FaceWarehouse: A 3D Facial Expression Database for Visual Computing

Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou

Abstract—We present FaceWarehouse, a database of 3D facial expressions for visual computing applications. We use Kinect, an off-the-shelf RGBD camera, to capture 150 individuals aged 7-80 from various ethnic backgrounds. For each person, we captured the RGBD data of her different expressions, including the neutral expression and 19 other expressions such as mouth-opening, smile, kiss, etc. For every RGBD raw data record, a set of facial feature points on the color image such as eye corners, mouth contour, and the nose tip are automatically localized, and manually adjusted if better accuracy is required. We then deform a template facial mesh to fit the depth data as closely as possible while matching the feature points on the color image to their corresponding points on the mesh. Starting from these fitted face meshes, we construct a set of individual-specific expression blendshapes for each person. These meshes with consistent topology are assembled as a rank-3 tensor to build a bilinear face model with two attributes: identity and expression. Compared with previous 3D facial databases, for every person in our database, there is a much richer matching collection of expressions, enabling depiction of most human facial actions. We demonstrate the potential of FaceWarehouse for visual computing with four applications: facial image manipulation, face component transfer, real-time performance-based facial image animation, and facial animation retargeting from video to image.

Index Terms—Face modeling, facial animation, face database, mesh deformation, RGBD camera

1 Introduction

Face models are of great interest to many researchers in computer vision and computer graphics. Different face models are widely used in many applications, including face replacement, face component transfer, image manipulation, face recognition, facial expression recognition, and expression analysis. In recent years, 3D face models became popular in increasingly complex visual computing applications due to the 3D nature of human faces, crucial in solutions to problems caused by ambiguities and occlusions. For instance, the latest approaches to face component transfer [1], video face replacement [2], and single-view hair modeling [3] are all based on 3D face models.

There exist numerous excellent 3D face databases for various purposes. Blanz and Vetter's 3D morphable model [4] is built on an example set of 200 3D face models describing shapes and textures. They then derived a morphable face model, applicable in 3D face reconstruction from a single image. Vlasic et al. [5] presented a multilinear model of 3D face meshes, which contains two separate face models: a bilinear model containing 15 subjects with the same 10 facial expressions and a trilinear one containing 16 subjects with five visemes in five different expressions. The multilinear face model can be linked to a face-tracking algorithm to extract pose, expression, and viseme parameters from monocular video or film footage, and drive a detailed 3D textured face mesh for a different target identity. Yin et al. [6] developed a 3D facial expression database, which includes both prototypical 3D facial expression shapes and 2D facial textures of 100 subjects with seven universal expressions (i.e., neutral, happiness, surprise, fear, sadness, disgust, and anger). The expression database can be used in facial expression recognition and analysis. All of the above face databases contain faces with different identities and expressions, but their expression spaces are not diverse enough for many applications in visual computing, such as the real-time performance-based facial image animation shown in this paper.

We introduce FaceWarehouse, a database of 3D facial expression models for visual computing. With an off-the-shelf RGBD depth camera, raw data sets from 150 individuals aged 7-80 from several ethnicities were captured (see Fig. 5 for a few examples). For each person, we captured her neutral expression and 19 other expressions, such as mouth-open, smile, angry, kiss, and eye-shut. For each RGBD raw data record, a set of facial feature points on the color image such as eye corners, mouth boundary, and nose tip are automatically localized, and manually adjusted if the automatic detection is inaccurate. We then deform a template facial mesh to the depth data as closely as possible while matching the feature points on the color image to their corresponding locations on the mesh. Starting from the 20 fitted meshes (one neutral expression and 19 different expressions) of each person, the individual-specific expression blendshapes of the person are constructed. This blendshape model contains 46 action units as described by Ekman's Facial Action Coding System (FACS) [7], which mimics the combined activation effects of facial muscle

• *C. Cao, Y. Weng, S. Zhou, and K. Zhou are with the State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou 310058, China. E-mail: {caochen, weng, zhoushun}@cad.zju.edu.cn, kunzhou@acm.org.*
• *Y. Tong is with the Department of Computer Science and Engineering, Michigan State University, Engineering Building, 428 S. Shaw Lane #3115, East Lansing, MI 48824. E-mail: ytong@msu.edu.*

Manuscript received 24 Nov. 2012; revised 6 July 2013; accepted 13 Sept. 2013; published online 25 Oct. 2013.
Recommended for acceptance by W. Matusik.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2012-11-0261.
Digital Object Identifier no. 10.1109/TVCG.2013.249.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

groups. It provides a good compromise between realism and control, and adequately describes most expressions of the human face. Owing to the consistent topology of the meshes in the data, we can subsequently organize all the blendshapes of different persons as a rank-3 tensor and construct a bilinear face model with two attributes, identity and expression, through a tensor version of singular value decomposition (SVD).

FaceWarehouse can be used in a wide range of applications in visual computing. In particular, the constructed bilinear face model is used to estimate the identity and expression parameters for faces in images and videos, based on which four applications were developed in this paper. The first application, facial image manipulation, allows users to change geometric facial attributes, such as the size of mouth, the length of face, and ethnicity in a single face image. The second application is face component transfer. Given two images of the same person with different expressions, we can transfer local components such as the mouth or eyes from one image to the other, keeping the transferred components compatible with the overall face shape and other components to give the synthetic image a natural look. The third is real-time performance-based facial image animation, allowing a user to animate a face image of a different person by performing in front of an RGBD camera, all in real time. The final application is facial animation retargeting from video to image. Given video footage with a continuously changing face and a still face image as input, we transfer the head motion and facial expression in the video to the still face in the image.

The main contribution of this paper is an extensive face expression database, which contains 150 persons with 47 different facial expressions for each person. To the best of our knowledge, FaceWarehouse is the most comprehensive 3D facial expression database for visual computing to date, providing data sorely needed in a multitude of applications in both computer graphics and computer vision. To facilitate future research on face related applications, we have made the database available at http://gaps-zju.org/facewarehouse. In addition, we describe how to use the constructed bilinear face model for face identity and expression estimation in facial images and videos. The estimations are accurate enough to support a wide range of applications including animating still face images using real-time RGBD data as well as video footage. We can generate visually plausible facial animations for any portrait image, including those shown in previous work.

In the rest of the paper, we first review some related work in the areas of face model database acquisition and face manipulation applications in Section 2. In Section 3, we elaborate on the pipeline for the construction of FaceWarehouse. Section 4 presents several applications showcasing the potential of FaceWarehouse.

2 Related Work

In this section, we first discuss related work on 2D and 3D face model databases. Some of them focused on neutral expression models, while others contain multiple expressions for dynamical applications. We then discuss applications involving face and head modeling, including face transfer, reanimation, and performance tracking in images and video.

2.1 Face Model Databases

As face databases are of great value in face-related research areas for both modeling and validation of methods, many researchers built their own face database for specific applications. Some of these face databases are composed only of 2D face images, while others contain 3D shape information. Some of them contain only one static expression (the neutral face), which is mostly used in applications involving only static faces, e.g., face recognition, while others also contain other expressions, which can be used in face motion-based applications, e.g., face expression recognition and tracking, and face reanimation in still images and video sequences. In the following, we only review several representative existing works and refer readers to the comprehensive surveys by Gross [8] and Yin et al. [6].

In computer vision, 2D face databases have been widely used as training and/or validation data in face detection and recognition and facial expression analyses. Yin et al. [6] mentioned that although some systems have been successful, performance degradation remains when handling expressions with large head pose rotation, occlusion, and lighting variations. To address the issue, they created a 3D facial expression database. They used a 3D face imaging system to acquire the 3D face surface geometry and surface texture of each subject with seven emotion-related universal expressions. Such a database with few expressions for each person works fine for their face expression recognition and analysis, but may fall short of the diversity required in some applications of visual computing, such as those shown in this paper.

Blanz and Vetter [4] model variations in facial attributes using dense surface geometry and color data. They built the *morphable face model* to describe the procedure of constructing a surface that represents the 3D face from a single image. Furthermore, they supplied a set of controls for intuitive manipulation of appearance attributes (thin/fat, feminine/masculine).

Vlasic et al. [5] proposed multilinear models in facial expression tracking and transferring. They estimate a model from a data set of three-dimensional face scans that vary in expression, viseme, and identity. This multilinear model decouples the variation into several modes (e.g., identity, expression, viseme) and encodes them consistently. They estimate the model from two geometric data sets: one with 15 identities each performing the same 10 expressions, and the other with 16 identities, each with five visemes in five expressions ($16 \times 5 \times 5$). To construct their multilinear models from data sets with missing data, they propose to fill the missing combinations (e.g., of identity, viseme, and expression) by an expectation-maximization approach.

