Expression Transfer: A System to build 3D Blend Shapes for Facial Animation

Chandan Pawaskar*, Wan-Chun Ma†, Kieran Carnegie*, J.P. Lewis* and Taehyun Rhee**
* Victoria University of Wellington
† Weta Digital
Wellington New Zealand
*taehyun.rhee@ecs.vuw.ac.nz

***Abstract*—Blending face geometry in different expressions is a popular approach for facial animation in films and games. The quality of the animation relies on the set of blend shape expressions, and creating sufficient blend shapes takes a large amount of time and effort. This paper presents a complete pipeline to create a set of blend shapes in different expressions for a face mesh having only a neutral expression. A template blend shapes model having sufficient expressions is provided and the neutral expression of the template mesh model is registered into the target face mesh using a non-rigid ICP (iterative closest point) algorithm. Deformation gradients between the template and target neutral mesh are then transferred to each expression to form a new set of blend shapes for the target face. We solve optimization problem to consistently map the deformation of the source blend shapes to the target face model. The result is a new set of blend shapes for a target mesh having triangle-wise correspondences between the source face and target faces. After creating blend shapes, the blend shape animation of the source face is retargeted to the target mesh automatically.**

I. Introduction

3D blend shape animation is popular for facial animation. It maps the high dimensional face mesh space (3 $\times$ number of vertices) to a low dimensional expression space (number of expressions). Blend shapes allow convenient user control and require much fewer parameters for storage of the animation sequence. Blend shape facial animation effectively blends expressions in the same mesh structure. The basic concept can be expressed as:

$$\tilde{v}_i = \sum_{j}^{N} w_j v_{ij}$$

Where, the resulting mesh's i^{th} vertex $\tilde{v}_i$ is a weighted sum of the i^{th} vertices of all of the N blend shape meshes and the weight w_j is the contribution of the j^{th} expression. [1] provides a detailed overview of blend shape animation.

Creating blend shapes is a highly skilful and laborious task [2]. Each blend shape needs to be visually meaningful for ease of control such as smile, left eyebrow raise, etc. Therefore, the blend shapes in the expression space are not all orthogonal to each other and changing the weight of one can adversely affect the contribution of other blend shapes [3]. In order to correct the interference, more blend shapes are added to the model in general. Also, any expression that cannot be represented by the current set of blend shapes has to be sculpted and augmented into the blend shape set. A production quality blend shape face model can easily consist of more than a hundred blend shapes. The efforts involved in building the blend shapes can easily be of the order of several weeks or even months for a skilful artist. Once the complete blend shape set is authored by the 3D artist, it is desirable to generate a similar set of blend shapes for a new character's face.

In this paper we present a system that can build a 3D face model comprising of blend shapes. We start with a template 3D face model having a set of blend shapes. From the template blend shape model, deformation gradients from the neutral expression to other expressions are transferred to the neutral expression of an arbitrary face. The deformation transfer algorithm [4] is utilized for this purpose.

The deformation transfer scheme requires triangle correspondences between the two meshes. We deform the template model in neutral expression to fit the arbitrary face as closely as possible based on the optimal step non-rigid ICP algorithm [5]. Triangle-wise correspondence with the template mesh (same number of vertices and triangles, and same triangle indices) is then achieved while maintaining appearance of the target arbitrary face. Therefore, the arbitrary face mesh need not have triangle-wise correspondence with the template mesh due to our non-rigid mesh registration step.

As a result, with our pipeline, we can simply transfer the animation authored for the template character to the new face model. Also, since the new character face can be made to have triangle-wise correspondence with a template model, it would be possible to blend from the template face mesh to the new face mesh while animating expression. Such a 3D face model would have two interactive controllable dimensions such as expression and identity. We can morph face meshes while blending identities. The system diagram is shown in Fig.1.

