Real Time Online Facial Expression Transfer with Single Video Camera

Abstract

Facial expression transfer is currently an active research field. However, 2D image wrapping based methods suffer from depth ambiguity and specific hardware is required for depth-based methods to work. We present a novel markerless, real time online facial transfer method that requires only a single video camera. Our method adapts a model to user specific facial data, computes expression variances in real time and rapidly transfers them to another target. Our method can be applied to videos without prior camera calibration and focal adjustment. It enables realistic online facial expression editing and performance transferring in many scenarios, such as: video conference; news broadcasting; lip-syncing for song performances; etc. With a low computational demand and hardware requirement, our method tracks a single user at an average of 38 fps. Our tracking method runs smoothly in web browsers despite their slower execution speed.

Keywords: facial tracking, expression transfer

1 Introduction

Facial performance tracking and editing have attracted attention from both research community and industries, due to its successful application in film making and game production. Real time methods in particular are being actively studied due to their effectiveness and many potential applications. Traditionally, such technologies were only available to institutional users with high-end hardwares such as laser scanners and motion capture devices. Nowadays, with the wide spread availability of video recording devices, there has been an increasing demand for real time systems that can track and edit facial performances using only video input. Such systems feel more natural to users because they offer instant feedback.

Color and depth information on their own do not carry much semantic meaning and cannot be used to directly track facial performance. In order to transfer an expression between the actor and the target, it has to be encoded in the same system. Virtually all methods are based on the assumption that all faces can be represented by a linear combination of blend shapes or training data, which is usually represented by coefficients. One can adjust the combination to fit the input and derive the expression coefficient from its discrepancy when compared to the neutral expression. Since methods that use color information only have access to the projected shape of the face, generally depth is estimated from the same face through different expressions and poses derived from multiple image frames. However, the facial mesh differs drastically between individuals. Some of the tracking methods require a preprocessing step to build a user specific facial mesh in advance to achieve real time performance.

The easiest way to animate and transfer facial expressions would only require a single depthless video camera, and the system would run in real time. As these methods would only use 2D images, they have the potential of processing videos collected in uncontrolled settings and using common consumer devices such as phone cameras and webcams. Online methods are particularly appealing as user can get instant feed back.

Still, after years of research in computer vision and graphics, many existing image-based facial tracking systems still are not able to produce results with high frequency detail, since they use only a coarse set of 2D or 3D facial landmarks. Facial regions that are not tracked by the systems are typically ignored and the information is discarded. However, high frequency detail is necessary to generate a convincing result. Most research on transferring expression to another target focuses on virtual avatars [26, 4], where the human actor drives the expression of a virtual character. Creating a realistic morphable model for a real human is difficult due to factors such as skin folding and wrinkling, and change in eye and mouth shape. Transferring expression using only color information is even more challenging, because without depth information as guidance it has to be robust against depth ambiguity, illumination variation, noisy background and occlusion.

In this paper, we present a fully real time online facial expression transfer method with a single RGB camera that can be applied to both static images and video. Our approach requires low computational resource and is robust in uncontrolled environments.

Contributions: Firstly, we propose a novel method to model the appearance variance of a target user with different facial expressions in real time. Secondly, a new 3D mesh tracking method that is both robust and simple is introduced. Finally, the learned appearance models can be used to validate the accuracy of new input, which prevents outliers from being taken into consideration.

2 Related Work

Facial performance capture and tracking is a well established research field[11, 12, 13, 14]. Traditionally, 3D facial performance capture methods use marker-based motion capture systems such as [15], which track a sparse set of markers on a person's face. These kind of methods are dependent upon using a specific environment and require the person to wear markers, which is both time consuming and invasive. Recently, with the development of stereo vision and depth camera, modern methods no longer require markers and are able to produce superior results under less constraints.