All of these databases contain face data with different identities, and some even with different expressions. However, they may still be inadequate for facial expression parameterization or *rig*. Due to their excellent performance, facial rigs based on blendshape models are particularly popular in expressing facial behaviors. Ekman's Facial Action Coding System [7] helps decompose facial behavior

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

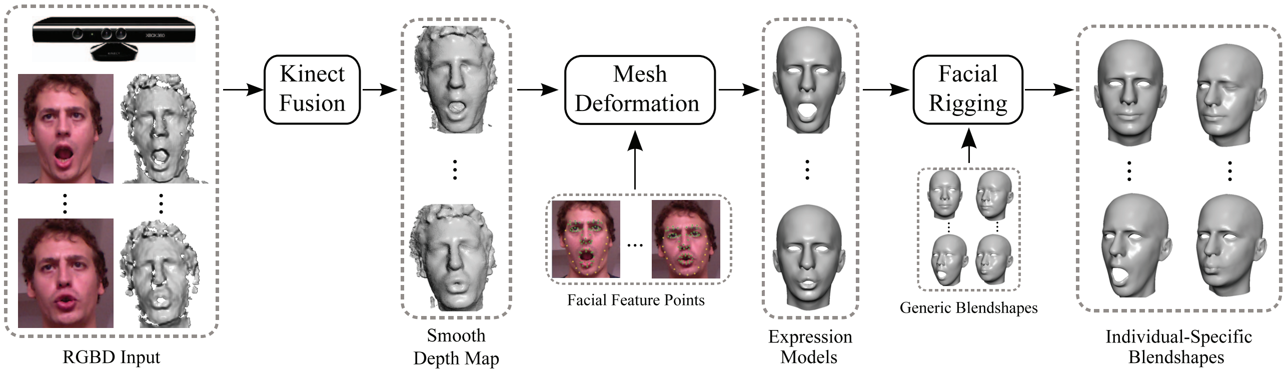


Fig. 1. The generation process of the individual-specific expression blendshapes for one person.

into 46 basic action units. Li et al. [9] developed a method for generating facial blendshape rigs from a set of example poses. Our database contains the full set of 46 expression blendshapes which comprise the linear blendshape model for each person.

2.2 Face Manipulation Applications

Face manipulation is a convenient tool for artists and animators to create new facial images or animations from existing materials, and hence of great research interest in computer animation. In the 2D domain, Leyvand et al. [10] enhance the *attractiveness* of human faces in frontal photographs. Given a face image as the target, Bitouk et al. [11] find a most similar face in a large 2D face database and use it to replace the face in the target image to conceal the identity of the face. Joshi et al. [12] improve the quality of personal photos by using a person's favorite photographs as examples.

Recently, 3D face models have become increasingly popular in complex face manipulation. Blanz et al. [13] reanimate the face in a still image or video by transferring mouth movements and expression based on their 3D morphable model [4] and a common expression representation. They also exchange the face between images across large differences in viewpoint and illumination [14]. Given a photo of person A, Kemelmacher-Shlizerman et al. [2] seek a photo of person B with similar pose and expression from a large database of images on the Internet. Yang et al. [1] derive an expression flow and alignment flow from the 3D morphable model between source and target photos, capable of transferring face components between the images naturally and seamlessly. Kemelmacher-Shilizerman et al. [15] generate face animations from large image collections. Dale et al. [16] replace the face in a portrait video sequence through a 3D multilinear model [5]. We demonstrate that FaceWarehouse can provide a rich collection of expression data to facilitate and improve these applications.

3 FACEWAREHOUSE

In this section, we describe our pipeline for constructing FaceWarehouse and the techniques involved. We use Microsoft's Kinect System to capture the geometry and texture information of various expressions of each subject. We register the frames from different views of the same expression to generate a smooth, low-noise depth map. The depth maps, together with the RGB images, are used to guide the deformation of a template mesh to generate the expression meshes. Once we obtain all the expression meshes of a single subject, we generate her individual-specific expression blendshapes. Fig. 1 shows the entire pipeline of processing one subject. Finally, the expression blendshapes from all subjects constitute our face database. As we have all the face models in a consistent topology, we can build a bilinear face model with two attributes: identity and expression.

3.1 Data Capture

A Kinect system is used as our only capturing device, capable of producing 640×480 2D images and depth maps at 30 frames per second. With the low-cost small-size acquisition device, we can capture a person's face in a nonintrusive way, as the person being captured is not required to wear any physical markers or be staged in a controlled environment.

For each person, we capture 20 different expressions: the neutral expression and 19 other specific expressions. These expressions are chosen to be common facial motions that vary widely among different individuals. They contain combinations of the facial muscle group action units in FACS and some asymmetric patterns. Specifically, they are mouth stretch, smile, brow lower, brow raiser, anger, jaw left, jaw right, jaw forward, mouth left, mouth right, dimpler, chin raiser, lip puckerer, lip funneler, sadness, lip roll, grin, cheek blowing, and eyes closed. Face meshes are created by an artist to serve as a guide for each specific expression: G_0 for the neutral expression, and $G_1, G_2, \ldots, G_{19}$ for the other expressions. The guide models are shown to each subject sequentially, and the person is asked to imitate each expression and rotate her head within a small angle range while keeping the expression fixed, assisted by our staff when necessary.

The advantages of the chosen Kinect system, such as low-cost and mobility, come at the price of low quality and severe noise in the captured data. To reduce noise, we aggregate multiple scans by using the Kinect Fusion algorithm [17] to register the 3D information of a specific expression of a person from different views (captured in the head rotation sequence), and generate a smooth, low-noise depth map.

During the capturing process, the person to be captured is required to position his (or her) face inside a rectangle region shown on the screen. The Kinect Fusion algorithm is applied in this rectangle region. When the person is rotating

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

his (or her) head, the algorithm automatically aligns and fuses all the depth data streamed from the camera into a single global implicit surface model, which is then ray traced to generate a smooth depth map for an arbitrary frame. In our experiment, the noise of raw depth data from Kinect has a range up to 4 mm (1.5 mm on average), and the smooth depth map generated by Kinect Fusion has an maximum error of 6 mm (1.8 mm on average). In total, the scanned data have a maximum error of 10 mm (3.3 mm on average). In practice, we found that the data can well capture the 3D geometry of the person's facial features (e.g., the mouth lips, the nose, and the eye sockets), but may lose some high-frequency details such as wrinkles.

3.2 Expression Mesh and Individual-Specific Blendshape Generation

From smooth depth maps and corresponding color images, we generate the associated expression meshes. For each expression data, we first use the Active Shape Model (ASM) [18] to locate 74 feature points on the color image, including the face contour, eye corners, brow boundary, mouth boundary, nose contour, and tip. The automatically detected locations may not be accurate in all cases, especially for those expressions with relatively large shape changes (e.g., mouth-open and smile). In the worst case, there are about 20 feature points that do not match the facial features in images. We thus require a small amount of user interaction to refine the positions of these points—the user interaction is as simple as drag-and-dropping the feature points on the image. In our experiment, this takes us about 1 minute for each image on average. Our mesh fitting algorithm is also robust to small errors in feature labeling as feature points are just one item in the mesh fitting energy (see (5)).

The 74 feature points are divided into two categories: the m_i internal feature points (i.e., features on eyes, brows, nose and mouth, cf. the green points in Fig. 3) located inside the face region, and the m_c contour feature points (the yellow points in Fig. 3). Given the correspondence between the color image and the depth map, we can easily get the corresponding 3D positions from the depth map for internal feature points. We classify all contour feature points in the image as 2D.

3.2.1 Neutral Expression

We first generate the face mesh for the neutral expression by using a two-step approach. Blanz and Vetter's morphable model is automatically fitted to produce an initial matching mesh. Then, a mesh deformation algorithm is employed to refine this mesh for better matching between the depth map and the feature points.

Blanz and Vetter's morphable model performs principal component analysis (PCA) on 200 neutral face models. Any face can be approximated as a linear combination of the average face and l leading PCA vectors: $V = \bar{F} + \sum_{i=1}^{l} \alpha_i F_i$, where $\bar{F}$ is the average face, and F_i is the ith PCA vector. Our goal is to compute the coefficients α_i to get the closest mesh in the PCA space. The energy to be minimized for feature point matching is defined as

$$E_{fea} = \sum_{j=1}^{m_i} \left\| \mathbf{v}_{i_j} - \mathbf{c}_j \right\|^2 + \sum_{k=1}^{m_c} \left\| \mathbf{M}_{proj} \mathbf{v}_{c_k} - \mathbf{s}_k \right\|^2. \quad (1)$$

The first term corresponds to internal feature matching. c_j is the 3D position of the jth feature point, while $\mathbf{v}_{i_j}$ is its corresponding vertex on the mesh V. The indices for these internal feature points on the mesh are simply marked on the average face in our implementation. The second term is for contour feature matching. $\mathbf{s}_k$ is a 2D feature point on the color image, $\mathbf{v}_{c_k}$ is its corresponding 3D feature vertex on the mesh V, and $\mathbf{M}_{proj}$ is the projection matrix of the camera. We use the method described in [1] to determine the indices of the contour feature points on the mesh V: We first project the face region of V to the image to get the 2D face mesh. Then, we find its convex hull to get the points along the contour of the mesh. Among these points, we find the nearest one for each contour feature on the image and assign it as the corresponding feature point on the mesh.

