Laplacian face blending

Mazala, Diego ; Esperança, Claudio ; Marroquim, Ricardo

DOI
10.1002/cav.2044

Publication date
2022

Document Version
Final published version

Published in
Computer Animation and Virtual Worlds

Received: 31 December 2021 | Revised: 14 February 2022 | Accepted: 14 February 2022

DOI: 10.1002/cav.2044

RESEARCH ARTICLE

WILEY

Laplacian face blending

Diego Mazala[1] | **Claudio Esperança**[1] | **Ricardo Marroquim**[2]

[1]Systems Engineering and Computer Science, Federal University of Rio de Janeiro, Rio de Janeiro, Brazil

[2]INSY, Delft University of Technology, Delft, Netherlands

Correspondence
Ricardo Marroquim, INSY, Delft University of Technology, Delft, Netherlands.
Email: r.marroquim@tudelft.nl

Abstract

Designing realistic tridimensional facial models is a challenging task, not only due to the effort and artistic abilities required but also because human visual perception is very tuned to the processing of facial features. For this reason, rather than creating face models from scratch, artists usually start from a scanned model of a real person. In this work, we present a novel method for blending human faces in order to create a new one. In a nutshell, our proposal uses Laplacian smoothing to segregate layers of details from one or more faces, which are then integrated into a base face with the help of an interactive and visual editor. In particular, our method supports blending multiple faces and multiple sub-regions in those faces. Since our approach is intuitive and relatively easy to implement, it can be integrated into artistic pipelines aiming at designing human face models from preexisting ones.

KEYWORDS

detail transfer, digital humans, face modeling

1 | INTRODUCTION

Throughout generations, we humans have been developing as a society through interaction with our peers. This interaction exposed us to countless different faces and their nuances. This, in turn, shaped our brains to become highly trained instruments for facial recognition, capable of detecting even minute imperfections. This advanced skill of our brains, however, makes creating believable digital faces a challenge.

Methods for creating digital models of human faces are in high demand, and have been extensively researched and improved over the years. The film and gaming industries,[1,2] for example, have played a leading role in this field. Notwithstanding the advances in scanning technologies and digital sculpting tools, it still takes a considerable amount of time and effort to develop a high-quality digital face. Here, we aim to contribute to this field by proposing an approach to generate new three-dimensional (3D) face models from preexisting ones. Our goal is to provide a simple, practical, and intuitive system to blend facial details, thus allowing the creation of alternative models in a well-controlled way. We have worked closely with professional artists and considered carefully how their modeling pipeline could best benefit from our method.

We specifically target background or secondary characters, and not main characters. In other words, we focus on techniques for creating faces quickly, with controllable variability and believable appearance. Hence, a system comprising only a few sliders to weight the blended features is ideal, as secondary characters rarely merit the time to tune all the minor details. For major characters, control of all aspects of the face would be preferable rather than a semi-automated system that can provide a believable 3D model in a short amount of time.

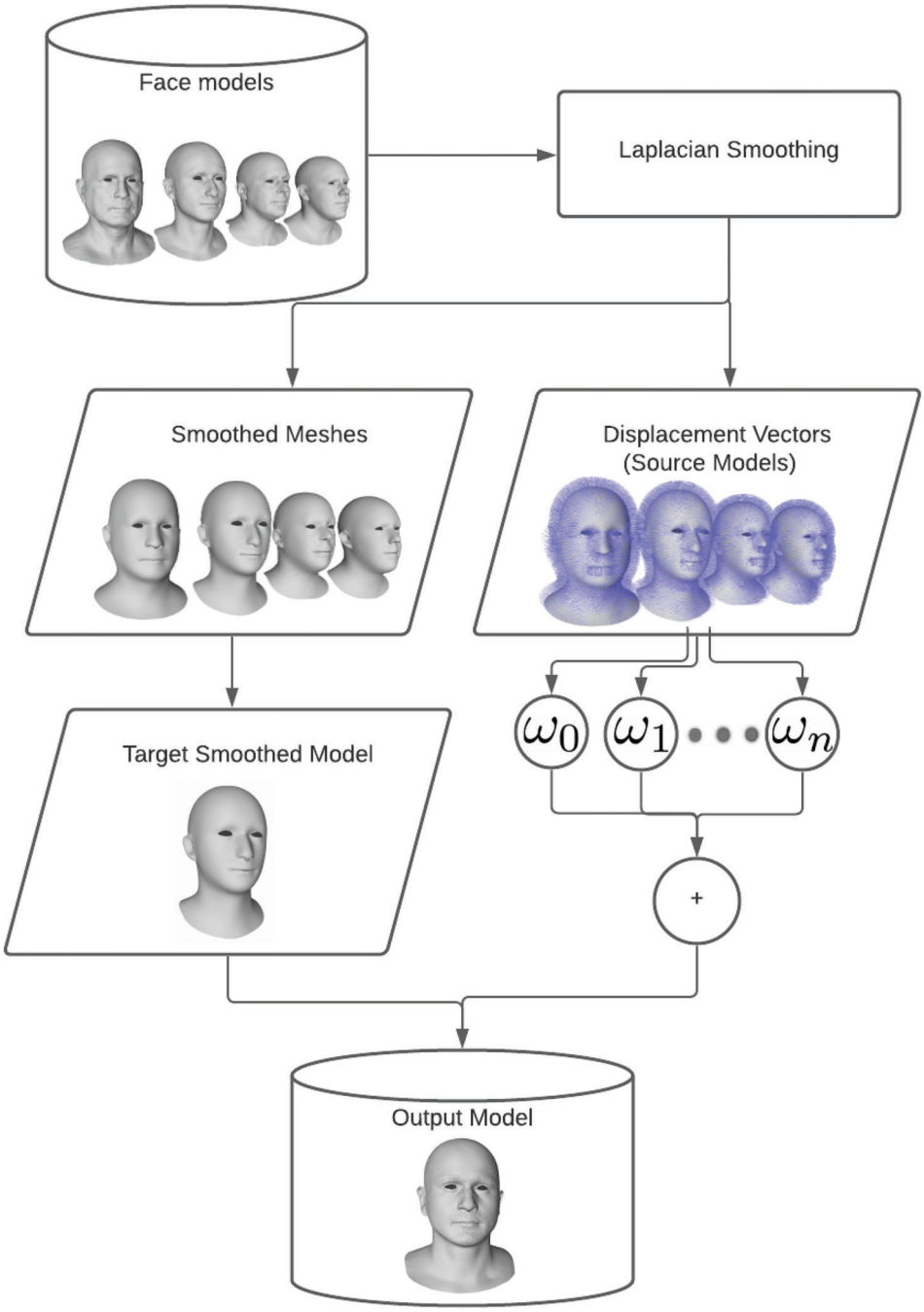

FIGURE 1 Overview of the whole process. The face models are smoothed by a Laplacian filter to separate low (smoothed) and high frequencies (displacement vectors). A target face is selected from the smoothed set and a weighted combination of displacement vectors is applied to generate new details for the output model

Briefly, the fundamental idea of our approach is to transfer the details from one face to another. We achieve this by smoothing the faces to separate the coarse and fine details, followed by a transfer of these details to a coarse mesh model taken from another face. Besides, we have developed an interactive system to test and validate our approach. This tool allows creating a new face in real-time by combining details over the entire model or mixing distinct features in smaller segmented areas. Figure 1 depicts the process workflow while Figure 2 shows one of our results using only two face models. Our work has the following main contributions:

- Seamlessly transfer details between two or more face models.
- Intuitive and simple parameters to control the detail transfer using a two-dimensional (2D) interface.
- Simple and efficient implementation that allows to blend multiple high-resolution faces in real-time.

FIGURE 2 Result of blending two faces. The model on the left provides the details (source), the one on the right the base anatomy (target), and the result is shown in the middle. The textures are for illustration purposes only, our method only deals with geometric blending

We organize this article in the following way: Section 2 describes related works; in Sections 3 and 4, we describe the acquisition pipeline to digitize faces and discuss mesh smoothing methods, respectively; in Section 5, we explain our proposed method followed by a description of our implementation in Section 6; in Section 7, we discuss our results and compare our results to previous work; and, finally, in Sections 8 and 9, we discuss limitations and conclude the work.