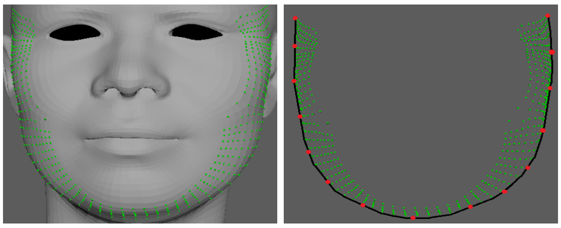

(a) Predefined set of contour vertices (b) Points uniformly sampled from convex hull

Figure 1: Contour index generation.

Currently, marker-less real time facial tracking methods can be categorized into two groups in terms of the input they use. One group, [1, 2, 3, 4] uses depth information and the other one relies solely on RGB input [5, 6, 7, 8, 9].

Lately consumer-level depth capture devices are becoming more widely available, and state-of-the-art methods [2, 3] that use depth information as input have achieved good performance. However, a vast majority of videos readily available online or captured by mobile devices are typically not accompanied with depth information.

RGB-based methods generally rely on a facial landmark detector to coarsely align the face. Numerous techniques have been proposed to locate facial landmarks from image input. These methods reduce the facial alignment error by using a linear combination of the training data or the descent direction to update the average face to explain the input image, e.g., Active Shape Models[16, 17], Cascaded Pose Regressors[18, 19, 20], Supervised Descent Methods[21, 22] and Deformable Part Based Models[23, 24, 25]. New techniques are in constant development, with higher robustness and accuracy. Fundamentally, our method can use any of these facial landmark detection methods provided that they provide reliable results.

Recently [5] proposed a method that tracks the 3D mesh of a user in real time using RGB cameras. Essentially, the mesh is driven by a linear combination of faces in a 3D face database [10] consisting of different individuals and expressions. However, it mainly focuses on tracking and driving the animation, and not transferring realistic expression onto a real human target from video streams. Thus the applications are limited to driving game characters or virtual avatars.

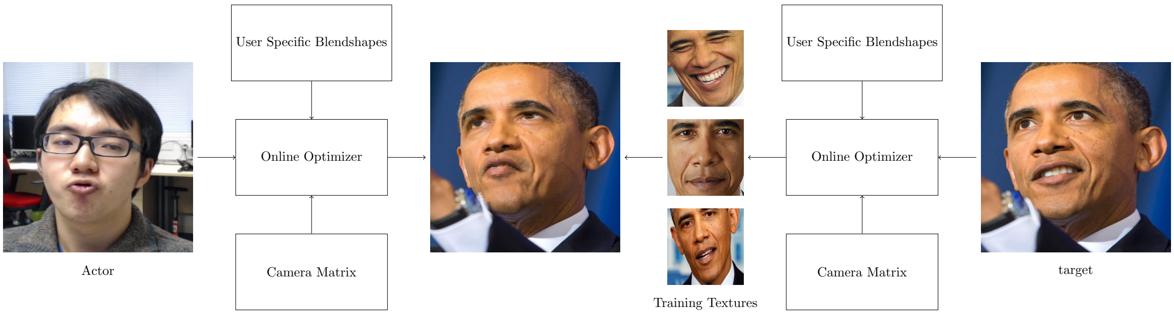


Figure 2: Overview of our method

3 Overview

The overview of our method is illustrated in Fig.2. To initialize, both images of the actor and the transfer target are first scanned by a face detector [27]. We use an object tracker [28] to get the bounding box of the face. First, we use a 2D landmark detector to localize the facial landmarks from within the bounding box of the detected face. Next, we solve the pose, expression coefficient and camera matrix by minimizing difference between the projection of their combination and the 2D landmarks using an average identity. After a sufficient number of frames are processed we solve for the user specific identity coefficients. This process can be repeated many times if necessary during the whole runtime. We learn the transfer target's appearance using all images available offline and solve for its pose, expressions and identity, or we can learn the textures online and update them iteratively. Finally, given the actor's expression we generate appropriate texture for the target and we render the target's face with the actor's expression.