The energy term for matching the depth map is defined as

$$E_{pos} = \sum_{j=1}^{n_d} \left\| \mathbf{v}_{d_j} - \mathbf{p}_j \right\|^2, \quad (2)$$

where $\mathbf{v}_{d_j}$ is a mesh vertex, $\mathbf{p}_j$ is the closest point to $\mathbf{v}_{d_j}$ in the depth map, and n_d is the number of the mesh vertices that have valid correspondences in the depth map. Note that not all mesh vertices are accounted for in this energy term as some vertices are occluded and cannot get valid positions from the depth map.

According to [4], another energy term is necessary to regularize the PCA coefficients α_i, based on the estimated probability distribution of a shape defined by α_i,

$$p(\alpha) \sim exp\left[-\frac{1}{2} \sum (\alpha_i / \sigma_i)^2 \right]. \quad (3)$$

where σ_i^2 is the eigenvalues of the face covariance matrix from PCA. Let $\Lambda = diag(1/\sigma_1^2, 1/\sigma_2^2, \ldots 1/\sigma_L^2)$; then, the Tikhonov regularization energy term is defined as

$$E_{coef} = \frac{1}{2} \alpha^T \Lambda \alpha. \quad (4)$$

Putting the three energy terms together, the total energy is defined as

$$E_1 = \omega_1 E_{fea} + \omega_2 E_{pos} + \omega_3 E_{coef}, \quad (5)$$

where ω_1, ω_2, and ω_3 balance the different energy terms. In our experiments, we set $\omega_1 = 2, \omega_2 = 0.5$. We found that $\omega_3 \in [0.6, 2]$ can achieve a good balance between fitting accuracy and face shape fidelity, and chose $\omega_3 = 1$ for our database construction. The energy can be minimized via a sequence of least squares optimizations. The least-squares step is iterated five to eight times in our construction. Note that between consecutive iterations, the mesh vertices corresponding to contour feature points in (1) and each mesh vertex's closest point in (2) need to be updated.

After an initial mesh is computed, it is refined by a Laplacian-based mesh deformation algorithm [19]. Similar to the optimization process described above, the deformation algorithm tries to minimize E_{fea} and E_{pos} to match the

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

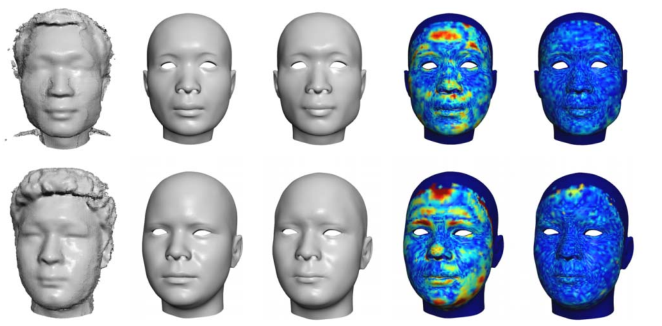


Fig. 2. Two examples of neutral mesh fitting. From left to right: input depth map; initial mesh produced by the morphable model; refined mesh using mesh deformation; error map of the initial mesh; error map of the refined mesh. The root-mean-square (RMS) errors for the first example are 1.19 mm (initial) and 0.39 mm (refined), and for the second example are 1.49 mm (initial) and 0.41 mm (refined).

feature points and the depth map. A Laplacian energy is used as the regularization term,

$$E_{lap} = \sum_{i=1}^{n} \left\| L\mathbf{v}_i - \frac{\delta_i}{|L\mathbf{v}_i|} L\mathbf{v}_i \right\|^2, \tag{6}$$

where L is the discrete Laplacian operator based on the cotangent form introduced in [20], δ_i is the magnitude of the original Laplacian coordinate before deformation, and n is the vertex number of the mesh. In (6), we follow [19] to constrain the target Laplacian coordinate to the direction of the Laplacian coordinate computed from the current mesh while keeping its original length. As pointed out in [21], such a nonlinear formulation of the target Laplacian coordinate can achieve results superior to linear approximations for large-scale deformations.

The mesh deformation energy is then computed as

$$E_2 = \omega_1' E_{fea} + \omega_2' E_{pos} + \omega_3' E_{lap}. \tag{7}$$

In our construction, we chose $\omega_1' = 0.5$, $\omega_2' = \omega_3' = 1$. This energy can be minimized using the inexact Gauss-Newton method as described in [19].

Mesh deformation helps fine-tune the initial guess. Fig. 2 shows two examples of neutral mesh generation. From the figure, we can see that the refinement process in the second step drastically reduces the mismatch, resulting in a better matching face mesh.

3.2.2 Other Expressions

Once we obtain the face mesh S_0 for the neutral expression, we proceed to compute the face meshes $S_1, S_2, \ldots, S_{19}$ for the other 19 expressions. We first use the deformation transfer algorithm described in [22] to generate the initial meshes for these expressions so that the deformation from S_0 to $S_i (i = 1 \ldots 19)$ mimics the deformation from the guide model G_0 to G_i as closely as possible. The same mesh deformation algorithm described above is then used to refine these initial meshes.

The mesh deformation algorithm uses all facial feature points on the color images as additional position constraints. We found in our experiments that these constraints not only greatly reduce the matching errors between the image feature points and their corresponding mesh vertices but also help avoid local minima in the deformation process and improve the matching between the mesh and the depth map, as demonstrated in Fig. 3.

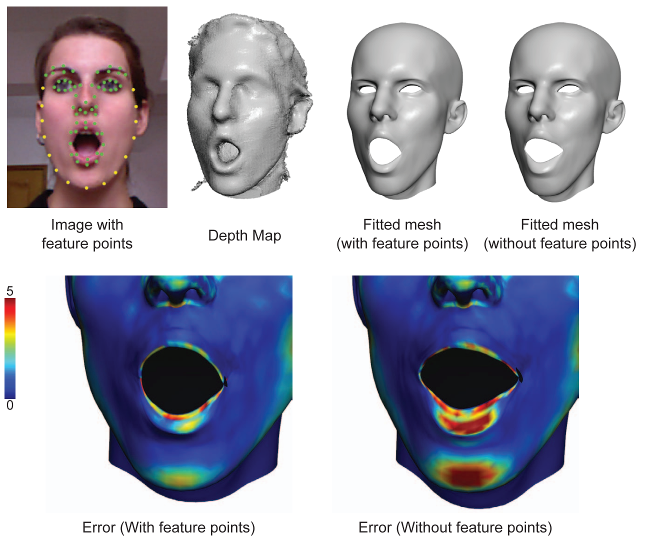
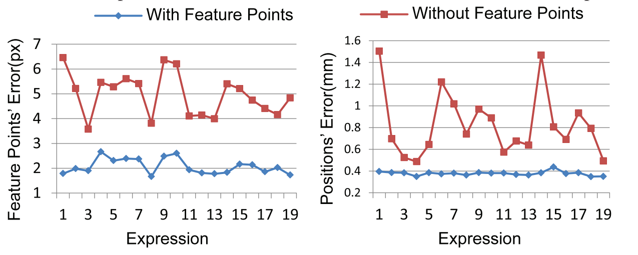


Image with feature points
Depth Map
Fitted mesh (with feature points)
Fitted mesh (without feature points)
5
0
Error (With feature points)
Error (Without feature points)

(a) *Without feature point constraints, the deformed mesh may not match the captured data well, such as the mouth in this example.*



(b) *The lack of feature point constraints leads to results trapped in local minima in the deformation process and causes greater matching error, for both the feature points of the color image and the depth map.*

Fig. 3. The effect of feature point constraints in mesh deformation.

3.2.3 Individual-Specific Expression Blendshapes

From the expression meshes generated for each person, we can use the example-based facial rigging algorithm proposed by Li et al. [9] to build a linear blendshape model representing the facial expression space of this person. The result is a neutral face plus 46 FACS blendshapes $\mathbf{B} = \{B_0, B_1, \ldots, B_{46}\}$ for each person, capable of replicating most human expressions through linear interpolation of the blendshapes. In other words, an expression H of the person can be expressed by a linear combination of the blendshapes: $H = B_0 + \sum_{i=1}^{46} \alpha_i (B_i - B_0)$, where α_i is the weight measuring how much the neutral face B_0 is deformed toward B_i. The rigging algorithm begins with a generic blendshape model $\mathbf{A} = \{A_0, A_1, \ldots, A_{46}\}$ and employs an optimization procedure to minimize the difference between each expression mesh S_j and the linear combination of B_i with the known weights for expression j as well as the difference between the relative deformation from B_0 to B_i and that from A_0 to A_i. See [9] for details of the algorithm.

