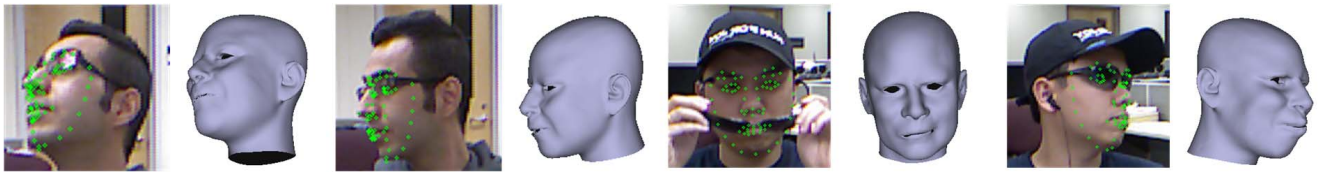
2016 Fourth International Conference on 3D Vision

Robust Real-Time 3D Face Tracking from RGBD Videos under Extreme Pose, Depth, and Expression Variations

Hai X. Pham Vladimir Pavlovic
Rutgers University, USA
{hxp1,vladimir}@cs.rutgers.edu



Abstract

We introduce a novel end-to-end real-time pose-robust 3D face tracking framework from RGBD videos, which is capable of tracking head pose and facial actions simultaneously in unconstrained environment without intervention or pre-calibration from a user. In particular, we emphasize tracking the head pose from profile to profile and improving tracking performance in challenging instances, where the tracked subject is at a considerably large distance from the camera and the quality of data deteriorates severely. To achieve these goals, the tracker is guided by an efficient multi-view 3D shape regressor, trained upon generic RGB datasets, which is able to predict model parameters despite large head rotations or tracking range. Specifically, the shape regressor is made aware of the head pose by inferring the possibility of particular facial landmarks being visible through a joint regression-classification local random forest framework, and piecewise linear regression models effectively map visibility features into shape parameters. In addition, the regressor is combined with a joint 2D+3D optimization that sparsely exploits depth information to further refine shape parameters to maintain tracking accuracy over time. The result is a robust on-line RGBD 3D face tracker that can model extreme head poses and facial expressions accurately in challenging scenes, which are demonstrated in our extensive experiments.

1. Introduction

Facial expressions tracking is an important task in computer vision, with recent methods [33, 4] achieving state-of-the-art results in capturing dynamic face performance using sophisticated, realistic 3D face models. Nonetheless, variations in pose, video quality, head movement and illumination, in addition to the richness of unique individual expressions, remain challenging factors that can adversely affect real-world tracking performance.

Among these problems, gross changes in pose (e.g., profile-to-profile), the distance of the face from the camera (beyond the limits of commodity sensors, resulting in low pixel/face area counts both in RGB and particularly depth [1]), and the expression of the face lead to articulated tracking challenges that have received less attention in the community. One reason for it is the difficulty in modeling nonlinear variations of the facial appearance at different angles, especially in (left, right) profile poses where half of the face details are missing. The most popular approach to deal with this is the multi-view face tracking/alignment, which uses a mixture of models to capture the varying poses [10, 39, 19, 41]. Other methods rely on optical flow to model the dynamics of facial movements [13], or combine with Iterative Closet Points (ICP) [20] procedure when the point cloud is available [33]. Pham et al. [23] have introduced an RGBD face tracker capable of tracking accurately at large distance, however it is not designed to model large head rotations. To the best of our knowledge, no prior work to date has addressed the difficulty in 3D face tracking at, simultaneously, large pose changes and large camera-face distance, where the face image has low resolution and is typically noisy, and the point cloud is sparse.

To solve this problem, which we deem the "extreme face tracking" problem, we propose a new unified RGBD face tracking framework. A critical aspect of this new framework for making the tracking robust across such "extreme" conditions is (1) the ability to accurately and rapidly determine the visibility of landmarks and (2) make the visibility estimation an integral part of the regressor training and online evaluation. Our approach generalizes the work in [23] to simultaneously capture 3D dynamic human facial perfor-

[1] For instance, assuming a typical human head size of $15cm \times 20cm$ (w x h), the first generation Kinect's RGB resolution of about 10×10 pixels per square degree (sd) suggests that a face at the distance of $200cm = 2m$ will be of about 40 pixels per linear dimension. The depth resolution will be even lower, about 20 pixels per linear dimension at 5×5 pixels/sd. Second generation Kinect roughly doubles the linear RGB resolution and increases the depth resolution by a factor of $7/5$.

978-1-5090-5407-7/16 $31.00 © 2016 IEEE
DOI 10.1109/3DV.2016.54

mance using dynamic expression models (DEM) as blendshapes, while being able to correctly handle extreme head poses.

Specifically, we propose two approaches to integrate pose and expression regression with visibility estimation. First, we leverage the local random forest framework of [23] to jointly predict landmark visibility and landmark displacement by a forest of decision trees that minimize a joint regression-classification entropy at no extra code. While traditional visibility estimation approaches, based on 3D object models, leverage geometric and deterministic visibility of surface points, they do not readily take into account the uncertainty of the head pose, motion, and the surface structure (expression, subject identity) that arise in tracking. As a consequence, landmarks predicted based on deterministic surface structure may be insufficiently accurate. In contrast, our forest framework can model the correlation between displacement of landmark and the probability of it being visible. These probability scores are used as the feature vector for the global parameter regression instead of the binary vector in [23]. Furthermore, a single regression mapping may not fit all data samples of different poses. Therefore we propose the second extension by learning a visibility specific mixture of regression experts, where the choice of the regressor is directly determined based on the visibility feature. Extensive experiments show that our 3D tracker performs robustly across a wide range of scenes, difficult poses and visual conditions, while maintaining or surpassing the tracking performance of other state-of-the-art methods.

