Displaced Dynamic Expression Regression for Real-time Facial Tracking and Animation

Chen Cao Qiming Hou Kun Zhou*

State Key Lab of CAD&CG, Zhejiang University

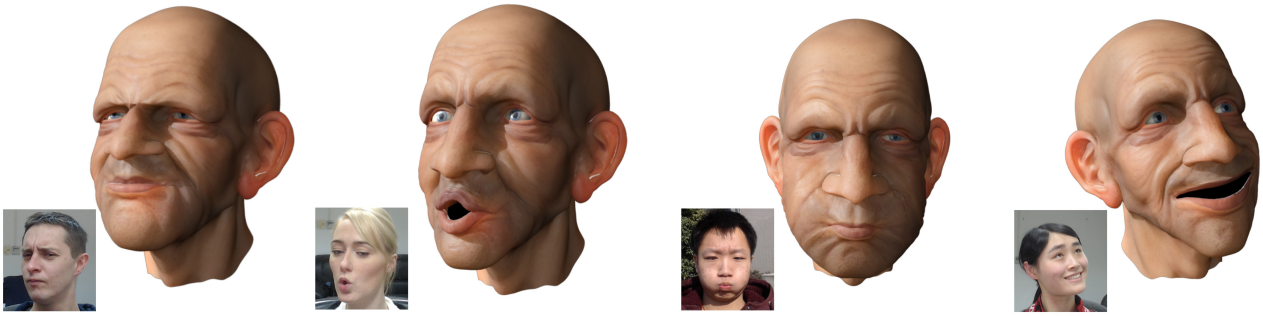


Figure 1: *Real-time facial tracking and animation for different users using a single camera.*

Abstract

We present a fully automatic approach to real-time facial tracking and animation with a single video camera. Our approach does not need any calibration for each individual user. It learns a generic regressor from public image datasets, which can be applied to any user and arbitrary video cameras to infer accurate 2D facial landmarks as well as the 3D facial shape from 2D video frames. The inferred 2D landmarks are then used to adapt the camera matrix and the user identity to better match the facial expressions of the current user. The regression and adaptation are performed in an alternating manner. With more and more facial expressions observed in the video, the whole process converges quickly with accurate facial tracking and animation. In experiments, our approach demonstrates a level of robustness and accuracy on par with state-of-the-art techniques that require a time-consuming calibration step for each individual user, while running at 28 fps on average. We consider our approach to be an attractive solution for wide deployment in consumer-level applications.

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

Keywords: face tracking, face animation, performance capture, blendshape models

Links: DL PDF WEB DATA

*Corresponding author (kunzhou@acm.org)

ACM Reference Format
Cao, C., Hou, Q., Zhou, K. 2014. Displaced Dynamic Expression Regression for Real-time Facial Tracking and Animation. ACM Trans. Graph. 33, 4, Article 43 (July 2014), 10 pages. DOI = 10.1145/2601097.2601204 http://doi.acm.org/10.1145/2601097.2601204.

DOI: http://doi.acm.org/10.1145/2601097.2601204

1 Introduction

With the wide spread of commodity RGBD cameras such as Microsoft's Kinect, performance-driven facial animation, a high end technology in film and game production, is now ready for deployment in consumer-level applications. State-of-the-art techniques [Weise et al. 2011; Bouaziz et al. 2013; Li et al. 2013] demonstrate impressive real-time facial tracking and animation results by registering a DEM (Dynamic Expression Model) with the depth data from an RGBD camera.

Video cameras, however, are more widely available on PCs and mobile devices than RGBD cameras, and video-based facial tracking remains a challenging problem. The recent regression-based algorithm proposed by Cao et al. [2013a] can reach the same level of robustness and accuracy as demonstrated in RGBD-based algorithms, while requiring only an ordinary web camera. It learns a 3D shape regressor to infer the 3D positions of facial landmarks from 2D video frames, which are then used to register the DEM. The regressor and DEM, however, are constructed for each specific user in a time-consuming calibration step, which greatly hinders its practical adoption in consumer-level applications. The latest calibration-free techniques (e.g., [Saragih et al. 2011a]) often assume weak perspective camera projection and can provide good tracking results, but cannot achieve the level of robustness and accuracy demonstrated in [Weise et al. 2011; Cao et al. 2013a].

In this paper, we propose a fully automatic approach to robust and efficient facial tracking and animation with a single video camera. Our approach does not need any calibration for each individual user. It learns a generic regressor from public image datasets, which can be applied to any user and arbitrary video cameras to infer accurate 2D facial landmarks as well as the 3D facial shape from 2D video frames, assuming the user identity does not change across frames. The inferred 2D landmarks are then used to adapt the camera matrix and the user identity to better match the facial expressions of the current user. The regression and adaptation are performed in an alternating manner, effectively creating a feedback loop. With more and more facial expressions observed in the video, the whole process converges quickly with accurate facial tracking and animation.

To facilitate the above regression and adaptation processes, we in-

troduce the DDE (Displaced Dynamic Expression) model as a novel facial shape representation suitable for video-based facial tracking. A unique feature of the DDE model is that it simultaneously represents the 3D geometry of the user's facial expressions (the final output of our algorithm), and the 2D facial landmarks which correspond to semantic facial features in video frames. We use the DEM to represent the 3D geometry because of its demonstrated robustness and efficiency in RGBD-based algorithms. The DEM, which is initialized for a generic face and not pre-calibrated to match the 3D facial geometry of the current user, cannot be well registered with the facial features observed in the video. To compensate for the inaccuracy caused by the DEM, we add a 2D displacement to the projection of each 3D facial landmark in the image space. The DDE model, combining the powers of DEM and landmark displacements, is capable of generating accurate 2D facial landmarks from video frames, which are then used to adapt the DEM and the camera matrix.

The value of the DDE displacements is beyond improving landmark accuracy. They allow us to train a 2D+3D regressor that achieves satisfactory landmark accuracy without specifying the DEM expression blendshapes (i.e., the user identity) beforehand. Such a design supports fast online updating of the actual user identity, which is a fundamental requirement of the adaptation process. The design also creates another advantage – it allows the *same* regressor to be trained using images from *different* users, without violating the *frame-invariant-identity* assumption in video tracking. Previous DEM based video trackers like [Cao et al. 2013a] require all training images to be taken from the same user, which practically limits their training dataset to a few dozen images. The broadened choice of training images is thus an algorithmically significant consequence of our model design.

Contributions. The main contribution of our work is a calibration-free approach to performance-driven facial animation with a single video camera. As shown in the supplementary video, the resulting system can be used by any user, without any training. It can robustly handle fast motions, large head rotations and exaggerated expressions. Compared with the state-of-the-art video-based tracking algorithm [Cao et al. 2013a] which needs a user-specific calibration process, our approach can reach the same level of tracking accuracy, and is even more robust under large lighting changes due to the significant lighting variations exhibited in our training images. Evaluation experiments also show that the 3D facial geometry tracked by our approach matches the ground truth depth acquired by an RGBD camera. The system performance is promising – it takes less than 20 miliseconds to process a video frame, making it very attractive for consumer-level applications.

We also contribute a new facial shape representation for video-based tracking, i.e., the DDE model, as well as a regression algorithm designed for the model. Experiments demonstrate that the DDE model can achieve more robust and accurate tracking than other alternative representations and algorithms.

The rest of the paper is structured as follows. The following section reviews related work. In Section 3, we give an overview of our system, including the DDE model and the system workflow. Section 4 introduces the regression algorithm for the DDE model. Section 5 describes how to use the DDE regression results to adapt the camera matrix and the DEM. Some important implementation details are discussed in Section 6. Section 7 presents results and Section 8 concludes the paper with some discussion of future work.

2 Related Work

Facial performance capture and face tracking have a long history in computer graphics and vision (e.g., [Williams 1990]). In this section we only review the most relevant references.