The generic blendshape model $\mathbf{A}$ is generated by an artist based on the Facial Action Coding System, the common standard to categorize the physical expression of emotions.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

The linear interpolation of these 47 expressions (including neutral expression) can represent most facial expressions very well. We do not capture and reconstruct these 47 expressions directly because among these 47 expressions there are many expressions that are either tiny (e.g., eye open) or asymmetric (e.g., left brow down), which are difficult for our capturing subjects to act out. The algorithm of [9] is designed to generate the 47 blendshapes that best reproduce the input 20 expressions while preserving the controller semantics of the generic blendshape model.

3.3 Bilinear Face Model

We obtained the facial geometry of 150 persons and each contains the same 47 facial expressions (one neutral and 46 others) by using the procedure described in the previous section. All these face meshes share the same topology and thus have the same number of vertices. Similar to [5], we can assemble the data set into a rank-3 (3-mode) *data tensor* T (11K vertices $\times 150$ identities $\times 47$ expressions). The data tensor is arranged in an obvious fashion, so that each slice with varying second factor and fixed third factor contains face vectors with the same expression (for different identities), and each slice with varying third factor and fixed second factor contains the same identity (with different expressions).

We use the N-mode singular value decomposition to decompose the tensor. As most visual applications only need to synthesize the entire face, we perform the decomposition without factoring along the vertex mode (mode-1). The N-mode SVD process is represented as

$$T \times_2 \mathbf{U}_{id}^T \times_3 \mathbf{U}_{exp}^T = C, \tag{8}$$

where T is the data tensor and C is called the *core tensor*. $\mathbf{U}_{id}$ and $\mathbf{U}_{exp}$ are orthonormal transform matrices, which contain the left singular vectors of the second mode (identity) space and second mode (expression) space, respectively. Third-mode SVD helps "rotate" the data tensor and sort the variance of C in decreasing order for each mode. This allows us to truncate the insignificant components of C and get a reduced model of the data set to approximate the original data tensor as

$$T \simeq C_r \times_2 \check{\mathbf{U}}_{id} \times_3 \check{\mathbf{U}}_{exp}, \tag{9}$$

where C_r is the reduced core tensor produced by keeping the top-left corner of the original core tensor. $\check{\mathbf{U}}_{id}$ and $\check{\mathbf{U}}_{exp}$ are the truncated matrices from $\mathbf{U}_{id}$ and $\mathbf{U}_{exp}$ by removing the trailing columns.

We call C_r the bilinear face model for FaceWarehouse. With C_r, any facial expression of any person can be approximated by the tensor contraction

$$V = C_r \times_2 \mathbf{w}_{id}^T \times_3 \mathbf{w}_{exp}^T, \tag{10}$$

where $\mathbf{w}_{id}$ and $\mathbf{w}_{exp}$ are the column vectors of identity weights and expression weights, respectively.

Fig. 4 shows an example of fitting a face mesh using different numbers of components in the core tensor. We found that choosing 50 knobs for identity and 25 knobs for expression provides satisfactory approximation results. Therefore, we use this reduced core tensor ($11\mathrm{K} \times 50 \times 25$) in the following applications.

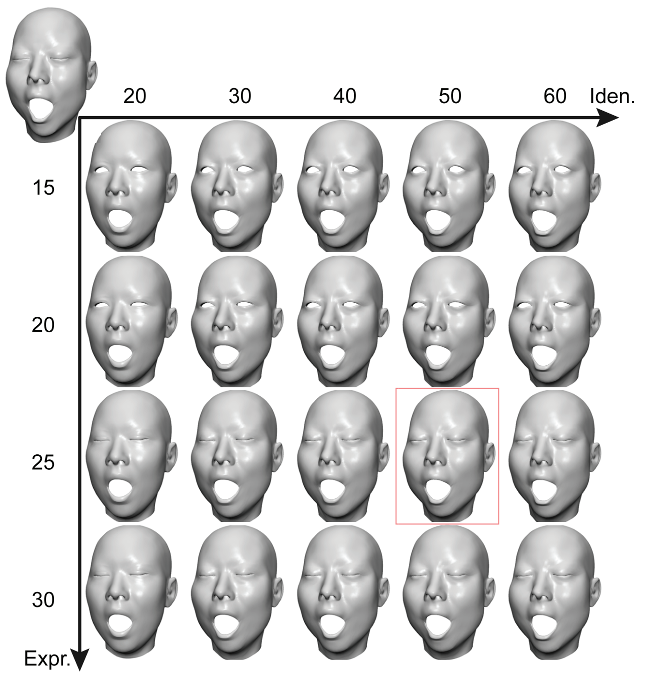


Fig. 4. Fitting a facial expression mesh with our bilinear model with different numbers of components. Top left is the input mesh, and the following shows the fitting results using different numbers of components in the identity attribute and expression attribute.

4 Applications

FaceWarehouse can be employed in various visual computing applications. In this section, we show four example applications: facial image manipulation, face component transfer, real-time performance-based facial image animation, and facial animation retargeting from video to image. Refer to the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2013.249, for the video demo.

4.1 Facial Image Manipulation

In this application, the user can manipulate facial attributes, such as the length of face, the size of mouth, the height of nose and ethnicity, directly in the single input face image (see Fig. 6 for an example). As we only have the two attributes of identity and expression in FaceWarehouse, we first learn a linear regression model that maps a set of user-specified facial attributes to the identity attribute in the bilinear face model. We then compute the identity and expression weights in our bilinear face model for the input face image. The changes to the user-defined attributes are mapped to the identity weights via the linear regression model, and then, a new 3D face mesh is reconstructed based on these weights. This new 3D face mesh is finally rendered with color textures from the input image to generate a new face image with the relevant features changed.

4.1.1 Facial Feature Analysis

To analyze facial attributes used in natural language (e.g., the width of mouth, the length of face), we use the algorithm of multi-variate linear regression [23] to map these attributes to the identity attribute in our bilinear face model. For every person captured in FaceWarehouse, we

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

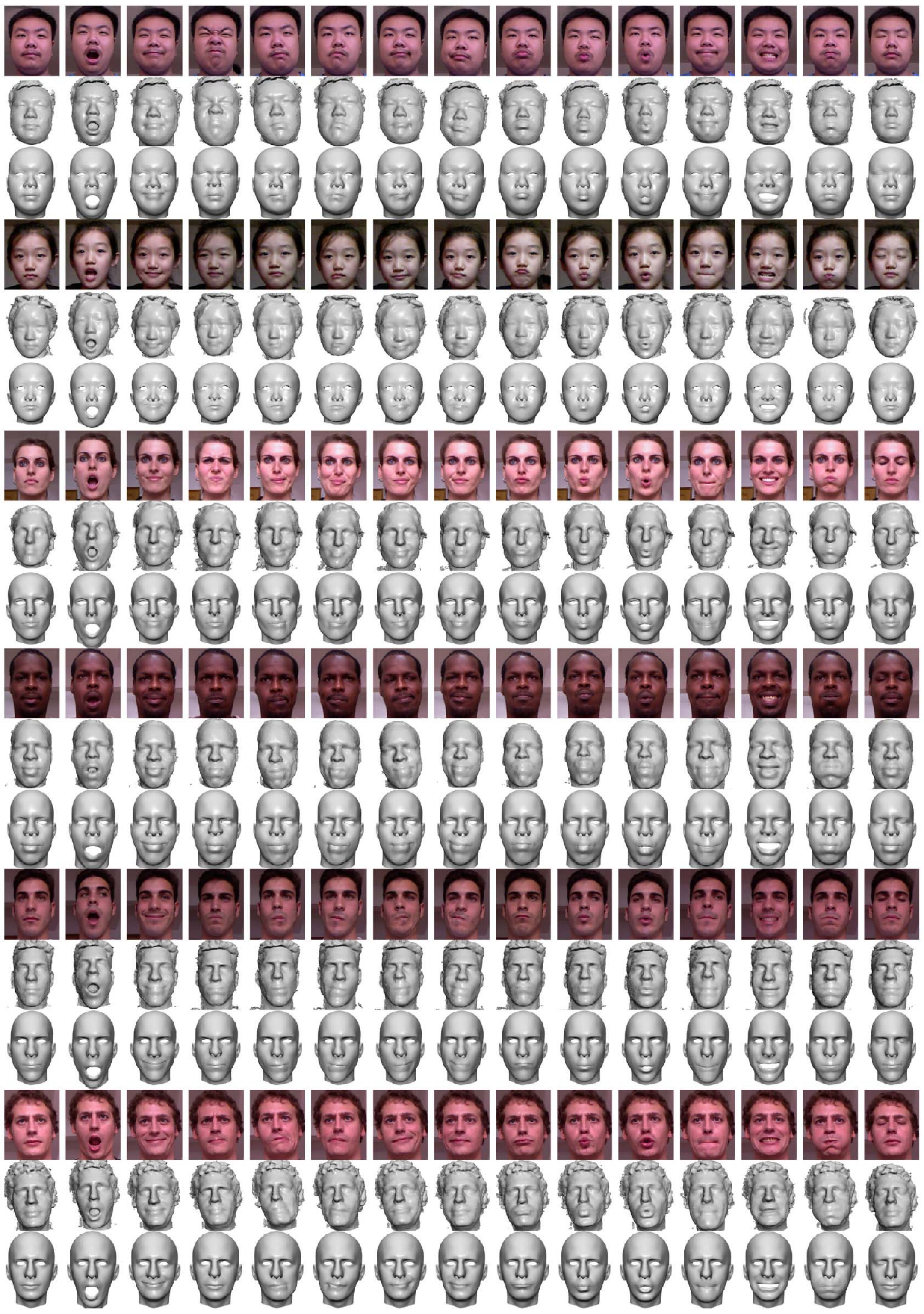


Fig. 5. Several expressions of five persons captured in FaceWarehouse. For each expression, we showed the color image, the depth map, and the reconstructed mesh.