II. Related Work

High fidelity animation is one of the key requirements for generating believable digital faces. [6] did the pioneering works to generate computerized head models for graphics and animation. For detailed reference related to a wide range of previous works in facial modeling and animation see [7], [8]. There are various techniques for achieving realistic facial animation. These include but not limited to: (1) blend shapes or

978-1-4799-0883-7/13/C$31.00

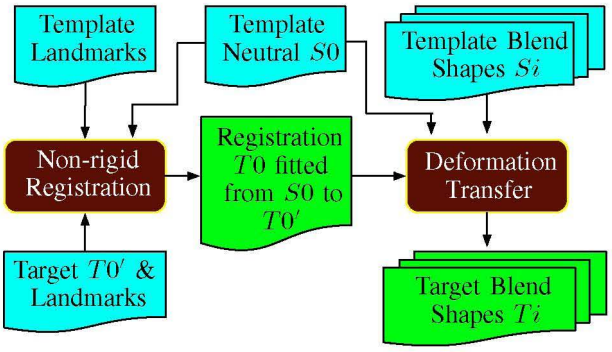

Fig. 1: System diagram.

morph targets, (2) physically based deformation and (3) facial motion capture.

Blend shape animation can be used to achieve high fidelity of expressions and fast playback. Due to these benefits, it has become the most popular method for facial animation used by movie and visual effects studios. Gollum, from The Lord of the Rings is an excellent example of the expressive power of blend shape animation. In blend shape animation (also called morph target animation or shape interpolation), a set of blend shapes (key facial expressions) are linearly combined to produce an arbitrary expression [9]. In other words, the arbitrary expression sits in the column-space of the matrix whose columns are the blend shapes. The dimensionality of this space can be reduced by performing principal component analysis (PCA). The PCA vectors are orthogonal, this is desirable as it minimizes the blend shape interference issue. However, unlike blend shapes sculpted by artists, the PCA vector contributions are no longer local. Also, the PCA vectors are un-intuitive and not suitable for manual edits [1]. [3] demonstrated an algorithm that addresses the blend shape interference and disadvantages of the PCA model approach.

In physically based systems, anatomical characteristics of bones, tissues and skin are physically simulated to provide realistic expressions [10], [11]. Such methods provide very high fidelity simulation of the facial expressions but suffer from high computational complexity.

Performance-driven facial animation captures actors performance and apply it to a 3D face model. In movie production setting, the actor's face is fitted with retroreflective markers. The motion of these markers is then recorded and used to drive deformation of the digital face [12], [13]. Marker-less motion capture has been made possible by recent research efforts. These are based on the use of structured light scanners [14], [15], depth cameras as well as conventional webcam with sufficient training captures of the actor [16] or methods like belief propagation and appearance matching [17].

As discussed, creating sufficient blend shapes takes a large amount of time and effort. There have been two notable works that can be used for greatly reducing effort for blend shape generation: expression cloning [18] and deformation transfer [4]. Expression cloning transfers vertex motion vectors from the source to the target face mesh. They use a set of user selected correspondence points. Complete correspondence is established by their novel search algorithm using the selected points. Deformation transfer, transfers deformation gradients from the source mesh to the target mesh. They also use a small set of user specified correspondence points. Although expression cloning is specialized for deformations that arise in facial expressions, deformation transfer can be used to transfer arbitrary non-linear deformations [4]. [19] indicates that expression cloning introduces interpolation artefacts in some cases and the choice of optimal landmarks is not trivial.

Our system is inspired by [20]. They presented a frame work to build an optimal blend shape model using a reduced set of example poses with initial blending weight. A template blend shapes model is used for a prior. [21] demonstrate a technique for transferring animation between blend shape models that do not have blend shape correspondence. A closely related work to ours is that of [22], their system allows compositing new faces and their blend shapes. Face features from different individuals are assembled to create a new composite face.

III. Automatic Blend Shape Transfer for corresponded meshes

If we have a set of blend shapes consisting of a neutral expression $S0$ and other expressions Si, this can be utilized to form a new set of blend shapes Ti. If each triangle of target mesh in neutral expression T_0 has correspondences with $S0$, we can build a set of blend shapes for the target face Ti using deformation transfer scheme [4]. The deformation gradients between $S0$ and other expressions (from $S1$ to Si) are transferred to the target face $T0$ to form its blend shapes $T1$ to Ti.