2. Related Work

Early work on articulated face tracking was based on classical Active Shape and Appearance Models (ASM, AAM) [9, 8, 21] and Constrained Local Models (CLM) [28] that fit a parametric facial template to the image. A common extension of these traditional methods to multi-view face tracking/alignment is to build multiple models for separate head poses, such as multi-view AAM [10], multi-view Direct Appearance Models [19]. TSPM [41] uses local detectors similar to CLM, but learns sparse tree structures of parts for different views. Yu et al. [37] use sparse TSPM to initialize CLM fitting. In practice, these parametric methods rely on the learned statistical shape model and may not generalize well to real world data.

In recent years, non-parametric shape models have achieved better results due to greater flexibility and efficiency. In one approach [12, 29, 38], individual landmarks are directly localized by creating response map for landmark locations, and landmarks are finally chosen by a mode seeking method. The second approach is shape regression, which maps visual features to landmark coordinates [34, 7, 25]. Although these shape regression methods have achieved state-of-the-art performance, there is little effort in tackling multi-view face alignment. Recently, Xiong et al. [35] propose to learn multiple descent maps on different subsets of data to implicitly generalize shape models to different views.

Another popular approach is to use 3D deformable models, which are typically controlled by a set of deformation units, as priors. Past works [3, 13, 16, 24] employed simple, coarse 3D wireframe models, but they lack the presentation power to realistically model high-fidelity human face. More sophisticated models and registration techniques have been developed to obtain state-of-the-art 3D face reconstruction [1, 26, 27]. Zhu et al. [40] utilized these deformable models in a 3D dense face alignment framework across large poses using Convolutional Neural Networks.

More interestingly, recent approaches use 3D facial blendshapes [6] for real-time high-fidelity 3D face tracking. Such techniques have gained more attention lately due to the proliferation of affordable commodity depth sensing devices, such as the Kinect [22]. Several approaches [33, 2, 18, 31] based on blendshape DEM have demonstrated state-of-the-art tracking performance on RGBD input, or only depth input [17]. However, incoming data is assumed to be of high quality, thus they only work well in close range where fine details of the face are preserved. The RGB-based approaches by Cao et al. [5, 4] learn 3D shape regressor as priors for robust DEM registration. Although RGB-only methods are not affected by inaccurate depth measures, it is still challenging to track with high fidelity at large object-camera distances. This is in part due to reduced reliability of regression-based updates at lower image resolutions, when there is less data for overcoming depth ambiguity. Pham et al. [23] address that shortcoming by incorporating depth information into a hybrid RGBD tracking framework. These methods [5, 4, 23] are robust by using the trained regressors, however, the shape regressors are unable to handle large angle poses, such as left/right profiles.

Our work is based on [23], focusing on handling large head angles by training robust 3D shape regressors that can infer the 3D transformation accurately at arbitrary pose. The result is a pose-robust high fidelity 3D face tracker able to work in challenging situations, such as tracking face at large distance in a telepresence setting.

3. Method Overview

3.1. Face Representation

In our framework, we use the blendshape face models from the FaceWarehouse database [6]. Let $V \in \Re^{3\times N_v}$ be the set of N_v 3D vertices describing the geometric structure of the face during tracking. Then, a facial expression of a person can be approximated by

$$V = C_r \times_2 w_{id}^T \times_3 w_{exp}^T, \quad (1)$$

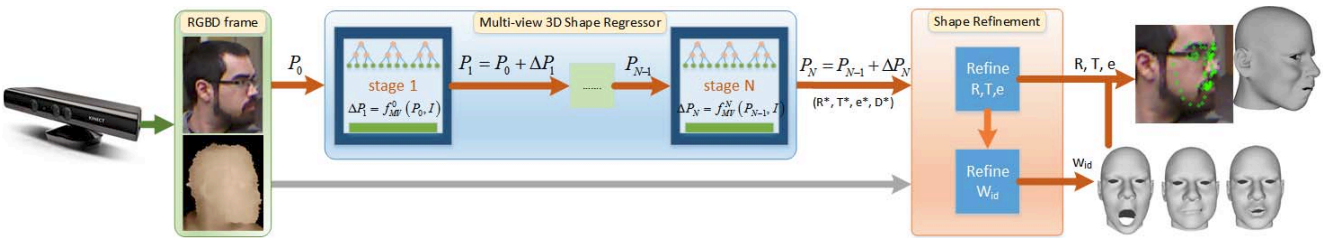


Figure 1. The multi-stage tracking pipeline. The multi-view regressor consists of several regression stages that are able to handle large poses. The fitted 3D blendshape and its corresponding 3D landmarks are shown on the top right corner; and three out of the 47 expression blendshape bases are displayed on the bottom right corner.

where $C_r \in \Re^{(3N_v)\times N_{id}\times N_e}$ is a 3D reduced core tensor matrix (corresponding to number of vertices, identities and expressions, respectively), $w_{id} \in \Re^{N_{id}}$ is a facial identity vector, $w_{exp} \in \Re^{N_e}$ is an expression vector, and $\times$ operators indicate the corresponding tensor products.

However, calculating a blendshape as a bilinear model in (1) is computationally expensive, because the identity of the person being tracked is constant in a large number of consecutive frames whereas her expressions can vary greatly. For real-time face tracking, given a particular identity vector w_{id} of a person, it is more convenient to reconstruct the N_e expression blendshapes for the person of identity w_{id} as

$$B_j = C_r \times_2 w_{id}^T \times_3 u_{\exp_j}^T, \tag{2}$$

where $u_{\exp_j}$ is a pre-computed ("basis") weight vector for the j-th expression mode. In this way, an arbitrary facial shape of the person can be represented as a linear sum of her expression blendshapes:

$$V = V(B, e) = B_0 + \sum_{j=1}^{N_e-1} (B_j - B_0)e_j, \tag{3}$$

where B_0 is the neutral shape, and $e_j \in [0, 1]$ is the blending weight of the j-th expression mode. Finally, a fully transformed 3D facial shape can be represented as

$$S = R \cdot V(B, e) + T, \tag{4}$$

with the parameters $\theta = (R, T, e)$, where R and T respectively represent global rotation and translation, and $e = \{e_j\}$ defined in (3) represents the expression deformation parameters.