have his (or her) identity weights (a k-D vector $\mathbf{w}_{id}$) and l user-specified attributes $\{f_1, f_2, \ldots, f_l\}$ which are calculated from the face geometry of this person. We need to construct a $k \times (l+1)$ matrix $\mathbf{M}_{fea}$ mapping these user-specified attributes to the identity weights,

$$\mathbf{M}_{fea}[f_1, \ldots, f_l, 1]^T = \mathbf{w}_{id}. \tag{11}$$

We assemble the vectors of identity weights and the vectors of user-defined attributes of all 150 persons into two matrices, i.e., $\mathbf{W}_{id}(k \times 150)$ and $\mathbf{F}$ $((l+1) \times 150)$, respectively. The mapping matrix $\mathbf{M}_{fea}$ can be solved as

$$\mathbf{M}_{fea} = \mathbf{W}_{id}\mathbf{F}^{+}, \tag{12}$$

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

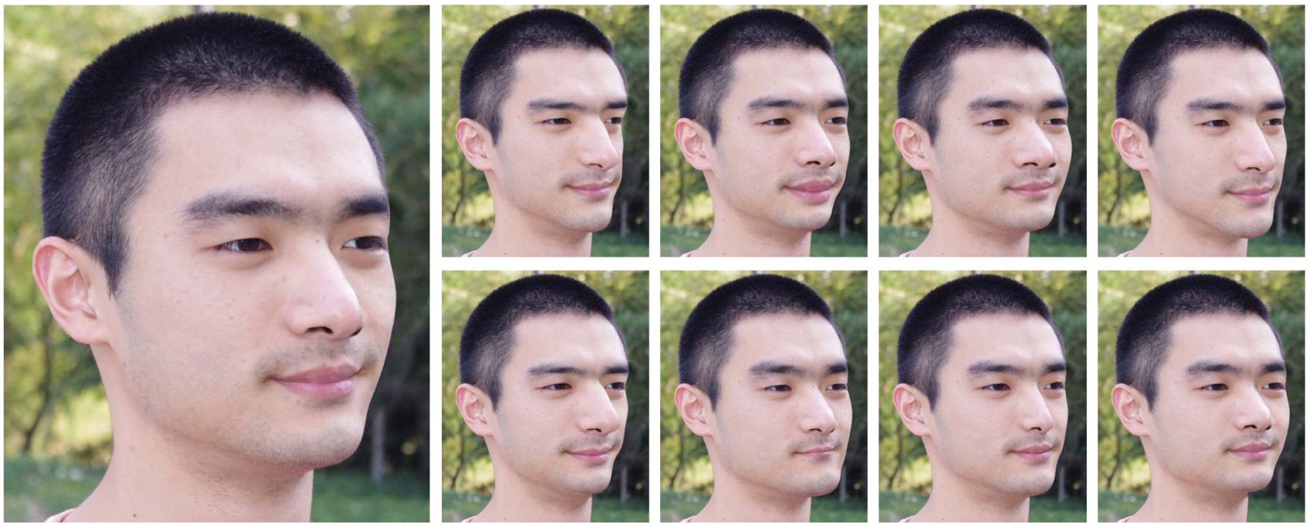


Fig. 6. Facial image manipulation. We can edit the facial attributes in the still image. Left: the original image. Top row, from left to right: more Caucasian-like, bigger mouth, wider nose, longer face. Bottom row, from left to right: higher nose, smaller mouth, narrower nose, shorter face.

where $\mathbf{F}^+$ is the left pseudoinverse of $\mathbf{F}$, i.e., $\mathbf{F}^+ = \mathbf{F}^T(\mathbf{F}\mathbf{F}^T)^{-1}$.

With $\mathbf{M}_{fea}$, we can directly map the changes of the user-specified attributes $\Delta\mathbf{f}$ to the changes of identity weights $\Delta\mathbf{w}_{id}$, i.e., $\Delta\mathbf{w}_{id} = \mathbf{M}_{fea}\Delta\mathbf{f}$. By adding $\Delta\mathbf{w}_{id}$ to the identity weights of the input face image, we can generate a new face with related attributes changed appropriately.

4.1.2 Fitting 3D Face Mesh to Image

To calculate a 3D face mesh using our bilinear model that can match the face image well, we first localize a set of facial feature points in the image in the same way as we suggest in Section 3.2. Then, we estimate the rigid transformation of the face model as well as the identity and expression weights in the bilinear face model to minimize the matching error between the feature points on the image and the face mesh.

Following previous work [5], we assume that the camera projection is weakly perspective. Every mesh vertex $\mathbf{v}_k$ is projected to the image space as

$$\mathbf{p}_k = s\mathbf{R}\mathbf{v}_k + \mathbf{t}, \tag{13}$$

where the rigid transformation consists of a scaling factor s, a 3D rotation matrix $\mathbf{R}$, and a translation vector $\mathbf{t}$. The mesh vertex position $\mathbf{v}_k$ can be computed from the bilinear face model according to (10).

The matching error of the feature points on the image and the mesh is defined as

$$E_k = \frac{1}{2} \cdot \left\| s\mathbf{R}\left(C_r \times_2 \mathbf{w}_{id}^T \times_3 \mathbf{w}_{exp}^T\right)^{(k)} + \mathbf{t} - \mathbf{s}^{(k)} \right\|^2, \tag{14}$$

where $\mathbf{s}^{(k)}$ is the feature point positions on the image.

This energy can be easily minimized using the coordinate-descent method as described in [5].

Compared with previous 3D face databases, our database provides a more diverse spaces of identity and expression for face-related applications. For example, the bilinear model presented in [5] contains the data of 15 persons (with 10 symmetric facial expressions). As their data are not publicly available, we randomly choose the data of 15 identities from our database, construct a bilinear model from these data, and use it in the image manipulation application. Fig. 7 compares the results using different number of identities. As shown, the fitted mesh using 15 identities does not match the input image very well, resulting in unnatural manipulation results (see Figs. 7d and 7e) when the mesh is changed. Note that choosing another set of 15 persons from our database may improve the results in this example, but could also generate worse results for other images.

In Fig. 8, we further analyze the influence of different number of identities in facial image manipulation. As

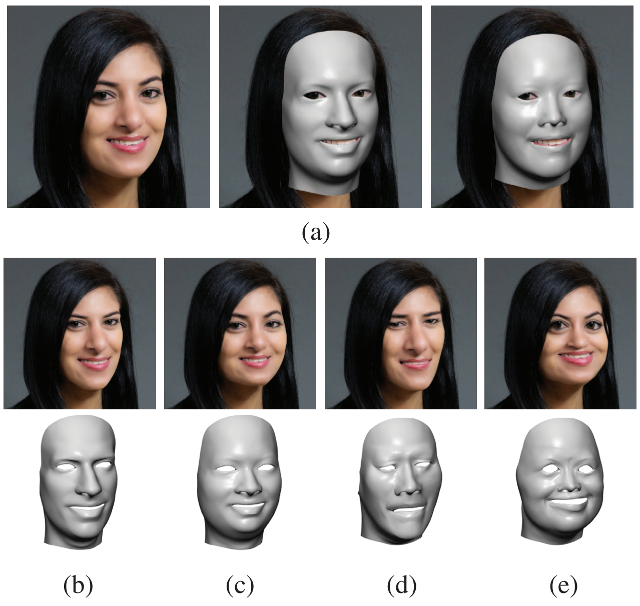


Fig. 7. Comparisons of different number of identities in facial image manipulation. (a) From left to right: the original image, the fitted mesh with 150 identities of 47 expressions (150×47), and the fitted mesh with 15 identities of 47 expressions (15×47). Manipulation results of longer face (b) and wider face (c) using the 150×47 data set. Manipulation results of longer face (d) and wider face (e) using the 15×47 data set. The corresponding face meshes for the manipulation results are shown in the bottom row.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

