Cayon R, Munaro M, Liaudanskas A, Chande K, Bertel T, Richardt C, JB A, Holzer S, Kar A (2020) Free-viewpoint facial re-enactment from a casual capture. In: SIGGRAPH Asia 2020 Posters, pp 1–2
36. Ribera RBI, Zell E, Lewis JP, Noh J, Botsch M (2017) Facial retargeting with automatic range of motion alignment. ACM Trans Graph (TOG) 36(4):1–12
37. Seo J, Irving G, Lewis JP, Noh J (2011) Compression and direct manipulation of complex blendshape models. ACM Trans Graph (TOG) 30(6):1–10
38. Seol Y, Lewis JP, Seo J, Choi B, Anjyo K, Noh J (2012) Spacetime expression cloning for blendshapes. ACM Trans Graph (TOG) 31(2):1–12
39. Seol Y, Ma WC, Lewis J (2016) Creating an actor-specific facial rig from performance capture. In: Proceedings of the 2016 Symposium on Digital Production, pp 13–17
40. Sumner RW, Popović J (2004) Deformation transfer for triangle meshes. ACM Trans Graph (TOG) 23(3):399–405
41. Thomas D, Taniguchi RI (2016) Augmented blendshapes for real-time simultaneous 3d head modeling and facial motion capture. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3299–3308
42. Toshpulatov M, Lee W, Lee S (2021) Generative adversarial networks and their application to 3d face generation: a survey. Image Vis Comput, p 104119
43. Wang S, Cheng Z, Deng X, Chang L, Duan F, Lu K (2020) Leveraging 3d blendshape for facial expression recognition using cnn. Sci China Inf Sci 63(120114):1–120114
44. Zhang J, Chen K, Zheng J (2020) Facial expression retargeting from human to avatar made easy. IEEE Trans Vis Comput Graph