Various techniques have been proposed to capture facial expressions of a subject and transfer them to a target model. In film and game production, special equipment, such as facial markers [Huang et al. 2011], camera arrays [Bradley et al. 2010; Beeler et al. 2011], and structured light projectors [Zhang et al. 2004; Weise et al. 2009], can be used to get 3D facial geometry of high fidelity. These techniques, however, are not suitable for consumer-level applications, where such special equipment are not available.

As a more practical solution for ordinary users, video-based facial tracking and animation have attracted much research effort. This category of techniques first locates semantic facial landmarks such as eyes, nose and mouth in video frames, and then uses the landmark positions to drive facial animation. Early techniques track each individual landmark by optical flow, which is unreliable under fast motions. To achieve more robust tracking, a few techniques have been proposed to incorporate geometric constraints to correlate the positions of all landmarks, such as feature displacements in expression change [Chai et al. 2003], physically-based deformable mesh models [Essa et al. 1996; DeCarlo and Metaxas 2000], and data-driven face models [Pighin et al. 1999; Blanz and Vetter 1999; Vlasic et al. 2005]. The majority of latest tracking techniques use CPR (Cascaded Pose Regression) [Dollar et al. 2010; Cao et al. 2012; Cao et al. 2013a], CLM (Constrained Local Model) [Saragih et al. 2011b; Saragih et al. 2011a; Baltrušaitis et al. 2012; Asthana et al. 2013] or SDM (Supervised Descent Method) [Xiong and De La Torre 2013] to track facial shapes from 2D images. We use CPR to regress our DDE model for two reasons. First, our experiments show that the limited local search radius in CLM can be problematic when handling fast motions. Second, CLM and SDM methods can only produce raw landmark positions as the output, which requires considerable additional processing before they can be applied to avatars [Saragih et al. 2011a].

The recent advent of RGBD cameras has enabled a number of depth-based facial tracking methods [Weise et al. 2011; Baltrušaitis et al. 2012; Bouaziz et al. 2013; Li et al. 2013]. In particular, Weise et al. [2011] construct a user-specific DEM in a preprocessing stage, and register the DEM with the observed depth data at runtime. This algorithm achieves real-time performance and shows more robust and accurate results than previous video-based methods. Bouaziz et al. [2013] and Li et al. [2013] further propose to combine the DEM construction with online tracking, and demonstrate impressive tracking and animation results for an arbitrary user, without any training or calibration.

Realizing the advantages of user-specific models, Cao et al. [2013a] propose to train a 3D shape regressor for each specific user, and use it to infer 3D facial shapes from 2D video frames at runtime. This video-based algorithm generates accurate and robust tracking results comparable to those created by RGBD-based techniques [Weise et al. 2011]. An improved version of the algorithm significantly reduces the computational cost by directly regressing the head poses and expression coefficients, but still requires a training process [Weng et al. 2013]. The training, however, needs to collect a set images of each specific user with predefined facial poses and expressions and then label them. This tedious and time-consuming process hinders the wide adoption of the technique in consumer-level applications. Our goal in this paper is to develop a video-based approach that does not need any calibration while keeping the same level of robustness, accuracy and efficiency as in [Weise et al. 2011; Cao et al. 2013a]. Unlike in RGBD-based tracking where the observed depth map can be used as the ground truth geometry to guide the online training and tracking, our problem is more challenging as we only have 2D video frames as input.

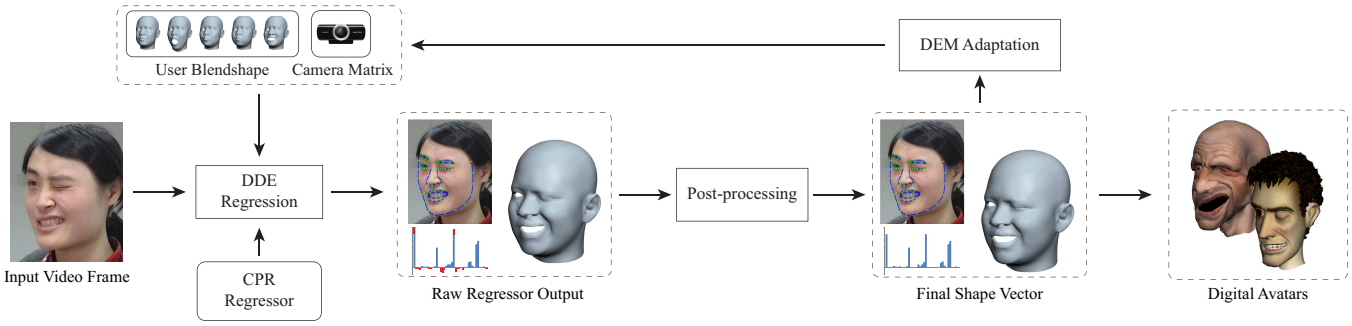


Figure 2: *The workflow of our approach.*

Various parametric shape models have been used to represent human faces, such as PCA models in ASM (Active Shape Model) [Cootes et al. 1995] and AAM (Active Appearance Model) [Cootes et al. 1998], multi-linear models [Vlasic et al. 2005] and blendshapes [Pighin et al. 1998; Lewis and Anjyo 2010]. Two or more parametric models can be used together in actual algorithms. For example, Xiao et al. [2004] introduced a 2D+3D algorithm for real-time facial tracking. They use a 3D PCA model for the 3D facial shape and fit its parameters using a 2D AAM on the projected shape. Our DDE model is a new facial shape representation that combines a 3D parametric model and 2D landmark displacements. It outperforms alternative representations in terms of tracking robustness and accuracy in our experiments.

3 Approach Overview

In this section, we first describe the DDE model, and then briefly overview the workflow of the tracking process.

3.1 The DDE Model

The DDE model is designed to simultaneously represent the 3D shape of the user's facial expressions and the 2D facial landmarks which correspond to semantic facial features in video frames. For the 3D shape, we use a DEM based on a set of blendshape meshes. Similar to [Weise et al. 2011], we represent the 3D facial mesh F as a linear combination of expression blendshapes $\mathbf{B} = [\mathbf{b}_0, ..., \mathbf{b}_n]$ plus a rotation $\mathbf{R}$ and a translation $\mathbf{t}$:

$$F = \mathbf{R}\left(\mathbf{B}\mathbf{e}^T\right) + \mathbf{t}, \tag{1}$$

where $\mathbf{e} = [e_0, ..., e_n]$ is the expression coefficients. As commonly assumed in blendshape models, $\mathbf{b}_0$ is the neutral face, and non-neutral blend weights $e_i, 1 \le i \le n$ are bounded between 0 and 1. All blend weights must sum to 1, leading to $e_0 = 1 - \sum_{i=1}^{n} e_i$. For simplicity of description, we ignore the dependent e_0 in the following equations and discussions, and assume the correct value is computed on demand. The rotation $\mathbf{R}$ is represented as a quaternion in our algorithm.

Our blendshape model is based on the FACS (Facial Action Coding System) [Ekman and Friesen 1978], which contains 46 action units (i.e., $n = 46$) that mimic the combined activation effects of facial muscle groups. This blendshape model adequately describes most expressions of the human face. As in [Cao et al. 2013a], we make use of FaceWarehouse [Cao et al. 2013b], a 3D facial expression database containing the data of 150 individuals from various ethnic backgrounds, to construct the blendshape meshes. Specifically, the expression blendshapes $\mathbf{B}$ of a certain user is constructed as:

$$\mathbf{B} = C \times_2 \mathbf{u}^T, \tag{2}$$

where $\mathbf{u}$ is the user identity vector, and C is the rank-3 core tensor from the database, of which the three modes are face mesh, identity and expression, respectively.

To represent the 2D facial landmarks $\{s_k\}$, we add a 2D displacement $\{d_k\}$ to the projection of each landmark's corresponding vertex $F^{(v_k)}$ on the facial mesh:

$$s_k = \Pi_{\mathbf{Q}}\left(F^{(v_k)}\right) + d_k, \tag{3}$$

where $\Pi_{\mathbf{Q}}$ is a perspective projection operator, parameterized by a projection matrix $\mathbf{Q}$. The 2D facial shape S is thus represented by the set of all 2D landmarks $\{s_k\}$. The displacement vector is denoted as $\mathbf{D} = \{d_k\}$.

