Realtime Facial Animation with On-the-fly Correctives

Hao Li* Jihun Yu† Yuting Ye‡ Chris Bregler§

Industrial Light & Magic

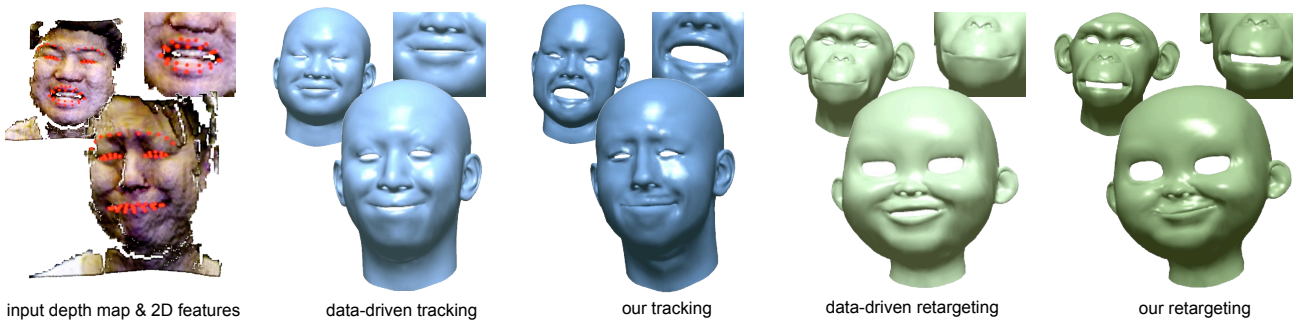

Figure 1: *Our adaptive tracking model conforms to the input expressions on-the-fly, producing a better fit to the user than state-of-the-art data driven techniques [Weise et al. 2011] which are confined to learned motion priors and generate plausible but not accurate tracking.*

Abstract

We introduce a real-time and calibration-free facial performance capture framework based on a sensor with video and depth input. In this framework, we develop an adaptive PCA model using shape correctives that adjust on-the-fly to the actor's expressions through incremental PCA-based learning. Since the fitting of the adaptive model progressively improves during the performance, we do not require an extra capture or training session to build this model. As a result, the system is highly deployable and easy to use: it can faithfully track any individual, starting from just a single face scan of the subject in a neutral pose. Like many real-time methods, we use a linear subspace to cope with incomplete input data and fast motion. To boost the training of our tracking model with reliable samples, we use a well-trained 2D facial feature tracker on the input video and an efficient mesh deformation algorithm to snap the result of the previous step to high frequency details in visible depth map regions. We show that the combination of dense depth maps and texture features around eyes and lips is essential in capturing natural dialogues and nuanced actor-specific emotions. We demonstrate that using an adaptive PCA model not only improves the fitting accuracy for tracking but also increases the expressiveness of the retargeted character.

CR Categories: I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: facial animation, real-time tracking, performance capture, facial retargeting, incremental PCA, correctives, depth sensor

Links: DL PDF

1 Introduction

The essence of high quality performance-driven facial animation is to capture every trait and characteristic of an actor's facial and verbal expression and to reproduce those on a digital double or creature. Even with the latest 3D scanning and motion capture technology, the creation of realistic digital faces in film and game production typically involves a very complex pipeline requiring intensive manual intervention. Long turn-around times are usually required for generating compelling results, resulting in high production costs. Consequently, the exploration of real-time facial performance capture as pre-visualization has gained increasing attention to help directors plan shots more carefully, animators quickly experiment with face models, and actors get into their characters when driving a virtual avatar. For all these applications, it is desirable to use a low impact and easily deployable acquisition setup, since performance capture often needs to be on-location, in an everyday environment, or even at an animator's desk.

While 2D video systems are often considered the most common and flexible solution, real-time 3D sensors such as Microsoft's Kinect have the ability to capture dense depth input data, while being robust to illumination changes and occlusions. For real-time facial tracking, linear models such as *blendshapes* or *PCA models* are often preferred due to their level of expressiveness and their compact representation for efficient processing. However, creating a linear model that can span the full spectrum of facial expressions for a specific person would require a large collection of expression measurements [Ekman and Friesen 1978] or a lengthy training session [Weise et al. 2009]. To improve deployability, data-driven

*e-mail:hao@hao-li.com (currently also with USC)
†e-mail:jihun.yu@gmail.com
‡e-mail:yuting.ye@gmail.com
§e-mail:chris.bregler@gmail.com (currently also with NYU)

techniques are often used to reduce the amount of pre-processing at the cost of synthesized reconstructions from priors that may look believable but not necessarily capture the characteristics of the specific actor [Li et al. 2010; Weise et al. 2011].

We propose a real-time facial animation framework where an *adaptive PCA model*, based on *correctives* (see Section 5), rapidly adapts to the expressions of the performing actor during the tracking. The process begins with an initial 3D scan of the subject in a neutral pose. From there, a customized digital model and an initial set of generic linear blendshape expressions are automatically generated. We track the face by solving for the best fit to the input data using these *generic blendshapes*. This step is followed by fitting a refined linear model onto the input data using *adaptive PCA shapes*. The adaptive PCA model consists of *anchor shapes* and *corrective shapes*. The anchor shapes are derived from the initial blendshapes to prevent the tracking model from converging to a bad model (drifting) due to noisy input data. The corrective shapes are used to learn the distinct look and expressions of the actor during tracking. The adaptive PCA model can therefore capture facial expressions that cannot be represented by the initial blendshapes. To train the correctives, we warp the result of the adaptive PCA fit to the current input depth map to discover new shapes that are outside of the adaptive PCA space. This tight *out-of-adaptive space* deformation uses 3D depth map and 40 2D facial features (lip contours, eye contours, and eyebrows). Our 2D facial features are obtained from *Live Driver*, an off-the-shelf real-time feature tracking technology from Image Metrics [ImageMetrics 2012]. Live Driver implements a data-driven tracking algorithm that works on any individual, under any lighting. While no calibration is needed for Live Driver, capturing a single neutral pose further improves tracking accuracy. The optimization of our adaptive PCA space is obtained through an incremental PCA learning approach based on the expectation-maximization (EM) algorithm of Roweis [1998].

Our depth sensor-based performance capture system (see Figure 2) requires no training phase and can be instantly used once a neutral expression has been captured. Whether in communication applications or interactive games involving virtual avatars, our system offers a general ease-of-use tool which makes it readily deployable, while *accurate* tracking is ensured with the on-the-fly correctives. In a more professional setting such as film production, actors may choose to use this algorithm as a calibration tool to build an optimal tracking model for their faces without going through a large and challenging set of prescribed facial action units—and immediately test the tracking quality. Since we train our model with fine-scale deformations around eyes and mouth regions, our framework is particularly effective in recovering emotions, conversations, and subtle nuances of actor-specific expressions. While this paper focuses on improving the tracking quality and performance capture workflow for a real-time system, we also demonstrate that more accurate and expressive facial retargeting can be achieved.

Contribution. Our real-time markerless facial animation framework can be instantly used by any subject—without training—and ensures accurate tracking using an adaptive PCA model based on correctives that adjusts to the user's expressions on-the-fly.

2 Related Work

Facial performance capture is a fundamental thread in computer animation, often involving many application-specific technologies for acquisition, modeling, tracking, and retargeting [Pighin and Lewis 2006]. This paper focuses mainly on the tracking problem, since we aim at capturing actor-distinct motions in a low impact setting. Still popular in the visual effects and gaming industry, many early methods involve marker dots for tracking due to their reliability [Williams 1990; Guenter et al. 1998]. Besides being limited to production environments, markers are typically sparsely distributed, requiring manual corrections or physical deformation priors [Bickel et al. 2008] to reintroduce the fine-scale dynamics.