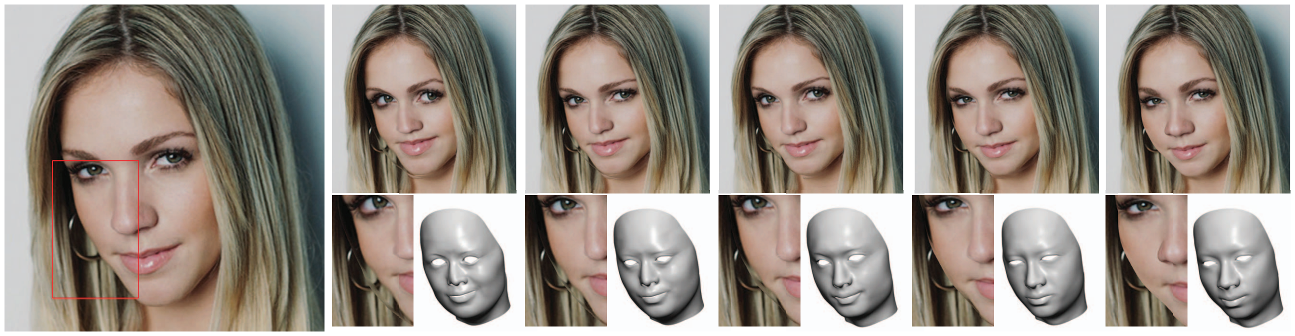


Fig. 8. Making the face wider using the bilinear models of different number of identities. From left to right: original image with highlighted region, manipulation results with 15, 30, 60, 90, and 150 identities. The highlighted regions and face meshes are shown in the second row.

shown, in this example, the face in the image is made wider. Increasing the number of identities can improve the manipulation results especially in regions around the face contour. However, compared with the result with 150 identities, some artifacts can still be observed around the face contour in the result with 90 identities.

Note that in all applications, we use the full-resolution mesh (11K vertices) in fitting the mesh to the image/video, but only display the frontal region of the mesh along with the image/video for the purpose of better visualization.

4.2 Face Component Transfer

This application performs face component copy-and-paste to modify the expression in a facial image. It takes two images of the same person as input: one is the target photo that contains an undesirable expression and the other one is the reference image that contains the desired expression, such as smiling. As modifying expression causes global changes in one's face, if we directly copy the local component from the reference image and paste/blend it to the target, the transferred component may not be compatible with the face contour or other components in the target image. A better approach proposed by Yang et al. [1] is to use 3D face models to guide the component transfer process. We follow this approach and use our bilinear face model to synthesize 3D face models matching the input images.

As the two input images represent the same person, their identity weights $\mathbf{w}_{id}^T$ should be the same. We, therefore, need to compute the unified identity weights $\mathbf{w}_{id}^T$ and the expression weights ($\mathbf{w}_{exp-1}^T$ and $\mathbf{w}_{exp-2}^T$) for the two images. This can be done by minimizing the following energy:

$$E_k^{joint} = \frac{1}{2}\sum_{j=1}^{2} \left\| s_j \mathbf{R}_j \left(C_r \times_2 \mathbf{w}_{id}^T \times_3 \mathbf{w}_{exp-j}^T \right)^{(k)} + \mathbf{t}_j - \mathbf{s}_j^{(k)} \right\|^2. \tag{15}$$

We first use the method described in the last section to compute an initial estimation of the identity and expression weights for each image separately. Then, we fix $\mathbf{w}_{exp-1}$ and $\mathbf{w}_{exp-2}$ and compute the unified identity weights $\mathbf{w}_{id}^T$ by minimizing (15). Next, $\mathbf{w}_{exp-1}$ and $\mathbf{w}_{exp-2}$ are solved again separately with $\mathbf{w}_{id}^T$ fixed. The latter two steps are performed iteratively until the fitting results converge. In our experiments, three iterations produce satisfactory results.

The two fitted face meshes can be reconstructed as

$$\begin{aligned} V_1 &= C_r \times_2 \mathbf{w}_{id}^T \times_3 \mathbf{w}_{exp-1}^T, \\ V_2 &= C_r \times_2 \mathbf{w}_{id}^T \times_3 \mathbf{w}_{exp-2}^T. \end{aligned} \tag{16}$$

Following [1], we use the two face meshes to calculate a 2D expression flow in the target image, which warps the target face to match the desired expression. A 2D alignment flow is also calculated from these two meshes to warp the reference face to an appropriate size and position for transferring. Finally, we select a crop region from the warped reference image and blend it to the warped target

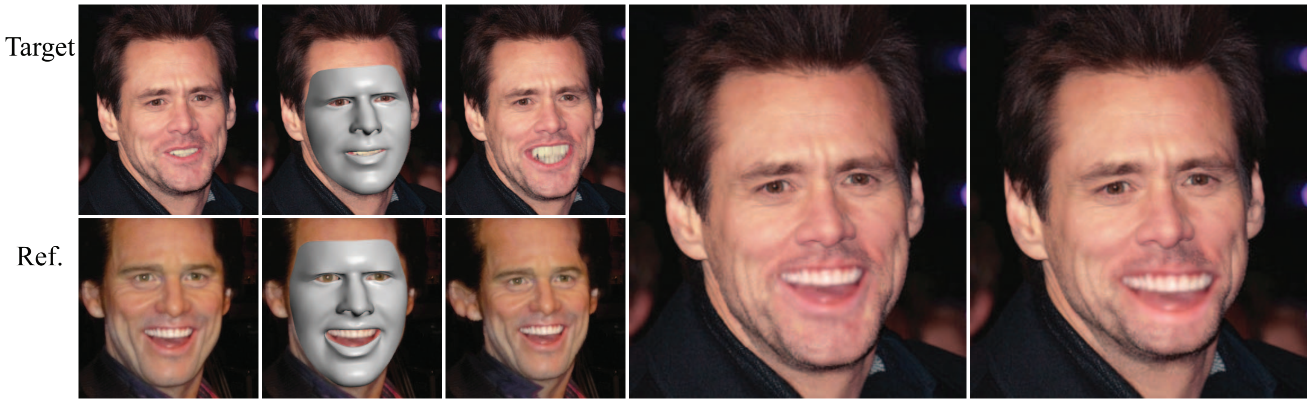


Fig. 9. Face component transfer. First column: two input images of the same person with different expressions. Second column: fitted meshes. Third column: target image warped by the expression flow and the reference image warped by the alignment flow. Fourth column: the transferred result by our method. Last column: the result produced by the 2D copy-and-paste method.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

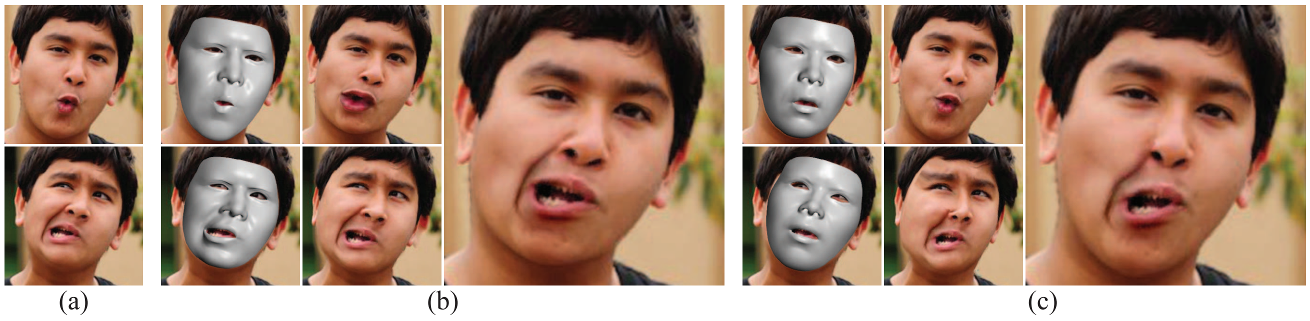


Fig. 10. Comparisons of different bilinear models in face component transfer. (a) Two input images of the same person with different expressions. (b) Fitted meshes and transfer results using our bilinear model constructed from the 150×47 data set. (c) Fitted meshes and transfer results using the bilinear model constructed from the 15×10 data set.

image to generate the transferred result. Fig. 9 shows one mouth-open example; our method produces a much more realistic result than the 2D copy-and-paste method.

In Fig. 10, we compare the face component transfer results using different bilinear models. In this example, the reference face image contains a very asymmetric expression. The bilinear model constructed from 15 persons with 10 symmetric expressions as in [5] fails to reconstruct a proper 3D face mesh for the asymmetric expression, resulting in an unnatural transfer result. Note that in this application, the number of identities does not affect the transfer results much, i.e., 150×10 gives a similar result to 15×10. The reason is that the algorithm does not alter the fitted meshes, but directly warps the images according to the computed expression flows and blends the warped images in the desired region. This can effectively hide the errors from an inaccurate fitted mesh caused by insufficient identities.