Following [Cao et al. 2013a], we assume an ideal pinhole camera, with the projection matrix represented as

$$\mathbf{Q} = \begin{pmatrix} f & 0 & p_0 \\ 0 & f & q_0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{4}$$

where f is the focal length and (p_0, q_0) is at the image center, making f the only unknown variable.

Note that for landmarks on the face contour, the vertex indices v_k may vary across frames and have to be recomputed when F changes. The index updating procedure is detailed in Section 6.

Combining the 3D part and the 2D part, our DDE model can be written as a function mapping unknown variables to a 2D facial shape:

$$\mathrm{DDE}(\mathbf{Q}, \mathbf{u}; \mathbf{e}, \mathbf{R}, \mathbf{t}, \mathbf{D}) = S. \tag{5}$$

Among the unknowns, the projection matrix $\mathbf{Q}$ and the identity coefficients $\mathbf{u}$ need to be treated in a special way as their ground truth values should be invariant across all frames for the same user and the same video camera during tracking. This fact is reflected by the semi-colon in Eq. (5). The remaining per-frame unknowns will be referred to collectively as the *shape vector* $\mathbf{P} = (\mathbf{e}, \mathbf{R}, \mathbf{t}, \mathbf{D})$.

3.2 Tracking Workflow

Our tracking workflow is illustrated in Fig. 2. The input video frame I is first sent into a CPR regressor [Cao et al. 2012], formulated as a function mapping a guessed shape vector P^{in} to a regressed shape vector $\mathrm{P}^{\mathrm{out}}$:

$$\mathrm{CPR}(I, \mathbf{Q}, \mathbf{u}; \mathbf{P}^{\mathrm{in}}) = \mathbf{P}^{\mathrm{out}}, \tag{6}$$

where $\mathbf{Q}, \mathbf{u}$ are known for the current frame. The regressor output is then post-processed to improve the temporal coherence and clamp expression coefficients within valid ranges. The post-processed output is optionally sent to an iterative optimization procedure to update the camera matrix and the identity if the current

frame is identified to contain representative facial motions. The optimized projection matrix and identity are sent back to the CPR regressor in a feedback loop. Finally, the post-processed output, including the rotation, translation and expression coefficients, can be directly transferred to a digital avatar to drive its facial animation.

As indicated by Eq. (6), the regressor only computes the *facial motion*, i.e., the expression, rigid transformation and displacements. The frame-invariant parameters $\mathbf{Q}$ and $\mathbf{u}$ are a part of the *regressor input*, as opposed to the output.

4 DDE Regression

In the following we first explain how to learn the DDE regressor indicated in Eq. (6) from a set of training images, and how to use it for runtime tracking. We then describe the postprocessing of the regression output.

4.1 Training

Training data preparation. Our training algorithm takes a set of facial images from public datasets as input. For each input image I, 73 2D landmarks are manually labeled to produce the 2D facial shape $S = \{s_k\}$. From the landmarks, we fit all the unknowns $(\mathbf{Q}, \mathbf{u}; \mathbf{P})$ by minimizing the total displacements $E_{\text{im}} = \sum_k \|d_k\|^2$, under the constraint $\text{DDE}(\mathbf{Q}, \mathbf{u}; \mathbf{P}) = S$. Images of the same person are manually identified and the corresponding shape vectors are optimized jointly with the same identity coefficients $\mathbf{u}$.

The above optimization process is similar to the 3D facial shape recovery and camera calibration steps described in [Cao et al. 2013a], with the exception that we need to recover the 2D landmark displacements in addition to the 3D shape. Given a value of the focal length f (and the corresponding $\mathbf{Q}$), we use the coordinate-descent method to solve the identity $\mathbf{u}$ and the shape vector $\mathbf{P}$, by alternately optimizing each parameter while fixing the others in each iteration. Different values of f result in different fitting errors of E_{im}. We thus use the binary search scheme to find the optimal focal length f that leads to the minimal E_{im}.

Training pair construction. The CPR training method requires creating *guess-truth pairs* for each image I_i, in order to relate parameter differences to image features. We use the notation $(I_i, \mathbf{Q}_{ij}, \mathbf{u}_{ij}; \mathbf{P}_{ij}, \mathbf{P}^g_{ij})$ to denote such pairs, where $\mathbf{P}_{ij}$ is the guessed shape vector, $\mathbf{P}^g_{ij}$ is the ground truth shape vector and j represents the indices of the guess-truth pairs for the image I_i.

For each input image I_i and the corresponding fitted ground truth $(\mathbf{Q}^g_i, \mathbf{u}^g_i; \mathbf{P}^g_i)$, where $\mathbf{P}^g_i = (\mathbf{e}^g_i, \mathbf{R}^g_i, \mathbf{t}^g_i, \mathbf{D}^g_i)$, we generate several classes of training pairs by perturbing individual parameters. In each training pair, we also set the guessed displacement vector with $\mathbf{D}^r_{ij}$, taken from a random image.

- **Random rotation.** Add a random rotation $\Delta\mathbf{R}_{ij}$, yielding $\mathbf{P}_{ij} = (\mathbf{e}^g_i, \mathbf{R}^g_i + \Delta\mathbf{R}_{ij}, \mathbf{t}^g_i, \mathbf{D}^r_{ij})$, $\mathbf{P}^g_{ij} = \mathbf{P}^g_i$;
- **Random translation.** Add a random translation $\Delta\mathbf{t}_{ij}$, yielding $\mathbf{P}_{ij} = (\mathbf{e}^g_i, \mathbf{R}^g_i, \mathbf{t}^g_i + \Delta\mathbf{t}_{ij}, \mathbf{D}^r_{ij})$, $\mathbf{P}^g_{ij} = \mathbf{P}^g_i$;
- **Random expression.** Choose a random image $I_{i'}$, assign its expression coefficients $\mathbf{e}_{ij} = \mathbf{e}^g_{i'}$ to the current image, yielding $\mathbf{P}_{ij} = (\mathbf{e}_{ij}, \mathbf{R}^g_i, \mathbf{t}^g_i, \mathbf{D}^r_{ij})$, $\mathbf{P}^g_{ij} = \mathbf{P}^g_i$;
- **Random identity.** Choose a random image $I_{i'}$, assign its fitted identity coefficients to the current training pair, yielding $\mathbf{u}_{ij} = \mathbf{u}^g_{i'}$, $\mathbf{P}_{ij} = (\mathbf{e}^g_i, \mathbf{R}^g_i, \mathbf{t}^g_i, \mathbf{D}^r_{ij})$. Since the identity coefficients are input parameters and cannot be changed during regression, the ground truth shape vector must be updated accordingly to $\mathbf{P}^g_{ij} = (\mathbf{e}^g_i, \mathbf{R}^g_i, \mathbf{t}^g_i, \mathbf{D}^g_{ij})$, where the landmark displacements $\mathbf{D}^g_{ij}$ are recomputed to match the ground truth landmarks under the changed identity;
- **Random camera.** Add a random offset to the focal length in the camera matrix $\mathbf{Q}^g_i$, yielding $\mathbf{Q}_{ij} = \mathbf{Q}^g_i + \Delta\mathbf{Q}$, $\mathbf{P}_{ij} = (\mathbf{e}^g_i, \mathbf{R}^g_i, \mathbf{t}^g_i, \mathbf{D}^r_{ij})$. Similar to the identity case, the ground truth shape vector must be updated accordingly to $\mathbf{P}^g_{ij} = (\mathbf{e}^g_i, \mathbf{R}^g_i, \mathbf{t}^g_i, \mathbf{D}^g_{ij})$, where the landmark displacements $\mathbf{D}^g_{ij}$ are recomputed to match the ground truth landmarks under the changed camera.