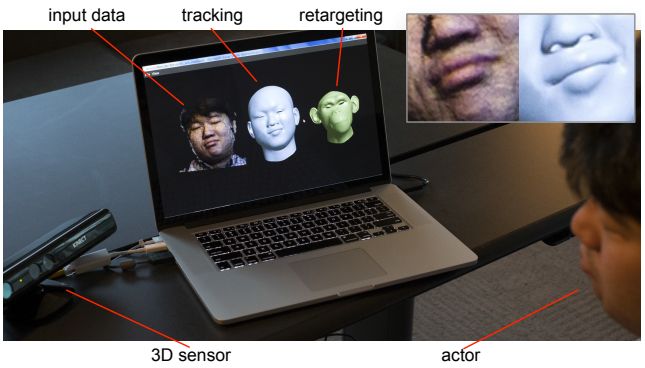

Figure 2: *Our system uses a Kinect depth sensor and runs in real-time on a consumer level computer.*

Purely video-based facial tracking systems that do not require any markers are aimed at fully uncontrolled environments where a 2D parametric shape model is used [Li et al. 1993; Bregler and Omohundro 1994; Black and Yacoob 1995; Essa et al. 1996; Bregler et al. 1997; Pighin et al. 1999; Decarlo and Metaxas 2000]. Although 2D facial tracking remains difficult, data-driven algorithms were introduced to achieve real-time performance. These include active appearance model [Cootes et al. 1998], Eigen points [Covell and Bregler 1996], the more recent landmark prediction techniques of Saragih and colleagues [Saragih et al. 2011], and the commercial solution, Live Driver, from Image Metrics [ImageMetrics 2012]. Even though video-based tracking can be used to drive 3D facial animations convincingly [Chuang and Bregler 2002; Chai et al. 2003], these methods only extract features around prominent regions (lips, nose, eyes, etc.) for a limited set of facial expressions. Detailed 3D facial tracking from video is generally achieved at the cost of a controlled environment, expensive computation, and the use of a carefully crafted 3D tracking model [Borshukov et al. 2005; Alexander et al. 2009; Vlasic et al. 2005].

Real-time 3D acquisition systems produce a continuous stream of high-resolution depth maps and can effectively capture fine geometric details such as wrinkles and folds. While high-quality 3D data can be obtained from multi-view or photometric stereo techniques [Bradley et al. 2010; Fyffe et al. 2011; Beeler et al. 2011; Valgaerts et al. 2012], active methods such as structured light [Rusinkiewicz et al. 2002; Zhang and Huang 2004] are more robust in general environments and can generate 3D points at interactive rates. To capture facial deformations, a preconstructed 3D template model is warped through a sequence of 3D input scans. Correspondences are computed using variants of non-rigid registration and texture tracking [Zhang et al. 2004; Li et al. 2009; Furukawa and Ponce 2009; Bradley et al. 2010; Beeler et al. 2011] or more recently with a joint optimization between capture and tracking [Valgaerts et al. 2012].

The ability to track detailed facial expressions from a 3D sensor in real-time has been demonstrated by Weise and coworkers [2009] using a linear PCA subspace that has been trained with a very large set of pre-processed facial expressions. Since an extended training session with a careful choice of facial action units is required for every actor, the system is less suitable for the general audience. Moreover, additional training sessions may be required to achieve satisfactory tracking quality. To reduce the amount of training and enable retargeting, Li and colleagues [2010] introduced an example-based blendshape optimization technique that only requires a limited number of random facial expressions as input to generate a full set of FACS blendshape expressions [Ekman and Friesen 1978]. While geometric details are recovered from the in-

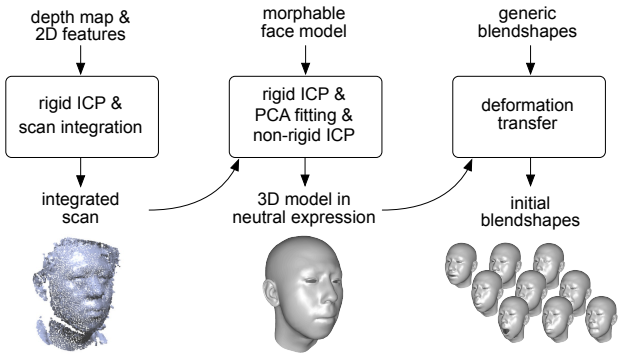

Figure 3: *Construction of initial blendshapes using morphable face models for the neutral expression. We use deformation transfer to produce FACS-based generic expressions.*

put training examples, the interpolated expressions often lack the subtle traits of the actor. As shown in the depth sensor-based facial animation pipeline of [Weise et al. 2011], the use of example-based blendshapes in combination with data-driven animation priors often result in poor tracking accuracy (especially around mouth and eye regions). While this capture method is sufficient to produce plausible results in the context of puppeteering, it fails to accurately capture the lip movements and high-frequency geometric details needed to convey nuanced emotions and micro-expression.

The general idea of updating a tracking model during a performance has been explored in the field of computer vision for tracking general objects. Collins and colleagues [2005] proposed a method that updates color features on-the-fly to improve object-to-background discrimination. Grabner and coworkers [2008] proposed a more elaborate feature selection scheme for on-line boosting; and [Kalal et al. 2009] demonstrated growing discriminative template databases during tracking. All these techniques focus on general 2D object detection problems with a level of accuracy which falls short of our requirements. While a pre-trained PCA subspace can enable real-time face tracking, constructing such a PCA model from a large database using traditional SVD is impractically slow [Pearson 1901; Kirby and Sirovich 1990]. There exist many incremental variants of the PCA algorithm that do not require a full PCA recomputation for a stream of incoming data, such as [Gu and Eisenstat 1993], [Chandrasekaran et al. 1997], and [Skocaj and Leonardis 2003]. We use an EM-based approach derived from [Roweis 1998] because of its fast convergence.

3 Pipeline Overview

Capturing the Neutral Face. Our pipeline begins with the construction of a *3D model* from a face scan in neutral expression (see Figure 3). Similar to [Weise et al. 2011], we first aggregate multiple input depth map frames using a rigid alignment based on the fast iterative closest point method (ICP) [Rusinkiewicz and Levoy 2001] and volumetric integration [Rusinkiewicz et al. 2002]. We obtain a merged 3D point cloud with better coverage of the neutral face. We then warp the model of a statistically average face onto the integrated scan using a linear fit of PCA modes obtained from 200 human subjects as described in [Blanz and Vetter 1999; Paysan et al. 2009]. The fitting consists of an optimization that solves for both the global rigid transformation and the PCA coefficients. To capture details that are not present in the PCA model, we shrink-wrap the resulting model onto the input scan using the non-rigid ICP algorithm of Li and coworkers [2009]. For both deformable alignment steps, PCA and non-rigid ICP, we use point-to-plane constraints on the input scans and point-to-point 2D feature constraints (lips, eyes, and eyebrows) obtained from Live Driver [ImageMetrics 2012].

Building Initial Blendshapes. Once the neutral face model of the actor is reconstructed, we automatically generate a set of initial blendshapes from a collection of 23 FACS inspired generic expressions using the deformation transfer algorithm of Sumner and colleagues [2004] (see Figure 3). The initial blendshapes are personalized but crude approximations of the actor's real expressions.