2 | RELATED WORK

2.1 | Blendshapes

Blendshapes are one of the most used approaches for facial animation and deformation. This method generates new face models by linearly combining facial expressions or deformations.[3]

Although there are several others alternatives, such as methods based on principal component analysis,[4,5] physically based modeling,[6,7] motion capture driven meshes,[8,9] and interpolation of an abstract pose,[10,11] Blendshapes are still popular due to its simplicity and ease of understanding. Even in cases when more sophisticated techniques are applied, Blendshapes are frequently used as a base layer over which nonlinear or physically based deformations are performed.

Two important advantages of Blendshapes are that the weights have intuitive semantics since they represent the influence of each facial expression, and that it is easy to avoid undesired deformations. Our approach has similar goals and targets a method to easily and intuitively generate new faces from a given input set.

2.2 | Facial scanning

Even when using a conceptually simple framework such as Blendshapes, creating a face from scratch is a labor intensive effort. For this reason, photogrammetry has become a useful tool in facial modeling. There is a vast literature on

facial reconstruction from images, but here we focus on methods geared for high-quality results. Fyffe et al.[12] present a facial capture technique that combines the benefits of single-shot and multi-shot techniques by introducing a slight delay between the cameras in order to capture several lighting conditions while considering the subject static. Tian et al.[13] present a method for accurately reconstructing faces using multiple viewpoints and shape priors. Fyffe et al.[14] describe a video based facial capture which deforms a common template model to match multi-view input images of the subject. Zhu et al.[15] propose a robust method using a multi-feature framework which includes SIFT features,[16] pixel intensity, and contours. Lin et al.[17] propose a method for visual sensor networks where the amount of required facial feature points is significantly reduced using self-adaptive morphable models. Dai et al.[18] propose a coarse-to-fine multi-view 3D face reconstruction method by taking advantage of the complementarity between facial feature points and occluding contours.

2.3 | Facial detail transfer

Transferring details from a source to a target shape is a well-studied problem for general digital 3D models. Sorkine et al.[19] show how details can be extracted from a single model via local vertex displacements. Sumner and Popović[20] describe a way to encode details and relate them to a reference pose. Although these methods were not particularly designed for facial details, they can be customized for this purpose.

Shin et al.[21] propose a method to extract and transfer expression wrinkles from a high resolution example face model to a target model in order to enhance the realism of facial animations. The detail maps contain the surface normal perturbations that are used to design expression wrinkles. In a different scenario, Romeiro et al.[22] propose a way to reconstruct faces from skulls for forensic purposes. They use a face template deformation and detail transfer to semi-automatically reconstruct the soft tissue structures.

Aiming at providing an intuitive way to blend detail between faces, Ma et al.[23] proposed a facial composite editor. It is an interactive editing system that, starting from a small number of given face models, allows digital modelers to create new Blendshape face models for primary or background characters. To avoid the limitations inherent to linear blending, they propose an approach similar to forensic software, in which face features (eyes, nose, mouth, etc.) from different individuals are assembled to create a composite likeness.

Booth et al.[24] present an automated pipeline to construct 3D morphable models from thousands of distinct facial identities. First, they establish dense correspondence using UV based interpolation methods. Then, they propose an approach for 3D landmark localization followed by dense correspondence estimation. Finally, they detect and exclude the cases of failures of dense correspondence and use PCA to construct the deformation basis. For a more extensive reading on 3D morphable models, please refer to the survey from Egger et al..[25]

Ploumpis et al.[26] propose methodology to fuse large-scale statistical model of the human head in terms of ethnicity, age and gender using both, a regression method based on latent shape parameters and a covariance combination approach. They utilize the combined models to perform full head reconstruction from unconstrained single images. Their approach builds new morphable models from meshes with different topology and that only partly overlap.

Guo et al.[27] presented a method to reconstruct face geometry and appearance from sequence of images. Albeit having a different goal since they do not generate new faces, they do employ a two scale process to recover the geometry where, in a similar manner to our work, the fine scale represents the geometric details in a displacement map.

More recently, Li et al.[28] presented a framework to generate face models from scan data. They base their method on a learning approach, training over an augmented dataset from 178 scanned faces. Their model combines anatomical and physically based face attributes to generate the new digitized faces at fine geometrical resolution.

The main difference from our proposed method in regards to the works above, is that we aim at providing a more handcrafted way to generate new faces while being intuitive and providing control. In fact, the inspiration for our method comes from the content generation pipeline from an artistic point-of-view, and the necessity to generate new models in a small amount of time. Moreover, since we do not need training data our method works with as few as two faces.

The approach proposed by Yoon et al.[29] is one of the most similar to ours. Starting from existing dense face models, they parameterize the 3D meshes onto a 2D unit domain. Then, they build a hierarchical representation of the face using a technique based on uniform cubic B-splines.[30] This generates a multi-scale representation where the first level is a very coarse and smooth approximation of the face. Using higher resolution grids, they repeatedly refit the errors and, thus, create increasingly better approximation surfaces. These surfaces are used to build a multiscale face model. They observe that if no self-intersection occurs, the original face model can be reconstructed by displacing the surface along its normal direction. The vector displacement is again approximated by a multilevel B-spline function. Finally they define a set of

FIGURE 3 The template mesh warped to an input head model. A low-resolution template is shown for illustrative purposes

multiscale continuous displacement maps, to store the level of details to be transferred between faces. We provide further comparison with the method from Yoon et al. in Section 7 since both works have similar goals.

3 | ACQUISITION PIPELINE

Although a digital artist can sculpt a human face from scratch, this process is more time-consuming than digital scanning methodologies. Here we describe the acquisition pipeline that was used to create the database of 3D human face models used in this work.

The photogrammetry rig is composed of 20 Canon EOS T5i cameras distributed in a circular manner around the subject. To align the photos and produce an initial raw version of the 3D facial model, we have used the commercial software Agisoft Photoscan.[31] In ideal conditions there are no alignment issues. However, in practice, areas with high specularity, like eyes and hair, frequently result in a poor and noisy mesh. Thus, a cleanup step is usually required to remove or adjust undesirable features. Since we work closely with professional artists, they were responsible for this step and prepared the final meshes using software such as Zbrush.[32]

The resolution and topology at this stage are arbitrary, thus the final step is to deform a template mesh to fit the digitized face. This is crucial in the production pipeline as, during other stages such as texturing and animation, artists can reuse effort between multiple models. Therefore, the topology for the mesh was designed considering subsequent work on animation and facial deformation. The initial template mesh is composed by quads and has 1.2 million vertices and we use a commercial software R3DS Wrap 3.3[33] to fit the template to any given input facial mesh. R3DS employs a two-step procedure, where, namely a rigid transformation followed by a nonrigid warping. Figure 3 shows an example of a head mesh being warped to fit the target head mesh.

Even though the acquisition pipeline is not the focus of this work and it could be improved in many ways, we found that it would be appropriate to describe it in order to place our work in the context of a professional content production pipeline. It also serves as further motivation for our work, since even with a semi-automatic pipeline, a lot of artistic intervention is still necessary to create each face. If each face is only used for a single character, the amount of time to generate many background characters may be prohibitively high in many cases. However, if each face can be used in the creation of many new faces in an efficient way, our method may have a significant impact on the creation pipeline without compromising the quality of the models.

