Acta Cybernetica **23** (2018) 1055–1069.

An Approximative and Semi-automated Method to Create MPEG-4 Compliant Human Face Models[*]

Ákos Tóth[a] and Roland Kunkli[a]

Abstract

In this paper, we introduce our method to facilitate the process of creating an MPEG-4 compliant face model based on a simple 3D mesh. The presented method is semi-automatic, and the user needs to choose several points on the model as a preprocessing step. We use a cage based deformation technique to approximate an input model with a generic one which already contains the required MPEG-4 parameters. In the paper, we also show how the cage can be constructed to surround the model completely and be close enough to the head to get better deformation results. After these steps, the resulting model can be used in any MPEG-4 based facial animation player.

Keywords: MPEG-4, talking head, facial animation, automation, cage based deformation

1 Introduction

The usage of virtual avatars—also called talking heads in the case of eliminating the body part—is widespread in HCI (Human-Computer Interaction), and they may play an essential role in the future as well [9].

Methods for creating a talking head based on the face of a real or a fictive person typically consist of two main steps. The first one is the creation of the model itself, while the second step is to prepare our model for further facial animations. There are several different approaches for this preparation from complex manual parameterizations [18, 26] all the way up to automatic motion capture based techniques [4].

Some of the main challenges are the portability and the reusability of the model prepared by the previously mentioned methods. Their principles can be very different; therefore, in most cases, the conversion between the outputs is nearly impossible. For solving this problem, many earlier systems [2, 7] support the well-known

[*]This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.

[a]University of Debrecen, Faculty of Informatics, E-mail: `{toth.akos, kunkli.roland}@inf.unideb.hu`

DOI: 10.14232/actacyb.23.4.2018.5

MPEG-4 facial and body animation standard [16]. Using MPEG-4, we can guarantee that the desired animation can be attached to any standard model automatically, but unfortunately, the creation of an MPEG-4 compliant face model requires a lot of manual user interactions. With all this in mind, our goal in this paper is to give a possible solution to this problem by reducing the amount of required interaction in a semi-automatic way. In the next section, we give a short overview related to MPEG-4 facial animation and its usage in science. Then, in Section 3, we discuss some previous works related to the area of MPEG-4 facial animation. We highlight their main advantages and disadvantages as well. In Section 4, we present our method, which can provide a solution to some of these mentioned disadvantages, based on a prepared generic model and a cage based mesh deformation technique. We demonstrate the results of our algorithm in Section 5. Then, in the last section, we discuss the conclusions and our future ideas as well.

2 Facial animation in MPEG-4

In March of 1999, the Moving Picture Experts Group announced MPEG-4 as an ISO standard also for facial animation. Nowadays, it is the only widely accepted standard for this kind of application; furthermore, the industry also pays close attention to it [23]. The standard defines the parameters, which control the animation of a human face, in detail. All possible natural facial expressions are available using these parameters.

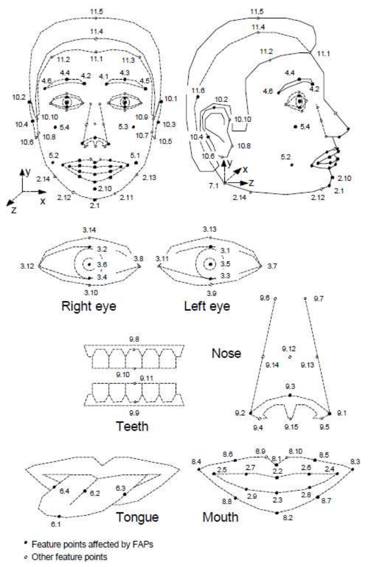

Figure 1: The MPEG-4 Feature Points [16].

MPEG-4 defines 84 Feature Points (FPs) on a neutral face as shown in Figure 1.