The local coordinate system of one triangle in $S0$ can be represented in terms of its two edges and its normal $\mathbf{n}$ as shown in equation (1). We can have a similar formulation for the local coordinate system matrix $\mathbf{V}^p$ for the corresponding triangle in Si. The deformation of the triangle in $S0$ to the corresponding triangle in Si can then be represented as a 3×3 affine transformation matrix called the deformation gradient $\mathbf{S}^p$ as shown in equation (2). By applying $\mathbf{S}^p$ to the corresponding triangle in $T0$ we can transfer the expression and obtain Ti. However, transforming individual triangles cannot respect connectivity between triangles and can fracture the mesh. Therefore, all the deformation gradients can be applied to the mesh and the target vertices can be solved for in a least squares sense by formulating a linear system of the form $\mathbf{S} = \mathbf{UX}$. Where $\mathbf{S}$ is made up of deformation gradients of $S0$ and Si, $\mathbf{U}$ is made up of deformation gradients of $T0$ and $\mathbf{X}$ is made up of the vertex positions we wish to solve for. Algorithm 1 describes this formulation. More mathematical details can be referred from [23].

$$\mathbf{V}^0 = [\mathbf{p}_2 - \mathbf{p}_1 \quad \mathbf{p}_3 - \mathbf{p}_1 \quad \mathbf{n}] \tag{1}$$

$$\mathbf{S}^p = \mathbf{V}^p \ \mathbf{V}^{0^{-1}} \tag{2}$$

We implemented the deformation transfer algorithm in the form of a Maya (3D authoring tool) plug-in as a convenient and practical solution. Solving the linear system involves factorizing a huge sparse matrix $[\mathbf{U}^T\mathbf{U}]_{n\times n}$. We use the Eigen

library for an efficient solution. An example MEL command for transferring expressions from a source mesh to a target mesh is as follows:

defTransfer -src "S0,S1,S2,S3" -tgt "T0,T1,T2,T3"

When this command is executed, deformation from mesh $S0$ to mesh Si is transferred to $T0$ and the resulting vertex locations are copied to Ti. Because deformation gradients are invariant to translation at least one vertex must be constrained [23]. We fix the first vertex by default. The default behaviour can be changed by specifying the required vertex IDs that need to be constrained. The following command will constrain the 0^{th}, 4^{th} and 120^{th} vertices to remain fixed at their locations.

defTransfer -src "S0,S1" -tgt "T0,T1" -cv "0,4,120"

Fig.2 shows a sample result.

Fig. 2: The top row is a template mesh and its blend shapes, the bottom row is target mesh (cyan) and its generated blend shapes (green).

IV. Blend Shape Transfer for Non-Corresponded Meshes

Usually corresponded meshes are difficult to obtain. For example, 3D scans of human faces are generally in the form of point cloud or polygon soup, not meshes with identical topology. To make deformation transfer work in our customized blend shape creation pipeline, we have to register a template mesh to the acquired 3D face scan. The template will deform into another shape such that it records the geometrical features of the face scan, however intrinsically it contains the same topology. We implemented the non-rigid deformation proposed by [5] for surface registration. Similar to [24], [5] solves per-vertex affine transformation that can bring the template mesh close to the target scan. Since finding the correct transformation is generally an ill-posed problem, a Laplacian regularization term is introduced to minimize the difference between transformation matrices of adjacent vertices. The algorithm acts like the iterative closest point (ICP) algorithm where it iterates between finding new correspondence and use the correspondence to compute new transformation matrices. To avoid falling into a local minimum of the objective function, we use a fitting strategy that is similar to simulated annealing: in the beginning of the fitting a large stiffness is used, thus the algorithm acts like an ICP, once the deformation of the template converges at the current stiffness term, we then reduce the stiffness, this will make the template more flexible, such that it will fit the target better. We continue to reduce the stiffness term until the template fits the target within a given threshold. For initialization and guidance of the registration, we used a small number of landmark vertices (around 6 to 8) in our cost function. Screenshot of our Maya plugin is shown in Fig.3. An example result is shown in Fig.4.