4 | MESH SMOOTHING

We call mesh smoothing the process of changing vertex positions in a mesh in order to improve the mesh according to some given criterion. It may be useful to improve mesh quality,[34] remove noise,[35] or to change its topology.[36] Mesh smoothing methods can be classified as optimization-based,[37,38] geometry-based,[39,40] physics-based,[41] and some combination thereof.[42,43]

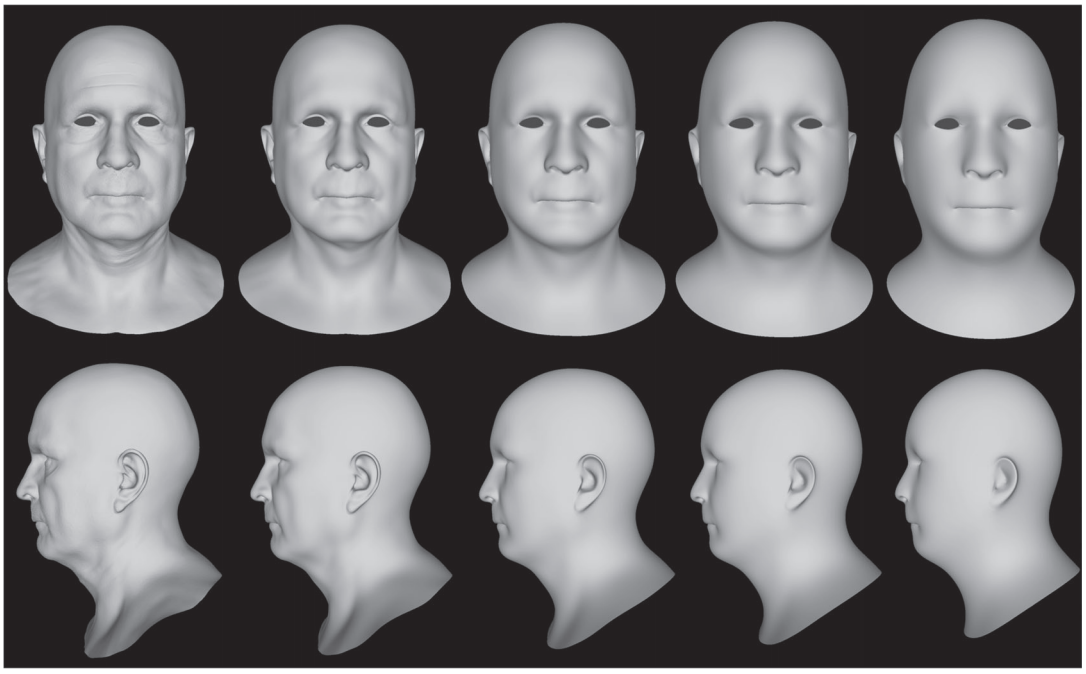

FIGURE 4 Four levels of smoothing for a mesh with approximately 1.2 million vertices. From the left to right, the original model is followed by smoothed meshes using 500, 2500, 5000, and 10,000 iterations, respectively

Due to its simplicity and speed, Laplacian smoothing can be considered one of the most popular smoothing methods.[34] It can be derived from a finite difference approximation of the Laplace operator.[44] Vartziotis et al.[40] discuss the efficiency and effectiveness of Laplacian smoothing and introduce a class of approaches known as geometric element transformation methods. It turns out that Laplacian smoothing of surface meshes maximizes a concave quality function, which is intimately related to the mean ratio quality measure.

Others common approaches for mesh smoothing use global operators, such as radial basis function,[45] B-splines,[29] or wavelets.[46] These are mathematical functions able to interpolate, on a distance basis, scalar information known only at discrete points (source points).[47] The quality and the behavior of the interpolation depend both on the function and on the kind of chosen basis functions.[48]

In our proposal, we adopt Laplace's equation for mesh smoothing as it satisfies the minimum/maximum principle. In other words, this means that the values of the interior displacements are bounded by the values on the boundary, ensuring that interior nodes will not cross the mesh boundaries. In the case of faces this is a key property since we want to preserve the boundaries of the eyes, mouths, and nostrils, for example. In addition, the Laplacian operator preserves the topology, which is also a requirement for our method since we need one-to-one vertex correspondence for any smoothness level.

Briefly, the Laplacian Smooth operator works by iteratively displacing each vertex based on local information, such as the position of neighbors. The Laplacian smoothing operation on vertex v_i may be defined as:

$$v_i' = \frac{1}{N}\sum_{j=1}^{N} v_j, \tag{1}$$

where N is the valence of vertex i, v_j is the position of the jth adjacent vertex of v_i, and v_i' is the new position for v_i. The operator can be applied multiple times on all vertices to achieve further mesh smoothing. Figure 4 shows an example of Laplacian smoothing being applied to a face's quad mesh.

5 | BLENDING DETAILS BETWEEN FACES

In this section, we describe our method to blend multiple faces by carrying the details of one or more source meshes to a target one. We start by describing how to blend two faces. Given a source C and a target B, we produce a new 3D model

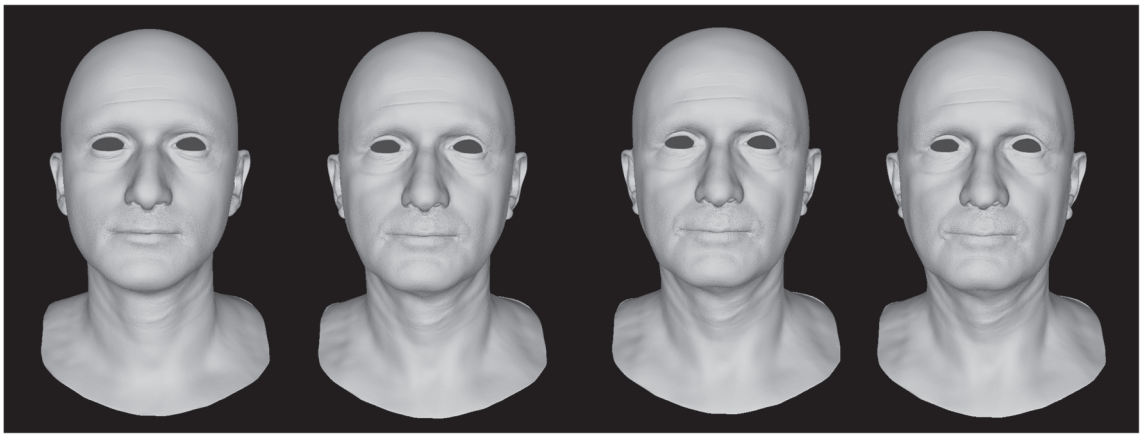

FIGURE 5 Blending the source and target faces from Figure 2 with the following number of smoothing iterations L from the left to right: 10,000, 14,000, 18,000, and 20,000

A by transporting the details from C to B in the following way:

$$A = B^L + (C - C^L), \tag{2}$$

where B^L is the face B smoothed by L Laplace iterations, and likewise for C^L.

Since the meshes have the same topology and a one-to-one correspondence between vertices, which also share the same uv texture coordinates, we only need a single parameter to control the blending results, the number of iterations L for the Laplacian smoothing. Higher values of L lead to more smoothing of the source and target faces and, consequently, to more details being transferred from the target to the source mesh. Figure 5 illustrates the process of blending two faces.

The method may be extended for multiple faces. In this case, Equation (2) becomes:

$$A = B^L + \frac{\sum_{i=1}^{n} \omega_i (C_i - C_i^L)}{\sum_{i=0}^{n} \omega_i}, \tag{3}$$

where n is the number of source faces to be used in the composition and ω_i is the weight for source face i. We normalize the contributions to avoid aberrations.

We define the weight ω_i of each face using Euclidean distances measured on a 2D plane. Faces are manually arranged on a 2D plane, where t_i is the position of face i. Then, for any 2D blending position m the weight of face i is defined as the inverse of the squared distance between m and t_i:

$$\omega_i = \frac{1}{||m - t_i||^2}. \tag{4}$$

To avoid computing B^L and C^L every time a new smoothness factor L is selected, we define approximated smoothed faces $\tilde{B}^L$ and $\tilde{C}^L$, as described in Section 5.1. In the same way, we define the approximated displacement vector that inserts details in $\tilde{B}^L$ as

$$\tilde{D} \approx \frac{\sum_{i=1}^{n} \omega_i (C_i - \tilde{C}_i^L)}{\sum_{i=0}^{n} \omega_i}. \tag{5}$$

Finally, the final mesh for any smoothness level L becomes

$$A = \tilde{B}^L + \tilde{D}. \tag{6}$$

Notice that the weights are normalized but, by removing this restriction, we can also create faces that extrapolate the details. This interesting side effect is achieved by modulating the weights by some factor, as illustrated in Figure 6.