3.2. Tracking Framework

Our tracker follows the coarse-to-fine design, as shown in Figure 1. First, the multi-view shape regressor, which consists of a cascade of regressions $\{f_{MV}^t\}_{t=1..N}$ learned from training data, rapidly but coarsely estimates shape parameters θ from the RGB frame (Section 4). In the second stage, a 2D+3D optimization is carried out to refine θ, and finally the identity parameter w_{id} is updated (if necessary) to improve shape fitting to the input RGBD frame (Section 5).

The 3D shape regressor is the key component of the tracker, which is robust to pose and distance and can predict 3D shape parameters directly from the RGB frame. Depth information is not used in the regressors as the depth quality of current commodity sensors decreases significantly at large distance and might introduce inaccuracies on the regressors as opposed to RGB images, which retain more consistency at different distances or resolutions.

The shape regressor does not predict identity parameters w_{id}. It instead embeds the errors between the projection of 3D landmark vertices and annotated 2D landmarks in the training data, which can be used to adapt the identity in a later stage [4, 23]. Indeed, the projection of N_l landmarks vertices of the 3D shape to the image plane typically does not accurately match the 2D landmarks annotated in the image. We therefore include 2D displacements $D = \{D_i\}$,

$$D_i = \Pi_p(S_i) - l_i, \tag{5}$$

where Π_p is the projection function, S_i is a 3D landmark vertex and l_i is the corresponding 2D landmark in the training data, into the parameter set and define a new global shape parameter vector $P = (\theta, D) = (R, T, e, D)$. The advantages of including D in regression are three-fold. First, it trains a heterogeneous regressor that can predict both 2D landmark positions $\{l_i^*\}$, similar to traditional 2D alignment methods [34, 25, 7], and 3D parameters θ. Second, the regressor can infer unseen identity and/or expression which does not appear in the training set by adjusting values of D. In such cases the displacement error D output may be large to compensate for the difference in identities/expressions. Third, these reconstructed 2D landmarks are subsequently used as target to adapt identity parameters w_{id}, in an optimization that minimizes the displacement errors D.

The coarse estimates by the regressor are refined further in the next stage, using energy optimization augmented with depth information. Specifically, $\theta = (R, T, e)$ are optimized to reduce both the error in 2D displacements to the target landmarks $\{l_i^*\}$ provided by the shape regressor, and the distance to the 3D point cloud. This additional depth information is beneficial, as it helps constrain the shape model, in particular its rigid parameters R & T, and prevents the drifting problem that often occurs during tracking a long sequence of complex movements. Lastly, the identity vector w_{id} is re-estimated given the current transformation

via another energy optimization as in (21). w_{id} would finally converge to stable values and remain constant for the rest of the tracking sequence of a specific person, therefore identity parameters are not included in the regression.

4. Multi-view 3D Shape Regression

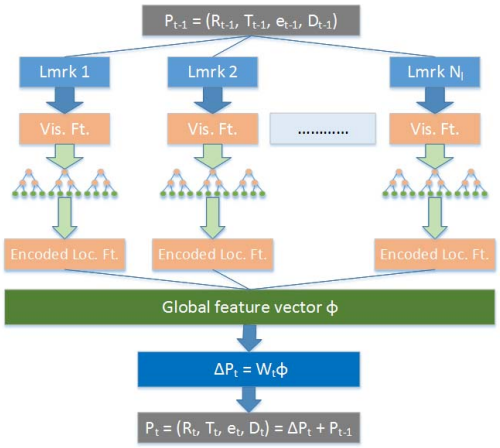


Figure 2. One regression stage, which updates global parameter vector P. It consists of two sub-stages: the local sub-stage encodes visual features around each individual landmark into local features, and the global sub-stage aggregates these local features to predict the parameter update ΔP_t.

Figure 2 graphically illustrates the underlying algorithm of one regression stage in the cascade proposed by Pham et al. [23], which is further improved in our work to handle large poses. The t^{th} regression stage predicts the parameter update $\Delta P_t = f^t_{MV}(P_{t-1}, I)$, where I is the input RGB frame and P_{t-1} is the output from the previous stage. Specifically, 2D landmark coordinates are first predicted from P_{t-1}. For each landmark l_i, pixel intensity-difference features [7] are randomly sampled within a radius r_i to train a local regression random forest $Forest_i$ [11], which predicts 2D displacement update for the i^{th} landmark (there are N_l local forests in total). r_i is scaled proportionally to face size in the training set, and inversely proportional to scene depth T_z at run-time, making the regressor robust to distance. Note that these local displacement updates $\{D_i\}$ are discarded, instead, the local forests encode visual features into local binary features. Indeed, every training sample x_k [2] is passed through the i-th local forest and recovers a binary vector $F_{k,i}$ whose length equals to the number of leaf nodes of $Forest_i$. Each leaf node that responds to x_k will be represented as 1 in $F_{k,i}$; otherwise the bit will be 0. Local binary vectors from N_l landmarks are concatenated to form a global binary vector Φ_k representing x_k and is used to learn a global linear regression matrix $\boldsymbol{W} \in \Re^{|P|\times|\Phi|}$ which predicts the parameter update $\Delta P_k = \boldsymbol{W} \cdot \Phi_k$. After that, P_k, including displacements D, is updated and enters the next iteration.