(a) (b) (c)

Figure 3: *Random identity example. For image (a), we use the identity from (b) but keep the original displacements, resulting in inaccurate 2D landmark positions. Therefore we need to recompute the displacements to get the correct landmarks (c).*

Our identity and camera perturbation is a significant divergence from conventional CPR methods, as the ground truth shape vector is perturbed alongside the guessed shape vector to avoid introducing a change in non-regressed input parameters (i.e., $\mathbf{Q}$ and $\mathbf{u}$). Such training pairs simulate cases where these input parameters are initially inaccurate, in which the regressor is expected to produce large displacements to get the correct landmark positions. Fig. 3 illustrates such a training pair. In Fig. 3(a), the identity in the ground truth shape vector is replaced with that of a different person (shown in Fig. 3(b)), which introduces significant changes in the landmark positions. In Fig. 3(c), the displacements $\mathbf{D}^g_{ij}$ are recomputed to move the landmarks back to the correct locations. The CPR regressor is trained to be able to reproduce such displacements at runtime.

We generate 5 training pairs for each class except the random expression class, for which we generate 15 pairs to better capture the rich variety of expressions. Another detail is that for each training pair, the landmark vertex indices v_k in Eq. (3) remain the same throughout the training process. The reason is that the training pair models facial shape changes within a single frame, within which contour vertex indices are not updated. The v_k values used in the training are computed according to the guessed shape vector $\mathbf{P}_{ij}$.

Training. Once the training pairs are constructed, we learn a regression function from $\mathbf{P}_{ij}$ to $\mathbf{P}^g_{ij}$ based on intensity information in the image I_i. We follow the two-level boosted regression approach proposed by Cao et al. [2012], and use 15 stages for the first level and 300 stages for the second level. The approach combines a set of weak regressors in an additive manner. Each weak regressor computes a shape increment from image features and updates the current shape vector. At a high level, the training process exploits the correlation between the error $\mathbf{P}^g_{ij} - \mathbf{P}_{ij}$ and appearance vectors extracted from the image I_i, which minimizes the following energy:

$$E_{\text{tr}} = \sum_{i,j} \left\| \mathbf{P}^g_{ij} - \mathbf{P}_{ij} \right\|^2 . \tag{7}$$

We only make a minor change to the appearance vector extraction

step, which will be described in Section 6. Please refer to [Cao et al. 2012] or [Cao et al. 2013a] for more details about the training algorithm.

4.2 Runtime Regression

Our runtime regression resembles the runtime procedure in [Cao et al. 2013a]. Specifically, for the t-th input video frame I^t and the current estimation of $\mathbf{Q}$ and $\mathbf{u}$, we compute a set of guessed shape vectors $\{\mathbf{P}_j^{t-1}\}$. For each $\mathbf{P}_j^{t-1}$, the regressor produces an updated shape vector $\mathbf{P}_j^t = \mathrm{CPR}(I^t, \mathbf{Q}, \mathbf{u}; \mathbf{P}_j^{t-1})$. Finally, we average all output vectors $\{\mathbf{P}_j^t\}$ to obtain the average shape vector $\bar{\mathbf{P}}^t$.

The guessed shape vectors $\{\mathbf{P}_j^{t-1}\}$ are computed from the shape vector computed in the previous frame, $\hat{\mathbf{P}}^{t-1}$. Following [Cao et al. 2013a], we first find $\mathbf{P}_{near}$, the nearest shape vector to $\hat{\mathbf{P}}^{t-1}$ among the training shape vectors $\{\mathbf{P}_i^t\}$. Here the distance between two shape vectors is defined by first aligning the centroids of their projected 2D shapes, and then computing the total RMS (Root Mean Square) distance between all corresponding landmark points. We then replace the rotation quaternion in $\hat{\mathbf{P}}^{t-1}$ with the one in $\mathbf{P}_{near}$, and compute for the resulting shape vector its K nearest vectors among the training shape vectors as $\{\mathbf{P}_j^{t-1}\}$.

For the first frame $t = 1$, no previous-frame shape vector is available and we have to compute $\hat{\mathbf{P}}^{t-1} = \hat{\mathbf{P}}^0$ from scratch. Specifically, we detect the face location using a real-time face detector [Viola and Jones 2004], use a 2D CPR method [Cao et al. 2012] to compute a 2D facial shape, and then fit a shape vector from this 2D shape using the same method as in Section 4.1.

For more details about the runtime regression algorithm, please refer to [Cao et al. 2013a].

Discussion. Although it is possible to simply regress $\mathbf{Q}$ and $\mathbf{u}$ along with other parameters, we leave them out of the regression and update them in the DEM adaptation step. The reasons are two-fold. First, the ground truth parameters of the camera matrix and identity are invariant across video frames (for the same camera and the same user). It is therefore unnecessary to compute these parameters per frame. Second, the identity and camera matrix estimated from a single frame are significantly less accurate than those computed from multiple representative frames in a joint optimization. Factoring $\mathbf{Q}$ and $\mathbf{u}$ out of the regressor would produce more accurate results in the long run, once the joint optimization converges.

4.3 Post-processing

Prior to post-processing, we compute a projected 2D shape S^t from the regressed shape vector $\bar{\mathbf{P}}^t$. S^t is regarded as the ground truth 2D shape of the current frame in all subsequent computations.

The regression output $\bar{\mathbf{P}}^t$ does not take temporal coherence into account, nor does it enforce valid parameter ranges. This may pose problems even if the landmark positions are accurate. For example, the expression coefficients could lie outside the range of $[0, 1]$. The displacements could become too large due to inaccurate estimations of the identity or the camera matrix, undermining the significance of the regressed facial motion. As a remedy, we use an additional optimization procedure to post-process the regression output.

The post-processing step computes the final output shape vector $\hat{\mathbf{P}}^t$ by minimizing a weighted combination of three energy terms, including a fitting error term E_{fit}, a regularization term E_{reg}, and a temporal coherence term E_{tm}:

$$\begin{aligned} E_{\mathrm{fit}} &= \textstyle\sum_{k=1}^{m} \left\| \Pi_{\mathbf{Q}} \left(\hat{\mathbf{R}}^t \left(\mathbf{B}\hat{\mathbf{e}}^{tT} \right) + \hat{\mathbf{t}}^t \right)^{(v_k^t)} - s_k^t \right\|^2, \\ E_{\mathrm{reg}} &= \left\| \hat{\mathbf{P}}^t - \bar{\mathbf{P}}^t \right\|^2, \\ E_{\mathrm{tm}} &= \left\| \hat{\mathbf{P}}^{t-2} - 2\hat{\mathbf{P}}^{t-1} + \hat{\mathbf{P}}^t \right\|^2. \end{aligned}$$

E_{reg} helps to make the tracking result more expressive, while E_{tm} is designed to make the animation more stable and smooth.

The final combined energy is:

$$E_{\mathrm{tot}} = E_{\mathrm{fit}} + \omega_{\mathrm{reg}} E_{\mathrm{reg}} + \omega_{\mathrm{tm}} E_{\mathrm{tm}}, \tag{8}$$

where ω_{reg} and ω_{tm} are the non-negative weights for the respective terms. E_{tot} is minimized using an off-the-shelf BFGS optimizer [Byrd et al. 1995]. We set $\omega_{\mathrm{reg}} = 5$ and $\omega_{\mathrm{tm}} = 1$ in our current implementation, which can provide good tracking results. The expression coefficients $\hat{\mathbf{e}}^t$ are constrained within the range $[0, 1]$.

5 DEM Adaptation

The DEM adaptation step takes as input the regressed 2D shape S^t and the post-processed shape vector $\hat{\mathbf{P}}^t$ to optimize the frame-invariant parameters $\mathbf{Q}$ and $\mathbf{u}$, i.e., the camera projection matrix and the user identity. After $\mathbf{u}$ is updated, the expression blendshape matrix $\mathbf{B}$ is regenerated according to Eq. (2).