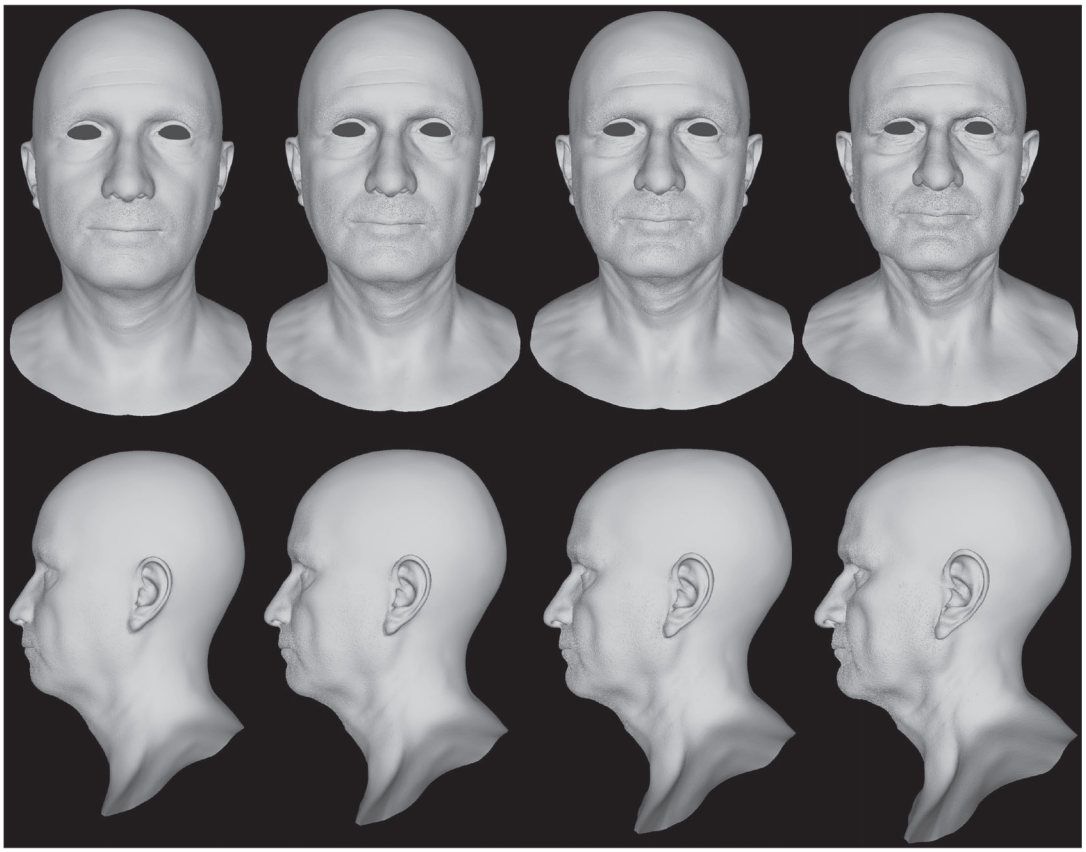

FIGURE 6 Detail extrapolation by manipulating the weights from the example shown in Figure 2. From the left to right scaling all weights by 0.75, 1.0, 1.2, and 1.4. Notice that not only details are intensified but the global shape of the face is also deformed for high values

5.1 | Approximating smoothness levels

To compute approximations $\tilde{B}^L$ and $\tilde{C}^L$ of the smoothed levels, we first define a maximum level $L_{\max}$. A naive way to approximate the intermediate smoothed levels is to linearly interpolate between the original face and $L_{\max}$. Naturally, linearly interpolating the displacement vector is not the same as computing the Laplacian smoothing operator a given number of times. With this approximation, some fine details remain after smoothing the mesh and are not properly transferred. Consequently, $\tilde{B}^L$ and $\tilde{C}^L$ result in poor approximations of B^L and C^L. Instead, we approximate the displacement of each vertex along with the multiple smoothing operations by a cubic polynomial using a least-squares method.[49,50] Moreover, since it has a fixed number of parameters, we also have a constant representation of the displacement vectors independently of $L_{\max}$.

Nevertheless, fitting the curve using all $L_{\max}$ samples would impose a large memory overhead as we need to store all intermediate smoothed faces. Thus, we select a few levels to be stored in order to accelerate the preprocessing and avoid storage issues. Since the finer details are smoothed out after a few iterations, as illustrated in Figure 4, we use more levels from the first iterations and less nearer $L_{\max}$ in order to preserve these details. In total we use around 40 levels to fit the curve, which is a massive reduction since $L_{\max}$ is typically larger than 10,000.

The maximum error between the curve fitted with all levels and the original smoothed faces is very low, around 1×10^{-7} cm. Using only 40 levels to fit the curve results in a slightly higher error, around 3×10^{-6} cm. Nevertheless, it is still negligible for any visual purposes.

5.2 | Multiple regions

To add more control and broaden the possible space of faces, we segment the face into separate regions, where for each region we can apply the method separately. This is, naturally, optional and one could decide to split the mesh into any

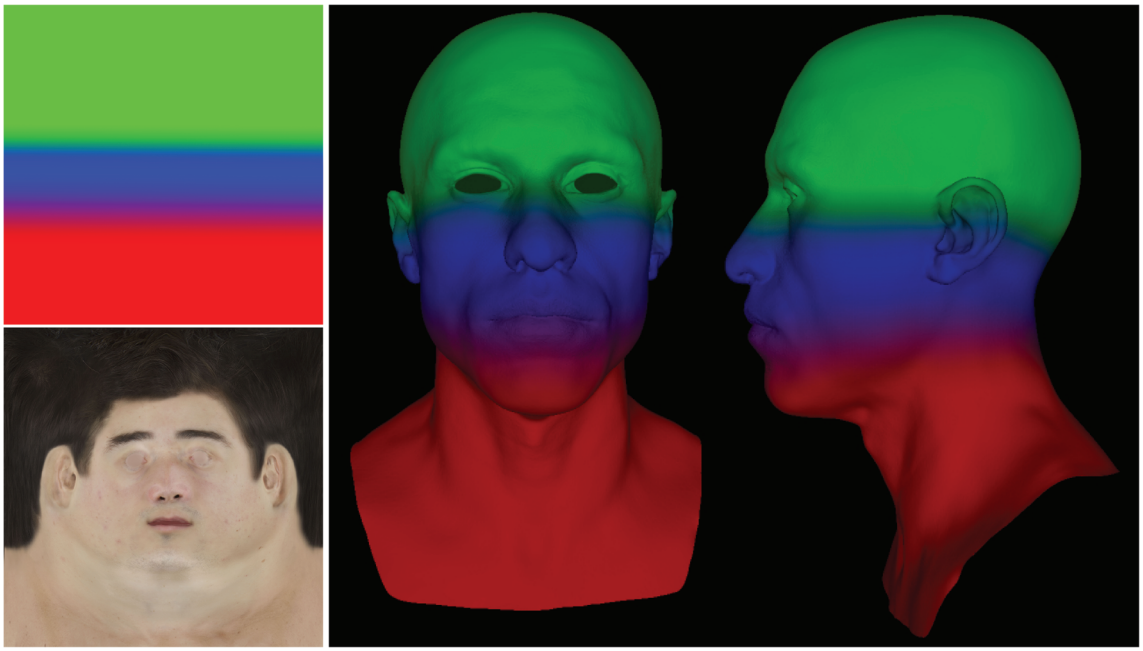

FIGURE 7 An example of using region masks manually defined. On the left, one of the original textures and the painted regions in texture space. Note the blending region between the top (red) and middle (blue), and middle (blue) and bottom (green) masks to achieve a smooth transition. On the right, the resulting regions applied on one of the heads

number of regions. For our test cases, we defined three horizontal regions by creating a color map on the texture uv space. We assigned three different regions with transition bands between them to avoid abrupt changes, as shown in Figure 7. The regions were manually defined using a simple image editor software, and we used a gradient brush for the transition bands. Nevertheless, there are no restrictions on how the regions may be defined and, for example, a 3D brush can be used to paint directly over the mesh to assign regions to vertices.

Taking into account the regions mask, Equation (6) now becomes

$$A = \sum_{k=1}^{r} (\tilde{B}^{L_k} + \tilde{D}_k) * \alpha_k, \tag{7}$$

where α_k is a scalar in the range $[0, 1]$ that defines the weight for region and is taken from the mask color channel, r is the number of defined regions, $\tilde{B}_k^{L_r}$ is the smoothed region k of the approximated target mesh $\tilde{B}$ and $\tilde{D}$ is the approximation displacement vector as defined in Equation (5). Note that we can define different levels of smoothness L_k for each region. Furthermore, we can define one blending position m_k for each region.