Besides that, the zone of the influence of each FP must be set for a realistic and believable animation. These FPs are described in <*group*>.<*index*> format, where *group* stands for the feature type like eyes, cheek, or mouth, and *index* is just a label. The FPs are employed to supervise the animation, but they have to be controlled at the same time in order to reach the desired facial expression properly. The 68 FAPs (Facial Animation Parameters) are used to manipulate these FPs on the face to produce natural facial expressions. FAPs are universal parameters, but we have to calibrate them before their usage. This calibration is feasible by using the so-called Face Animation Parameter Units (FAPU). FAPUs provide the interpretation of FAPs on any facial models. Each unit corresponds to the fraction of distances between fundamental facial features (e.g., the distance between the mouth and the nose). So a 3D model together with its FPs and FAPUs is perfectly enough to be able to animate the model in any MPEG-4 compatible facial animation player, e.g., [2, 7, 20, 19]. Owing to the previously mentioned advantages of the standard, researchers like to use MPEG-4 facial animations in several different projects, e.g., for supporting psychosocial treatments [10] or for speech synthesis frameworks [11].

3 Previous work

Nowadays, there are standards and descriptions, which can help in the parameterization based facial animation process, like FACS [17] (Facial Action Coding System) or the MPEG-4 facial and body animation standard. The latter one is often used because it provides a foundation not only for facial animation but also for speech animation by defining control points in the mouth region as well. Therefore, the many parameters on the lips create exact and granular shapes and also provide a useful toolkit for speech articulation systems. Also, owing to the parameters, we can reuse previous facial expressions or animation sequences. There are numerous projects [2, 6, 7, 20] with which we can play an MPEG-4 based facial animation or create the parameterization of the face model.

The face model adaptation (parameterization) heavily depends on the topology of the considered mesh. In the case of simple models, which consist of a few hundred vertices, it can be executed easily manually. However, if the models are highly detailed, relatively complex, and include thousands of vertices, then the calibration is not practical. Unfortunately, the adaptation of an existing parameterization to a new facial model is not possible. To solve some of these problems (e.g., the parameterization of the models), Sheng et al. have been released a PDE (Partial Differential Equations) based method [21], but unfortunately it does not support the MPEG-4 standard.

MPEG-4 compliant deformation-based methods have been also published. Escher et al. have been proposed a solution [8] which can create standard faces by using a generic model. If all the feature points of the calibration model are available, then the method fits the generic model with Dirichlet Free Form Deformation. In other cases, a pre-processing step is required to compute the missing feature points using a cylindrical projection and a Delaunay triangulation. Lavagetto et al. introduced a method [13] which can help in the calibration process. Based on the feature points, they used an RBF (Radial Basis Function) to reshape the geometry of the neutral face. However, due to the limitations of the used deformation algorithm (i.e., RBF and Free Form Deformation [22]), the techniques cannot provide a satisfactory deformation of the generic model, especially in the case of the nose, the eyes, and the ears (see Figure 2).

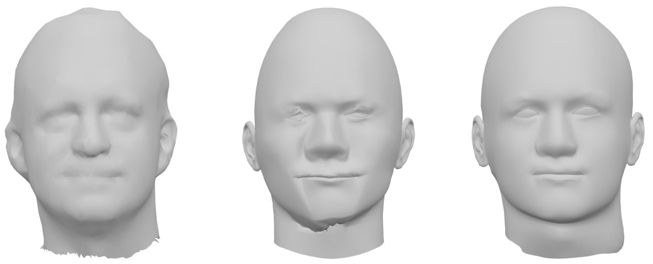
Figure 2: Input model (left), the results of the deformation using Surface-Oriented Free-Form Deformation (middle) and cage based deformation technique (right) which is the basis of our approach.

As we discussed in Section 1, our primary goals are to ease the adaptation and avoid possible errors caused by the users. We suggest a semi-automatic method based on a generic model which should be deformed in order to approximate the input one. MPEG-4 parameters are already specified on the generic model. Thus, the face model calibration can be skipped. For the manipulation of the generic model, we used a deformation method called cage based deformation technique (see Figure 3). One of the main advantages of these methods against other deformation solutions is that using them we can work in real-time, and we have an easy to use, smooth, and intuitive control over the mesh with the defined cage [15].

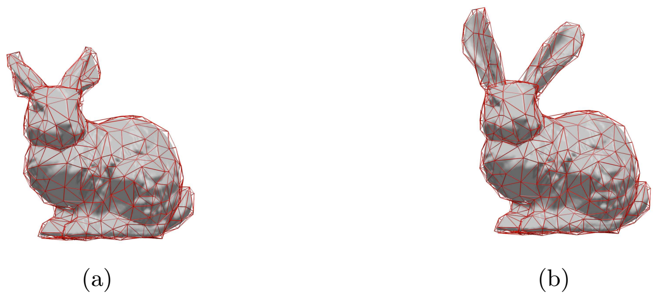
Figure 3: (a) An original mesh and its surrounding cage. (b) We can move the vertices of the cage, thereby generating a smooth deformation of the model.