Initialization. When a new user enters the camera's field of view, we initialize $\mathbf{Q}$ and $\mathbf{u}$ to average values. The identity vector $\mathbf{u}$ is initialized to the average identity in FaceWarehouse. For the camera matrix, we first compute the initial 2D facial shape as described in Section 4.2. We then use the binary search scheme described in [Cao et al. 2013a] to compute an initial focal length f, from which we construct the initial camera matrix $\mathbf{Q}$.

5.1 Representative Frame Selection

We use a joint optimization across multiple frames to update $\mathbf{Q}$ and $\mathbf{u}$. As it is computationally expensive and unnecessary to use all input frames in this optimization, we need a way to select representative frames to obtain maximal accuracy within a fixed computational budget.

We perform the frame selection by incrementally appending to a representative frame set $\{(S^l, \hat{\mathbf{P}}^l)\}$. The initial L frames are always added to the set. After that, we only append a frame to the set if its expression coefficients and rigid rotation parameters are sufficiently distant from the linear space formed by existing frames.

Specifically, we first define for each frame an *expression-rotation vector* $\mathbf{V}^l = (\mathbf{R}^l, \mathbf{e}^l)$. We perform a mean-removed PCA (Principal Component Analysis) dimension reduction on all $\mathbf{V}^l$ vectors in the current frame set, yielding a mean vector $\bar{\mathbf{V}}$ and an eigenvector matrix $\mathbf{M}$. The PCA discards the last 5% eigenvectors in terms of energy. For each incoming frame with vector $\mathbf{V}^t$, we compute its PCA reconstruction error:

$$E_{\mathrm{rec}} = \left\| \mathbf{V}^t - \left(\bar{\mathbf{V}} + \mathbf{M}\mathbf{M}^T \left(\mathbf{V}^t - \bar{\mathbf{V}} \right) \right) \right\|^2. \tag{9}$$

We only accept the new frame if E_{rec} is larger than a threshold (0.1 in our experiments). Accepted frames are appended to the frame set and the PCA space is updated accordingly.

5.2 Optimization

We start the optimization when a new representative frame is identified. We first optimize the identity vector $\mathbf{u}$, then the camera matrix $\mathbf{Q}$, and finally we re-fit shape vectors for all frames in the representative frame set under the updated identity and camera matrix.

The optimization of the identity coefficients $\mathbf{u}$ is similar to the fitting process in training data preparation (Section 4.1). We fix the camera matrix $\mathbf{Q}$ and the shape vectors $\{\hat{\mathbf{P}}^l\}$ of all representative frames, and minimize the total displacements $E = \sum_{l,k} \|d_k^l\|^2$ in these frames, under the constraint $\mathrm{DDE}(\mathbf{Q}, \mathbf{u}; \hat{\mathbf{P}}^l) = S^l$.

When optimizing the camera matrix, we fix the identity $\mathbf{u}$ and the shape vectors $\{\hat{\mathbf{P}}^l\}$, generate the 3D mesh F^l for each frame according to Eq. (1), and optimize the following energy:

$$E_{\mathrm{im}} = \sum_{l,k} \left\| \Pi_{\mathbf{Q}} F^{l,(v_k)} - s_k^l \right\|^2 . \tag{10}$$

Substituting Eq. (4) into Eq. (10) yields:

$$E_{\mathrm{im}} = \sum_{l,k} \left\| \frac{x_k^l}{z_k^l} f + p_0 - p_k^l \right\|^2 + \left\| \frac{y_k^l}{z_k^l} f + q_0 - q_k^l \right\|^2 , \tag{11}$$

where $s_k^l = (p_k^l, q_k^l)$ are the 2D coordinates of landmark s_k^l, $F^{l,(v_k)} = (x_k^l, y_k^l, z_k^l)^T$ are the 3D coordinates of the corresponding vertex on mesh F. Since Eq. (11) is a least squares problem, we simply compute f analytically.

Note that we solve for $\mathbf{Q}$ and $\mathbf{u}$ only once in each DEM adaptation step. Due to the limited computational budget, we cannot optimize them in an alternating manner or use the binary search scheme to compute the optimal focal length f.

After updating $\mathbf{Q}$ and $\mathbf{u}$, the existing shape vectors $\hat{\mathbf{P}}^l$ no longer match the ground truth 2D shapes S^l and have to be updated. We use the same method as in training data preparation (Section 4.1) to re-fit $\hat{\mathbf{P}}^l$ from S^l. The PCA space for frame selection is updated accordingly.

As new representative frames can always occur throughout the video, the DEM adaptation step may be executed anytime. In our experiments, we found that with more facial expressions and head poses observed, the number of frames in the representative frame set becomes stable. Note that the adaptation result may become less accurate if a newly added representative frame contains some inaccurate landmarks, and we do not have a good way to handle such contaminations. Fortunately, this situation rarely happens in practice. The natural response of a new user is to check his/her own "reflection" in the camera preview. This creates a few seconds of adjustment period with near-frontal facial motions, which are ideally suited for our CPR regressor. The resulting 2D landmarks are highly accurate and allow the adaptation to reach near-convergence before the adjustment period ends. Typically, the optimization converges quickly in less than 20 seconds, with fewer than 50 representative frames appended.

6 Implementation Details

Vertex index update. When reconstructing the 2D facial shape from our shape vector, we need to compute for each 2D landmark its corresponding vertex index v_k on the 3D mesh. For the internal landmarks (e.g., eyes and nose), the indices are pre-defined and fixed. However, for landmarks along the face contour, we need to compute the mesh's silhouette under the current camera projection and sample the silhouette to get the correct indices.

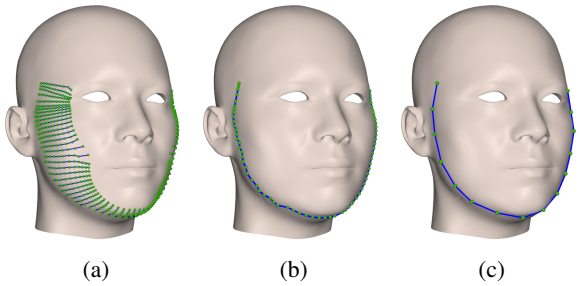


Figure 4: *Vertex index update for landmarks along the face contour. We organize the mesh vertices into a series of horizontal lines in (a), and choose a vertex for each line to construct the silhouette curve (b), which is then projected to the image plane and uniformly sampled to locate the corresponding vertex indices (c).*

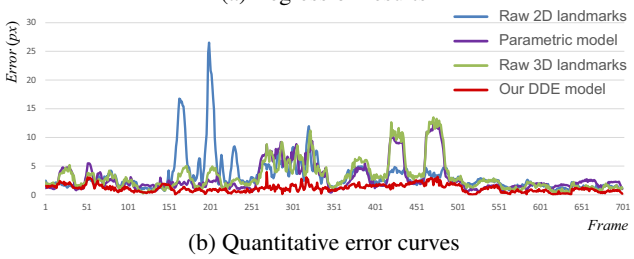
The main challenge is that we need to compute the silhouette as a single connected curve for the landmark sampling. To avoid costly connectivity computation, we organize the mesh vertices into a series of horizontal lines as shown in Fig. 4(a). Note that we have already discarded the vertices that cannot be part of the silhouette in any possible head poses that are supported in our tracking system. During the on-the-fly silhouette extraction, we choose the vertex with the smallest $|N \cdot V|$ from each horizontal line, and connect the chosen vertices *vertically* to construct the silhouette curve as shown in Fig. 4(b). Here N is the vertex normal and V is the view direction. The silhouette curve is then projected to the image plane and uniformly sampled at predefined intervals to get the 2D landmark and their corresponding vertex indices on the mesh as shown in Fig. 4(c).

Appearance vector extraction. The regression algorithm needs to extract appearance vectors from images, which are composed of the image intensities at a set of randomly selected points (called feature points). These appearance vectors are used to represent the images during the training and runtime processes. Cao et al. [2012] define a feature point as a landmark position plus an image space offset. This approach, however, cannot robustly handle large rotations in our scenario. The 3D feature points used by the 3D shape regressor in [Cao et al. 2013a] can effectively handle large rotations, but cannot be directly used in our case as our shape representation is a combination of a 3D shape and 2D dispacements. Burgos-Artizzu et al. [2013] propose to represent a feature point as a linear combination of two 2D landmarks. This approach can also handle large rotations, but restricts feature points to lie on the line segments connecting two landmarks.

