音声を用いて生成する３次元顔動画システム

陳 炳 宇[†] 黄 鈞 澤[†]
黄 輔 中[†] 荘 永 裕[†]

Speech-Driven 3D Facial Animation

Bing-Yu Chen,[†] Jun-Ze Huang,[†] Fu-Chung Huang[†]
and Yung-Yu Chuang[†]

It is often difficult to animate a face model speaking a specific speech. Even for professional animators, it always takes a lot of time. In this paper, we provide a speech-driven 3D facial animation system which allows the user to easily generate facial animations. The user only needs to give a speech as the input. The output will be a 3D facial animation relative to the input speech. There are three components in our system. The first part is the multidimensional morphable model (MMM). MMM is build from the pre-recorded training video using machine learning techniques. People can use MMM to generate realistic speech video respect to the input speech. The second part is facial tracking. Facial tracking can extract the feature points of a human subject in the synthetic speech video. The third part is Mesh-IK (mesh based inverse kinematics). Mesh-IK takes the motion of feature points as the guide to deform the 3D face models, and makes the result model have the same looking in the corresponding frame of the speech video. Thus we can have a 3D facial animation as the output.

1. Introduction

As the popularity of 3D animation movies and video games, the facial animation recently plays an important role in those applications. However, even for a professional animator, to creat a 3D facial animation with the correct lip shape relative to the input speech is still a difficult task. Therefore, our goal is to generate a 3D facial animation by an input speech, so that the users can create a 3D facial animation as simple as just speaking.

When the user input a speech voice, we first generate a corresponding speech video. Then, we use the speech video to drive a 3D face model to produce the final facial animation. An overview of our system is shown in Fig. 1. For an input speech, we first use MMM (multidimensional morphable model)[1] to generate the corresponding speech video. Then, we use facial tracking to find the feature points of a human subject in the synthetic speech video, and then use the motion of the feature points to move the feature vertices on a 3D face mesh in 2D space. Finally, Mesh-IK (mesh-based inverse kinematics)[2] is introduced to take the 2D positions of the feature vertices as the constraints to produce the final 3D facial animation.

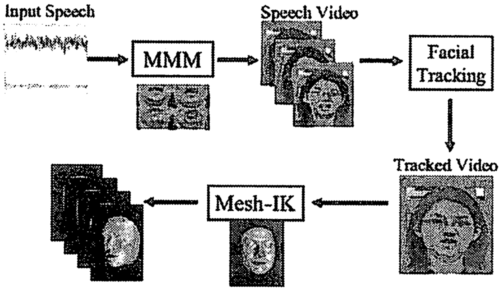

図１ システムの概要
Fig. 1 The system overview.

The facial tracking and Mesh-IK can also take a real speech or facial expression video as the input, and generate a speech or expression facial anima-

[†] 台湾大学
National Taiwan University

tion. The only requirement is the subject must be recorded at frontal-parallel view.

2. Related Work

To synthesize a speech video, Ezzat and Poggio propose a learning network[3]. They use the central image as a reference image, and find a correspondence function from the reference image to other images in the network. With this function, their system is capable to synthesize different head poses inside the range of the network. They further divide a face in some small regions, and construct a network for each sub-region. By combining those separate regions, they can synthesize different expression faces. They also map different phonemes that have a similar mouth shape to the same viseme image to reduce the size of viseme set[4]. Then, they apply morphing technique to the transformation defined by optical flow to get the result video.

Bregler et al. describe a system called Video Rewrite[5]. They break the training video into a small set of audiovisual basis units. Each unit is a triphone segment. A new video is synthesized by decomposing the input audio into phonemes and concatenating the appropriate triphone sequences. Although Video Rewrite can produce amazing results, it requires a large database. Hence, Ezzat et al. proposed an image-based videorealistic speech animation approach with machine learning techniques[1]. They select 46 prototype images from the recorded corpus. Those images' textures and motion flows corresponding to a reference image are used to build the space of mouth appearances with the multidimensional morphable model (MMM). When a phonetically transcribed audio is given, they transfer the audio into a trajectory in the MMM space. Then a videorealistic speech video is synthesized according to the trajectory.