Tracking with Blendshapes and Adaptive PCA. After creating the initial blendshapes of the actor, we begin tracking the actor's face (see Figure 4). We first solve for a global rigid transformation using fast rigid ICP [Rusinkiewicz and Levoy 2001] as before; we then perform an *initial blendshape fit* for every input frame using both 3D point constraints on the input scans and 2D facial features [ImageMetrics 2012]. The fitting is refined using a Laplacian deformation algorithm [Botsch and Sorkine 2008] with the same constraints, followed by a projection onto an adaptive PCA space since the input data are noisy and incomplete. The adaptive PCA space is an orthonormal basis and consists of A *anchor shapes* and K additional *corrective shapes*.

Training the Correctives. We initialize the anchor shapes with $A = 23$ orthonormalized vectors from the initial blendshapes and learn K corrective shapes to improve the fitting accuracy over time. To train the correctives, we first collect new expression samples that fall outside of the currently used adaptive PCA space. These samples are obtained by warping the result of the initial blendshape fit to fit the current input depth map and 2D facial features using again a per-vertex Laplacian deformation algorithm. These samples are used to refine the corrective shapes using the incremental PCA technique described in Section 5.

Expression Retargeting. The initial blendshape coefficients that are solved in the beginning could be immediately used for expression retargeting. However, we found that more expressive retargeting is obtained by re-solving for blendshape coefficients using the final mesh output.

4 Real-time Tracking

The first step of our tracking pipeline (see Figure 4) consists of rigidly aligning the tracked 3D model of the previous frame to the current input frame. Next, we fit the initial blendshapes using 3D depth map and 2D facial feature constraints. The resulting blendshape fit is represented by a mesh with vertices $\mathbf{v}^1 = \mathbf{b}_0 + \mathbf{B}\mathbf{x}$, where $\mathbf{b}_0$ is the neutral expression mesh, the columns of $\mathbf{B}$ are the FACS-based expression meshes, and $\mathbf{x}$ forms the blendshape coefficients. We further refine the result using our progressively updated adaptive PCA model. This fitting is divided in two stages. First, we perform a Laplacian deformation using the same input constraints and obtain the mesh with vertices $\mathbf{v}^2 = \mathbf{v}^1 + \Delta\mathbf{v}^1$, where $\Delta\mathbf{v}^1$ are the per vertex displacements. The Laplacian deformed mesh is then projected to the adaptive PCA model producing the mesh $\mathbf{v}^3 = \left(\mathbf{M}\mathbf{M}^\top(\mathbf{v}^2 - \mathbf{b}_0)\right) + \mathbf{b}_0$, where the columns of $\mathbf{M}$ are the bases of the adaptive PCA model. Our final output is an additional Laplacian deformation on top of the adaptive PCA result and we obtain $\mathbf{v}^4 = \mathbf{v}^3 + \Delta\mathbf{v}^3$. Since depth maps are noisy, we only use 2D facial features as constraints. However, for the training samples of the incremental PCA algorithm, we do use the 3D depth constraints in addition to 2D features. In practice, we use $\mathbf{v}^2$ as training samples. Each stage is designed for real-time performance as well as resistance to large occlusions and outliers.

4.1 Rigid Motion Tracking

To reduce the amount of data captured by the depth map, we first crop the input frame with a face bounding window (200×200 pix-

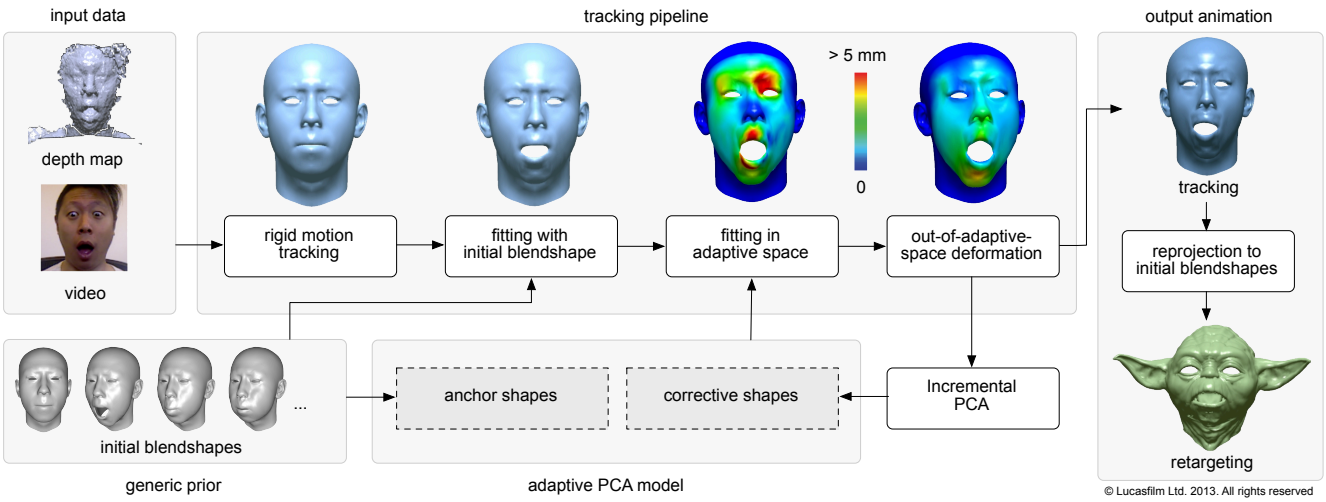

Figure 4: *Our real-time facial animation pipeline based on an adaptive tracking model. The heat map of the fitting in adaptive space shows the difference between our adaptive PCA model to the initial blendshape fit. The color of the out-of-adaptive-space deformation illustrates the difference between the tracking output and the current adaptive PCA model. This error is decreased with every incoming training sample.*

els) centered around the tracked model from the previous frame. The initial face detection is obtained from the 2D bounding box of the facial features and re-detection is only performed whenever the face model is lost (i.e., when convergence fails during rigid motion tracking). We then track the result using the fast projection variant of the rigid ICP algorithm with the point-to-plane metric [Rusinkiewicz and Levoy 2001]. We prune correspondences that are 25 mm away. Rigid ICP finishes when the average distance of unpruned correspondences to the depth map is below 3 mm.

4.2 Fitting with Initial Blendshape

Similar to ICP, we fit the linear blendshape model to the input scans by alternating between finding per-vertex correspondences and solving for blendshape coefficients. Let $\mathbf{v}_i^1(\mathbf{x}) = (\mathbf{b}_0 + \mathbf{B}\mathbf{x})_i$ be the i-th blendshape mesh vertex, $\mathbf{b}_0$ the neutral expression mesh, the columns of $\mathbf{B}$ the $A = 23$ meshes of FACS-based expressions, and $\mathbf{x}$ the blendshape coefficients. We then use the following *point-to-plane fitting term*:

$$c_i^S(\mathbf{x}) = \mathbf{n}_i^\top (\mathbf{v}_i^1(\mathbf{x}) - \mathbf{p}_i)\,, \tag{1}$$

where $\mathbf{p}_i$ is the point of the depth map that has the same camera space coordinate as $\mathbf{v}_i^1$, and $\mathbf{n}_i$ the surface normal of $\mathbf{p}_i$. We use point-to-plane instead of point-to-point constraints for more robust convergence in the optimization.