The aforementioned approach assumes that all landmarks must be visible in the camera field of view and does not take into account various out-of-plane head rotations, where a number of landmarks are self-occluded. Thus, the regressor cannot effectively predict shape parameters in such cases. One way to address this issue is to augment the regressor with landmark visibility to indirectly model large head poses. Specifically, the visibility of the i^{th} landmark is represented by a Bernoulli random variable $v_i \in \{0, 1\}$ together with its associated probability $p(v_i)$. Two different scenarios to integrate the landmark visibility information into the local regression framework are discussed in details in Section 4.1 and 4.2, and a novel global piecewise regression to improve the error rate is explained in Section 4.3.

4.1. 3D Pose-deterministic Shape Regression

As a baseline model for visibility assessment, we used a traditional visibility test based on fixed, deterministic structure of the current face geometry S. Specifically, we change the way local binary features $F_{k,i}$ are aggregated. Given current parameters θ_k, we calculate the 3D shape S_k, and use z-buffer to determine which landmarks of S_k are visible ($p(v_{k,i} = 1) = 1$), then the local binary features is

$$\hat{F}_{k,i} = F_{k,i} \cdot p(v_{k,i} = 1) \quad (6)$$

This modification trains a global regression $\boldsymbol{W}$ capable of inferring shape parameters despite missing landmarks.

4.2. Joint Landmark Visibility - Displacement Prediction Local Forest

The extension described in Section 4.1 is straightforward and works well in most cases. There are two drawbacks, however. First, calculating the 3D shape and z-buffer requires substantial computing power, especially with dense face models that we use, making the regressor less efficient. Second, if the transition between frames is not smooth enough, visibility of landmarks might not be predicted accurately. We therefore propose a data-driven approach, in which we train a joint classification-regression random forest [14] for each landmark i to predict both its 2D displacement D_i and the visibility v_i together with its associated probability $p(v_i)$. Random decision trees are trained using standard information gain maximization procedure. The information gain from a split at node $\mathbb{S}$ is defined as

$$I(\mathbb{S}) = H(\mathbb{S}) - \sum_{i\in\{Left,Right\}} \frac{|\mathbb{S}_i|}{|\mathbb{S}|} H(\mathbb{S}_i), \quad (7)$$

where $H(\mathbb{S})$ is the joint entropy of D and v[3]

$$H(\mathbb{S}) = -\sum_v \int_D p(D, v|x) \log p(D, v|x) dD. \quad (8)$$

[2] Only one particular regression stage is considered here, so the subscript t is dropped. For example, we denote P_k as the parameter vector of training sample x_k in this stage, instead of $P_{k,t}$.

[3] We drop subscripts i, k for clarity.

$H(\mathbb{S})$ can be decomposed into two terms, $H_v(\mathbb{S})$ and $H_r(\mathbb{S})$, which are the Shannon and joint differential entropy, respectively:

$$H_v(\mathbb{S}) = -\sum_v p(v|x)\log p(v|x), \quad (9)$$

$$H_r(\mathbb{S}) = -\sum_v p(v|x)\int_r p(D|v,x)\log p(D|v,x)dD, \quad (10)$$

where $p(D|v,x) \triangleq \mathcal{N}(D;\mu_{D|v},\Sigma_{D|v}|v,x)$, $\mu_{D|v}$ and $\Sigma_{D|v}$ are the mean and covariance of D w.r.t. v. Thus, the joint differential entropy H_r in (10) can be rewritten as

$$H_r = \sum_v p(v|x)\left(\frac{1}{2}\log\left[(2\pi e)^2\left|\Sigma_{D|v}\right|\right]\right). \quad (11)$$

$H_v(\mathbb{S})$ and $H_r(\mathbb{S})$ have different value ranges. In order to balance two tasks of classification and regression in training, we calculate a normalized joint entropy term $\mathcal{H}(\mathbb{S})$ over the entropies of root node $\mathbb{S}_0$:

$$\mathcal{H}(\mathbb{S}) = \frac{1}{2}\left(\frac{H_v(\mathbb{S})}{H_v(\mathbb{S}_0)} + \frac{H_r(\mathbb{S})}{H_r(\mathbb{S}_0)}\right). \quad (12)$$

Local features $F_{k,i}$ are extracted as before, but the 1 bit is replaced by the probability $p_\tau\left(v_i = 1|x_k\right)$,

$$F_{k,i} = p_\tau\left(v = 1|x_k\right) = \frac{\aleph_{\ell(x_k)}(v=1)}{\sum_{v=0,1}\aleph_{\ell(x_k)}(v)}, \quad (13)$$

of a landmark being visible at the leaf node $\mathbb{S}_{\ell(x_k)}$ of tree τ where x reaches; the leaf indicated by index $\ell(x_k)$ contains histogram $\aleph$ of v. Global features Φ_k are then collected to learn the global regression matrix $\boldsymbol{W}$.

4.3. Piecewise Global Regression

A single global linear regression matrix $\boldsymbol{W}$ as described above may not be able to map the feature space to the parameter space correctly due to the nonlinearly varying poses in training data. We exploit the collection of landmark visibility probability from Section 4.2 to partition the training data into different subsets, to form a series of piecewise linear regressions to better model the data.

The averaged probability of landmark i of sample x_k being visible is measured as

$$\bar{p}_{k,i} = \bar{p}(v_i = 1|x_k) = \frac{1}{\mathbb{T}_i}\sum_\tau p_\tau(v_i = 1|x_k), \quad (14)$$

where $\mathbb{T}_i$ is the number of trees in the local forest of landmark i. Probabilities $\{\bar{p}_{k,i}\}$ from N_l landmarks are aggregated into a feature vector $\bar{p}_k$ to be used in clustering. The data $(\bar{p}_1,..,\bar{p}_N)$ is split into N_c subsets $\{\bar{P}_1,..,\bar{P}_{N_c}\}$ with centers $\{\mu_1,..,\mu_{N_c}\}$. Finally, N_c regression matrices $\{W_c\}_{c=1}^{N_c}$ are learned, one for each subset of data, similar to [35].