4 Facial Mesh

We use Facewarehouse [10] as our underlying 3D mesh database. It consists of the facial geometry of 150 persons, and 47 expression for each of the said people. This database is accompanied with a dataset of 2D facial images and their corresponding manually labeled 74 facial landmarks. We train a 2D landmark detector on this dataset and use it to track 2D landmarks. Since all faces shared the same topology we select only the frontal facial vertices and rearrange them into a rank-three(3 mode) data tensor. The tensor is compressed into a 4k vertices × 50 identities × 25 expressions core using [29]. Uncompressed original data is represented as blendshapes and has semantic meanings such as closing eyes, opening mouth and frowning. However, the computational cost of fitting new expression and identity is smaller on the compressed core. The compressed core also carries no semantic meaning and we cannot use it to directly transfer facial expressions. Since the landmark prediction produced by the detector can be noisy at times, in order to generate realistic results we have to regularize the expression coefficients to make sure they are within valid ranges. Thus, we reconstruct the original expression blendshapes and weights given an identity coefficient, and use them instead of the compressed weights. Uncompressed blendshapes can be reconstructed using the product of 150 × 50 or 47 × 25 orthonormal matrices and the product of compressed coefficient.

$$B_{exp} = C \times U_{id} \tag{1}$$

$$B_{id} = C \times U_{exp} \tag{2}$$

Intuitively, B_{exp} represents a person with different facial expressions, whereas B_{id} represents the same expression performed by different individuals. When solving for identity we always use the compressed core with early stopping to prevent over-fitting while transferring expression and animating virtual characters we solve with the reconstructed blendshapes, with early stop and clamping as a regularization to generate plausible results.

5 Mesh Tracking

5.1 Pose Estimation

First we rigidly align the 3D mesh to the 2D landmarks. 3D coordinates of the inner landmarks such as the ones on nose, eyes, mouth and eyebrow are located using fixed indices. However the vertex indices on the face contour have to be updated in regard to different poses and expressions. We can find the contour indices by projecting the vertices onto a plane and sampling its convex hull uniformly. To reduce computational cost we only project a predefined set of vertices as in Fig.1. First we project the predefined set of vertices, which are organized into horizontal lines in a clockwise order, onto a plane and compute their convex hull. Next we find the closest point to its first and last point. Finally we connect the two points with uniformly sampled points on the convex hull, and the indices are copied from the closest vertices to these points.

After all the landmarks have been found we estimate the pose by minimizing distance between its projection and 2D landmarks. We formulate rotation in the pose estimation step as a compact 3D Rodrigues vector. First we use Direct Linear Transform [30] to get an initial estimate of rotation and translation. Then we use L-BFGS [31] algorithm to iteratively refine the Rodrigues and translation vector r and T by summing the errors and residuals of every points. Finally we convert the Rodrigues vector to a 3 × 3 rotation matrix R.

5.2 Expression Estimation

Next we solve for the expression coefficient. We define the projection operator of a vertex V as $\prod$, which is a scaled orthogonal

$$\prod(V) = (V \times R + T)_{xy}/T_z \tag{3}$$

for the actor and perspective

$$\prod(V) = (f \times V \times R + T)_{xy}/(V \times R + T)_z \tag{4}$$

for the target. f stands for the focal length. A vertex on facial mesh of different expressions can be represented by the product of a expression coefficient E and a set of identity specific blendshapes B, as in

$$V_{xyz} = E \times B \tag{5}$$

Given an identity coefficient I, which represent the same individual performing the different expressions. We minimize squared distance between the 2D landmarks L as in

$$D = \frac{1}{2}||L - \prod(E \times B)|| \tag{6}$$

and save the blended vertices as F_{xyz} and their projection as P which are reused when computing the derivative.