We pre-associate the 40 sparse 2D facial features to a fixed set of mesh vertices of our tracked 3D model. We then formulate the *facial feature fitting term* as vectors between the 2D facial features and their corresponding mesh vertices in camera space:

$$\mathbf{c}_j^F(\mathbf{x}) = \begin{bmatrix} 1 & 0 & -u_j^x \\ 0 & 1 & -u_j^y \end{bmatrix} \mathbf{P}\,\mathbf{v}_j^1(\mathbf{x})\,, \tag{2}$$

where $\mathbf{u}_j = [u_j^x, u_j^y]^\top$ is the j-th 2D facial feature position and $\mathbf{P}_{3\times3}$ the camera projection matrix.

We solve for the blendshape coefficients $\mathbf{x} = [x_1 \ldots x_A]^\top$ using the terms from Equation (1) and (2) and minimize the following L^2 energy:

$$\min_{\mathbf{x}} \sum_i (c_i^S(\mathbf{x}))^2 + w \sum_j \|\mathbf{c}_j^F(\mathbf{x})\|_2^2\,,$$

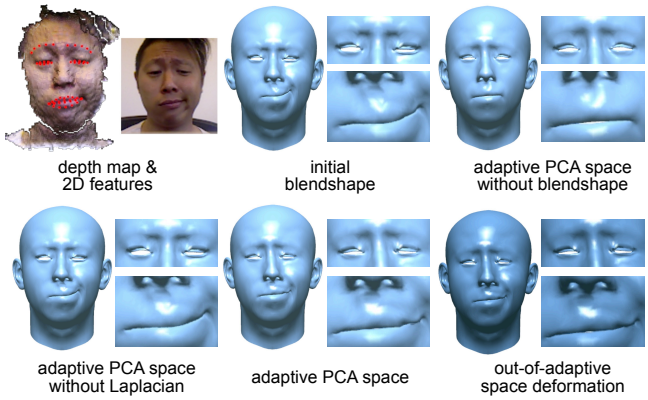

Figure 5: *Influence of each stage on the full tracking pipeline.*

where $w = 5 \cdot 10^{-5}$ is the weight of the facial feature constraints and $x_l \in [0,1]$ are the blendshape coefficients. We use the fast iterative projection method from Sugimoto and colleagues [1995] to solve the bounded dense linear system. Due to the localized nature of blendshapes and because the coefficients are bounded, large motions can be recovered very effectively as shown in Figure 5. We perform a fixed number of iterations between this optimization and the closest point search (in our case 3 times). We apply the standard Kalman filter [Welch and Bishop 1995] to blendshape coefficients to effectively reduce the noise caused by the depth-sensor data. The linear state variable for predicting the position and velocity has a process noise of 0.05 and measurement noise of 0.02.

4.3 Fitting in Adaptive PCA Space

We refine the tracking by warping the blendshape model to a linear adaptive PCA model whose space is spanned by the $A + K$ PCA shapes which are stacked to form a matrix $\mathbf{M} = [\mathbf{M}_A, \mathbf{M}_K]$. The A columns of $\mathbf{M}_A$ are the orthonormalized vectors of the initial blendshapes and form the *anchor blendshapes*. The K columns of $\mathbf{M}_K$ lie in the null space of $\mathbf{M}_A$ and represent the *corrective shapes* $\mathbf{M}_K$. The correctives capture the fine-scale details, not present in the initial blendshape model. Bounded linear optimization as used with the initial blendshape fitting is not possible with the adaptive PCA shapes because the deformations are now global (each PCA mode

affects the entire face). Our solution is to use a Laplacian deformation to establish reliable correspondences between the tracked model and the input scan before projecting the solution to the adaptive PCA space.

Fast Laplacian Deformation. We perform a Laplacian deformation [Botsch and Sorkine 2008] $\mathbf{v}_i^2 = \mathbf{v}_i^1 + \Delta\mathbf{v}_i^1$ using both depth and 2D facial feature constraints. The Laplacian smoothing term regularizes the vertex displacements $\Delta\mathbf{v}_i^1$ constrained by the sparse 2D features, and also reduces the spatial high-frequency noise introduced by the 3D depth sensor. The deformation optimization solves for all the vertices that are in the frontal part of the head model subject to Dirichlet boundary conditions.

Even though the linear systems of Laplacian deformations are sparse, matrix factorization such as Cholesky decomposition can still be problematic for real-time performance (our tracking meshes have 6918 vertices where the frontal 3480 are used for deformation). We therefore rely on optimization constraints where pre-factorization is possible (i.e., the left hand sides of the sparse linear system remain constant over time). For instance, point-to-plane constraints are not suitable because the left hand side needs to be recomputed in each iteration.

We use a point-to-point fitting term between every mesh vertex and its corresponding projection into camera space:

$$\mathbf{c}_i^P(\Delta\mathbf{v}_i^1) = \Delta\mathbf{v}_i^1 - (\mathbf{p}_i - \mathbf{v}_i^1)\,, \tag{3}$$

and 2D feature terms using a weak projection formulation:

$$\mathbf{c}_j^W(\Delta\mathbf{v}_j^1) = \begin{bmatrix} 1 & 0 & -(u_j^x - v_j^{1,x}) \\ 0 & 1 & -(u_j^y - v_j^{1,y}) \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P}\,\Delta\mathbf{v}_j^1 - \begin{bmatrix} 0 \\ 0 \\ (p_j^z - v_j^{1,z}) \end{bmatrix}, \tag{4}$$

where $\mathbf{p}_j = [p_j^x, p_j^y, p_j^z]$ and $\mathbf{v}_j^1 = [v_j^{1,x}, v_j^{1,y}, v_j^{1,z}]$. We rephrase this term compactly as:

$$\mathbf{c}_j^W(\Delta\mathbf{v}_j^1) = \mathbf{H}_j(\mathbf{u}_j)\mathbf{P}\,\Delta\mathbf{v}_j^1 - \mathbf{d}_j\,. \tag{5}$$

We define cotangent weights [Botsch and Sorkine 2008] w.r.t. the neutral mesh for the Laplacian smoothing terms:

$$\mathbf{C}^L(\Delta\mathbf{v}^1) = \mathbf{L}(\mathbf{b}_0)\begin{bmatrix} \Delta\mathbf{v}_1^1 \\ \vdots \\ \Delta\mathbf{v}_N^1 \end{bmatrix} = \mathbf{L}(\mathbf{b}_0)\Delta\mathbf{v}^1\,. \tag{6}$$

We can stack Equation (3), (5), and (6) into a single over-constrained linear system.

$$\left[\begin{array}{ccc} \mathbf{H}_1 & & \\ & \ddots & \\ & \mathbf{H}_F & \\ \hline & w_1\mathbf{I} & \\ \hline & & w_2\mathbf{I} \end{array}\right] \begin{bmatrix} \mathbf{Q} \\ \mathbf{I} \\ \mathbf{L} \end{bmatrix} \Delta\mathbf{v}^1 = \mathbf{a}\,, \tag{7}$$

where $\mathbf{Q}$ is a $3F \times 3N$ matrix stacked from the projection matrix $\mathbf{P}$ from Equation (4), $\mathbf{I}$ denotes a $3N \times 3N$ identity matrix, $w_1 = 0.1$ is the weight for the point-to-point constraints, $w_2 = 100$ is the weight for the Laplacian regularization constraint, and $\mathbf{a}$ contains all the constant terms from the constraints. The above system can be rewritten as $\mathbf{GK}\Delta\mathbf{v}^1 = \mathbf{a}$, where the least-square solution can be readily computed using the Moore-Penrose pseudoinverse:

$$\Delta\mathbf{v}^1 = \mathbf{K}^\top(\mathbf{K}\mathbf{K}^\top)^{-1}\mathbf{G}^{-1}\mathbf{a}\,.$$