To determine cluster assignments at run-time, we use the cluster representatives μ_i in the visibility feature space. Specifically, the most likely cluster that the incoming sample $\hat{p}$ belongs to is determined using nearest neighbor search:

$$c^*(\hat{p}) = \arg\min_c \|\hat{p} - \mu_c\|. \quad (15)$$

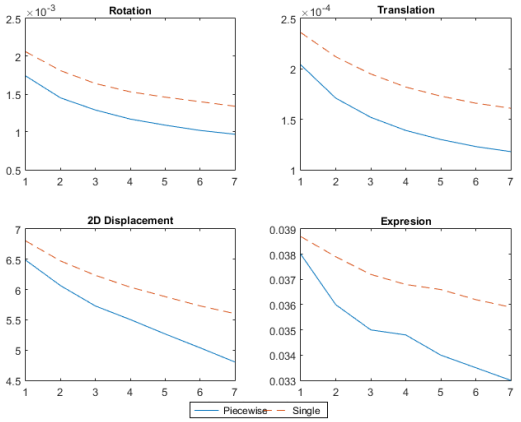


Figure 3. Comparing training error rates of single and piecewise regressions across seven iterations. The curves show *squared* errors w.r.t. different parameters. Rotation is represented by a unit quaternion, whereas translation is measured in meters.

Figure 3 demonstrates the advantage of using piecewise regression, leading to lower error rate compared to using a single linear regression across all training samples, especially in the case of translation, which is particularly important in tracking fast moving faces.

5. Facial Shape Refinement

At this stage, we further refine the shape parameters θ fully utilizing both the regressor output (θ^*, D^*) and depth image, and also update the shape identity.

First, we refine θ by alternatingly optimizing the following energy w.r.t. (R,T) and e:

$$\widehat{R},\widehat{T},\widehat{e} = \arg\min_{R,T,e} E_{2D} + \omega_{3D}E_{3D} + E_{reg}, \quad (16)$$

where ω_{3D} is a tradeoff parameter, E_{2D} is the 2D error term measuring the 2D displacement errors, E_{3D} is the 3D ICP energy term measuring the geometric matching between the 3D face shape model and the input point cloud, and E_{reg} is the regularization term to ensure the shape parameter refinement is smooth across the time. Particularly, E_{2D}, E_{3D} and E_{reg} are defined as

$$E_{2D} = \frac{1}{N_l}\sum_{i=1}^{N_l}\|\Pi_p\left(S_i\left(R,T,e\right)\right) - l_i^*\|^2, \quad (17)$$

$$E_{3D} = \frac{1}{N_d}\sum_{k=1}^{N_d}\left(\left(S_k\left(R,T,e\right) - d_k\right)\cdot n_k\right)^2, \quad (18)$$

$$E_{reg} = \alpha\|\theta - \theta^*\|^2 + \beta\left\|\theta - 2\theta^{(-1)} + \theta^{(-2)}\right\|^2 \quad (19)$$

In (17), the 2D landmarks $\{l_i^*\}$ are computed from the raw regressor output (θ^*, D^*). In (18), N_d is the number of ICP corresponding pairs that we sample from the blendshape and the point cloud, and d_k and n_k denote point k in

the point cloud and its normal, respectively. In (19), $\theta^{(-1)}$ and $\theta^{(-2)}$ are the shape parameters from the previous two frames, and α and β are tradeoff parameters. The non-linear energy function is minimized using bounded gradient descent to keep e in the valid range of [0,1].

In the last step, the identity paramaters are updated to better adapt the expression blendshapes to the true appearance of the currently tracked subject. We solve for w_{id} by minimizing the following objective function, using coordinate descent:

$$\widehat{w}_{id} = \arg\min_{w_{id}} \omega_{2D} E_{id2D} + \omega_{3D} E_{id3D}, \tag{20}$$

where

$$E'_{2D} = \frac{1}{N_l} \sum_{i=1}^{N_l} \left\| \Pi_p \left(R\left(C_r \times_2 w_{id}^T \times_3 \gamma^T\right)_i + T \right) - l_i^* \right\|^2,$$

$$E'_{3D} = \frac{1}{N_d} \sum_{k=1}^{N_d} \left\| R\left(C_r \times_2 w_{id}^T \times_3 \gamma^T\right)_k + T - d_k \right\|^2, \tag{21}$$

with $\gamma = (1 - \sum_{j=1}^{N_e-1} e_j) u_{\exp_0} + \sum_{j=1}^{N_e-1} e_j u_{\exp_j}$.

6. Experiments

6.1. Regressor Training Data

We use real RGB image samples from the FaceWarehouse [6], Labeled Face in the Wild [15] and GTAV [30] datasets. However, these datasets do not contain samples depicting large head poses. Thus, after fitting the 3D blendshape model to each sample to extract shape parameters [23], the sample image is artificial triangulated around the 3D head shape and rotated to create synthetic large-posed images from real images. We also render synthetic large-posed samples from the BU4DFE database [36] in order to increase the variation of face poses in the training set.

6.2. Experiment Settings

We carried out extensive tracking experiments on synthetic RGBD sequences and real videos captured by a Kinect(v1) camera. We compared the tracking performance of our two proposed models: the tracker with single regression is henceforth denoted as **3DLGSR** while the one using the piecewise regression is denoted as **3DLGMR**, against the baseline tracker using 3D geometry described in Section 4.1, called **3DLGR-Dt**, as well as the near-frontal 3D face tracker 3DLGR [23]. We measure the 2D landmark error as RMSE (in pixels) w.r.t. 2D ground truth landmarks. Due to the lack of a publicly available profile-to-profile 3D face tracking software, we alternatively compare our methods to other 2D face alignment methods PMCDS [37] and TSPM [41], and a recent 3D dense face alignment method 3DDFA [40]. All three methods have been proven as capable of profile face registration. Particularly, PMCDS is configured in tracking mode, in which the facial landmarks of the previous frame are used to initialize the registration algorithm on the new frame. On the other hand, 3DDFA requires the exact bounding box of face region to be provided for each frame in order to work properly. We use the pretrained models provided by the authors in all of our tests.