We omit the focal length for the orthogonal operator. Accordingly, derivative of the two operators in respect to the ith expression coefficient are

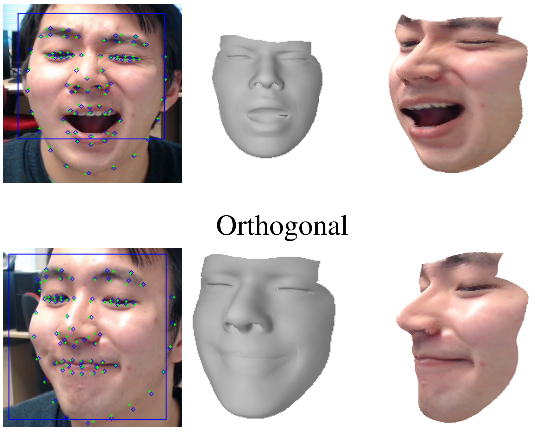
$$\frac{B^{(i)}_{xy}}{T_z}(L - \frac{1}{T}(F_{xy} + T_{xy})) \tag{7}$$

for orthogonal projection and

$$(f \times P - L)(f\frac{B^{(i)}_{xy} + B^{(i)}_z \times P}{F_z}) \tag{8}$$

for perspective projection. Since rotation matrix is orthogonal, we rotate the blendshapes in advance for both projection operator and scale the 2D landmark by $\frac{1}{T}$ the orthogonal case, which makes its derivative become $B^{(i)}_{xy} \times D$. Again, we use a L-BFGS solver for the expression coefficient, which allows early stopping which speeds up the estimation and prevents over-fitting.

Using fixed pose parameters, we use the orthogonal projection operator on the actor as its computational cost is lower. For the transfer target we use the perspective operator as it produces more realistic results as shown in Fig.3. The rendering in the third column generated by slightly rotating the tracked face to show depth. Since the orthogonal operator use a uniform depth assuming the object is small enough that its appearance will not be greatly affected by focal length and depth, the fitted mesh seems squashed with rotation. Nevertheless, we are able to successfully recover expressive facial expression for both projection operator. Although the two operators are different but the expression coefficients are interchangeable.



Orthogonal

Perspective

Figure 3: Projection Operator

As discussed in Section 4, when transferring the users' expression we need to use the uncompressed original data. By subtracting the neutral expression from every other expressions the vertex can be represented similarly

$$V_{xyz} = E \times (B - N) + N \tag{9}$$

, where N is the neutral face. We don't need to change the way we compute derivative as we replace the parts involving adding the neutral with blended vertices saved as F_{xyz}.

5.3 Identity Estimation

The identity coefficient can be solved similarly. Given a specific expression, we can create a set of blendshapes which represent different individuals performing the same expression. However, unlike expression coefficient solving identity on a single frame is not sufficient and often leads to weird looking results. Thus, we solve it using a set of frames with distinctive poses and expressions. We denote the ith blendshapes with expression j as $B^{(i,j)}$ and minimize the distance

$$\sum_{i,j,k} \frac{1}{2} ||L^{(j,k)} - \prod(I^{(i)} \times B^{(i,j,k)})|| \tag{10}$$

while fixing all the other parameters.

It is also unnecessary to solve for the focal length for single frames. Given a set of 2D landmarks and projected vertices, we can solve the focal length by minimizing

$$\sum_{j,k} \frac{1}{2} || \prod(F^{(j,k)}_{xy}) * f_{xy} - L^{(j,k)}|| \tag{11}$$

The routine of updating the focal length and identity coefficient is summarized in Algorithm 1. How to update the texture is described in Section 6.

```
Algorithm 1: User Adaption
  Data: New Frame
  while not converged do
      if New frame is valid then
          Add frame;
      end
      forall the frames do
          Solve pose;
          Solve identity;
          Solve expression;
      end
      Solve identity for all frames;
      Solve focal length for all frames;
      Update texture;
  end
  Result: Updated Identity
  Result: Updated Focal Length
  Result: Updated Texture
```

5.4 User Adaptation

We regularly select distinctive frames to solve for users' identity coefficients. The first few initial expression, pose and the flattened landmark vector joined together in row major order to form a parameter base A. When a new frame arrives, we try to use this base to reconstruct the solved parameters. If the reconstruction error is larger than a threshold we add this frame to the base. Once there are enough newly added frames we recompute the identity coefficient focal length, which can be repeat multiple time during runtime if necessary. We denote b as the new frame's parameters.