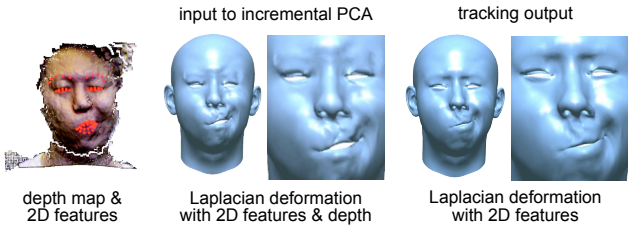

Figure 6: *Out-of-adaptive-space deformation for tracking output and input to incremental PCA.*

Because $\mathbf{K}$ is sparse and constant, we only need to pre-factorize $\mathbf{K}\mathbf{K}^\top$ once. Moreover, $\mathbf{G}^{-1}$ can be trivally computed thanks to its square and sparse structure. Therefore, we can solve the entire linear system efficiently. We use the sparse LDLT solver from the C++ Eigen Library for the pre-factorization.

PCA Subspace Projection. The second step consists of projecting the Laplacian deformed mesh onto the continuously improving adaptive PCA space. The PCA subspace projection is formulated as follows:

$$\mathbf{v}^3 = \left(\mathbf{M}\mathbf{M}^\top(\mathbf{v}^2 - \mathbf{b}_0)\right) + \mathbf{b}_0\,,$$

where $\mathbf{v}^3 = [\mathbf{v}_1^3 \ldots \mathbf{v}_N^3]^\top$ and $\mathbf{v}^2 = [\mathbf{v}_1^2 \ldots \mathbf{v}_N^2]^\top$. In particular, $\mathbf{v}^3 = \mathbf{M}\mathbf{y} + \mathbf{b}_0$, where $\mathbf{y} = [y_1 \ldots y_A, y_{A+1} \ldots y_{A+K}]^\top$ are the resulting adaptive PCA coefficients. This projection is necessary because the mesh obtained from the Laplacian deformation may contain outliers and visible artifacts due to the noisy and incomplete input depth map. The adaptive PCA model however is trained from a large number of samples that are not outliers. Moreover, the accumulation of multiple frames during incremental PCA also averages out the artifacts caused by incomplete input data.

4.4 Out-of-adaptive-space Deformation

As illustrated in Figure 4, we use two additional warping steps to generate out-of-adaptive-space deformations: (1) to feed the incremental PCA algorithm (Section 5) with reliable data and (2) to generate the final output mesh. To train the adaptive PCA model, we could deform the mesh $\mathbf{v}^3$ to fit the current frame using the Laplacian deformation by solving the system defined by Equation (7) where $w_1 = 0.1$ and $w_2 = 100$. In practice, the resulting mesh would be very similar to $\mathbf{v}^2$, so we use $\mathbf{v}^2$ directly to save the cost of an extra computation of Laplacian deformation. While the results from the adaptive PCA space are already superior to those obtained from the initial blendshapes, we further refine the accuracy using an additional Laplacian deformation with only the 2D facial features as constraints and obtain $\mathbf{v}^4 = \mathbf{v}^3 + \Delta\mathbf{v}^3$ by setting $w_1 = 0$ (or simply eliminating the corresponding rows) and $w_2 = 100$ (see Figure 6).

5 Incremental PCA

We use the resulting out-of-adaptive-space meshes from Section 4.4 as input data to train the adaptive PCA model for improved tracking accuracy (see Figure 7). As mentioned previously, the adaptive model $\mathbf{M} = [\mathbf{M}_A, \mathbf{M}_K] = [\mathbf{m}_1 \ldots \mathbf{m}_A, \mathbf{m}_{A+1} \ldots \mathbf{m}_{A+K}]$ consists of anchor and corrective shapes. The anchor shapes prevent the adaptive tracking model from drifting and are computed as the principle components of the initial blendshapes; hence $\mathbf{B}\mathbf{B}^\top = \mathbf{M}_A\mathbf{D}\mathbf{M}_A^\top$.

For every incoming out-of-adaptive-space mesh $\mathbf{v}^2$, the incremental PCA algorithm verifies that the sample is "*valid*" (see definition below) before updating $\mathbf{M}_K$. Because we only want to update the

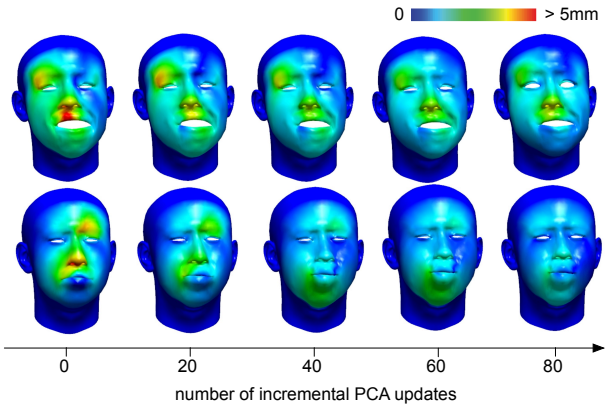

Figure 7: *The effect of correctives during tracking with continuously updated adaptive PCA model. The heat map shows the difference between the final tracking and the adaptive PCA model. The error decreases rapidly when new expressions are learned.*

corrective space spanned by $\mathbf{M}_K$, we define the input samples $\mathbf{s} = (\mathbf{v}^2 - \mathbf{b}_0) - (\mathbf{M}_A\mathbf{M}_A^\top)(\mathbf{v}^2 - \mathbf{b}_0)$ as the projected residuals onto the anchor space. A sample is considered "*valid*" if it is sufficiently far away from the anchor space $\mathbf{M}_A$, (i.e., $\|\mathbf{s}\|_2^2 > 1$) and if it is not an outlier (i.e., $\|\mathbf{s}\|_2^2 < 100$).

Because a continuous computation of the correctives cannot be achieved in real-time using standard PCA, we use the iterative EM algorithm of [Roweis 1998] which progressively approximates a solution of $\mathbf{M}_K$ given a collection of new valid samples $\mathbf{S}$. The samples are collected with a buffer $\mathbf{S} = [\mathbf{s}_1 \dots \mathbf{s}_S]$, where $S = 200$. The first incremental PCA update happens once the buffer is full; then $\mathbf{M}_K$ is recomputed for every *valid* incoming sample.

We initialize $\mathbf{M}_K$ with the first K *valid* samples and orthogonalize them via standard QR factorization. Because the samples are already orthogonal to $\mathbf{M}_A$, $\mathbf{M}$ is semi-orthonormal; the columns are normalized and orthogonal to each other.