4 Our method for automating the face model calibration

As we mentioned in Section 1, the creation of an MPEG-4 compliant face model requests a lot of time if we do it manually because we need to define all of the 84 feature points and the zone of influence for each FP. Thus, we suggest a semi-automatic solution to create a standard face if its simple 3D mesh is given. At first, we consider a specified generic model together with all its predefined standard points, and then a cage based method is used to approximate the input model with the generic one. Therefore, the whole procedure of the face model adaptation does not need to be executed.

4.1 Generic model

In the area of facial animation and human modeling, the generic model has to satisfy some requirements. In order to get a realistic talking head animation, the generic mesh must be fine triangulated in high-curvature regions, while low-curvature areas are allowed to contain larger triangles. Naturally, if the goal is a cartoon talking head, the usage of a less detailed generic mesh is more practical. Besides that, the lower and upper lips must be separated from each other so that the speech can be animated as well. We integrated the generic model (see Figure 4) into our system from an open-source application called Xface [2]. We calibrated it to be MPEG-4 compliant, and we modified the geometry of it to get a more neutral face, and we removed the hair, the tongue, and the teeth in order to handle it easily for future deformations. Thus, using this generic model and its FDP (Face Definition Parameters) file we can execute any deformations, and the resulting model will be usable in any MPEG-4 based facial animation player.

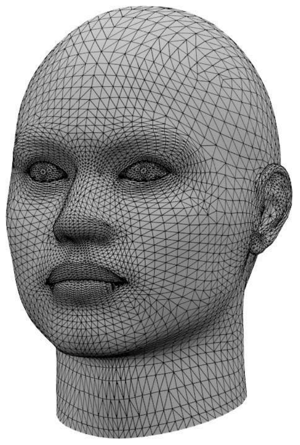
Figure 4: Our generic model that contains 5570 vertices and 11032 triangles.

4.2 Input model

The input head model can be any human-like head model (e.g., realistic models from 3D reconstructions or animation characters). Compared with the generic model, input models are allowed to have lower resolution or defects as well.

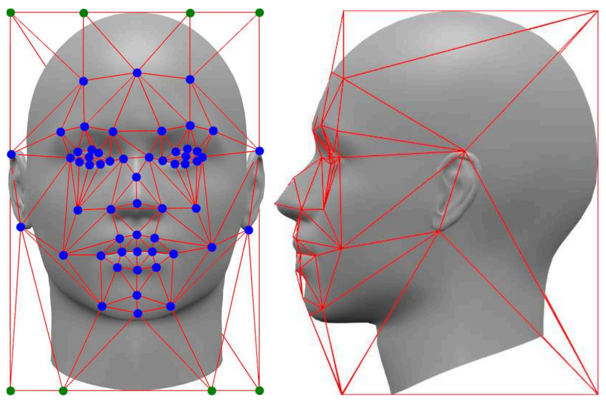

Figure 5: The control cage which has been created for the generic model. Blue dots are the facial feature points of the model, which need to be marked manually, while the green ones are the auxiliary points. These latter points (top four and bottom four on the left image) are calculated from the bounding box of the model automatically.

4.3 Our deformation method

The necessary modifications of the generic model are achieved by a cage based deformation technique called harmonic coordinates [12]. As we mentioned earlier, a topologically flexible cage (or also known as control mesh) is used to control the deformation of the interior object. If we move the cage vertices C_i to the new positions C'_i, an interior point p moves to the new location p', and it is computed as

$$p' = \sum_i h_i(p) C'_i,$$

where $h_i(p)$ is the harmonic coordinate of p respect to C_i. Therefore, at first, we define simple control cages (shown in Figure 5) surrounding the generic and the input models in the same way, based on pre-marked facial feature points on the heads. These cages are responsible for the deformation. Then, we modify the cage of the generic model by translating each point to the corresponding position on the input model's cage separately. As a result, the generic model sufficiently approximates the input one, and it still remains MPEG-4 compliant.