4.3 Real-Time Performance-Based Facial Image Animation

In this application, a still face image is animated in real time by the facial performance of an arbitrary user (see Fig. 11). Again, we first use the algorithm described in Section 4.1 to compute the identity and expression weights to produce a 3D face mesh matching the feature points in the input image. We then generate the individual-specific expression blendshapes for this face mesh using the bilinear face model. Finally, we implement the Kinect-based facial animation system to capture the user's facial expressions expressed as blendshape coefficients, which are then transferred to the individual face model fitted for the image.

Assuming that the identity weights computed from the face image are $\mathbf{w}_{id}$, we can construct the expression blendshapes for the person of identity $\mathbf{w}_{id}$ as follows:

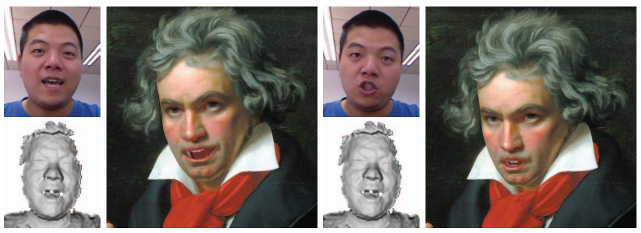


Fig. 11. Using a Kinect camera to track a user's facial expressions, which are then transferred to a still facial image, all in real time. This application allows a user to create an image avatar from a single facial image of another person.

$$B_i = C_r \times_2 \mathbf{w}_{id} \times_3 (\check{\mathbf{U}}_{exp}\mathbf{d}_i), \quad 0 \le i \le 47, \tag{17}$$

where $\check{\mathbf{U}}_{exp}$ is the truncated transform matrix for the expression mode as described in Section 3.3, and $\mathbf{d}_i$ is the expression weight vector with value 1 for the ith element and 0 for other elements.

The generated expression blendshapes, which we call *image blendshapes*, can then be used to generate new expressions of the same identity by setting different coefficients β_i for these blendshapes: $V = B_0 + \sum_{i=1}^{46} \beta_i(B_i - B_0)$.

We then use a real-time performance-based facial animation system [24] to capture the dynamic expressions of an arbitrary user, who has another set of expression blendshapes $(\mathbf{U} = \{U_0, U_1, \ldots, U_{46}\})$ constructed by the system during preprocessing. The system is able to track the rigid transformation of the user's head and the facial expressions expressed in the format of blendshapes coefficients β_i, which are then easily transferred to image blendshapes $\mathbf{B}$ to synthesize facial animations that mimic the user's performance.

To render the image animations in a realistic manner, the hair and teeth need to be processed in a proper way. We use a single-view hair modeling technique [3] to reconstruct a strand-based 3D hair model, which is then transformed together with the face mesh and rendered into the image. The teeth are handled using the algorithm described in [13]. Starting from a generic teeth model, we deform it to match the 3D face model. During animation, the motion of the teeth is easy to simulate: the upper jaw teeth are connected and moved with the upper part of the face, while the lower jaw teeth are connected and moved with the tip of the chin.

In Fig. 12, we compare the effects of different bilinear models in this application. The bilinear model (150×10) constructed from 10 symmetric expressions fails to track the facial animation of some asymmetric expressions (the third row in Fig. 12c). On the other hand, with only 15 identities (15×47), the fitted mesh does not match the input video frames well in regions around the mouth and face contour, resulting in implausible artifacts (see Fig. 12c).

4.4 Facial Animation Retargeting from Video to Image

The final application takes a video clip containing a continuously changing face as input, extracts the coefficients of expression blendshapes in all frames, and retargets them to a still face image (see Fig. 13). It needs to estimate the face identity and construct the expression blendshapes

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

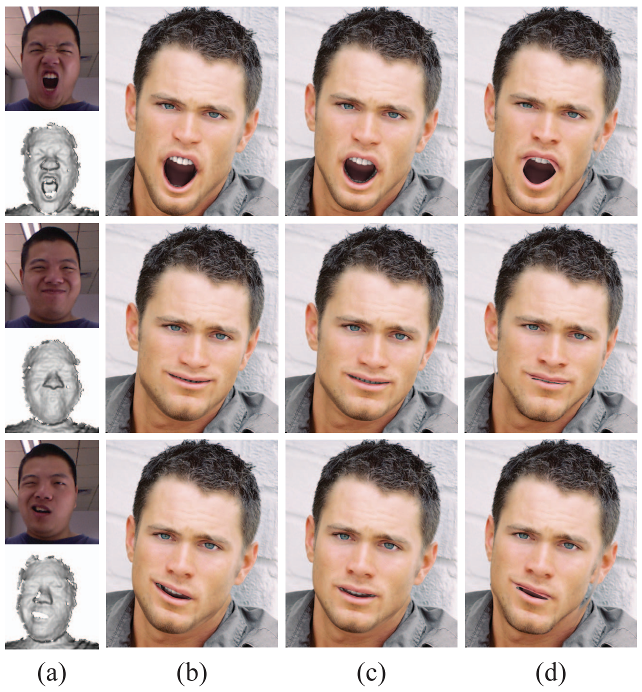


Fig. 12. Facial image animation comparison. (a) Input RGBD data. (b), (c), (d) animation results generated from 150×47, 150×10, and 15×47 data sets, respectively.

for both the face video and face image. The expression coefficients fitted for the face video are then transferred to the face image (see Fig. 13).

The face identity and expression of the image can be estimated using the algorithm described in Section 4.1. For the video, we need to fit a unified face identity for all frames using a simple extension of the joint fitting algorithm described in Section 4.2. We first locate the facial feature points on all frames using the Active Shape Model [18], and then extend the joint fitting algorithm to multiple frames to account for the matching errors in all frames. After the face identities of the video and image are fixed, we use the method described in Section 4.3 to construct their expression blendshapes.

Note that in this application (and the application in Section 4.3), we transfer the coefficients of the expression blendshapes, instead of the expression weights of the bilinear model as in [5]. Each face shape in the expression blendshapes corresponds to a meaningful expression. As pointed out in [25], transferring the coefficients of these shapes could generate more pleasant results than transferring the weights of basis shapes of the bilinear model

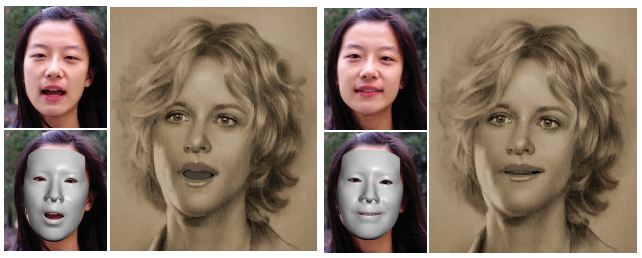


Fig. 13. Facial animation retargeting from video to image. Fitted face meshes and retargeting results for two frames.

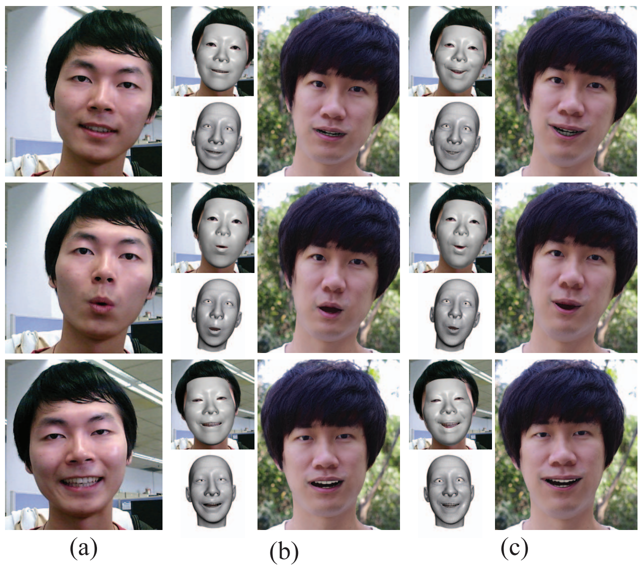


Fig. 14. Facial animation retargeting comparison. (a) Input video. (b) Fitted face meshes and retargeting results using expression blendshapes. (c) Fitted face meshes and retargeting results using the bilinear model.

which do not have any physical meaning. In Fig. 14, we compare the animation transfer results using blendshapes and the bilinear model. While both results are visually plausible, we can still notice that the blendshape results look more natural and better follow the facial expressions in the input video. Moreover, expression bendshapes are widely used in both game and film production. It is convenient for artists to create the FACS blendshape expressions for an arbitrary character, which can be used as 3D avatars in the retargeting applications [24].