6 | THE SYSTEM

In order to perform tests and validate our method, we have developed a real-time visualization system with a simple and intuitive interface. With this tool, it is possible to load the head meshes and textures, select the meshes we want to blend and dynamically move the controllers and change the blend parameters. For example, parameter L in Equations (2) and (3) is controlled by a slider as a percentage value $\frac{L}{L_{\max}}$. The blending position m that defines the blending weights is controlled by moving the position of a circle. The location of each thumbnail representing a model can also be dynamically changed by moving them and, thus, influencing more or less the final blending. Figure 8 shows the interface and some of its functionalities.

In our prototype, we have included a controller for each face region, in order to manipulate each weight m_k independently. Furthermore, we include one slider to set the smoothness level L_k for each region.

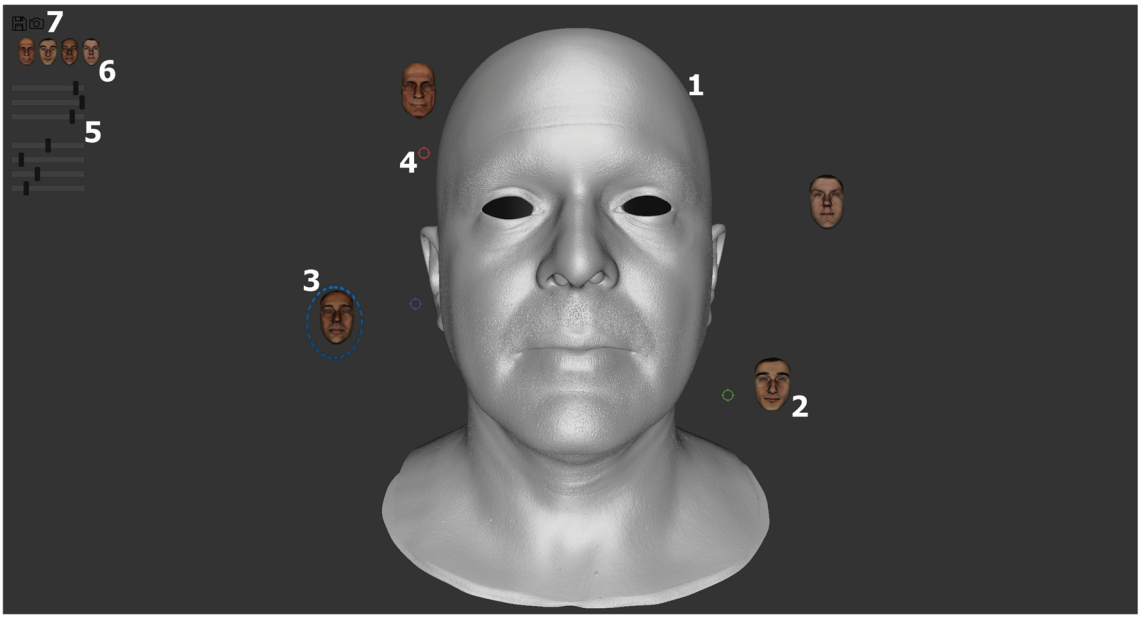

FIGURE 8 The main screen of our interface. (1) resultant mesh A; (2) one of the thumbnails representing a source face; (3) dashed blue circle indicates selected target face B; (4) a controller for a specific region defining position m_k; (5) sliders to control the smoothness level for each region; (6) available textures that can be applied to the models; (7) buttons to save resulting model or image

The only preprocessing required is smoothing each face with $L_{\max}$ iterations and fitting the cubic curves to approximate the vertex displacements. We have not made any efforts to further optimize this preprocessing since it can be seen as a final step of the acquisition pipeline which, in turn, may take in total a couple of days considering the manual artistic efforts involved. In addition, the face needs only to be prepared once and can be used multiple times within the system to generate new faces. Consequently, the time to prepare the smoothed head model is negligible within the production pipeline.

We use the original quad meshes for all operations and only transform into a triangular mesh for rendering purposes. In cases of very high resolution meshes or poor hardware, a potential strategy to keep a real-time responsive interface is to use only the front part of the subject's face instead of the whole head. This approach would reduce the computational costs but still maintain the most significant part of the face in regards to details. Once the desired face is achieved through the interface, the system can easily export the parameters and generate the final solution for the whole head and optionally use directly B^L and D^L in Equation (7), instead of the approximated $\tilde{B}^L$ and $\tilde{D}^L$ versions using the cubic curve. An example of a post-processed full head blend can be seen in Figure 2.

We have implemented and tested our method in a desktop with an Intel Core i7 7700 with 32 GB of RAM and an nVidia RTX 2080 Ti with 11 GB. After loading all smoothed meshes and associated displacement vectors, the system runs entirely in GPU using GLSL shaders. All blending operations are performed interactively with results being immediately composed and visualized. The maximum number of loaded faces is initially bounded by the graphics card video memory. Even though we have not reached this limit in any of our experiments, this restriction can be circumvented by employing a multi-pass strategy. Another option is to lower the mesh resolutions and only apply the blending to high-resolution meshes in CPU when the face creation process is finished. Nevertheless, so far we have not noticed the need for such solutions since we tested our method with up to 15 high-resolution faces while maintaining real-time frame rates.

7 | RESULTS

In this section, we present some results obtained using head meshes with 1.2 million vertices. Preprocessing each face takes approximately 12 min for each face but needs to be done only once.

Although, the method can produce general results as shown in Figure 9, specific combination of faces may also carry other high-level semantic meanings. Note in Figure 2 how by transferring details from an older to a younger person the result can be seen as an ageing effect.

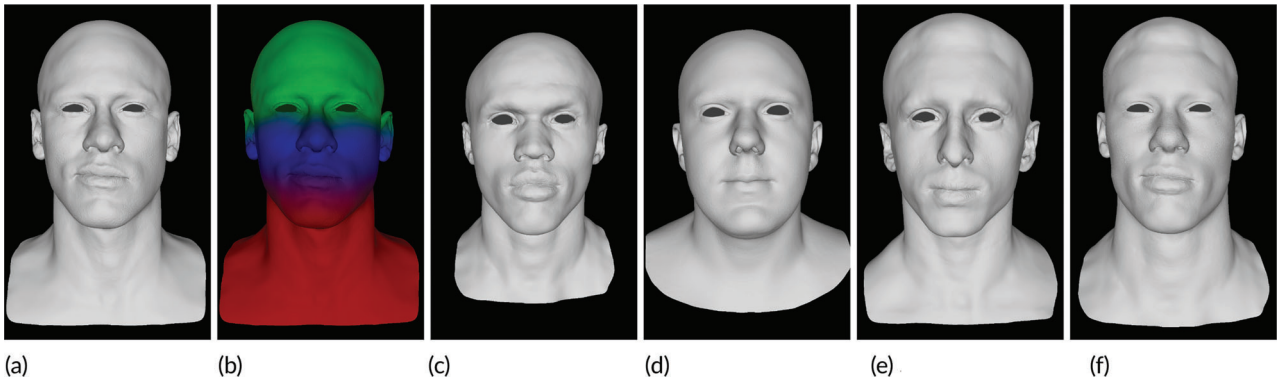

FIGURE 9 Example of using multiple faces with different weights per region. Here, each source face has full weight for only one specific region to better illustrate the effect

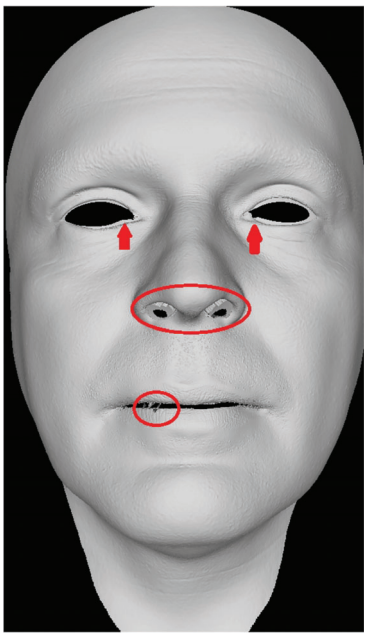

FIGURE 10 The circles and arrows in red highlight boundaries issues with the method from Yoon et al.[29]

TABLE 1 Performance when increasing the number of faces.