Algorithm 1 Deformation Transfer

procedure DEFORMATIONTRANSFER
  qr refers to QR factorization.
  $\mathbf{M}_s^k$, $\mathbf{M}_t^k$ are k^{th} source and target meshes respectively.
  n_p is the num. of expressions.
  n_t and n_v are the num. of tri. and vert. in template.
  Allocate $\mathbf{S}_{3n_t \times 3}$ and $\mathbf{U}_{3n_t \times n_v}$, initialize $\mathbf{U}$ to zeros.
  for $k \leftarrow 1, n_p$ **do**
    for $j \leftarrow 1, n_t$ **do**
      $\mathbf{V}_j^0 \leftarrow$ LOCALTRANSFORMATION$(\mathbf{M}_s^0, j)$
      $\mathbf{V}_j^p \leftarrow$ LOCALTRANSFORMATION$(\mathbf{M}_s^k, j)$
      $\mathbf{S}_j^p \leftarrow \mathbf{V}_j^p \mathbf{V}_j^{0-1}$
      $\mathbf{S}[j \times 3, :] \leftarrow \mathbf{S}_j^{p\top}$
      $\mathbf{T}_j^0 \leftarrow$ LOCALTRANSFORMATION$(\mathbf{M}_t^0, j)$
      $[\mathbf{Q}_{3\times3}, \mathbf{R}_{2\times2}] \leftarrow \mathbf{qr}(\mathbf{T}_j^0[1:2, :])$
      $\mathbf{W}_{2\times3} \leftarrow \mathbf{R}^{-1}\mathbf{Q}[:, 1:2]^\top$
      $[i_1, i_2, i_3] \leftarrow$ VERTEXINDICES$(\mathbf{M}_s^0, j)$
      for $t \leftarrow 1, 3$ **do**
        $r \leftarrow 3 \times (j-1) + t$
        $\mathbf{U}[r, i_1] = -\mathbf{W}(1,t) - \mathbf{W}(2,t)$
        $\mathbf{U}[r, i_2] = \mathbf{W}(1,t)$
        $\mathbf{U}[r, i_3] = \mathbf{W}(2,t)$
      end for
    end for
    $n_c =$ NUMBEROFCOLUMNS$(\mathbf{U})$
    $\hat{\mathbf{U}} = \mathbf{U}[:, 2:n_c]$
    $\mathbf{v}_1$ is the first vert. position in $\mathbf{M}_t^0$.
    $\hat{\mathbf{S}} = \mathbf{S} - \mathbf{U}[:, 1]\mathbf{v}_1^\top$
    $\mathbf{X} = (\hat{\mathbf{U}}^\top \hat{\mathbf{U}})^{-1} \hat{\mathbf{U}}^\top \hat{\mathbf{S}}$
    return X ▷ $\mathbf{X}$ is the vert. positions for $\mathbf{M}_t^k$.
  end for
end procedure

procedure LOCALTRANSFORMATION($\mathbf{M}$, i_t)
  $\mathbf{M}$ is a mesh structure.
  i_t is the index of a triangle in $\mathbf{M}$.
  i_t^1, i_t^2, i_t^3 are the vertex indices of the i_t triangle.
  $\mathbf{p}(M, i_v)$ returns the 3D position of i_v vertex in $\mathbf{M}$.
  $\mathbf{n}(M, i_t)$ returns the normal vector of i_t triangle in $\mathbf{M}$.
  $[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3] \leftarrow [\mathbf{p}(\mathbf{M}, i_t^1), \mathbf{p}(\mathbf{M}, i_t^2), \mathbf{p}(\mathbf{M}, i_t^3)]$
  $\mathbf{V} \leftarrow [\mathbf{p}_2 - \mathbf{p}_1 \quad \mathbf{p}_3 - \mathbf{p}_1 \quad \mathbf{n}(\mathbf{M}, i_t)]$
  return V
end procedure

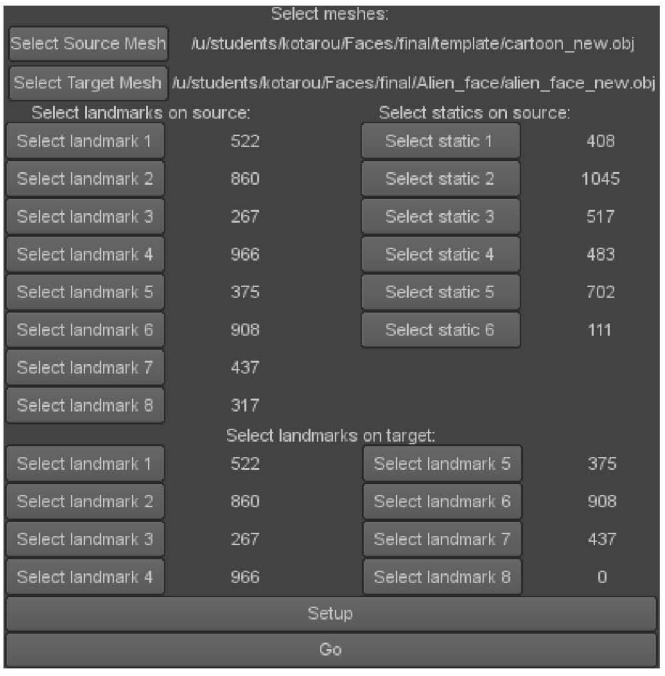

Fig. 3: A maya plugin developed for the registration process.