4.4 Generating the control cages

In order to generate the cages of the models efficiently, we have created a Blender [3] add-on called *Standardize Me* [25] (see Figure 6).

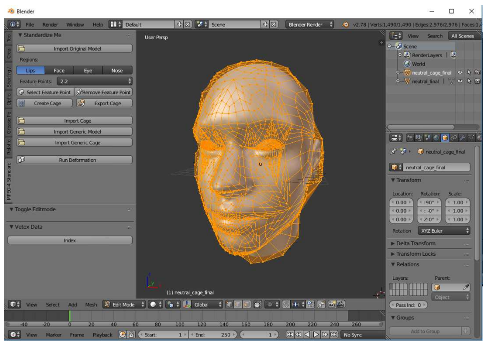

Figure 6: The user interface of our *Standardize Me* Blender add-on.

With our Blender add-on, the users can load a 3D head model and mark the vertices of it; these are required for the generation of its control cage. The vertices to be marked are derived from the feature points of the MPEG-4 standard because the characteristic of the human head can be defined properly with these points. Using the add-on, users have to mark 56 positions on the model to generate its control cage automatically.

Because of the harmonic coordinates method requires that all points of the object to be deformed must lie within the cage, we use 12 additional auxiliary points—beyond the user-defined points—for the cage construction as well, which are automatically found by the add-on. The position of these points is based on the minimum and maximum coordinates of the model. We assume that the line of the centers of the eyes is orthogonal to the plane of y and z axes, and the origin is located at the center of mass of the model.

After the above-mentioned points have been defined, we have to apply a triangulation on them to construct the cage itself. The used one (see in Figure 5) is uniform for any head models. Thus, we can define a one-to-one correspondence between the points of the generic cage and the input one. Then, in the last step of the cage generation, we scale the cage to avoid intersections with the input model itself.

4.5 Improvement in the usage of harmonic coordinates

In our view, the method of harmonic coordinates has two important properties. First, the quality of the deformation depends on the number of control cage vertices. So, if we increase the number of these vertices, we can get smoother results, and this is essential in areas where facial animation plays a central role, such as virtual reality, computer games, or the animation film industry. Therefore, we apply a subdivision technique—which can handle a triangulated mesh—on both cages to improve the influence of harmonic coordinates. The used scheme is the following: In each iteration step, an original face of the model is divided into new triangle faces by connecting the midpoints of its edges. It is important to note that in the used subdivision method, the positions of the original cage vertices do not change.

The other good property is that the reduction of the distance between the control mesh and the 3D model results a much more efficient deformation. To this end, we reduce this distance using the following technique. We define rays from the origin (which is at the model's center of mass) through the cage vertices. Then, in the case of the generic model, we translate all cage vertices with equal length vectors on its ray to the maximum level, while keeping the whole model within the cage.

In the case of an input model, the positions of its cage vertices will be the intersection points of the corresponding rays and the model. It follows from this construction that the resulting cage will intersect the input model, but this is not a problem for us because we do not need to apply the method of harmonic coordinates on the input model, unlike the generic one.

The above-mentioned distance reduction technique may give undesirable results around the ears of the models (especially, when the ears are highly detailed). The reason is that the intersections are not located evenly, so the structure of the cage may be changed. Thus, the new positions of the cage vertices, which are close to the ears, will be the minimum or maximum x coordinates of the model, so the smooth deformation of the model's ears can be achievable.

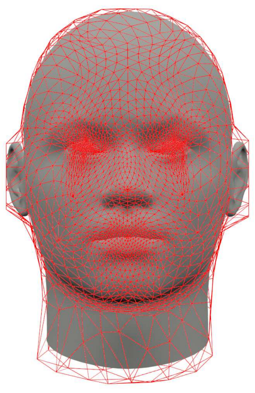

Figure 7: The new control cage of the generic model after applying two iterations of subdivision and the distance reduction technique.

After these modifications, we are able to get a cage which conforms the small changes in the model's geometry as well, and it also fulfils the required conditions. The new cages (see Figure 7) contain approximately 1500 vertices and 3000 triangles, which are achieved by two iterations of subdivision, while the harmonic coordinates and the deformation itself still can be computed in real-time.