The EM algorithm [Dempster et al. 1977] is generally used to estimate probabilistic models with hidden states, such as Hidden Markov Models, Mixture of Gaussians, or Mixture of Experts. In our case, the *hidden state variables* are the coefficients $\mathbf{Y}_K = [\mathbf{y}_1 \dots \mathbf{y}_S]$ of our model represented by the corrective shapes $\mathbf{M}_K$. EM iterates between an E-step (to find the "E"xpected value of the hidden state, given a model guess) and an M-step (to "M"aximize the expected model likelihood given the hidden states). In particular, [Roweis 1998] proves that the iterative estimations of $\mathbf{M}_K$ converge to the true PCA solution using the EM algorithm. We sketch out the EM algorithm for our domain:

1. **E-step.** compute the corrective space coefficients $\mathbf{Y}_K$ from the input samples $\mathbf{S}$ given a guess of the corrective shapes $\mathbf{M}_K$:

$$\mathbf{Y}_K = (\mathbf{M}_K^\top \mathbf{M}_K)^{-1} \mathbf{M}_K^\top \mathbf{S}\,.$$

2. **M-step.** update the corrective shapes $\mathbf{M}_K$ from the input samples $\mathbf{S}$ given the corrective space coefficients $\mathbf{Y}_K$:

$$\mathbf{M}_K = \mathbf{S}\mathbf{Y}_K^\top (\mathbf{Y}_K \mathbf{Y}_K^\top)^{-1}\,.$$

We repeat the above EM steps twice and use QR factorization again to orthonormalize $\mathbf{M}_K$ (EM does not have any orthonormal constraints). The resulting $\mathbf{M}_K$ replaces the old corrective shapes in $\mathbf{M}$. When the dimension K of the corrective space is too small, our algorithm fails at capturing fine-scale actor-specific details. If K is too large, undesired high-frequency noise can be learned into the adaptive PCA model. We use $K = 7$ for all our examples. In general the buffer size S should be chosen as large as possible, as it keeps a longer history of collected samples; but if S is too large, the algorithm slows down.

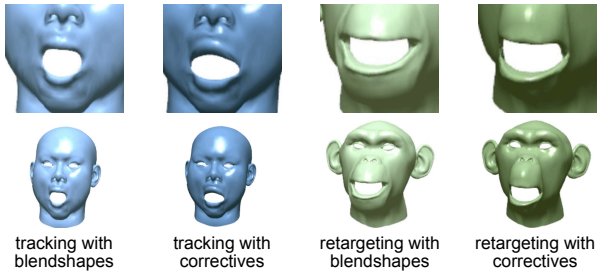

Figure 8: *We transfer blendshape coefficients for retargeting.*

6 Facial Retargeting

While our first initial blendshape result provides blendshape coefficients that can be used for facial retargeting, re-solving for the blendshape coefficients using the optimization in Section 4.2 and the final output mesh vertices as constraints inevitably improves the retargeting quality. Mapping back to blendshape space is necessary for retargeting since the adaptive tracking model lies in the PCA space. More elaborate techniques such as example-based facial rigging [Li et al. 2010] could be used, but our system would no longer be calibration free. The extracted blendshape coefficients $\mathbf{x}$ are simply transferred to a compatible blendshape model of a target character (see Figure 8).

7 Results

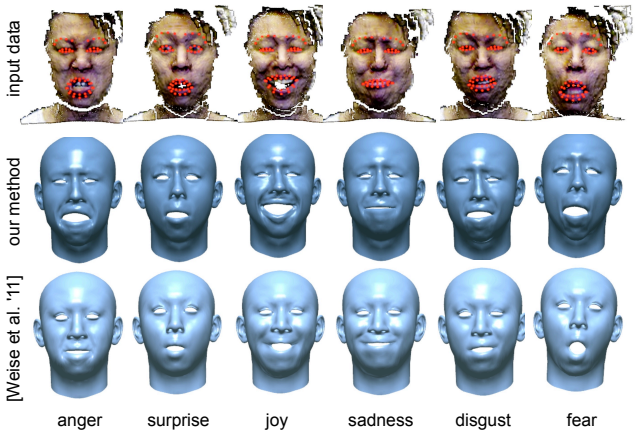

Figure 9: *Even though no training is involved, our adaptive model captures emotions more accurately than [Weise et al. 2011]*

Evaluation. Figure 5 compares the result of the full tracking pipeline to that obtained with individual steps omitted. All stages capture more details than a pure blendshape fit. Also see the benefit of using an initial blendshape fit to improve the optimization of the adaptive PCA fitting stage. When the Laplacian term is left out during this optimization, the 2D facial feature constraints become less effective. The final out-of-adaptive space deformation can produce expressions that are not yet trained in the adaptive PCA model. Figure 6 shows the two versions of the out-of-adaptive space deformation. The warp for the tracking output uses only 2D feature con-

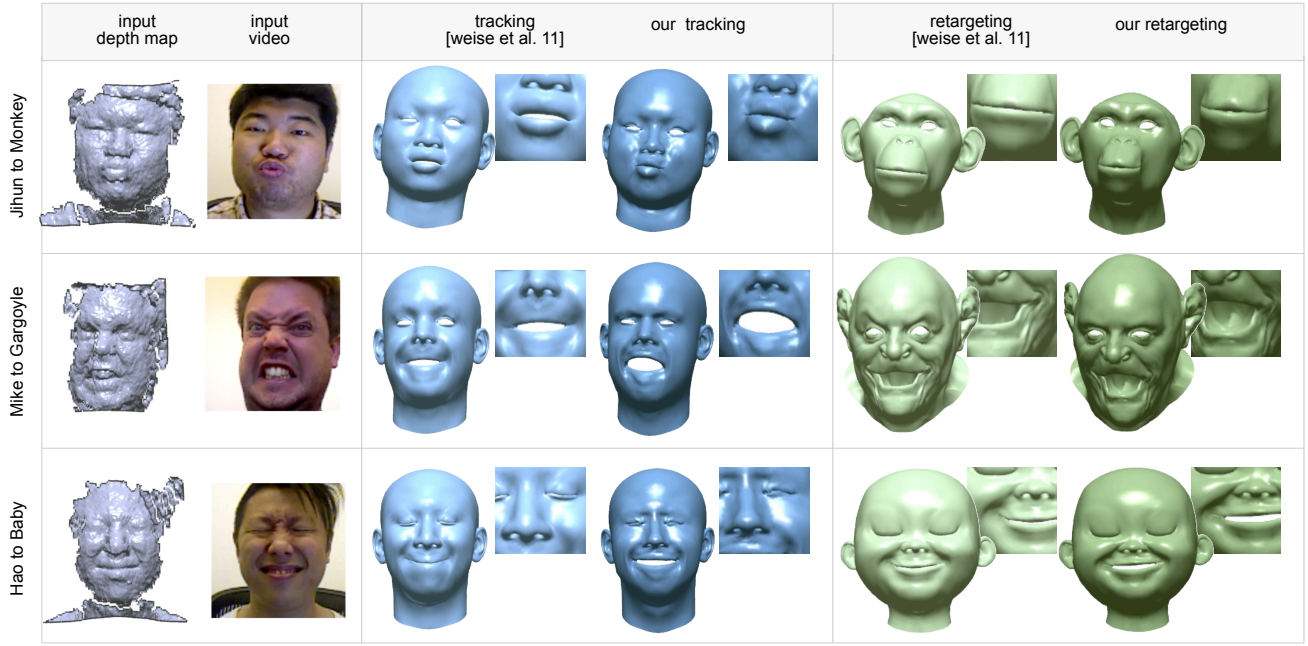

Figure 10: *Additional results and comparisons of our real-time facial animation system.*