For the facial animation, Zhang et al. use synchronized video cameras and a structured light projector to record the videos of a moving face from multiple viewpoints[6]. Then, they create the 3D face meshes and texture materials using spacetime stereo algorithm. They also propose a data-driven inverse kinematics technique called FaceIK for the user to edit the face in real time. FaceIK solves a blending weight vector for each control point, and uses the normalized radial basis function (RBF) to compute blending weights for the rest vertices.

Chai et al. develop a real time facial tracking system to extract a set of animation control parameters from a video such as head pose and face features[7]. Then, by using the expression retargeting technique and a preprocessed motion capture database, they can synthesize the mesh relative to the user performing expression.

Vlasic et al. develop face transfer with multilinear models[8] that allows the users to easily change the visemes, expressions, or even the identity of the target of a face mesh. Face transfer is based on the multilinear space of 3D face meshes that parameterizes the space of geometric variations due to different attributes like the expression or the identity.

3. Algorithm

3.1 MMM

The architecture of MMM contains two parts: one is the building of the system, including corpus recording, pre-processing, and analysis. The other part is the run time of MMM, including trajectory synthesis, MMM synthesis, and post-processing.

Our corpus contains a human subject speaking English sentences in a neutral emotion. The content of the sentences is to make all phonemes appear the same times and durations in the corpus. When recording, we ask the subject try not to move her head. The total length of the corpus is 9 minutes 55 seconds. The corpus is recorded by an analog camera, and then digitized at frame rate 29.97, resolution of 720 × 480. Hence, finally there are 17833 frames. We then analyze the recorded corpus by using the CMU Sphinx system ☆, so that every frame can be corresponded to its phoneme.

Then, to normalize the corpus is necessary, so that the only movement region of the corpus is the mouth part. Because we ask the subject try not to move her head during recoding, we can assume that the movement of the head is a perspective motion of a plan lying on the surface of the face. We

☆ http://cmusphinx.sourceforge.net/sphinx2/

use planar perspective deformation[9] to remove this movement. To fix the position of head, the first frame in the corpus is set to be the reference frame. Then, the robust optical flow[10] is applied to find the correspondences between the reference and current frames.

In order to speed up the computation of optical flow, we downgrade the resolution from 720 × 480 to 320 × 240, and only the correspondences of pixels in the head mask are used. The pixels in the mouth and eye part are also not used because their movement is a non-rigid motion, and the pixels in the background are not used because they do not have motion at all. Then we use least squares to solve the over-determined function to get the 8 parameters of the perspective warping. After finding those parameters, we warp the frames so that the head position through the corpus is still. Finally, we extract the lower part of face from the head normalized corpus for later process.

MMM is a model which applies warping and blending techniques on various lip images to synthesize new, previously unseen lip images. The MMM assumes that the complete set of lip images associated with human speech lies in a low dimensional space which has axes represent as lip texture variations and lip shape variations. The lip textures in the MMM are represented by a set of prototype images $\{I_i\}_{i=1}^{N}$. One of the prototype images is arbitrarily designated as the reference image. The lip shapes in the MMM are represented by a set of prototype flows $\{C_i\}_{i=1}^{N}$, meaning the correspondences from reference image I_1 to other prototype image I_i. The correspondence is defined as relative displacement vectors: $C_i(p) = \{d_x^i(p), d_y^i(p)\}$, Therefore, MMM is a 92 dimensional space, and can be parameterized by shape parameters α and texture parameters β.

Once we build MMM, we can use it for two tasks. One is analysis: given a lip image, MMM can be used to compute the (α, β) parameters that represent the image position in MMM space. The shape parameters α describe how to warping the prototype images using the prototype flows, and the texture parameters β describe how to blend the warped prototype images to generate the same image as the input image. Another task is synthesis: given a (α, β) parameters, MMM can be used to synthesize the lip image in the shape-texture configuration. Hence, we can use the input speech voice to get corresponding parameters (α, β) and use MMM to synthesize the lip images.

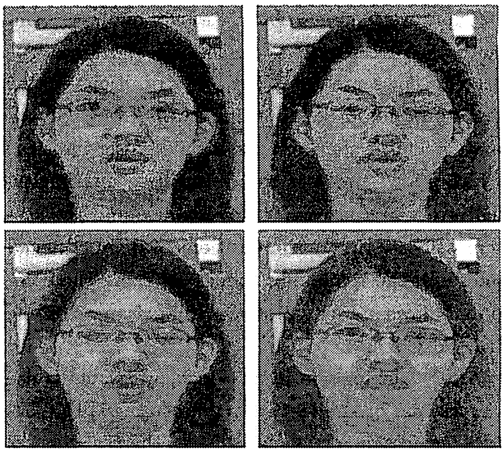

図 2　特徴点を追跡した結果
Fig. 2　Some frames of the tracked video.