Since the landmarks produced by the detector are often noisy, we use double exponential smoothing to smooth them and accounts for its trend. We also use double exponential smoothing to smooth the fitted result, after converting rotation matrix to quaternion than back. Although we don't have cross frame regularization as in [5], our system is still able to produce smooth and accurate result. Furthermore, we use our learned appearance model to quickly verifies whether a frame is valid by thresholding the sum of pixel differences of the extracted texture to the texture produced from the learned appearance model. After a while base size become stable and most of the possible combinations of users' expression and pose will be collected.

6 Appearance Learning

After the identity coefficients and corresponding blendshapes have been computed, we can directly transfer low frequency information from actor to target. However, details such as wrinkles cannot be transfered because these high frequency information are not captured by the 2D facial landmarks.

6.1 Appearance Learning

The original uv coordinates of the models in Facewarehouse includes the whole head, again we crop the frontal facial uv coordinates and use the holes formed by the eyeballs and inner mouth to place our uv coordinate for transferring eyes and inner mouth texture. Texture is extracted by rendering the uv coordinates as the vertices and their corresponding vertex projection as uv coordinate. This helps us to normalize different textures to the same format, which makes learning possible. Since the structure of our uv map is the same for different person and expression, we can transfer our the new texture to different targets.

After the 3D mesh's projection is matched to its corresponding image, we use the texture map as vertex position and the projected vertex of the mesh to extract a texture. We categorized the extracted textures by their corresponding expressions. The appearance model is updated by finding the most similar expression from the collected parameter base and using the new texture, which is passed through a low-pass filter to avoid outliers and exponential smoothing to void outliers.

Finally, we compute an average texture from all the collected textures. Each textures can be represented by a linear combination of their difference from the average texture. We denote the differences as B_d and use it to construct a texture variance base.

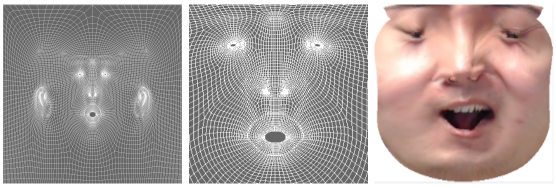


Figure 4: UV coordinates.

6.2 Texture Generation

When transferring the expression of the actor to the target, we need to generate appropriate texture for the actor's expression. Given a learned appearance texture variance base, we compute a vector that contains the distance between the given expression and the expressions from parameter base. We normalize this vector and use it as a coefficient to generate a texture, which is a linear combination of the texture base and overlayed onto the texture of the new frame. This new texture contains high frequency details and makes the result realistic. To keep the computational cost low we need keep the length of the texture coefficient short. Thus, instead of using dimension reduction methods such as Principle Component Analysis[32], we omit textures with low variance from the average texture and keep the expression indices unchanged, which allows us to deliver real time performance.

The generated texture can also be used to validate whether the parameters estimated for the current frame is valid by checking its sum of pixel differences to the extracted texture. We reject inaccurate frames from the texture updating, identity and focal length solving process. The texture variance base is shown in Fig. 5, where the first row are the input/query frames, the second row are the extracted textures and the third row are their variances from the base texture colored in JET colormap. Note that the color intensity of the third row between each frames does not necessarily reflect its real value as we normalize them individually instead of jointly to make regions with different high frequency details highlighted.

7 Results

We use all the frames from a certain user to solve for the identity, texture and focal length offline.

By treating it as ground truth we produce Fig. 7 to show the its difference from the identity, texture and focal length solved online. Our method is able to converge to good result within 400 hundred frames.