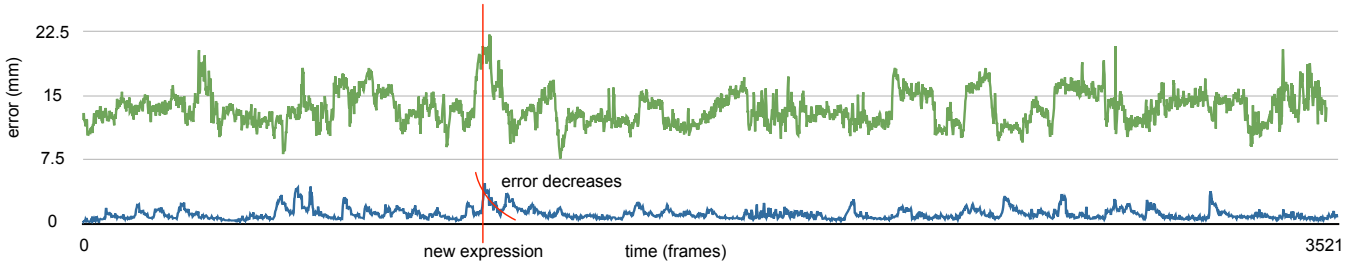

Figure 11: *The vertices* $\mathbf{v}^4$ *of the out-of-adaptive space deformation are the final facial tracking output. The average distance between these vertices and the anchor shape space* $\mathbf{M}_A$ *is shown in green; and the full adaptive PCA space* $\mathbf{M}$ *in blue. We show that tracking is more accurate with our adaptive tracking model than with initial blendshapes. Whenever a new expression occurs, the error drops rapidly (red).*

straints and contains less high-frequency noise as opposed to the one for the incremental PCA, which also uses depth data.

The fast convergence of our incremental PCA algorithm is demonstrated in Figure 11. We observe that very few updates are necessary to produce a good fit between the adaptive PCA model and the input scans. Also shown in the accompanying video, Figure 7 visualizes the evolution of the correctives for each update on two examples. Whenever the actor performs a new expression that could not be captured by the current adaptive PCA model, we measure a large error between the adaptive PCA fit and final tracking. This error decreases rapidly as new training samples are provided to our incremental PCA algorithm. We also learn that a higher number of anchor shapes improve the fitting accuracy since a more faithful shape prior is introduced.

Comparison. We show that our real-time facial tracking algorithm can adapt to a wide variety of actors and produce retargeting results that match the real performance more accurately than the current state-of-the-art method [Weise et al. 2011] (see Figure 1 and 10). The adaptive PCA model learns all the fine-scale details after only a few updates, faithfully capturing and reproducing emotional actor-specific expressions and subtle lip movements. Figure 9 compares both methods on the six basic human emotions and shows that an accurate tracking model is essential to capture these characteristics. We refer to the accompanying video to best appreciate the dynamics and geometric nuances captured by our method.

While our method does not require any training except for a single neutral face scan, the example-based facial rigging [Li et al. 2010] step of [Weise et al. 2011] requires at least 7 input examples to build a reasonable blendshape for tracking. Their entire process takes around 5 minutes, involving some manual corrections to adjust the non-rigid alignment. Our method can be used instantly and the full range of expressions can be performed in less than a few seconds without any manual intervention.

Performance. For every input frame, rigid motion tracking takes 3.2 ms, fitting with initial blendshapes 3 ms, fitting in adaptive space 9 ms, and each out-of-space deformation 4.2 ms. One update of incremental PCA (two iterations of EM steps) takes 3 ms. Our code does not involve GPU computation, but is multi-threaded on CPU. Timings (including I/O operations) are executed on a 2.6 Ghz quad-core Intel Core i7 with 16GB RAM (2012 MacBook Pro). We obtained slightly more stable results using the short range Primesense Carmine 1.09 depth sensor than with Microsoft's Kinect.

8 Discussion

Compared to existing state-of-the-art techniques, our real-time facial animation system achieves *superior tracking fidelity* and *does not require expression calibration* since our adaptive tracking model rapidly personalizes to different actors on-the-fly.

We have shown that combining 3D depth maps and sparse 2D facial features is essential for accurate tracking using our adaptive PCA model. Realistic real-time facial performance capture for high-end pre-visualization in feature films and game production becomes possible. Previously, actor specific emotions and natural dialogues could only be achieved with considerable manual corrections by an artist, or with more sophisticated acquisition systems. Because of the increased stability for real-time retargeting, our system is also suitable for ADR sessions or actors to practice their performances with interactive visual feedback of their digital characters.

Broader Impact. Our method allows the actor to use the system instantly once a neutral face pose has been captured. As a result, our technology can be immediately applied to general consumer applications such as virtual avatars in communication or performance-driven gaming experiences. Our technique also opens up new domains for capturing non-cooperating subjects that refuse to comply with any training protocols, such as babies, patients with movement disorders, or rapidly changing users in public places such as kiosks, amusement parks, and other venues. Certain social and psychological studies, such as facial analysis for behavioral economics, even require the subject to be unaware of the capture process.

Limitations and Future Work. Our system still requires a single scan of the actor as input. Ideally, we would like to estimate the actor's neutral shape together with other expressions. We plan to investigate ways to automatically identify the expressions of the actor without the need of a reference neutral expression shape. Our system adds correctives but does not improve the meshes of the initial blendshapes which encode expression semantics. Consequently, we are still using the initial blendshape coefficients for transfer. The subtle details that are captured in corrective space are not transferred to the target character. If both source and target are human-like characters, one could combine blendshape coefficient transfer with per-vertex deformation transfer algorithms to retarget fine-scale details, such as folds and wrinkles. Because our system uses a vision-based 2D facial feature tracking algorithm, it works best when the actor is facing front to the sensor. When turning away, the 2D features may lose accuracy and introduce suboptimal data to the incremental PCA algorithm. We plan to build a 3D variant of the real-time facial feature tracker that is directly coupled to our 3D tracking model.

Acknowledgements

We thank the modelers: Gio Nakpil and Michael Koperwas for the gargoyle (Hag); Hieu Phan for the baby; Hiroki Itokazu for the monkey; and Lee Perry Smith for the generic tracking model. We also thank Thomas Vetter for providing the morphable face model database; Mike Jutan for the facial performance; Jonas Rabbe, Karen Wong, and Eric Dillinger for the user interface; Yoojin Jiang and Jens Fursund for the rendering; Florian Kainz for shooting the live footages; Image Metrics for their support for Live Driver; Cyrus Wilson, Etienne Vouga, and Chris Twigg for proofreading; and Zoran Kačić-Alesić, Kirk Haller, Steve Sullivan, and Kim Libreri for their support and supervision. This work is presented by Industrial Light & Magic, Lucasfilm Ltd. and dedicated to all the former members of the R&D group. **May the force be with you!**

References

ALEXANDER, O., ROGERS, M., LAMBETH, W., CHIANG, M., AND DEBEVEC, P. 2009. The digital Emily project: photo-real facial modeling and animation. In *ACM SIGGRAPH 2009 Courses*, 12:1–12:15.

BEELER, T., HAHN, F., BRADLEY, D., BICKEL, B., BEARDSLEY, P., GOTSMAN, C., SUMNER, R. W., AND GROSS, M. 2011. High-quality passive facial performance capture using anchor frames. *ACM Trans. Graph. 30* (August), 75:1–75:10.

BICKEL, B., LANG, M., BOTSCH, M., OTADUY, M. A., AND GROSS, M. 2008. Pose-space animation and transfer of facial details. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

BLACK, M. J., AND YACOOB, Y. 1995. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *Proceedings of the Fifth International Conference on Computer Vision*, IEEE Computer Society, Washington, DC, USA, ICCV '95, 374–.

BLANZ, V., AND VETTER, T. 1999. A morphable model for the synthesis of 3d faces. In *Proceedings of ACM Siggraph 99*, ACM Press/Addison-Wesley Publishing Co., 187–194.