Fig. 4: Each row shows non-rigid registration for three different faces. From the left, source template face, target face, registered source mesh to target, and it's wireframe. Magenta dots shows landmarks.

V. Results

A. Test and Application

In this paper, we created a complete pipeline to build new blend shapes of a neutral face mesh from a template blend shapes model. Our pipeline starts with the non-rigid registration stage followed by the deformation transfer stage. Fig.6 shows blend shape expressions transferred from a template blend shape model to three arbitrary faces. Our pipeline is integrated into Maya as a form of a plug-in. After deformation transfer, vertex normals are recomputed using Maya. The correspondences between the source and target meshes were obtained by the non-rigid registration step described in section IV.

The template face consists of 2178 triangles and 1138 vertices. The faces in the second column of Fig.4 consist of 70314, 6638, 2840 triangles and 35645, 3424, 1481 vertices respectively. It took an average 2.7 seconds to generate faces in the third column. A single blend shape creation in Fig.6 took 0.15 seconds, doing a batch creation of 22 blend shapes took 1.5 seconds. The hardware setup was: i7 2.4 GHz processor, 8 GB ram running in Windows 8. Our implementation was single threaded without GPU acceleration.

Our pipeline results in fully corresponded set of blend shapes between the source and target face. Therefore, it is possible to make an interesting animation sequence such that the face identity is gradually changing from source to target while animating expression variations naturally. An example of the animation is shown in Fig.8 while selecting important frames. We simply implemented it within Maya while adding another blending parameter and control for identity changes. Also, expression retargeting is automatically obtained, since blend shape animation created using source blend shapes can be directly applied to the target face as shown in Fig.8.

B. Minimizing Artefacts

Original deformation transfer [4] can result in self intersecting meshes as shown in Fig.5. This is simply because it does not fully consider geometric differences between source and target faces. Instead of transferring deformation gradients of expressions, we demonstrated a simple but novel idea to transfer deformation gradients of identity to the new expressions. The new idea is that, instead of transferring deformation from $S0$ to $S1$ onto $T0$ we transfer deformation from $S0$ to $T0$ onto $S1$ to update $T1$. In fact, this approach gives better results as shown in Fig.7 without adding virtual triangles as in [19]. In Fig.7, the model in the first column has wider eyes compared to the template. Therefore, original deformation transfer shows artefacts such that eyelid closes more than it should. Model in the second column has taller eyes and eyelid does not close fully. Clearly, result in the second row based on our new approach shows better results.

VI. Conclusion

We presented a complete pipeline that is capable of building fully corresponded blend shape models starting from a single template blend shape model. Since our non-rigid registration step solves mesh correspondence, our system can be applied to any arbitrary target mesh. In order to minimize artefacts around eye-lids, we transferred deformation due to identity and demonstrated that it can give better results compared to transferring deformation due to change in expression. Apart from the initial allocation of landmark vertices in our non-rigid registration step, our entire pipeline is fully automated.

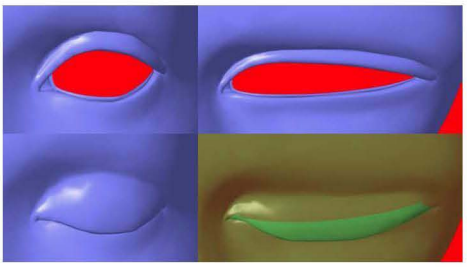

Fig. 5: The target mesh has much wider eyes, deformation transfer [4] moves the eyelid way past the expected position. The bottom-right mesh is rendered semitransparent to better visualize the artefact.