# faces	2	4	7	10	13	15
# vertices (millions)	2.4	4.8	8.4	12.0	15.6	18.0
ms/frame	14.1	15.6	17.5	20.4	23.2	25.0

Figure 10 shows a result of blending three different faces, where each face only influences one of the mask regions. Note how for the lower region (green) the lips become thicker but the crooked smiled is preserved, and the fine details from the chin are also transferred. For the nose and eye region (blue), the nose keeps its general shape but retrieves the details such as nasal bone format. The eyes maintain the thin format but receive the contouring lines from the source face. Finally, the forehead region (red) becomes more triangular shaped.

For all tests we have achieved real-time performance. Apart from a constant overhead, the time increases linearly with the number of faces, as expected. As shown in Table 1, even with 15 faces our systems runs at 40 FPS.

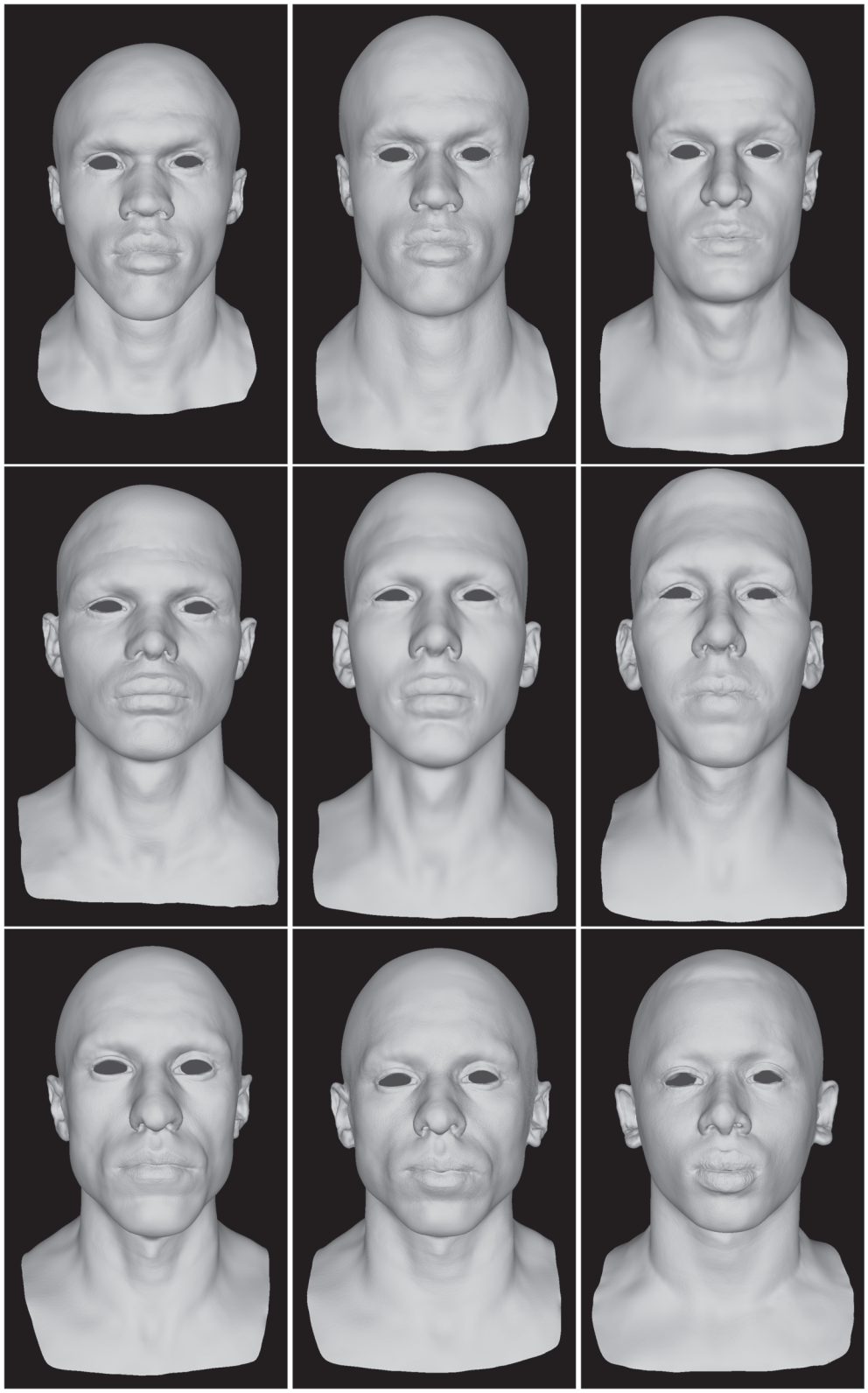

FIGURE 11 Some results of our approach applied on meshes with 1.2 million vertices. For each row, column (a) shows the source face from the details are extract, column (c) shows the target face and column (b) the blending result