BORSHUKOV, G., PIPONI, D., LARSEN, O., LEWIS, J. P., AND TEMPELAAR-LIETZ, C. 2005. Universal capture - image-based facial animation for "the matrix reloaded". In *ACM SIGGRAPH 2005 Courses*.

BOTSCH, M., AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14*, 1 (Jan.), 213–230.

BRADLEY, D., HEIDRICH, W., POPA, T., AND SHEFFER, A. 2010. High resolution passive facial performance capture. *ACM Trans. Graph. 29* (July), 41:1–41:10.

BREGLER, C., AND OMOHUNDRO, S. 1994. Surface learning with applications to lipreading. *Advances in neural information processing systems*, 43–43.

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: Driving visual speech with audio. In *Proceedings of Computer graphics and interactive techniques*.

CHAI, J.-X., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3d facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '03, 193–206.

CHANDRASEKARAN, S., MANJUNATH, B. S., WANG, Y.-F., WINKELER, J., AND ZHANG, H. 1997. An eigenspace update algorithm for image analysis. *CVGIP: Graphical Model and Image Processing*, 5, 321–332.

CHUANG, E., AND BREGLER, C. 2002. Performance driven facial animation using blendshape interpolation. Tech. rep., Stanford University.

COLLINS, R., LIU, Y., AND LEORDEANU, M. 2005. Online selection of discriminative tracking features. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 27*, 10, 1631–1643.

COOTES, T. F., EDWARDS, G. J., AND TAYLOR, C. J. 1998. Active appearance models. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Springer, 484–498.

COVELL, M., AND BREGLER, C. 1996. Eigen-points. In *Image Processing, 1996. Proceedings., International Conference on*, vol. 3, IEEE, 471–474.

DECARLO, D., AND METAXAS, D. 2000. Optical flow constraints on deformable models with applications to face tracking. *Int. J. Comput. Vision 38*, 2 (July), 99–127.

DEMPSTER, A., LAIRD, N., AND RUBIN, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1–38.

EKMAN, P., AND FRIESEN, W. 1978. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press, Palo Alto.

ESSA, I., BASU, S., DARRELL, T., AND PENTLAND, A. 1996. Modeling, tracking and interactive animation of faces and heads using input from video. In *Proceedings of the Computer Animation*, IEEE Computer Society, CA '96, 68–.

FURUKAWA, Y., AND PONCE, J. 2009. Dense 3d motion capture for human faces. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, IEEE, 1674–1681.

FYFFE, G., HAWKINS, T., WATTS, C., MA, W.-C., AND DEBEVEC, P. 2011. Comprehensive facial performance capture. In *Eurographics 2011*.

GRABNER, H., LEISTNER, C., AND BISCHOF, H. 2008. Semi-supervised on-line boosting for robust tracking. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, Springer-Verlag, Berlin, Heidelberg, ECCV '08, 234–247.

GU, M., AND EISENSTAT, S. C. 1993. A Stable and Fast Algorithm for Updating the Singular Value Decomposition. Tech. Rep. YALEU/DCS/RR-966, Yale University, New Haven, CT.

GUENTER, B., GRIMM, C., WOOD, D., MALVAR, H., AND PIGHIN, F. 1998. Making faces. In *Proceedings of SIGGRAPH '98*, ACM, 55–66.

IMAGEMETRICS. 2012. Image metrics live driver SDK http://www.image-metrics.com/livedriver/.

KALAL, Z., MATAS, J., AND MIKOLAJCZYK, K. 2009. Online learning of robust object detectors during unstable tracking. In *In International Conference on Computer Vision*.

KIRBY, M., AND SIROVICH, L. 1990. Application of the karhunen-loeve procedure for the characterization of human faces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 12*, 1, 103–108.

LI, H., ROIVAINEN, P., AND FORCHEIMER, R. 1993. 3-d motion estimation in model-based facial image coding. *IEEE Transactions on PAMI 15*, 6, 545–555.

LI, H., ADAMS, B., GUIBAS, L. J., AND PAULY, M. 2009. Robust single-view geometry and motion reconstruction. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009) 28*, 5.

LI, H., WEISE, T., AND PAULY, M. 2010. Example-based facial rigging. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2010) 29*, 3 (July).

PAYSAN, P., KNOTHE, R., AMBERG, B., ROMDHANI, S., AND VETTER, T. 2009. A 3d face model for pose and illumination invariant face recognition.

PEARSON, K. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11, 559–572.

PIGHIN, F., AND LEWIS, J. P. 2006. Performance-driven facial animation. In *ACM SIGGRAPH 2006 Courses*, ACM, New York, NY, USA, SIGGRAPH '06.

PIGHIN, F. H., SZELISKI, R., AND SALESIN, D. 1999. Resynthesizing Facial Animation through 3D Model-based Tracking. In *Proc. 7th International Conference on Computer Vision, Kerkyra, Greece*, 143–150.

ROWEIS, S. 1998. EM algorithms for pca and spca. In *in Advances in Neural Information Processing Systems*, MIT Press, 626–632.

RUSINKIEWICZ, S., AND LEVOY, M. 2001. Efficient variants of the icp algorithm. In *International Conference on 3-D Digital Imaging and Modeling*.

RUSINKIEWICZ, S., HALL-HOLT, O., AND LEVOY, M. 2002. Real-time 3D model acquisition. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3 (July), 438–446.

SARAGIH, J. M., LUCEY, S., AND COHN, J. F. 2011. Deformable model fitting by regularized landmark mean-shift. *Int. J. Comput. Vision 91*, 2 (Jan.), 200–215.

SKOCAJ, D., AND LEONARDIS, A. 2003. Weighted and robust incremental method for subspace learning. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, IEEE, 1494–1501.

SUGIMOTO, T., FUKUSHIMA, M., AND IBARAKI, T. 1995. A parallel relaxation method for quadratic programming problems with interval constraints. *Journal of Computational and Applied Mathematics 60*, 12, 219 – 236.

SUMNER, R. W., AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph. 23*, 3 (Aug.), 399–405.

VALGAERTS, L., WU, C., BRUHN, A., SEIDEL, H.-P., AND THEOBALT, C. 2012. Lightweight binocular facial performance capture under uncontrolled lighting. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2012)*, vol. 31.

VLASIC, D., BRAND, M., PFISTER, H., AND POPOVIĆ, J. 2005. Face transfer with multilinear models. In *ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, SIGGRAPH '05, 426–433.

WEISE, T., LI, H., GOOL, L. V., AND PAULY, M. 2009. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer animation*.

WEISE, T., BOUAZIZ, S., LI, H., AND PAULY, M. 2011. Realtime performance-based facial animation. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2011) 30*, 4 (July).

WELCH, G., AND BISHOP, G. 1995. An introduction to the kalman filter. Tech. rep., Chapel Hill, NC, USA.

WILLIAMS, L. 1990. Performance-driven facial animation. *SIGGRAPH Comput. Graph. 24*, 4 (Sept.), 235–242.

ZHANG, S., AND HUANG, P. 2004. High-resolution, real-time 3d shape acquisition. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 3 - Volume 03*, IEEE Computer Society, Washington, DC, USA, CVPRW '04, 28–.

ZHANG, L., SNAVELY, N., CURLESS, B., AND SEITZ, S. M. 2004. Spacetime faces: High-resolution capture for modeling and animation. In *ACM Annual Conference on Computer Graphics*, 548–558.