The synthesis frames of our system contain only lower part of the face, so we paste these frames to a normalized head video which still has eye movement. Because the head is still in the same position, the synthesis frames of lower face part can be directly pasted on the video.

3.2 Facial Tracking

The input of facial tracking can be a synthetic speech video generated by MMM, a real speech video, or a real facial expression video. The human subject in all input videos can be anyone, but the face of the subject must be at a frontal-parallel view. Facial tracking will find out the feature points in every frame of the input video, and use the motion of feature points to move the feature vertices on a face mesh. The output of the facial tracking is the 2D positions of the feature vertices.

Our facial tracking requires the user to manually click 48 points on the first frame. In order to capture the mouth's dynamics when the subject speaks or makes expressions, we use more features points around the mouth, and put some feature points inside the mouth.

Then, we use the robust optical flow[10] to compute the corresponding flow between all frames. After the flows are computed, the positions of the feature points in every frame can be roughly estimated. Our system provide a simple user interface which allows the user to adjust those mis-tracked feature points in certain key frames, and the system will re-estimate the positions of those adjusted points in the remaining frames by using the computed flows. Fig. 2 shows some frames of the tracked video.

After tracking the input video, we want to use the motion of the feature points to move the feature vertices on a face mesh. First, the user chooses a reference image I_r from the input video and a reference mesh M_r from the example meshes such that I_r and M_r have similar appearance. Then, the user chooses 48 vertices $\{v_j\}_{j=1}^{48}$ in M_r which have an one-to-one mapping to the feature points in I_r. Finally, when we want to synthesize the result mesh M_i from the reference mesh M_r corresponding to the frame I_i, the motion vectors d_j^i of the feature points $\{f_j\}_{j=1}^{48}$ in I_r and I_i are computed by $d_j^i(x,y) = f_j^i(x,y) - f_j^r(x,y)$.

Then, the motion vectors d_j^i are added to the projection of the feature vertices $\{v_j\}_{j=1}^{48}$. Because the face model and the face in the input video are different, the motion vectors must be multiplied by a scale value. Hence, we divide a face into four parts: the left eye, the right eye, the mouth, and the whole part as shown in Fig. 3. Each part of a face is surrounded by a bounding box, then the scale ratio $S_k \in R^2 (1 \leq k \leq 4)$ is defined by the ratio of the bounding boxes on the image and on the mesh.

Then, the scaled motion vectors $S_k d_j^i$ are added to the projection of the feature vertices $\{v_j\}_{j=1}^{48}$ to get the un-projected vertex positions of $\{v_j'\}_{j=1}^{48} = \Pi(v_j) + S_k d_j^i$ (where Π is the projection function) and use them as the position constrains $\{c_j\}_{j=1}^{48} = \Pi^{-1}(v_j')$.

We assume the video is viewed in the orthographic projection, and the mesh is also viewed in the same projection, so that we can directly add the rescaled motion vectors to the projected feature vertices, and assign any value to the z parameter of the projected feature vertices when we need to un-project them. Finally, we output the x and y coordinates of $\{c_j\}_{j=1}^{48}$ for the feature vertex constrains in Mesh-IK[2].

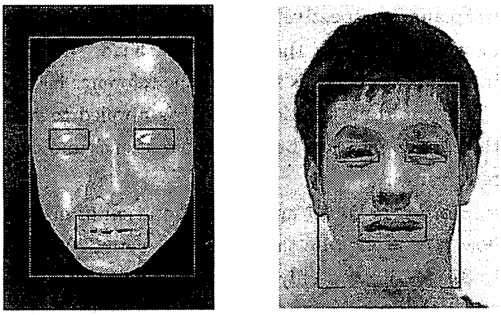

図3 画像と対応しているメッシュ上の4つのバウンディングボックス

Fig. 3 The four bounding boxes on the image and the corresponding mesh.