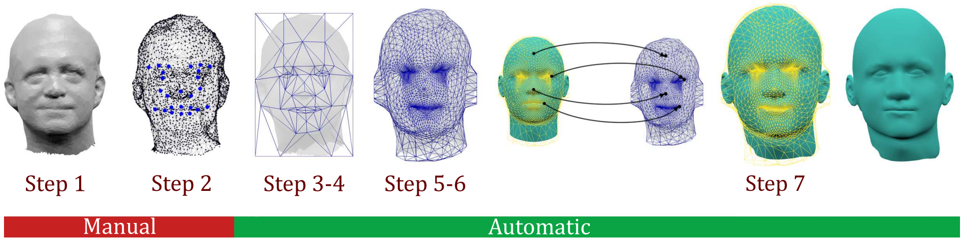

Figure 8: Our semi-automatic MPEG-4 calibration workflow. Each subfigure shows the result of the actual step. Only the first two steps are manual, the user has to define an input model, then mark the necessary points on the model. The further steps are automatic; the algorithm generates the proper control cage for the input model in steps 3–6. Then it translates the vertices of the generic model's control cage to the corresponding ones on the input model's control cage. Because of the applied harmonic coordinates on the generic model, the algorithm deforms the generic model (step 7) which approximates the input one.

4.6 The step by step process of our semi-automatic calibration

We can summarize our method in the following steps. Only the first two steps require user interactions; the remaining ones are automatically executed. Before running the algorithm, we must have a generic model with its control cage and FDP file.

1. Creating a 3D head model of a person or a fictive character, whose talking head we want to create.
2. Marking all necessary points on the 3D head model.
3. Generating the control cage for the input model from the marked and the auxiliary points by using a triangulation.
4. Applying a scaling transformation to the generated control cage in order to avoid possible intersections.
5. Applying two iterations of subdivision on the control cage.
6. Minimizing the distance between the control cage and the 3D head model using the previously mentioned distance reduction technique.
7. Translating the vertices of the generic model's control cage to the corresponding ones on the input model's control cage.

The above-mentioned steps can be seen in Figure 8.

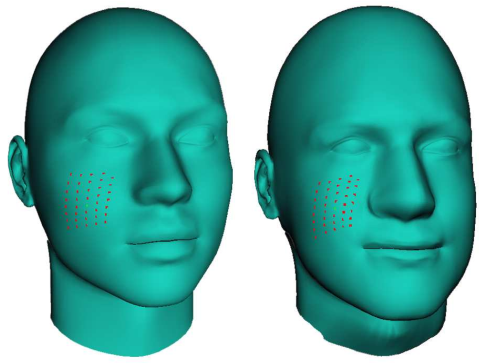

Figure 9: The generic and the deformed model. The green dot marks the 5.4 MPEG-4 FP, while the red ones mark the zone of influence of it.

As we mentioned above, at the end of the process, the cage of the generic model will be transformed into the cage of the input one. The deformation of the generic model depends on the positions of the pre-marked facial features on the input model. The size of the input and the deformed generic model may be a bit different, but we can eliminate this problem by using the dimensions of the bounding box of the original input model. Therefore, we receive a deformed generic model which is similar to the person's face; and all MPEG-4 standard points are already defined on it (see Figure 9).

5 Results

5.1 Implementation

We created the necessary input models in three different ways. We used Microsoft's *Kinect* sensor, an open-source 3D modeling software called *MakeHuman* [24], and a photo based reconstruction application called Autodesk® 123D Catch® [1]. We tried our method on twenty models which are originated from the previously mentioned sources. Considering these systems, *MakeHuman*'s models gave the best results, because they were detailed enough to mark the expected positions of facial features easily, without any defects. In *MakeHuman*, there is a race-related setting, which let us test our solution thoroughly. In the case of the other two, reconstruction based methods, the resolution of the models became a serious limitation. Therefore, we employed a subdivision technique on the meshes before the generation of their cages.

As we mentioned above, our solution is implemented in our self-developed *Blender* add-on, called *Standardize Me*, which is a Python script. The script can be installed in *Blender* as an add-on, and it can be used for executing the whole process of creating an MPEG-4 compliant face model.