We define the position of a feature point as the barycentric coordinates in a triangle formed by 2D landmarks. We first generate a set of feature points (400 points in our current implementation by sampling a Gaussian distribution on the unit square. For each point, we find a triangle in a reference 2D facial shape whose center is closest to the point, and compute the point's barycentric coordinates in the triangle. During appearance vector extraction, this point will be located according to the triangle positions and the barycentric coordinates. We call such feature points *triangle-indexed features*. The reference 2D facial shape is computed by averaging the 2D shapes of all training images. The triangles are created as the Delaunay triangulation of all landmarks.

Face tensor preparation. The original FaceWarehouse database uses n-mode SVD to compress both the identity mode and the ex-

(a) Regression results



(b) Quantitative error curves

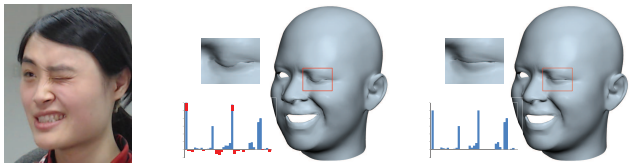
Figure 5: *Shape representation comparison. In (a) from left to right respectively: raw 2D landmarks. the parametric model. raw 3D landmarks, our DDE model.*

pression mode. As the coefficient range enforcement in Section 4.3 requires the expression weights to be within $[0, 1]$, we cannot use their compressed tensor directly. Instead, we re-compress their original data tensor using a simple matrix SVD along the identity mode, leaving the expression mode uncompressed. Specifically, the SVD "rotates" the original tensor and sorts its variance in decreasing order for the identity mode. We then truncate the insignificant components of the rotated tensor and get the reduced tensor C. We found that keeping 75 knobs for the identity provides satisfactory results in our experiments.

7 Experimental Results

We have implemented the described appproach in native C++ with OpenMP parallelization. The test system runs on a quad-core Intel Core i5 (3.0GHz) CPU, with an ordinary web camera producing 640×480 video frames at a maximum of 30 fps. The regressor training takes about 6 hours. For each video frame, our approach takes about $12ms$ to regress the shape vector and $3ms$ to post-process the regression result. Before the frame-invariant parameters (i.e., the camera matrix and the identity) converge, we also need an additional $5ms$ to execute the DEM adaptation. Our end-to-end frame rate is about 28 fps on average, bounded by the camera.

Our training images are from three public image datasets: FaceWarehouse [Cao et al. 2013b], LFW [Huang et al. 2007] and GTAV [Tarres and Rama]. Excluding the images that are too blurry to label, we collected $14,460$ images in total, each of which is manually



(a) (b) (c)

Figure 6: *For an input frame (a), the raw output from the regressor generates a physically implausible mesh (b). The post-processing improves the result by restricting the expression coefficients to the valid range as in (c).*

labeled. Table 1 shows the relevant statistics. All theses training images with labeled landmarks can be downloaded from our website[1].

Database	FW	LFW	GTAV
Individuals	150	3,010	44
Labeled images	5,904	7,258	1,298

Table 1: *Training data statistics.*

7.1 Algorithm Validation

The supplementary video provides a live demonstration of our system. As seen, any user can instantly be tracked once he (or she) enters the camera's field of view, without any calibration. The user switch does not cause any noticeable lag. The tracking results are accurate in terms of the 2D landmarks from the beginning, and the 3D shape quickly converges to match the user's facial geometry. The tracked head motions and expressions can be transferred to a digital avatar, producing convincing facial animation in real time. Note that none of the performers appeared in the video and paper was included in our training dataset.

In the following, we evaluate the main components in our approach. Comparisons with other algorithms are given in the next sub-section.

The DDE model. We compare our DDE model with three alternatives: raw 2D landmarks [Cao et al. 2012], the parametric model that does not incorporate the 2D displacements [Weng et al. 2013], and raw 3D landmarks [Cao et al. 2013a]. We use the same training process and datasets to train a regressor for each formulation. The resulting regressors are tested on the same video sequence. The ground truth landmark positions are manually labeled. Fig. 5(a) compares the tracking results generated by the four representations.

As shown, relying completely on raw 2D landmarks without any 3D constraints can generate implausible 2D shapes. While the parametric model is able to maintain a consistently plausible result, without displacements the tracked 2D landmarks cannot be accurately aligned with corresponding image features. Using raw 3D landmarks can improve the alignment. However, without a fixed identity constraint, the landmark depths cannot be accurately inferred, making the tracking results unstable. Fig. 5(b) shows a quantitative comparison of the tracking errors. As illustrated, our DDE model outperforms all the other representation by a fair margin.

We also study the effectiveness of the large number of images in training the DDE regressor. We learn three DDE regressors based randomly sampled image subsets, and then test them on the sequence shown in Fig. 5. The average errors (in pixels) for different image subsets are: 2.9 (4750 images), 2.2 (9540 images), and 1.5

[1] http://gaps-zju.org/DDE

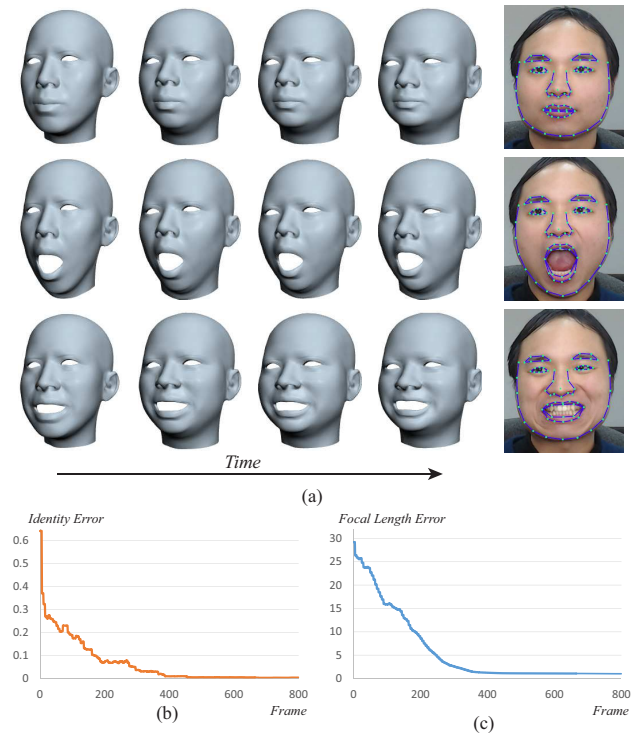


Figure 7: *DEM adaptation. Each row in (a) shows the progressive adaptation of a specific expression blendshape. We also plot the L^2 difference between the current identity vector (and the focal length) and the ground truth in (b) (and (c)). Both the identity and focal length converge quickly within* 400 *frames.*

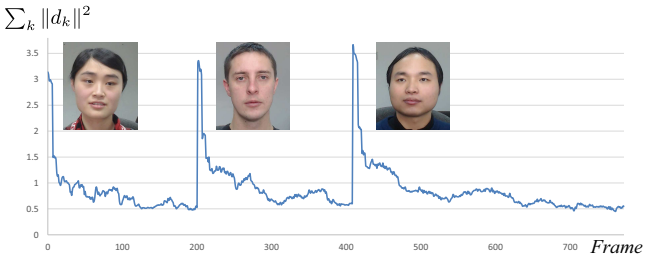
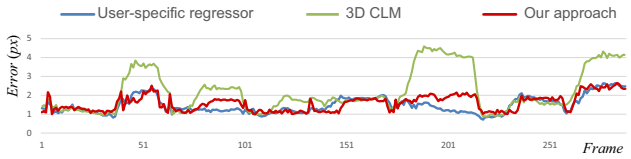