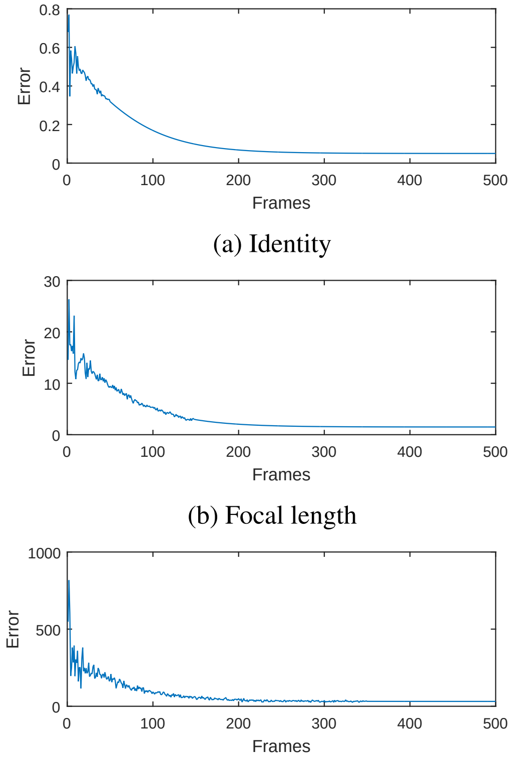
The tracked mesh are shown in Fig.8, which shows that our system can robustly track the rough movement and expression. Although high frequency details are missing, combined with the texture generated from our learned appearance model we can still deliver realistic results as shown in 6.

8 Conclusion

We have introduced a novel online facial expression transfer method using a single video camera. A novel online appearance learning method has been proposed to generate more convincing results. We have shown that our method is robust and accurate. It has relatively low computational cost and hardware requirement. Not only does our method provides facial expression transfer but it also can drive virtual characters, the learned appearance learning scheme we proposed allows us to generate high frequency details on the target.

Our facial mesh tracking pipeline is robust against occlusion and illumination changes. Even if it fails in extreme situations it is able to quickly recover and our appearance model is able to reject the outliers from contaminating the long term parameters such as identity, focal length and expression specific textures. Nevertheless, although we have accounted for possible noise and brief occlusion, our method cannot deal with long term partial occlusion and gadgets such as glasses will influence the final transfer result.

With our demo we have already set a up live demo demonstrating that our mesh tracking method is simple enough to run smoothly on web browsers. For future work, we will save the learned model on client machines as a live puppetry and transfer only expression coefficient to greatly reduce bandwidth requirement for real time video conference.



(a) Identity

(b) Focal length

(c) Texture

Figure 7: Online tracking error.

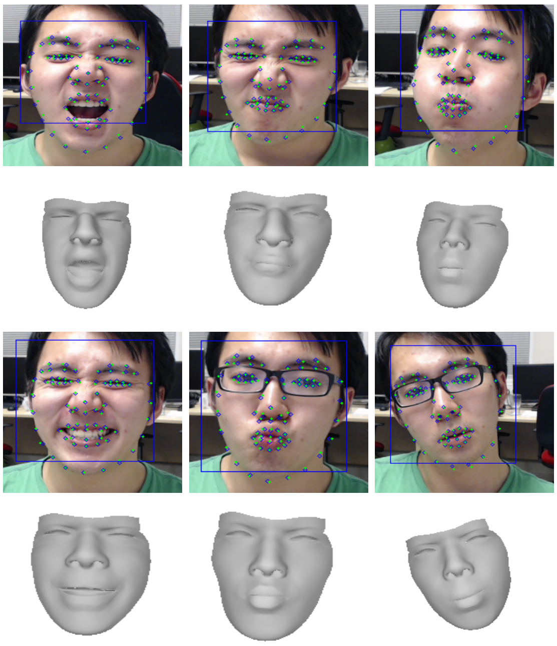


Figure 8: Tracked mesh.