6.2.1 Implementation Details and Running Time

In the first frame when the tracker starts or restarts, the face is localized using the OpenCV face detector [32]. The tracker is written in native C++, parallelized with Intel Thread Building Blocks (TBB). We measure the running time of three tracker implementations (each shape regressor has five sub-stages), excluding the identity adaptation process in (2) whose speed depends on the GPU (about 7ms on a Tesla K40c). Specifically, tested on an Intel Core i7 quad-core 3.4GHz CPU, 3DLGSR and 3DLGMR can process one frame in roughly 30ms, they are as fast as the 3DLGR tracker proposed by Pham et al. [23]. The baseline 3DLGR-Dt using z-buffer is slower, since each regression sub-stage takes additional 4ms to perform depth test in our implementation.

6.3. Evaluation on Synthetic Data

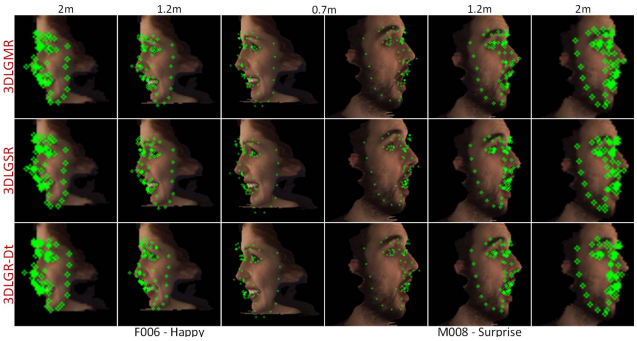


Figure 4. Tracking results on two sample frames from the BU4DFE dataset, rendered at different distances. From top to bottom, the figure shows results of 3DLGMR, 3DLGSR and 3DLGR-Dt, respectively. Sub-images were rescaled to the same size, therefore the landmark circles in the 2m samples appear larger than those in the 1.2m samples, which in turn are larger than those in the 70cm samples. Notice the differences in eyebrow and mouth areas in the 70cm frames, where 3DLGMR accurately registers both poses and mouth deformations.

The BU4DFE dataset [36] contains sequences of high-resolution 3D dynamic facial expressions. We rendered these sequences into RGBD to simulate the Kinect(v1) [22] at six distances: 70cm, 1.2m, 1.5m, 1.75m, 2m and 2.5m with profile-to-profile head rotations, hence we can observe the same sequence of facial movements at different distances and in different resolutions as shown in Figure 4. In total, we collected tracking results from 480 sequences. The dataset does not provide ground truth, hence we used

Table 1. Evaluation results on the profile-to-profile BU4DFE sequences. *RMSE* is measured in raw pixels, best results are marked in bold. The 2D error becomes smaller as the scene depth increases because of the camera perspective projection. TSPM did not work with low-resolution face images.

Dataset	PMCDS[37]	3DDFA[40]	TSPM[41]	3DLGR[23]	**3DLGR-Dt**	**3DLGSR**	**3DLGMR**
BU4D (70 cm)	24.05 ± 9.68	3.86 ± 0.69	5.62 ± 0.82	5.85 ± 4.38	4.30 ± 1.01	4.59 ± 1.02	**3.81** ± 0.95
BU4D (1.2 m)	14.90 ± 7.60	2.36 ± 0.41	3.68 ± 0.48	2.88 ± 1.31	2.37 ± 0.59	2.51 ± 0.62	**2.28** ± 0.61
BU4D (1.5 m)	12.08 ± 6.54	1.96 ± 0.34	3.75 ± 0.46	2.32 ± 0.85	1.90 ± 0.42	1.99 ± 0.39	**1.72** ± 0.33
BU4D (1.75 m)	10.42 ± 5.85	1.74 ± 0.30	n/a	2.15 ± 1.50	1.76 ± 0.58	1.81 ± 0.44	**1.55** ± 0.31
BU4D (2m m)	9.31 ± 6.70	1.57 ± 0.27	n/a	2.65 ± 2.38	1.68 ± 0.74	1.83 ± 0.91	**1.38** ± 0.23
BU4D (2.5 m)	7.63 ± 8.35	**1.36** ± 0.23	n/a	2.64 ± 2.38	1.61 ± 0.38	1.74 ± 0.59	1.39 ± 0.28

Table 2. Evaluation results on real profile-to-profile sequences. *RMSE* was measured in raw pixel values. PMCDS performed poorly on *ro01* amd *sj01*, thus its results are not included. TSPM also did not work with *sj01*, since this sequence captures low-resolution face.

Dataset	PMCDS[37]	3DDFA[40]	TSPM[41]	3DLGR[23]	**3DLGR-Dt**	**3DLGSR**	**3DLGMR**
ad01	4.23 ± 5.83	4.67 ± 2.76	**3.69** ± 1.57	3.93 ± 3.09	4.26 ± 2.28	3.96 ± 1.76	3.89 ± 1.82
bn01	6.74 ± 5.89	2.73 ± 2.06	3.86 ± 1.39	2.53 ± 1.12	2.68 ± 0.82	2.79 ± 0.93	**2.36** ± 0.62
bn02	3.71 ± 2.62	7.03 ± 4.25	3.36 ± 0.52	**2.74** ± 1.29	3.43 ± 1.79	3.69 ± 2.92	3.26 ± 1.73
jp01	3.08 ± 3.32	3.90 ± 2.45	**2.94** ± 0.68	3.13 ± 2.62	3.60 ± 3.34	6.09 ± 7.62	**2.95** ± 1.07
ro01	n/a	8.29 ± 6.68	**5.33** ± 6.26	7.94 ± 8.83	8.30 ± 11.02	12.54 ± 15.76	6.21 ± 2.02
sj01	n/a	5.04 ± 2.01	n/a	3.11 ± 2.09	**1.49** ± 0.53	1.90 ± 0.96	1.76 ± 0.76
sj02	3.49 ± 1.21	5.07 ± 2.54	3.70 ± 1.57	3.24 ± 1.12	**2.55** ± 0.62	2.99 ± 0.80	2.89 ± 0.72
sy01	7.17 ± 5.53	8.11 ± 4.94	4.06 ± 1.62	3.37 ± 1.19	3.71 ± 1.54	3.24 ± 1.04	**3.08** ± 0.81