5 Conclusion and Future Work

We introduce FaceWarehouse, a 3D facial expression database for visual computing applications. The database contains the facial geometry and texture of 150 subjects, each with 20 expressions. This raw data set is used to construct 47 expression blendshapes for each person, capable of representing most expressions of human faces. All these blendshapes are then assembled into a rank-3 tensor, which is decomposed to build a bilinear face model. This bilinear face model can be used to accurately estimate face identities and expressions for facial images and videos. We demonstrate its relevance in a wide range of applications, such as facial image manipulation, face component transfer, real-time performance-based facial image animation, and facial animation retargeting from video to image. We expect many other applications to benefit from FaceWarehouse in the future, such as face tracking in motion capture and expression recognition/analysis.

Due to the low precision in depth information provided by the current Kinect system, our face data do not contain detailed facial geometries such as wrinkles. In the future, it is possible to employ capturing techniques and more expensive equipment like multiple high-resolution cameras for high-quality facial geometry (e.g., [26], [27], [28]) to acquire data with finer details.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.

ACKNOWLEDGMENTS

The authors would like to thank the image owners for letting us use their work. This work was partially funded by NSFC (No. 60825201, No. 61003145, and No. 61272305) and the 973 program of China (No. 2009CB320801).

REFERENCES

[1] F. Yang, J. Wang, E. Shechtman, L. Bourdev, and D. Metaxas, "Expression Flow for 3d-Aware Face Component Transfer," *ACM Trans. Graphics,* vol. 30, no. 4, pp. 60:1-60:10, July 2011.

[2] I. Kemelmacher-Shlizerman, A. Sankar, E. Shechtman, and S.M. Seitz, "Being John Malkovich," *Proc. 11th European Conf. Computer Vision: Part I (ECCV '10),* pp. 341-353, 2010.

[3] M. Chai, L. Wang, Y. Weng, Y. Yu, B. Guo, and K. Zhou, "Single-View Hair Modeling for Portrait Manipulation," *ACM Trans. Graphics,* vol. 31, no. 4, pp. 116:1-116:8, July 2012.

[4] V. Blanz and T. Vetter, "A Morphable Model for the Synthesis of 3d Faces," *Proc. ACM SIGGRAPH,* pp. 187-194, 1999.

[5] D. Vlasic, M. Brand, H. Pfister, and J. Popović, "Face Transfer with Multilinear Models," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 426-433, July 2005.

[6] L. Yin, X. Wei, Y. Sun, J. Wang, and M.J. Rosato, "A 3d Facial Expression Database for Facial Behavior Research," *Proc. IEEE Intl. Conf. Face and Gesture Recognition,* pp. 211-216, 2006.

[7] P. Ekman and W. Friesen, *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press, 1978.

[8] R. Gross, "Face Databases," *Handbook of Face Recognition,* A.S. Li, ed., Springer, Feb. 2005.

[9] H. Li, T. Weise, and M. Pauly, "Example-Based Facial Rigging," *ACM Trans. Graphics,* vol. 29, no. 4, pp. 32:1-32:6, July 2010.

[10] T. Leyvand, D. Cohen-Or, G. Dror, and D. Lischinski, "Data-Driven Enhancement of Facial Attractiveness," *ACM Trans. Graphics,* vol. 27, no. 3, pp. 38:1-38:9, Aug. 2008.

[11] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S.K. Nayar, "Face Swapping: Automatically Replacing Faces in Photographs," *ACM Trans. Graphics,* vol. 27, no. 3, pp. 39:1-39:8, Aug. 2008.

[12] N. Joshi, W. Matusik, E.H. Adelson, and D.J. Kriegman, "Personal Photo Enhancement Using Example Images," *ACM Trans. Graphics,* vol. 29, no. 2, pp. 12:1-12:15, Apr. 2010.

[13] V. Blanz, C. Basso, T. Poggio, and T. Vetter, "Reanimating Faces in Images and Video," *Computer Graphics Forum,* vol. 22, no. 3, pp. 641-650, 2003.

[14] V. Blanz, K. Scherbaum, T. Vetter, and H.-P. Seidel, "Exchanging Faces in Images," *Computer Graphics Forum,* vol. 23, no. 3, pp. 669-676, 2004.

[15] I. Kemelmacher-Shlizerman, E. Shechtman, R. Garg, and S.M. Seitz, "Exploring Photobios," *ACM Trans. Graphics,* vol. 30, no. 4, pp. 61:1-61:10, July 2011.

[16] K. Dale, K. Sunkavalli, M.K. Johnson, D. Vlasic, W. Matusik, and H. Pfister, "Video Face Replacement," *ACM Trans. Graphics,* vol. 30, no. 6, pp. 130:1-130:10, Dec. 2011.

[17] R.A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A.J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-Time Dense Surface Mapping and Tracking," *Proc. 10th IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR '11),* pp. 127-136, 2011.

[18] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham, "Active Shape Models—Their Training and Application," *Computer Vision and Image Understanding,* vol. 61, no. 1, pp. 38-59, 1995.

[19] J. Huang, X. Shi, X. Liu, K. Zhou, L.-Y. Wei, S.-H. Teng, H. Bao, B. Guo, and H.-Y. Shum, "Subspace Gradient Domain Mesh Deformation," *ACM Trans. Graphics,* vol. 25, no. 3, pp. 1126-1134, July 2006.

[20] M. Desbrun, M. Meyer, P. Schroder, and A.H. Barr, "Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow," *Proc. ACM SIGGRAPH,* pp. 317-324, 1999.

[21] M. Botsch and O. Sorkine, "On Linear Variational Surface Deformation Methods," *IEEE Trans. Visualization and Computer Graphics,* vol. 14, no. 1, pp. 213-230, Jan./Feb. 2008.

[22] R.W. Sumner and J. Popović, "Deformation Transfer for Triangle Meshes," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 399-405, Aug. 2004.

[23] B. Allen, B. Curless, and Z. Popović, "The Space of Human Body Shapes: Reconstruction and Parameterization from Range Scans," *ACM Trans. Graphics,* vol. 22, no. 3, pp. 587-594, July 2003.

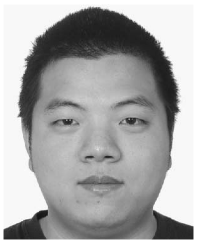
[24] T. Weise, S. Bouaziz, H. Li, and M. Pauly, "Realtime Performance-Based Facial Animation," *ACM Trans. Graphics,* vol. 30, no. 4, pp. 77:1-77:10, July 2011.

[25] E. Chuang and C. Bregler, "Performance Driven Facial Animation Using Blend Shape Interpolation," technical report, Stanford Univ., 2002.

[26] H. Huang, J. Chai, X. Tong, and H.-T. Wu, "Leveraging Motion Capture and 3D Scanning for High-Fidelity Facial Performance Acquisition," *ACM Trans. Graphics,* vol. 30, no. 4, pp. 74:1-74:10, July 2011.

[27] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross, "High-Quality Single-Shot Capture of Facial Geometry," *ACM Trans. Graphics,* vol. 29, pp. 40:1-40:9, July 2010.

[28] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R.W. Sumner, and M. Gross, "High-Quality Passive Facial Performance Capture Using Anchor Frames," *ACM Trans. Graphics,* vol. 30, no. 4, pp. 75:1-75:10, July 2011.

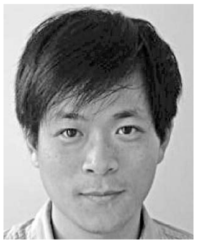


Chen Cao is working toward the PhD degree at the State Key Lab of CAD&CG, Zhejiang University. His research interest includes computer animation.

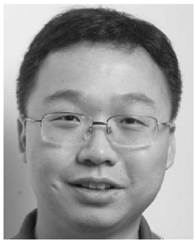
Yanlin Weng received the PhD degree from the University of Wisconsin, Milwaukee, in 2008. She is currently an assistant professor in the Computer Science Department, Zhejiang University. Her research interests include geometric modeling and computer animation.

Shun Zhou is working toward the undergraduate degree in the Computer Science Department, Zhejiang University. His research interest includes computer animation.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.



Yiying Tong received the PhD degree from the University of Southern California in 2004. He is an assistant professor at Michigan State University (MSU). Prior to joining MSU, He worked as a postdoctoral scholar at Caltech. His research interests include discrete geometric modeling, physically based simulation/animation, and discrete differential geometry. He received the U.S. National Science Foundation (NSF) Career Award in 2010.



Kun Zhou received the BS and PhD degrees in computer science from Zhejiang University in 1997 and 2002, respectively. He is a Cheung Kong Distinguished Professor in the Computer Science Department, Zhejiang University, and a member of the State Key Lab of CAD&CG, where he leads the Graphics and Parallel Systems Group. Prior to joining Zhejiang University in 2008, he was a leader researcher of the Internet Graphics Group at Microsoft Research Asia. His research interests include shape modeling/editing, texture mapping/synthesis, animation, rendering, and GPU parallel computing.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.

Authorized licensed use limited to: Qingdao University. Downloaded on January 10,2023 at 07:09:35 UTC from IEEE Xplore. Restrictions apply.