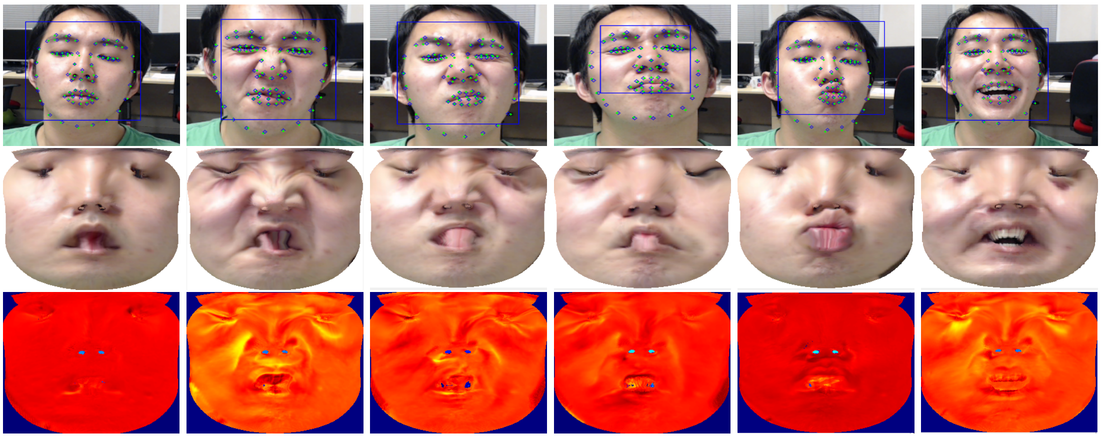


Figure 5: Texture generation

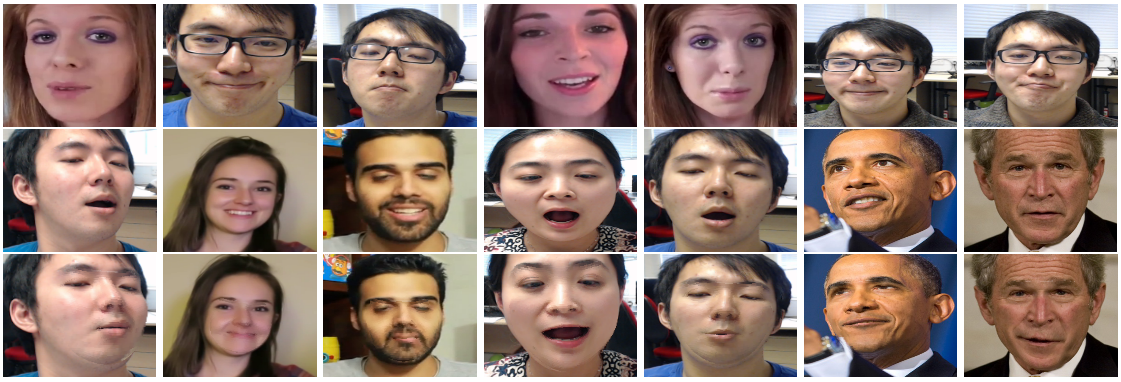


Figure 6: Expression transfer

References

[1] Matthias Hernandez, Jongmoo Choi, and Gérard Medioni. Laser scan quality 3-d face modeling using a low-cost depth camera. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1995–1999. IEEE, 2012.

[2] Justus Thies, Michael Zollhöfer, Matthias Nießner, Levi Valgaerts, Marc Stamminger, and Christian Theobalt. Real-time expression transfer for facial reenactment. *ACM Transactions on Graphics (TOG)*, 34(6):183, 2015.

[3] Pei-Lun Hsieh, Chongyang Ma, Jihun Yu, and Hao Li. Unconstrained realtime facial performance capture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1675–1683, 2015.

[4] Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. Realtime performance-based facial animation. In *ACM Transactions on Graphics (TOG)*, volume 30, page 77. ACM, 2011.

[5] Chen Cao, Qiming Hou, and Kun Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Transactions on Graphics (TOG)*, 33(4):43, 2014.

[6] Pablo Garrido, Levi Valgaerts, Chenglei Wu, and Christian Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph.*, 32(6):158–1, 2013.

[7] Yanlin Weng, Chen Cao, Qiming Hou, and Kun Zhou. Real-time facial animation on mobile devices. *Graphical Models*, 76(3):172–179, 2014.