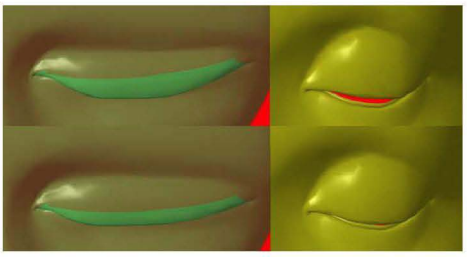

Fig. 7: The first row is result of original deformation transfer. The second row shows results from our method.

Although, the change in identity transfer can give better results for the eye region as seen in Fig.7, it cannot handle very wide variations from the template. Our system could benefit from the incorporation of virtual triangles as described in [19]. Also, our non-rigid registration step had difficulty with the eye regions where triangle density is high. Other non-rigid deformation methods that are based on the Laplacian deformation [25], [26] could be explored in the future.

The system performance could be further improved by leveraging multiple cores or GPU acceleration. Our system can be used to build bi-linear and multi-linear face models. However, building such models in the form of a database is beyond the scope of this paper. We leave it for our future works.

References

[1] J. P. Lewis and K.-i. Anjyo, "Direct manipulation blendshapes," *IEEE Comput. Graph. Appl.*, vol. 30, no. 4, pp. 42–50, Jul. 2010.

[2] J. Osipa, *Stop staring: facial modeling and animation done right, 3rd ed.* Hoboken: John Wiley & Sons Inc, 2010.

[3] J. P. Lewis, J. Mooser, Z. Deng, and U. Neumann, "Reducing blendshape interference by selected motion attenuation," in *Proceedings of the 2005 symposium on Interactive 3D graphics and games*, ser. I3D '05. New York, NY, USA: ACM, 2005, pp. 25–29.

[4] R. W. Sumner and J. Popović, "Deformation transfer for triangle meshes," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 399–405, Aug. 2004.

[5] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid icp algorithms for surface registration," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007, pp. 1–8.

[6] F. I. Parke, "Computer generated animation of faces," in *Proceedings of the ACM annual conference - Volume 1*, ser. ACM '72. New York, NY, USA: ACM, 1972, pp. 451–457.

[7] F. I. Parke, K. Waters, and T. R. Alley, *Computer facial animation*. AK Peters Wellesley, 1996, vol. 55.

[8] Z. Deng and J. Noh, "Computer Facial Animation: A Survey," 2007, pp. 1–28.

[9] Z. Deng and U. Neumann, *Data-Driven 3D Facial Animation*. Springer-Verlag London Limited, 2007.

[10] D. Terzopoulos and K. Waters, "Physically-based facial modelling, analysis, and animation," *The Journal of Visualization and Computer Animation*, vol. 1, no. 2, pp. 73–80, 1990.

[11] K. Kähler, J. Haber, and H.-P. Seidel, "Geometry-based muscle modeling for facial animation," in *No description on Graphics interface 2001*, ser. GRIN'01. Toronto, Ont., Canada, Canada: Canadian Information Processing Society, 2001, pp. 37–46.

[12] L. Williams, "Performance-driven facial animation," *SIGGRAPH Comput. Graph.*, vol. 24, no. 4, pp. 235–242, Sep. 1990.

[13] H. Huang, J. Chai, X. Tong, and H.-T. Wu, "Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 74:1–74:10, Jul. 2011.

[14] L. Zhang, N. Snavely, B. Curless, and S. M. Seitz, "Spacetime faces: high resolution capture for modeling and animation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 548–558, Aug. 2004.

[15] T. Weise, H. Li, L. Van Gool, and M. Pauly, "Face/off: live facial puppetry," in *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '09. New York, NY, USA: ACM, 2009, pp. 7–16.

[16] C. Cao, Y. Weng, S. Lin, and K. Zhou, "3d shape regression for real-time facial animation," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 41:1–41:10, Jul. 2013.

[17] T. Rhee, Y. Hwang, J. D. Kim, and C. Kim, "Real-time facial animation from live video tracking," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '11. New York, NY, USA: ACM, 2011, pp. 215–224.

[18] J. Noh and U. Neumann, "Expression cloning," in *ACM SIGGRAPH 2006 Courses*, ser. SIGGRAPH '06. New York, NY, USA: ACM, 2006.

[19] J. Saito, "Smooth contact-aware facial blendshapes transfer," in *Proceedings of the Symposium on Digital Production*, ser. DigiPro '13. New York, NY, USA: ACM, 2013, pp. 7–12.