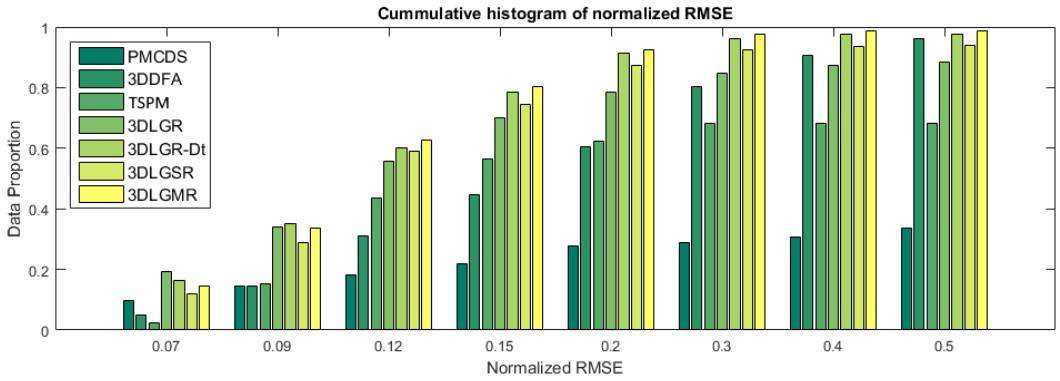


Figure 5. Cummulative Error Historgrams of seven methods. Overall, 3DLGMR achieves the best tracking performance, followed by 3DLGR-Dt. In fact, 3DLGMR has the fastest CEH growth before reaching equilibrium of 98.6% at the 0.34 error bin, while 3DLGR-Dt reaches 97.7% at 0.36.

the 3DLGR tracker [23] to recover 3D landmarks on the frontal images of stationary head pose rendered at 0.7m, which were then rotated, readjusted using ICP, and reprojected to different distances and treated as ground truth.

As shown in Table 1, 3DLGMR achieves the lowest RMSE on average compared to other blendshape-based methods, including 3DLGSR. It also performs the best overall on sequences at distances ranging from 70cm to 2m, and only slightly worse than 3DDFA at 2.5m. In general, performance of 3DLGSR is not as good as 3DLGR-Dt and 3DLGMR as expected, because single regression model is not fully capable of modeling highly non-linear input features, while 3DLGR-Dt only takes binary features. Figure 4 demonstrates a few sample frames from the test set, where 3DLGMR was able to achieve stable and highly accurate tracking results. These results reflect the advantage of combining local visibility prediction with piecewise linear regression models to predict shape parameters.

6.4. Evaluation on Real Data

Eight RGBD sequences capturing facial expressions, profile-to-profile head movements at different distances were recorded with a Kinect(v1) camera for testing, and ground truth landmarks were manually annotated on one among every 10 frames. The tracking results on these videos are shown in Table 2.

In these tests, 3DLGMR has the best results or only slightly below the best performers. Specifically, 3DLGR-Dt has slightly smaller errors on *sj01* and *sj02*. The fact that the tracked subjects in these two videos have darker skin tones, which affect the visibility prediction, may explain the lower performance of 3DLGMR as it maps the visibility features into shape parameters. Additionally, TSPM achieves lower average errors on *ad01*, *jp01* and *ro01* sequences, compared to 3DLGMR. This reflects the nature of the technique, because TSPM performs exhaustive search across all possible poses, thus it performs well when it is able to localize the