3.3 MeshIK

By taking the 2D position of the feature vertices as the input, we deform the face models based on MeshIK (Mesh-Based Inverse Kinematics)[2]. MeshIK can deform the models in meaningful ways that respects to the deformation explored by the input example models while satisfying the vertex constrains specified by the user. The output of MeshIK is a face mesh which subjects to the vertices constrains and looks like the corresponding frame in the speech video.

MeshIK uses some example meshes as the learning set to provide meaningful deformation. The only requirement of the example meshes is that all example meshes must have the same connectivity. The example face meshes we used are generated by Spacetime Faces[6]. We have a sequence of 384 face models under different expressions. The original mesh contains 23725 vertices and 46853 faces. For easily manipulating, we simplify the meshes to contain 3201vertices and 5825 faces, and only use 15 ∼ 20 models as the example meshes.

For mesh editing, presenting an example mesh as 3D vertex coordinates lacks of local shape property. We do not know a vertex's relation to other vertices under this presentation, so in MeshIK the geometry of an example mesh is represented as a feature vector. A feature vector contains the deformation gradient of each triangle. By randomly choosing a mesh from example meshes as the reference mesh, deformation gradient describes the transformation of a triangle on the reference mesh to another example mesh. All feature vectors relative to the example meshes forms the feature space. MeshIk uses the feature space to produce meaningful transformation while subjecting to vertex constrains.

For a reference model P_0 and a deformed model P, where both of them have n vertices and m triangles, we want to find the feature vector f corresponding to P. The deformation gradient of a triangle of P is the Jacobian of the affine transformation matrix which maps the positions of the triangle's vertices in P_0 to their positions in P. The affine transformation of the triangle does not uniquely define the rotation way of the triangle. The result model may be fractured when a triangle has different rotation as nearby triangles.

Thus, a fourth vertex is added to each triangle of a model. The fourth vertex is defined as: $V_4 = V_1 + n/\left\|n\right\|$, where n is the normal vector of the triangle and V_1, V_2, V_3 are the vertices of the triangle, so the total vertex size of P_0 and P become $n' = n + m$.

The affine transformation Φ of a vertex v of the j-th triangle operates as: $\Phi_j(v) = T_j v + t_j$, where T_j is a 3×3 matrix that contains the rotation, scaling, skewing information, and t_j is a vector which defines the displacement. The deformation gradient is the Jacobian matrix $D_p \Phi_j(v) = T_j$ which can compute from the four vertices of the j-th triangle in P_0 and P.

Because T_j is linear with the vertices $\{v_k^j\}_{k=1}^4$, a linear operator G that extracts a feature vector from the example mesh P can be defined as: $f = Gx$, where the feature vector f contains the unrolled and concatenated elements of the deformation gradient T_j for m triangles, the vector $x = (x_1, ..., x_{n'}, y_1, ..., y_{n'}, z_1, ..., z_{n'})$ stores the $\{v_k^j\}$ vertices positions of mesh P, and the coefficients of matrix G only depends on the vertices of the reference mesh P_0.

The mapping from a feature vector f of a mesh back to its 3D global coordinates involves solving a minimize function: $x = \arg\min_x \left\|\tilde{G}x - (f + c)\right\|$. Because the feature vector is invariant to global translations, the vector c which adds to f represents a constant vertex position that makes the solution unique. The matrix $\tilde{G}$ is a modified matrix of the matrix G which deletes the three columns that multiply the fixed vertex.

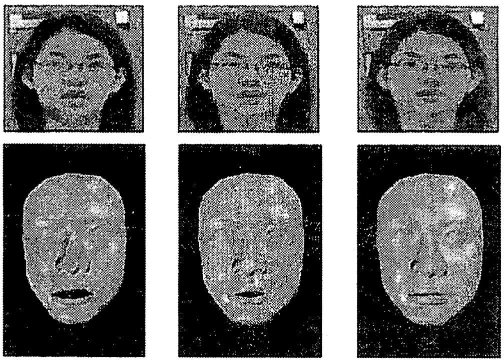

図 4　合成の音声ビデオを用いて生成した顔動画

Fig. 4 The facial animation driven by a synthetic speech video.

Given 2D positions of feature vertices, MeshIK can compute the z values of the constrained vertices and other unconstrained vertices that form the result mesh. We will then have desiring deformed meshes which form the final facial animation.