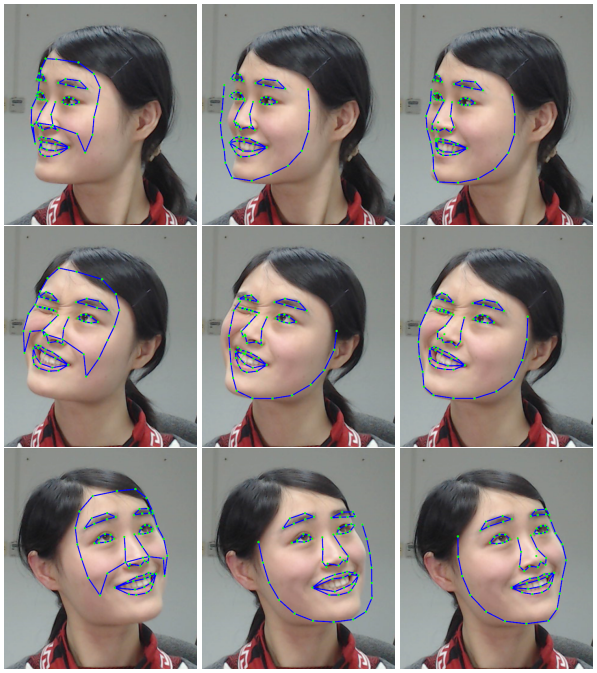
Figure 8: *DEM adaptation for three different users. The vertical spike in 2D displacements indicates that a new user has entered the camera's field of view. The 2D displacements decrease significantly as the frame-invariant parameters are updated, and quickly converge in several seconds.*

(all images). With more training images used, the regressor can provide more accurate tracking results. Note that the training set size does not have a significantly effect on the regressor size or the runtime processing speed.

Post-processing. In Fig. 6(a), we show the effects of our post-processing on the regression output. For an input frame shown in Fig. 6(a), the raw regressor output may produce out-of-bounds expression coefficients, as illustrated by the physically implausible mesh in Fig. 6(b). As discussed in Section 4.3, the post-processing step constrains the expression coefficients within the valid range, yielding a better result as shown in Fig. 6(c).



(a) Quantitative error curves



(b) Tracking results

Figure 9: *Comparisons with the user-specific algorithm [Cao et al. 2013a] and 3D CLM [Saragih et al. 2011a]. Our approach (right column) can generate accurate tracking results comparable to the user specific algorithm (left column), which needs* 60 *pre-captured facial images of the user to train a user-specific regressor. The 3D CLM (middle column) produces inaccurate results under large rotations. Note that the error computation in (a) is restricted to the internal landmarks shared by all three implementations, with the face contour excluded.*

7.2 Comparisons

DEM adaptation. In Fig. 7, we show the effects of the DEM adaptation. Starting from a deliberately inaccurate initialization of the camera matrix $\mathbf{Q}$ and the identity $\mathbf{u}$, our approach updates $\mathbf{Q}$ and $\mathbf{u}$ on the fly, and converges in approximately 400 frames. Fig. 7(b) and (c) illustrate the convergence trend of identity and focal length. As illustrated, both converge rapidly to near the respective final values within a few video frames. Note that to get the ground truth values of the identity and focal length, we took 60 images of the subject under a set of pre-defined pose and expression configurations, and used Cao et al. [2013a]'s approach to compute the identity and focal length.

We also validate the effectiveness of the adaptation by plotting the total displacement magnitude with respect to frame numbers. As illustrated in Fig. 8, the 2D displacements decrease significantly as $\mathbf{Q}$ and $\mathbf{u}$ become more accurate. The displacements, however, still exist even after $\mathbf{Q}$ and $\mathbf{u}$ converge to stable values. This is caused by the limited expressive power of the FaceWarehouse database and

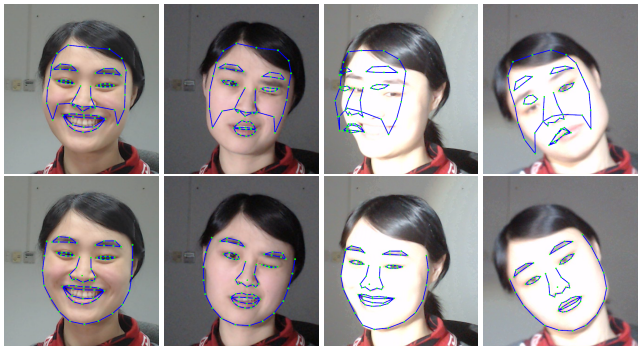


Figure 10: *Our approach (bottom row) is more robust than the user-specific algorithm [Cao et al. 2013a] (top row) under significant lighting changes.*

the blendshape model we used for 3D facial shapes. This fact, from the other side, supports our use of 2D displacements in the DDE model.

We compare our approach with two state-of-the-art techniques, the user-specific regression algorithm [Cao et al. 2013a] and the 3D CLM approach described in [Saragih et al. 2011a]. As in the previous section, we run all methods on a manually labeled video sequence and compare the computed 2D landmarks with the ground truth. For [Cao et al. 2013a], a user-specific regressor is trained using 60 images taken of the same subject immediately before the test video is recorded. The CLM model is trained using the same training images as in our approach. Our training data is substantially larger than the data used in the authors' implementation and the resulting model generates more accurate results.

Fig. 9 compares the tracking results of the three methods. As shown, the tracking accuracy of our approach is comparable to that of the user-specific method, while the 3D CLM approach produces inaccurate results for large rotations. Such inaccuracy is especially pronounced around the face contour, where local features are hard to distinguish. Note that the landmarks corresponding to the face contour significantly affect the fitting results of the identity and expressions. It is thus important to accurately locate their positions.

Fig. 10 compares our method with [Cao et al. 2013a] in handling lighting changes. If the current lighting is significantly different from that in the training images, the user-specific method may fail to get good tracking results (Fig. 10(a)). Based on a generic regressor trained from a large number of images taken under different lighting environments, our approach demonstrates better robustness under lighting changes (Fig. 10(b)).

Following [Cao et al. 2013a], we use the depth acquired from a Kinect camera to validate the accuracy of our approach. Specifically, we take an RGBD video from the Kinect camera and apply our approach to the color channels without using any depth information. We then reconstruct the 3D facial mesh F for each frame and compare the reconstructed depth values with the ground truth at a few representative vertices. As shown in Fig. 11, although the initial inaccurate identity and camera matrix created a noticeable difference between our reconstructed mesh and the acquired depth, the difference decreases to an insignificant level once the frame-invariant parameters converge through our DEM adaptation.

8 Conclusion

We have introduced a calibration-free approach to real-time facial tracking and animation with a single video camera. It works by

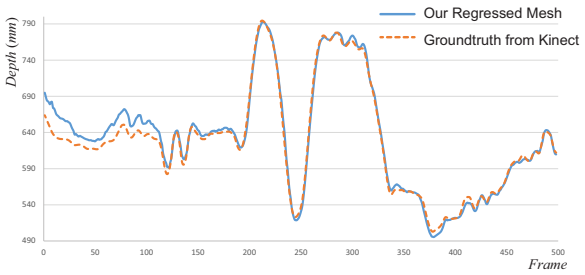


Figure 11: *Comparison of the depth of our 3D regression with the ground truth depth from Kinect. Here we use a vertex at the nose tip. Other vertices have similar curves.*

alternately performing a regression step to infer accurate 2D facial landmarks as well as the 3D facial shape from 2D video frames, and an adaptation step to correct the estimated camera matrix and the user identity (i.e., the expression blendshapes) for the current user. Our approach can achieve the same level of robustness, accuracy and efficiency as demonstrated in state-of-the-art tracking algorithms. We also contributed the DDE model, a new facial shape representation with the the combined advantages of 3D DEM and 2D landmarks. We consider our approach to be an attractive solution for wide deployment in consumer-level applications.