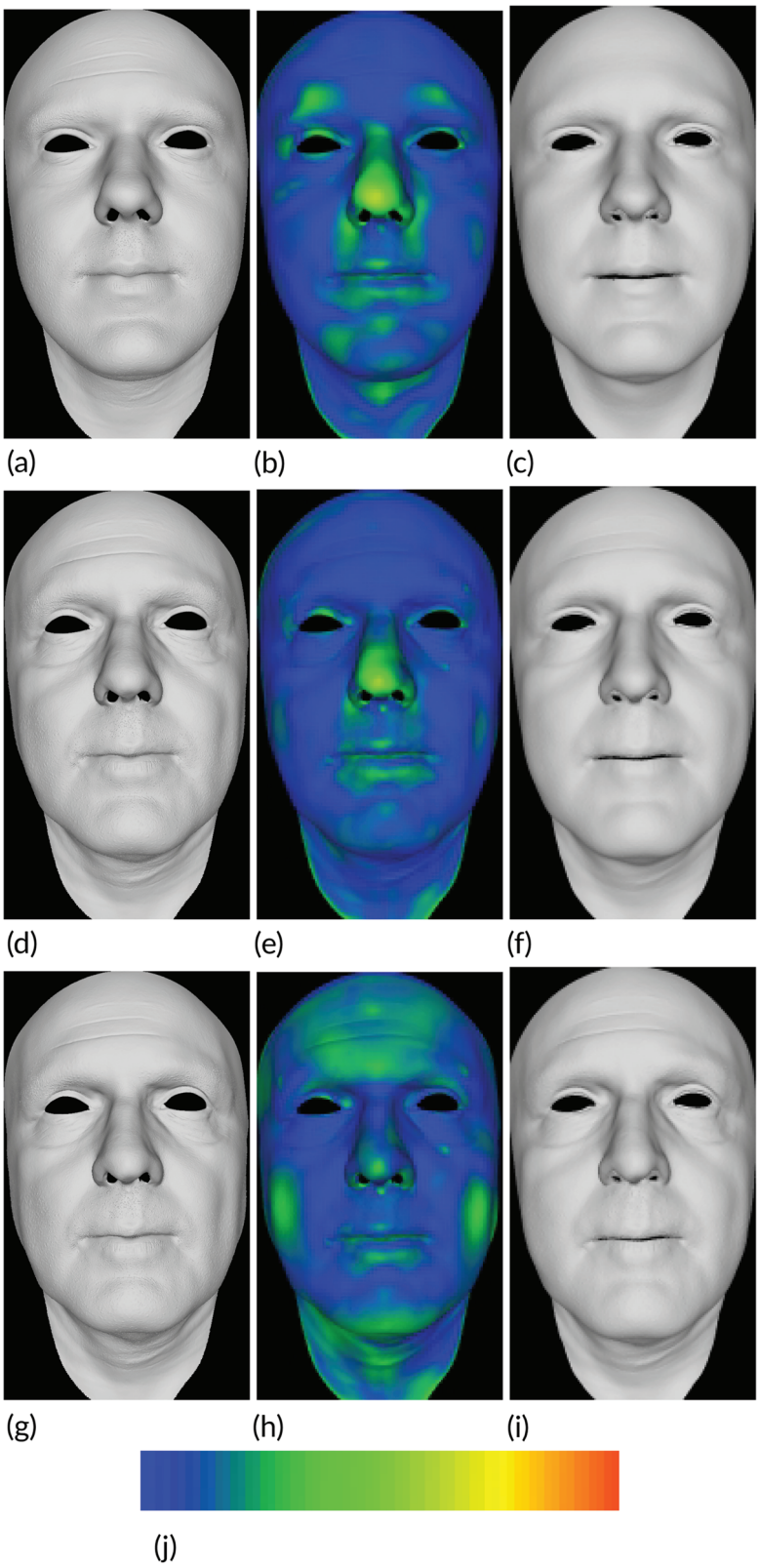

FIGURE 12 Comparison between our method (left column) and our implementation of Yoon et al.[29] method (right column) for three different combinations. The central column shows the difference in a colour scale the vertices of the two meshes when using as similar as possible settings (blue small difference and red large difference). The number of iterations used in our model are (a) 100, (d) 7500 and (g) 19000. For Yoon's method the weights for the base and detail head are, respectively, (d) 0.92 and 0.08, (f) 0.5 and 0.5 and (i) 0.08 and 0.92. Note that the regions with maximum errors, around 0.78cm, are located inside the noses. These results were rendered in Meshlab[51] for a better side-by-side comparison

Furthermore, we compare our method with the approach by Yoon et al.[29] As aforementioned, Yoon's method has a goal very similar to ours, which is to provide a method for artists to create new faces from existing ones. Their method is also based on a hierarchy and by setting a base smoothness level for the target face and attributing weights to the source faces. Nevertheless, their method requires more user intervention as the displacement maps need to be manually composed. Unfortunately, without any extra information on how these maps are composed it is not possible to reproduce the results precisely or to estimate the time the artists must invest in generating new faces. Nonetheless, we have implemented their method and tried to match the initialization of both approaches as close as possible for a fair comparison.

As can be seen from Figure 11, the overall result is similar, but our method is capable of preserving the fine details considerably better. Another advantage of our method is the simplicity, as we can work with a large number of faces and dynamically change the smoothness levels and weights by dragging controllers on a 2D plane. The fact that no manual preparation step is necessary for our method also may lead to benefits within a content generation pipeline, because it allows the artist to quickly test different combinations of faces, and incorporate new faces as soon as they undergo the acquisition pipeline.

Due to the boundary preserving property of Laplacian smoothness, our method can deal with holes in the mesh. This is another important advantage as face models usually contain cavities on eyes, mouth, and nostrils. Our method can handle these issues seamlessly. On the other hand, we found no trivial way to solve this issue with Yoon's method (see Figure 12) and it cannot be noticed from the results in their paper since no examples contain such cavities. Finally, even though we chose to use only the front part of the face in our interface to achieve real-time interaction, this is not an actual limitation of the method and we have shown the results are easily extended to the whole face in a post-processing stage. In the case of Yoon's method using the face only is actually a limitation of the parameterization and we have not found easy ways to extend the result to the whole head.

In contrast, Yoon's method handles faces with different parameterizations while our method depends on an external reparameterization method performed during the acquisition pipeline. Yet, we view this as an orthogonal problem that can be tackled as a preprocessing step, as it is actually done with Yoon's method. As a last remark, our method is remarkably simpler to implement and allows for a very straightforward and efficient GPU implementation.

8 | LIMITATIONS

As discussed, our method cannot handle faces with different mesh topologies. This is not a problem in many content creation pipelines since all faces are digitized the same way and are mapped to a common parameterization, but it still imposes a limitation when incorporating external models. Even though this was not the focus of this article, any reparameterization solution could be coupled to our method to solve this issue.

Although we support segmenting the face into regions, our method is oblivious to other semantics at other levels of the details, thus it becomes hard to extract selected features such as wrinkles or facial hair. Finally, region masks need to be created manually, and a manner to define masks during blending could make the creation process more agile.

9 | CONCLUSION

We have presented a method to generate new faces from a set of digitized high-resolution models. Our method provides an intuitive and simple way to create new faces by transferring details from one or more source faces to a target. Our approach is inserted into a content creation pipeline where faces are digitized to produce background characters. In this way, our method provides a fast way to generate new realistic characters. We place our method in a category of hand-crafted methods, where more control is given and less data are needed, in contrast to methods that rely on learning approaches. We have also provided a real-time interface for rapid creation of new faces.

As a natural future step we will look into blending the textures as well as the geometry. The challenge is that the texture details must be aligned and properly identified to avoid blurring the texture during blending. Furthermore, even if reparameterization is considered as preprocessing in our case, it might be interesting to investigate more automatic solutions in order to allow incorporating models that are not produced by our pipeline, or even non-humanoid faces.

Moreover, we believe that with further research we can tune the method for specific purposes. For example, to automatically produce a new set of original faces to create more diversity among the background characters. Alternatively, it should also be possible to produce faces that maintain some resemblance to produce, say, characters from the same ethnic

background, or that might belong to the same family. Finally, the current method has been applied to faces, we can explore the same strategies to transfer details between different models such as other body parts or even non-animated objects.