4. Result

The system is running on a desktop PC with a Intel Pentium 4 3.4 GHz CPU with 1GB memory. To synthesize a result mesh takes about 17 ∼ 25 seconds depending on the size of the example mesh. We use the synthetic speech video generated by MMM as an input speech video to generate the corresponding 3D facial animation. Fig. 4 shows some frames of the animation.

The facial tracking and Mesh-IK can also take real speech video as the input. Fig. 5 shows the generated result models. Because a real speech video has better resolution, we can have a better tracking result. Thus the output 3D facial animation looks more realistic.

Our system can also take real expression video as the input. Because our example meshes are the face models under different expression variations, the generated 3D facial animation looks better then the

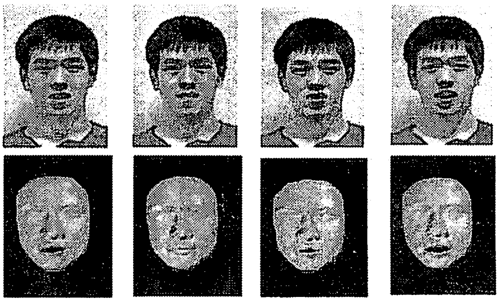

図5 本物の音声ビデオを用いて生成した顔動画
Fig. 5 The facial animation driven by a real speech video.

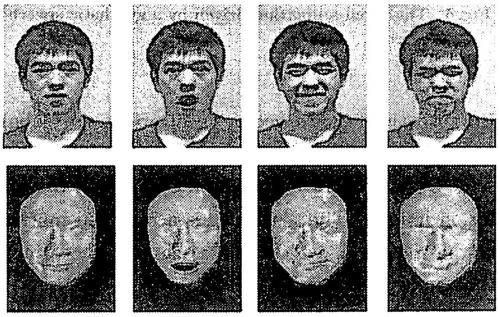

図6 本物の表情ビデオを用いて生成した顔動画
Fig. 6 The facial animation driven by a real expression video.

result driven by speech video. Fig. reffig:expression shows the synthesized result models.

5. Conclusion and Futurework

In this paper, we proposed a speech driven 3D facial animation. Based on the training video and example meshes, we can generate the 3D facial animation corresponding to the input speech. The MMM can generate video realistic speech videos, and the facial tracing and Mesh-IK can produce expression or speech facial animations by the input speech or expression video.

The example meshes used in the current system are mainly for making facial expressions. By using more example meshes related to speech, the speech video driven 3D facial animation may looks more realistic.

謝辞 The authors would like to thank Li Zhang[6] to provide the 3D face models.

参考文献

1) T.Ezzat, G.Geiger, and T.Poggio, "Trainable videorealistic speech animation," in *ACM SIGGRAPH 2002 Conference Proceedings*, 2002, pp. 388–398.

2) R.W. Sumner, M.Zwicker, C.Gotsman, and J.Popović, "Mesh-based inverse kinematics," *ACM Transactions on Graphics*, vol.24, no.3, pp. 488–495, 2005, (SIGGRAPH 2005 Conference Proceedings).

3) T.Ezzat and T.Poggio, "Facial analysis and synthesis using image-based models," in *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, 1996, pp. 116–121.

4) ——, "Visual speech synthesis by morphing visemes," *International Journal of Computer Vision*, vol.38, no.1, pp. 45–57, 2000.

5) C.Bregler, M.Covell, and M.Slaney, "Video rewrite: driving visual speech with audio," in *ACM SIGGRAPH 1997 Conference Proceedings*, 1997, pp. 353–360.

6) L. Zhang, N. Snavely, B. Curless, and S. M. Seitz, "Spacetime faces: high resolution capture for modeling and animation," *ACM Transactions on Graphics*, vol.23, no.3, pp. 548–558, 2004, (SIGGRAPH 2004 Conference Proceedings).

7) J. xiang Chai, J. Xiao, and J. Hodgins, "Vision-based control of 3d facial animation," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003, pp. 193–206.

8) D.Vlasic, M.Brand, H.Pfister, and J.Popović, "Face transfer with multilinear models," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 426–433, 2005, (SIGGRAPH 2005 Conference Proceedings).

9) G.Wolberg, *Digital Image Warping.* IEEE Computer Society Press, 1994.

10) M. J. Black, "Robust incremental optical flow," Ph.D. dissertation, 1992.