As a video based technique, our approach will fail to track the face if many of the facial features cannot be observed in the video frames. As shown in Fig. 12, our approach can handle some partial occlusions, but may fail if the face is largely occluded. Moreover, prolonged landmark occlusions during the DEM adaptation period may negatively impact the overall accuracy.

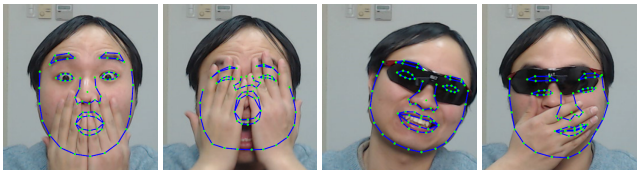


Figure 12: *Our approach can handle some partial occlusions, but may fail if the face is largely occluded.*

The 3D facial mesh reconstructed by our approach is optimized to match a set of facial features and does not contain high-frequency geometric details. If such details are required, one could extract them by sending our tracking result to an off-line shape-from-shading technique like [Garrido et al. 2013].

In the future, it would be interesting to see whether the DDE model can be applied to other problems, e.g., face recognition. The calibration-free nature of our approach can also facilitate multi-user scenarios, which user-specific approaches cannot handle. Finally, we plan to investigate how well our method could perform on mobile devices, where we would face additional challenges such as low image quality and a low computational budget.

Acknowledgements

We thank Xuchao Gong, Libin Hao and Siqiao Ye for making the digital avatars used in this paper, Eirik Malme, Carina Joergensen, Meng Zhu, Hao Wei, Shun Zhou and Yang Shen for being our performers, Steve Lin for proofreading the paper and the SIGGRAPH reviewers for their helpful comments. This work is partially sup-

ported by NSFC (No. 61103102 and No. 61272305) and the National Program for Special Support of Eminent Professionals.

References

ASTHANA, A., ZAFEIRIOU, S., CHENG, S., AND PANTIC, M. 2013. Robust discriminative response map fitting with constrained local models. In *IEEE CVPR*, 3444–3451.

BALTRUŠAITIS, T., ROBINSON, P., AND MORENCY, L.-P. 2012. 3D constrained local model for rigid and non-rigid facial tracking. In *Proceedings of IEEE CVPR*, 2610–2617.

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*, 4, 75:1–75:10.

BLANZ, V., AND VETTER, T. 1999. A morphable model for the synthesis of 3d faces. In *Proceedings of SIGGRAPH*, 187–194.

BOUAZIZ, S., WANG, Y., AND PAULY, M. 2013. Online modeling for realtime facial animation. *ACM Trans. Graph. 32*, 4 (July), 40:1–40:10.

BRADLEY, D., HEIDRICH, W., POPA, T., AND SHEFFER, A. 2010. High resolution passive facial performance capture. *ACM Trans. Graph. 29*, 4, 41:1–41:10.

BURGOS-ARTIZZU, X. P., PERONA, P., AND DOLLÁR, P. 2013. Robust face landmark estimation under occlusion. In *Proceedings of ICCV*, 117–124.

BYRD, R. H., LU, P., NOCEDAL, J., AND ZHU, C. 1995. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput. 16*, 5 (Sept.), 1190–1208.

CAO, X., WEI, Y., WEN, F., AND SUN, J. 2012. Face alignment by explicit shape regression. *Proceedings of IEEE CVPR*, 2887–2894.

CAO, C., WENG, Y., LIN, S., AND ZHOU, K. 2013. 3d shape regression for real-time facial animation. *ACM Trans. Graph. 32*, 4 (July), 41:1–41:10.

CAO, C., WENG, Y., ZHOU, S., TONG, Y., AND ZHOU, K. 2013. Facewarehouse: a 3D facial expression database for visual computing. *IEEE TVCG*, PrePrints.

CHAI, J.-X., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3d facial animation. In *Symp. Comp. Anim.*, 193–206.

COOTES, T. F., TAYLOR, C. J., COOPER, D. H., AND GRAHAM, J. 1995. Active shape models - their training and application. *Computer Vision and Image Understanding 61*, 38–59.

COOTES, T. F., EDWARDS, G. J., AND TAYLOR, C. J. 1998. Active appearance models. In *Proceedings of ECCV*, 484–498.

DECARLO, D., AND METAXAS, D. 2000. Optical flow constraints on deformable models with applications to face tracking. *Int. Journal of Computer Vision 38*, 2, 99–127.

DOLLAR, P., WELINDER, P., AND PERONA, P. 2010. Cascaded pose regression. In *Proceedings of IEEE CVPR*, 1078–1085.

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

ESSA, I., BASU, S., DARRELL, T., AND PENTLAND, A. 1996. Modeling, tracking and interactive animation of faces and heads: using input from video. In *Computer Animation*, 68–79.

GARRIDO, P., VALGAERT, L., WU, C., AND THEOBALT, C. 2013. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph. 32*, 6 (Nov.), 158:1–158:10.

HUANG, G. B., RAMESH, M., BERG, T., AND LEARNED-MILLER, E. 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep. 07-49, University of Massachusetts, Amherst, October.

HUANG, H., CHAI, J., TONG, X., AND WU, H.-T. 2011. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. *ACM Trans. Graph. 30*, 4, 74:1–74:10.

LEWIS, J. P., AND ANJYO, K. 2010. Direct manipulation blendshapes. *IEEE CG&A 30*, 4, 42–50.

LI, H., YU, J., YE, Y., AND BREGLER, C. 2013. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph. 32*, 4 (July), 42:1–42:10.

PIGHIN, F., HECKER, J., LISCHINSKI, D., SZELISKI, R., AND SALESIN, D. H. 1998. Synthesizing realistic facial expressions from photographs. In *Proceedings of SIGGRAPH*, 75–84.

PIGHIN, F., SZELISKI, R., AND SALESIN, D. 1999. Resynthesizing facial animation through 3d model-based tracking. In *Int. Conf. Computer Vision*, 143–150.

SARAGIH, J. M., LUCEY, S., AND COHN, J. F. 2011. Real-time avatar animation from a single image. In *IEEE International Conference on Automatic Face Gesture Recognition and Workshops*, 117–124.

SARAGIH, J., LUCEY, S., AND COHN, J. 2011. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision 91*, 2, 200–215.

TARRES, F., AND RAMA, A. GTAV Face Database. A`http://gps-tsc.upc.es/GTAV/ResearchAreas/UPCFaceDatabase/GTAVFaceDatabase.htm`.

VIOLA, P., AND JONES, M. 2004. Robust real-time face detection. *International Journal of Computer Vision 57*, 2, 137–154.

VLASIC, D., BRAND, M., PFISTER, H., AND POPOVIĆ, J. 2005. Face transfer with multilinear models. *ACM Trans. Graph. 24*, 3, 426–433.

WEISE, T., LI, H., GOOL, L. V., AND PAULY, M. 2009. Face/off: Live facial puppetry. In *Symp. Computer Animation*, 7–16.

WEISE, T., BOUAZIZ, S., LI, H., AND PAULY, M. 2011. Realtime performance-based facial animation. *ACM Trans. Graph. 30*, 4 (July), 77:1–77:10.

WENG, Y., CAO, C., HOU, Q., AND ZHOU, K. 2013. Real-time facial animation on mobile devices. *Graphical Models*, PrePrints.

WILLIAMS, L. 1990. Performance-driven facial animation. In *Proceedings of SIGGRAPH*, 235–242.

XIAO, J., BAKER, S., MATTHEWS, I., AND KANADE, T. 2004. Real-time combined 2d+3d active appearance models. In *Proceedings of IEEE CVPR*, 535–542.

XIONG, X., AND DE LA TORRE, F. 2013. Supervised descent method and its applications to face alignment. In *Proceedings of IEEE CVPR*, 532–539.

ZHANG, L., SNAVELY, N., CURLESS, B., AND SEITZ, S. M. 2004. Spacetime faces: high resolution capture for modeling and animation. *ACM Trans. Graph. 23*, 3, 548–558.