5.2 Validation

The comparison of the models (the input model and the resulting one) played a serious role in the improvement of our algorithm. For measuring and visualising the difference between the two mentioned models we used the *one-sided Hausdorff distance* in Meshlab [5] with the following formula:

$$h(\mathcal{A},\mathcal{B}) \equiv \max_{a\in\mathcal{A}} \left(\min_{b\in\mathcal{B}} d(\mathbf{a},\mathbf{b}) \right),$$

where $\mathcal{A}$ is the resulting model, while $\mathcal{B}$ is the input one.

It returns both numerical and visual evaluations of meshes' likeness (see in Table 1). We also computed the *two-sided Hausdorff distance* (see in the last row of Table 1) between the two models by the following way:

$$H(\mathcal{A},\mathcal{B}) \equiv \max(h(\mathcal{A},\mathcal{B}), h(\mathcal{B},\mathcal{A})).$$

	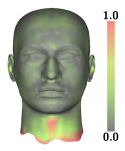	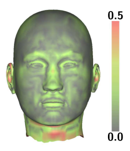	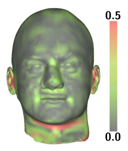	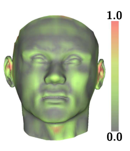	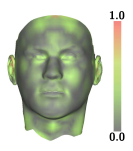
Min	1.1×10^{-5}	4.5×10^{-5}	0.3×10^{-5}	6.6×10^{-5}	0.4×10^{-5}
Max	3.4282	1.2055	1.5712	1.2969	2.0151
Mean	0.1650	0.1163	0.1681	0.2285	0.2033
Avg.	0.3783	0.2621	0.3083	0.2894	0.2989
Std. dev.	0.3175	0.1283	0.1762	0.1710	0.1959
Hausdorff	3.4282	1.6747	3.4446	1.9193	3.6731

Table 1: Distance computations between our results and the approximated input head models. The last row of the table shows the *two-sided Hausdorff distances*, while the values in the other rows are computed by the *one-sided Hausdorff distance*. In the case of the last two reconstructed model, we cut the back of the head model to get accurate measurements. For the sake of better comparison, we also measured the height of the models, which are 24.5233, 23.1186, 24.3174, 25.3531, and 23.5528, from left to right, respectively.

To validate our resulting models, we generated FAPs files using 3D reconstructed human facial expressions from the BU-3DFE (Binghamton University 3D Facial Expression) Database [27]. Then, these FAPs files were played on our resulting face models to get the same expressions. The original reconstructed model and our deformed model with different facial expressions can be seen in Figure 10.

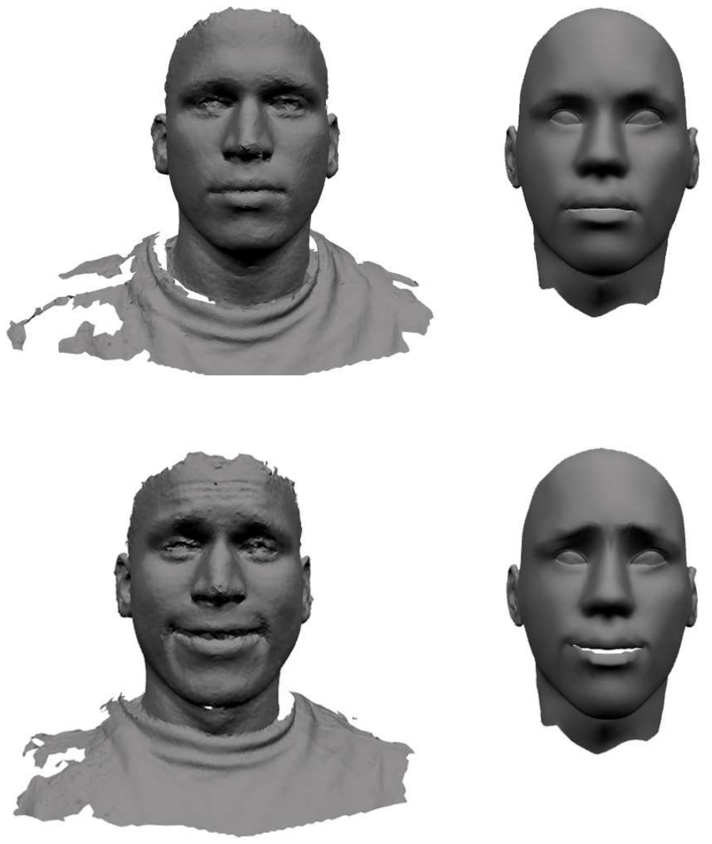

Figure 10: Left: 3D reconstructed human facial expressions from [27]. Top right: Our resulting model with neutral expression. Top left: Our resulting model using the generated FAPs file.