[20] H. Li, T. Weise, and M. Pauly, "Example-based facial rigging," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 32:1–32:6, Jul. 2010.

[21] Y. Seol, J. Lewis, J. Seo, B. Choi, K. Anjyo, and J. Noh, "Spacetime expression cloning for blendshapes," *ACM Trans. Graph.*, vol. 31, no. 2, pp. 14:1–14:12, Apr. 2012.

[22] W. Ma, M. Barbati, and J. P. Lewis, "A facial composite editor for blendshape characters," in *Proceedings of the Digital Production Symposium*, ser. DigiPro '12. New York, NY, USA: ACM, 2012, pp. 21–26.

[23] R. W. Sumner, "Mesh modification using deformation gradients," Ph.D. dissertation, Cambridge, MA, USA, 2006.

[24] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "Scape: shape completion and animation of people," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 408–416, 2005.

[25] M. Botsch and O. Sorkine, "On linear variational surface deformation methods," *IEEE Trans. on Visualization and Computer Graphics*, vol. 14, no. 1, pp. 213–230, 2008.

[26] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Proc. of the Eurographics Symposium on Geometry Processing*, 2007, pp. 109–116.

Fig. 6: Top row is template face and its blend shapes. Each subsequent row is a new face and its blend shape set. All the green face meshes were auto generated.

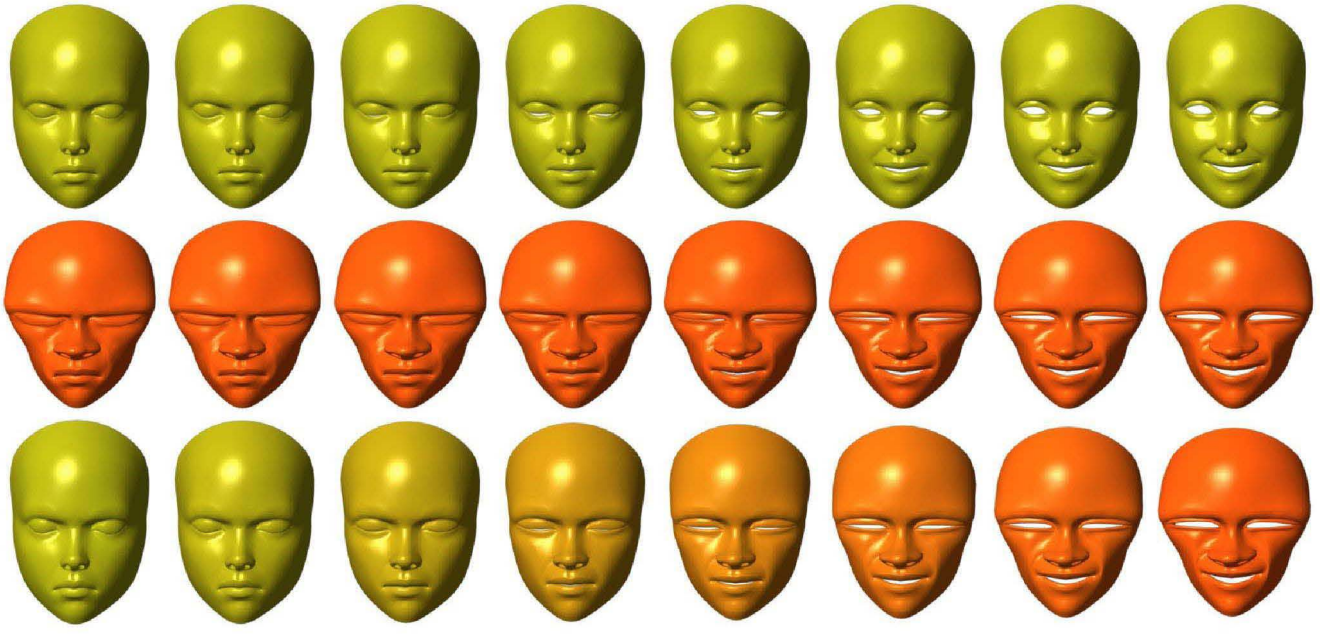

Fig. 8: First and second rows show same expression animation for two different faces. Third row shows the same animation but the face gradually changes from the first to the second face.