[8] Supasorn Suwajanakorn, Ira Kemelmacher-Shlizerman, and Steven M

Seitz. Total moving face reconstruction. In *Computer Vision–ECCV 2014*, pages 796–812. Springer, 2014.

[9] Fuhao Shi, Hsiang-Tao Wu, Xin Tong, and Jinxiang Chai. Automatic acquisition of high-fidelity facial performances using monocular videos. *ACM Transactions on Graphics (TOG)*, 33(6):222, 2014.

[10] Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. Facewarehouse: A 3d facial expression database for visual computing. *Visualization and Computer Graphics, IEEE Transactions on*, 20(3):413–425, 2014.

[11] Lance Williams. Performance-driven facial animation. In *ACM SIGGRAPH Computer Graphics*, volume 24, pages 235–242. ACM, 1990.

[12] Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W Sumner, and Markus Gross. High-quality passive facial performance capture using anchor frames. *ACM Transactions on Graphics (TOG)*, 30(4):75, 2011.

[13] Mark Sagar. Facial performance capture and expressive translation for king kong. In *ACM SIGGRAPH 2006 Sketches*, page 26. ACM, 2006.

[14] Wan-Chun Ma, Andrew Jones, Jen-Yuan Chiang, Tim Hawkins, Sune Frederiksen, Pieter Peers, Marko Vukovic, Ming Ouhyoung, and Paul Debevec. Facial performance synthesis using deformation-driven polynomial displacement maps. In *ACM Transactions on Graphics (TOG)*, volume 27, page 121. ACM, 2008.

[15] Brian Guenter, Cindy Grimm, Daniel Wood, Henrique Malvar, and Fredric Pighin. Making faces. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 55–66. ACM, 1998.

[16] Bram Van Ginneken, Alejandro F Frangi, Joes J Staal, Bart M Romeny, and Max A Viergever. Active shape model segmentation with optimal features. *medical Imaging, IEEE Transactions on*, 21(8):924–933, 2002.

[17] Timothy F Cootes, Gareth J Edwards, and Christopher J Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):681–685, 2001.

[18] Piotr Dollár, Peter Welinder, and Pietro Perona. Cascaded pose regression. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1078–1085. IEEE, 2010.

[19] Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun. Face alignment at 3000 fps via regressing local binary features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1685–1692, 2014.

[20] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. Face alignment by explicit shape regression. *International Journal of Computer Vision*, 107(2):177–190, 2014.

[21] Xuehan Xiong and Fernando Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013.

[22] Xuehan Xiong and Fernando De la Torre. Global supervised descent method. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2664–2673, 2015.

[23] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886. IEEE, 2012.

[24] Georgios Tzimiropoulos and Maja Pantic. Gauss-newton deformable part models for face alignment in-the-wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2014.

[25] Golnaz Ghiasi and Charless Fowlkes. Occlusion coherence: Localizing occluded faces with a hierarchical deformable part model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2385–2392, 2014.

[26] Natasha Kholgade, Iain Matthews, and Yaser Sheikh. Content retargeting using parameter-parallel facial layers. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 195–204. ACM, 2011.

[27] Lun Zhang, Rufeng Chu, Shiming Xiang, Shengcai Liao, and Stan Z Li. Face detection based on multi-block lbp representation. In *Advances in biometrics*, pages 11–18. Springer, 2007.

[28] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.

[29] Tamara G Kolda and Jimeng Sun. Scalable tensor decompositions for multi-aspect data mining. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 363–372. IEEE, 2008.

[30] YI Abdel-Aziz. Direct linear transformation from comparator coordinates in close-range photogrammetry. In *ASP Symposium on Close-Range Photogrammetry in Illinois, 1971*, 1971.

[31] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[32] Kwang In Kim, Keechul Jung, and Hang Joon Kim. Face recognition using kernel principal component analysis. *Signal Processing Letters, IEEE*, 9(2):40–42, 2002.