6 Conclusions and future works

In this paper, we have proposed a robust and semi-automatic method for creating an MPEG-4 compliant face model. Only the definition of the feature points needs user interaction, but it takes just a few minutes. In the case of a high-resolution face model ($\approx$3500 vertices, $\approx$7000 faces), the process takes 8 minutes approximately. After applying our method, the resulting talking heads can be used in any MPEG-4 compatible facial animation player immediately.

Our solution uses a cage based deformation method called harmonic coordinates to approximate the input model with the generic one. The main difference from earlier systems is that we do not have to define the MPEG-4 parameters. Therefore, calibration errors do not influence the quality of the animation of the model. Based

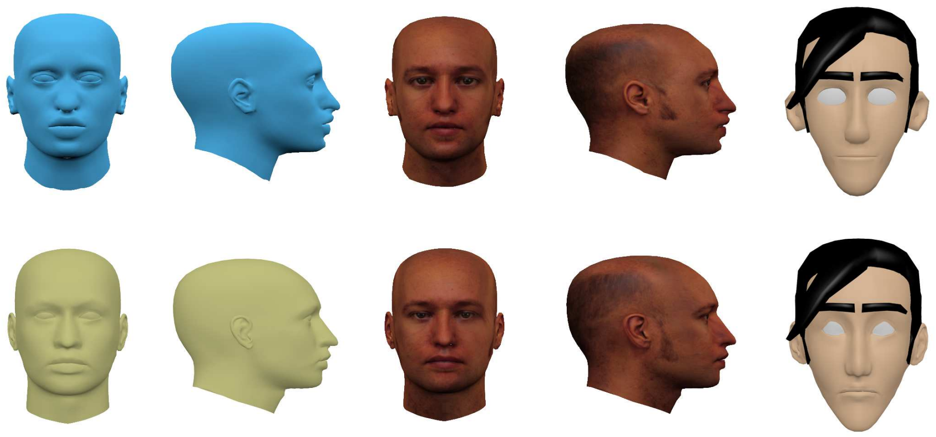

Figure 11: Top: Input models. Bottom: The resulting models after the calibration. The used textures are from [14]. In the last case, the hair is a separate model that has been attached to the result by hand.

on our measurements (see Section 5.2), we can say, that the resulting face models (shown in Figure 11) are similar to the original input, but there may be small differences in some parts of the faces, especially around the eyes, the ears, and the necks. We think that these errors stem from the applied distance reduction technique. Therefore, we would like to try different techniques for minimizing the distance between the cage and the model. A self-organizing neural network may be able to solve this problem. Additionally, we would like to attempt to minimize the number of user interactions by using a face tracking method which can mark the necessary facial feature positions for us. We assume that our method can work with different parameterization (e.g., FACS), but we used the MPEG-4 standard in this paper to create an operating prototype.

References

[1] Autodesk, Inc. Autodesk® 123D Catch®. `http://www.123dapp.com/catch/`. Accessed: 28 May 2015.

[2] Balci, K. Xface: MPEG-4 Based Open Source Toolkit for 3D Facial Animation. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 399–402, 2004.

[3] Blender Online Community. Blender - a 3D modelling and rendering package. `http://www.blender.org/`. Accessed: 7 March 2015.

[4] Cao, C., Weng, Y., Lin, S., and Zhou, K. 3D Shape Regression for Real-time Facial Animation. *ACM Transactions on Graphics*, 32(4):41:1–41:10, 2013.

[5] CNR, Visual Computing Lab ISTI. MeshLab. `http://meshlab.sourceforge.net/`. Accessed: 1 February 2015.

[6] Cosi, P., Fusaro, A., and Tisato, G. LUCIA a New Italian Talking-Head Based on a Modified Cohen-Massaro's Labial Coarticulation Model. In *INTERSPEECH*, pages 2269–2272, 2003.