ORCID

Ricardo Marroquim https://orcid.org/0000-0001-8299-7067

REFERENCES

1. Statham N. Use of photogrammetry in video games: a historical overview. Games Culture. 2018;15(3):1555412018786415.
2. Zollhofer M, Thies J, Garrido P, Bradley D, Beeler T, Pérez P, et al. State of the art on monocular 3D face reconstruction, tracking, and applications. Comput Graph Forum. 2018;05(37):523–50.
3. Lewis JP, Anjyo K, Rhee T, Zhang M, Pighin F, Deng Z. Practice and theory of blendshape facial models. In: Lefebvre S, Spagnuolo M, editors. Eurographics 2014-State of the art reports. Lugano, Switzerland: The Eurographics Association; 2014. p. 199–218.
4. Blanz V, Vetter T. A morphable model for the synthesis of 3D faces. Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques SIGGRAPH '99. New York, NY: ACM Press/Addison-Wesley Publishing Company; 1999. p. 187–94.
5. Blanz V, Basso C, Poggio T, Vetter T. Reanimating faces in images and video. Comput Graph Forum. 2003;09(22):641–50.
6. Zhang Y, Sung E, Prakash E. A physically-based model for real-time facial expression animation. Proceedings 3rd International Conference on 3-D Digital Imaging and Modeling; 2001. p. 399–406.
7. Terzopoulos D, Waters K. Physically-based facial modelling, analysis, and animation. J Vis Comput Animat. 1990;1:73–80.
8. Beeler T, Hahn F, Bradley D, Bickel B, Beardsley P, Gotsman C, et al. High-quality passive facial performance capture using anchor frames. ACM Trans Graph. 2011;30:75:1–75:10.
9. Beeler T, Bickel B, Beardsley P, Sumner B, Gross M. High-quality single-shot capture of facial geometry. Proceedings of the ACM SIGGRAPH 2010 Papers SIGGRAPH '10, New York, NY: ACM; 2010. p. 40:1–9.
10. Rhee T, Hwang Y, Kim JD, Kim C. Real-time facial animation from live video tracking. In: Bargteil A, van de Panne M, editors. Eurographics/ACM SIGGRAPH Symposium on Computer Animation. Lugano, Switzerland: The Eurographics Association; 2011. p. 215–24.
11. Ma WC, Jones A, Chiang JY, Hawkins T, Frederiksen S, Peers P, et al. Facial performance synthesis using deformation-driven polynomial displacement maps. ACM Trans Graph. 2008;12(27):121.
12. Fyffe G, Graham P, Tunwattanapong B, Ghosh A, Debevec P. Near-instant capture of high-resolution facial geometry and reflectance. Comput Graph Forum. 2016;35(2):353–363.
13. Tian L, Liu J, Guo W. Three-dimensional face reconstruction using multi-view-based bilinear model. Sensors. 2019;01(19):459.
14. Fyffe G, Nagano K, Huynh L, Saito S, Busch J, Jones A, et al. Multi-view stereo on consistent face topology. Comput Graph Forum. 2017;36(2):295–309.
15. Zhu X, Yi D, Lei Z, Li SZ. Robust 3D morphable model fitting by sparse SIFT flow. Proceedings of the 2014 22nd International Conference on Pattern Recognition; 2014. p. 4044–9.
16. Lowe DG. Distinctive image features from scale-invariant keypoints. Int J Comput Vis. 2004;60(2):91–110.
17. Lin K, Wang X, Tan Y. Self-adaptive morphable model based collaborative multi-view 3D face reconstruction in visual sensor network. Multimed Tools Appl. 2016;75(18):11469–91.
18. Dai P, Wang X, Zhang W. Coarse-to-fine multiview 3D face reconstruction using multiple geometrical features. Multimed Tools Appl. 2018;77(1):939–66.
19. Sorkine O, Cohen-Or D, Lipman Y, Alexa M, Rössl C, Seidel HP. Laplacian surface editing. Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing SGP '04; New York, NY: ACM; 2004. p. 175–84.
20. Sumner RW, Popović J. Deformation transfer for triangle meshes. ACM Trans Graph. 2004;23(3):399–405.
21. Shin IK, Oztireli AC, Kim HJ, Beeler T, Gross M, Choi SM. Extraction and transfer of facial expression wrinkles for facial performance enhancement. In: Keyser J, Kim YJ, Wonka P, editors. Pacific graphics short papers. Lugano, Switzerland: The Eurographics Association; 2014. p. 113–8.
22. Romeiro R, Marroquim R, Esperanca C, Breda A, Figueredo C. Forensic facial reconstruction using mesh template deformation with detail transfer over HRBF. Proceedings of the Brazilian Symposium of Computer Graphic and Image Processing; 2014. p. 266–73.
23. Ma WC, Barbati M, Lewis JP. A facial composite editor for blendshape characters. Proceedings of the Digital Production Symposium DigiPro '12; New York, NY: ACM; 2012. p. 21–26.
24. Booth J, Roussos A, Ponniah A, Dunaway D, Zafeiriou S. Large scale 3D morphable models. Int J Comput Vis. 2018;126(2-4):233–54.
25. Egger B, Smith WAP, Tewari A, Wuhrer S, Zollhoefer M, Beeler T, et al. 3D morphable face models - past, present and future. ACM Trans Graph. 2020;39(5):1–38.
26. Ploumpis S, Wang H, Pears NE, Smith WAP, Zafeiriou S. Combining 3D morphable models: a large scale face-and-head model. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019:10934–10943.
27. Guo Y, Zhang J, Cai J, Jiang B, Zheng J. CNN-based real-time dense face reconstruction with inverse-rendered photo-realistic face images. IEEE Trans Pattern Anal Mach Intell. 2019;41(6):1294–307.
28. Li R, Bladin K, Zhao Y, Chinara C, Ingraham O, Xiang P, et al. Learning formation of physically-based face attributes. Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020. p. 3407–16.

29. Yoon S, Lewis J, Rhee T. Blending face details: synthesizing a face using multiscale face models. IEEE Comput Graph Appl. 2017;37(6):65–75.
30. Lee S, Wolberg G, Shin SY. Scattered data interpolation with multilevel B-splines. IEEE Trans Visual Comput Graph. 1997;3(3):228–44.
31. Agisoft. Metashape. Available from: https://www.agisoft.com/ (2021). Accessed 15 Dec 2021.
32. Pixologic. Zbrush. Available from: http://pixologic.com/. (2021). Accessed 15 Dec 2021.
33. R3DS Wrap 3.3. Available from: https://www.russian3dscanner.com/. (2022). Accessed 14 Feb 2022.
34. Chen L, Zheng Y, Chen J, Liang Y. An improved Laplacian smoothing approach for surface meshes. In: Shi Y, van Albada GD, Dongarra J, Sloot PMA, editors. Computational science – ICCS 2007. Berlin, Heidelberg: Springer; 2007. p. 318–25.
35. Vollmer J, Mencl R, Müller H. Improved Laplacian smoothing of noisy surface meshes. Comput Graph Forum. 1999;18:131–8.
36. Cignoni P, Montani C, Scopigno R. A comparison of mesh simplification algorithms. Comput Graph. 1997;22:37–54.
37. Freitag L, Jones M, Plassmann P. An efficient parallel algorithm for mesh smoothing. Argonne, IL: Argonne National Lab; 1995;47–58.
38. Parthasarathy VN, Kodiyalam S. A constrained optimization approach to finite element mesh smoothing. Finite Elem Anal Des. 1991 Sep;9(4):309–20.
39. Field DA. Laplacian smoothing and delaunay triangulations. Commun Appl Numer Methods. 1988;4(6):709–12.
40. Vartziotis D, Athanasiadis T, Goudas I, Wipper J. Mesh smoothing using the geometric element transformation method. Comput Methods Appl Mech Eng. 2008;08(197):3760–7.
41. Shimada K, Yamada A, Itoh T. Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. Int J Comput Geometry Appl. 2000;08(10):417–40.
42. Canann SA, Tristano JR, Staten ML. An approach to combined Laplacian and optimization-based smoothing for triangular, quadrilateral, and quad-dominant meshes. IMR. 1998;1:479–94.
43. Freitag LA. On combining Laplacian and optimization-based mesh smoothing techniques. American society of mechanical engineers, applied mechanics division, AMD; 1999. p. 220.
44. Buell WR, Bush BA. Mesh generation — A survey. J Eng Ind. 1973;02(95):332.
45. Biancolini M. Mesh morphing and smoothing by means of radial basis functions (RBF): a practical example using fluent and RBF. Vol 15. Morph Hershey, PA: IGI Global; 2011.p. 347–80.
46. Knipping J, Du Toit J, Vuik C. Comparison of wavelets for adaptive mesh refinement. Technical report TR1/20 of Technical report NAG; 2020.
47. Groth C, Chiappa A, Biancolini M. Shape optimization using structural adjoint and RBF mesh morphing. Proc Struct Integr. 2018;01(8):379–89.
48. Boer A, Schoot M, Bijl H, Bijl H. Mesh deformation based on radial basis function interpolation. Comput Struct. 2007;85:784–95.
49. Bjorck A. Numerical methods for least squares problems. Society for Industrial and Applied Mathematics. 1996. https://doi.org/10.1137/1.9781611971484
50. Weisstein EW, Least squares fitting–polynomial. Available from: https://mathworld.wolfram.com/LeastSquaresFittingPolynomial.html. (2021). Accessed 15 Dec 2021.
51. Cignoni P, Callieri M, Corsini M, Dellepiane M, Ganovelli F, Ranzuglia G. MeshLab: an open-source mesh processing tool. In: Scarano V, Chiara RD, Erra U, editors. Eurographics Italian Chapter Conference. Lugano, Switzerland: The Eurographics Association; 2008. p. 129–36.

AUTHOR BIOGRAPHIES

Diego Mazala Diego Mazala is a software engineer with experience working in computer graphics areas, such as augmented and virtual reality. He graduated from Petropolis Institute of Technology and has a master's degree in Computer Graphics from the Federal University of Rio de Janeiro. He is currently a PhD candidate at the same institute.

Claudio Esperança is an Electric Engineer, graduated from the Federal University of Rio de Janeiro (UFRJ) in 1980. He received a MSc degree at the Systems Engineering and Computer Science (PESC). In 1995, he earned a PhD degree in Computer Science at University of Maryland. In 1998 he became a professor at PESC, where he pursued his academic interests. These include all things related to Computer Graphics. In 2021 he also joined UFRJ's Graduate Program in Design as a professor.

Ricardo Marroquim received the Ph.D. degree from the Federal University of Rio de Janeiro (UFRJ) in 2008. He was an ERCIM Post-Doctoral Fellow with the Visual Computing Lab in Pisa and in 2009, he joined as a Professor with UFRJ. Since 2019 he is an Assistant Professor at the Delft University of Technology with the Computer Graphics and Visualization Group.

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Mazala D, Esperança C, Marroquim R. Laplacian face blending. Comput Anim Virtual Worlds. 2022;e2044. https://doi.org/10.1002/cav.2044