[7] de Rosis, F., Pelachaud, C., Poggi, I., Carofiglio, V., and De Carolis, B. From Greta's mind to her face: modelling the dynamics of affective states in a conversational embodied agent. *International Journal of Human Computer Studies*, 59(1–2):81–118, 2003.

[8] Escher, M., Pandzic, I., and Thalmann, N. M. Facial deformations for MPEG-4. In *Proceedings Computer Animation '98 (Cat. No.98EX169)*, pages 56–62, June 1998.

[9] Feng, A., Rosenberg, E. S., and Shapiro, A. Just-in-time, viable, 3-D avatars from scans. *Computer Animation and Virtual Worlds*, 28(3-4):e1769, 2017.

[10] Fergus, P., El Rhalibi, A., Carter, C., and Cooper, S. Towards an avatar mentor framework to support physical and psychosocial treatments. *Health and Technology*, 2(1):17–31, 2012.

[11] Jia, J., Zhang, S., Meng, F., Wang, Y., and Cai, L. Emotional Audio-Visual Speech Synthesis Based on PAD. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(3):570–582, 2011.

[12] Joshi, P., Meyer, M., DeRose, T., Green, B., and Sanocki, T. Harmonic Coordinates for Character Articulation. *ACM Transactions on Graphics*, 26(3):71, 2007.

[13] Lavagetto, F. and Pockaj, R. The facial animation engine: toward a high-level interface for the design of MPEG-4 compliant animated faces. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(2):277–289, 1999.

[14] Lundqvist, D., Flykt, A., and Öhman, A. The Karolinska Directed Emotional Faces (KDEF). CD ROM from Department of Clinical Neuroscience. Psychology section, Karolinska Institutet, ISBN 91-630-7164-9. 1998.

[15] Nieto, J. R. and Susín, A. Cage based deformations: A survey. In *Deformation Models: Tracking, Animation and Applications*, pages 75–99. Springer Netherlands, Dordrecht, 2013.

[16] Pandzic, I. S. and Forchheimer, R., editors. *MPEG-4 Facial Animation: The Standard, Implementation and Applications.* John Wiley & Sons, Inc., New York, NY, USA, 2003.

[17] Parke, F. I. and Waters, K. *Computer Facial Animation.* CRC Press, 2008.

[18] Pyun, H., Kim, Y., Chae, W., Kang, H. W., and Shin, S. Y. An Example-based Approach for Facial Expression Cloning. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 167–176, 2003.

[19] Rácz, R., Tóth, Á., Papp, I., and Kunkli, R. Full-body animations and new faces for a WebGL based MPEG-4 avatar. In *CogInfoCom 2015: 6th IEEE International Conference on Cognitive Infocommunications*, pages 419–420, 2015.

[20] Rhalibi, A. E., Carter, C., Cooper, S., Merabti, M., and Price, M. Charisma: High-performance Web-based MPEG-compliant Animation Framework. *Computers in Entertainment*, 8(2):8:1–8:15, 2010.

[21] Sheng, Y., Willis, P., Gonzalez Castro, G., and Ugail, H. PDE-Based Facial Animation: Making the Complex Simple. In *Advances in Visual Computing*, pages 723–732. Springer Berlin Heidelberg, 2008.

[22] Singh, K. and Kokkevis, E. Skinning Characters using Surface Oriented Free-Form Deformations. In *Proceedings of the Graphics Interface 2000 Conference*, pages 35–42, 2000.

[23] Technologies, Visage. Visage Technologies - Face Tracking and Analysis. `http://visagetechnologies.com/`. Accessed: 15 March 2017.

[24] The MakeHuman team. MakeHuman. `http://www.makehuman.org/`. Accessed: 26 September 2014.

[25] Tóth, Á. and Kunkli, R. *Standardize Me.* `https://arato.inf.unideb.hu/kunkli.roland/software/standardize-me-toolkit/` and `https://github.com/dragostej/standardizeme`.
Accessed: 2 October 2017.

[26] Weise, T., Li, H., Van Gool, L., and Pauly, M. Face / Off : Live Facial Puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 7–16, 2009.

[27] Yin, L., Wei, X., Sun, Y., Wang, J., and Rosato, M. J. A 3D Facial Expression Database for Facial Behavior Research. In *7th International Conference on Automatic Face and Gesture Recognition (FGR06)*, pages 211–216, 2006.