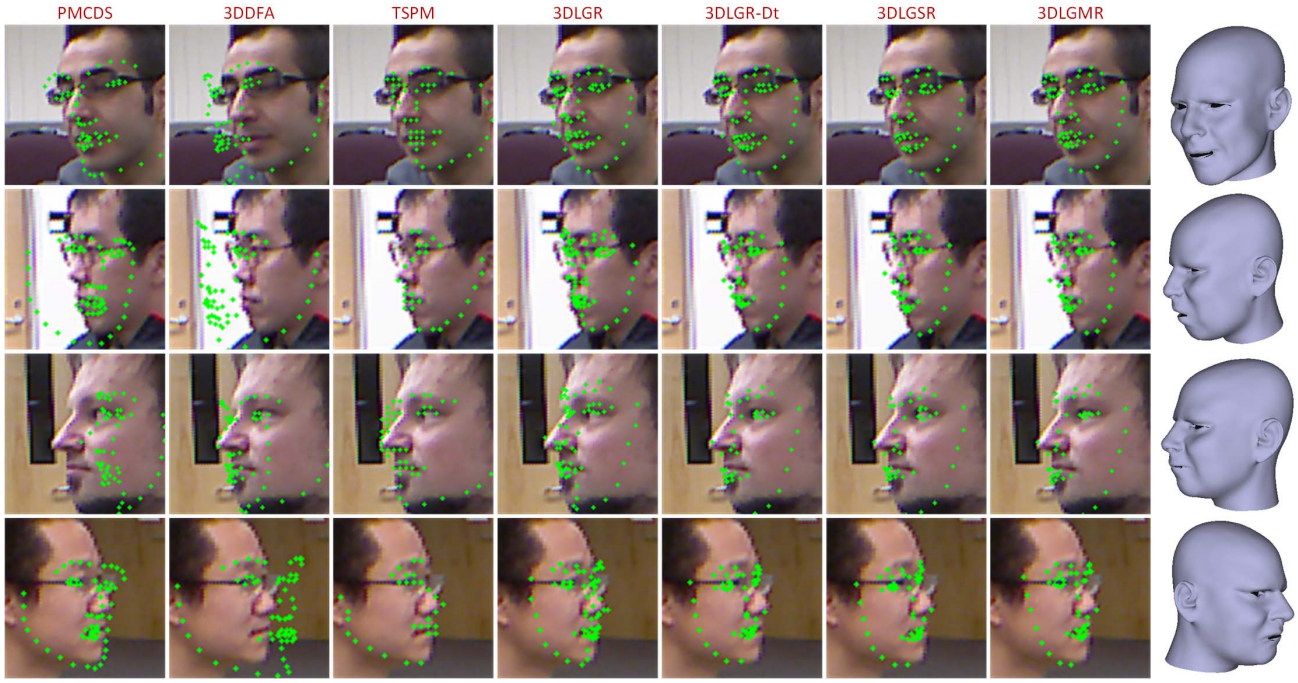


Figure 6. A few samples from real sequences. The 3D facial blendshapes produced by 3DLGMR are visualized in the last column.

Table 3. Tracking errors on large-posed frames from different datasets. 3DLGMR has substantial gain in majority of tests, except for *sj01*, *sj02* and *sy01*. In general, errors on large-posed frames are indeed larger than the average errors in Table 1 and 2.

Dataset	**3DLGR-Dt**	**3DLGSR**	**3DLGMR**
BU4D(70cm)	4.98 ± 1.07	5.14 ± 1.12	**3.97** ± 0.74
BU4D(1.2m)	2.69 ± 0.66	2.74 ± 0.62	**2.27** ± 0.62
BU4D(1.5m)	2.14 ± 0.49	2.16 ± 0.39	**1.77** ± 0.33
BU4D(1.75m)	1.97 ± 0.69	2.12 ± 1.10	**1.58** ± 0.33
BU4D(2m)	1.96 ± 0.86	1.99 ± 0.83	**1.59** ± 0.74
BU4D(2.5m)	1.95 ± 1.23	2.05 ± 1.30	**1.61** ± 0.83
ad01	5.50 ± 1.94	4.77 ± 1.45	**4.64** ± 1.48
bn01	2.79 ± 0.83	2.95 ± 0.89	**2.56** ± 0.57
bn02	3.50 ± 0.81	3.42 ± 0.69	**3.31** ± 1.23
jp01	4.17 ± 1.19	7.36 ± 7.97	**3.14** ± 0.68
ro01	9.11 ± 11.99	14.27 ± 15.56	**6.59** ± 2.05
sj01	**1.82** ± 0.54	2.55 ± 1.10	2.22 ± 0.93
sj02	**2.76** ± 0.60	3.35 ± 0.90	2.81 ± 0.60
sy01	4.30 ± 1.70	**3.38** ± 1.21	3.43 ± 0.65

face position, albeit at the cost of being considerably slower than 3DLGMR.

These errors were calculated from only successfully registered frames that do not account for tracking loss, which would very likely happen in unconstrained settings. To better understand the tracking performance, we measure additional Cummulative Error Histogram (CEH) metrics. CEH quantizes tracking error into κ bins, where the κ^{th} bin counts the number of frames with the error less than a threshold ϵ_k. The errors must be normalized over face size, which is specified as distance between outer-eye and mouth corners in our work. CEH is shown in Figure 5.

In smaller error bins, both 3DLGR and 3DLGR-Dt have a slight advantage over 3DLGMR, but overall, 3DLGMR has the best histogram among all tested methods. This can be explained as these small error bins consist of near-frontal frames with which 3DLGR has been demonstrated to achieve really good performance. However, 3DLGR incurs larger errors in frames with large poses. 3DLGR-Dt is better at handling these poses, but with additional cost of using z-buffer. 3DLGMR performs better than 3DLGR-Dt in some cases, whilst retaining the computational advantage of 3DLGR. Table 3 shows that 3DLGMR outperforms 3DLGR-Dt in majority of frames with large head poses. All experiments on real RGBD videos demonstrate that 3DLGMR performs consistently well across different pose variations.

7. Conclusion

We presented a novel RGBD face tracking framework that uses 3D facial blendshapes to simultaneously model head movements, including profile poses, as well as facial expressions in unconstrained environment. The tracker is driven by an efficient multi-view shape regressor, which is robust to distance and pose by taking visibility of landmarks into account in the training procedure. The global regression model is further extended to piecewise linear regression to better capture the varying head poses. Through extensive experiments on synthetic and real RGBD videos, our tracker performed consistently well in complex conditions. Being fully automatic, our tracker can be readily deployed into various tasks in human machine interaction or virtual reality.

References

[1] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[2] S. Bouaziz, Y. Wang, and M. Pauly. Online modeling for realtime facial animation. In *SIGGRAPH*, 2013.

[3] Q. Cai, D. Gallup, C. Zhang, and Z. Zhang. 3d deformable face tracking with a commodity depth camera. In *Europ. Conf. Comput. Vision (ECCV)*, 2010.

[4] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. In *SIGGRAPH*, 2014.

[5] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3d shape regression for real-time facial animation. In *SIGGRAPH*, 2013.

[6] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. FaceWarehouse: A 3D Facial Expression Database for Visual Computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, March 2014.

[7] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In *CVPR*, 2012.

[8] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE Trans. Pat. Anal. Mach. Intel.*, (23):681–684, 2001.

[9] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models - their tranining and applications. *Comput. Vis. Image Underst.*, (61):39–59, 1995.

[10] T. F. Cootes, G. V. Wheeler, K. N. Walker, and C. J. Taylor. View-based active appearance models. *Image and Vis. Comput.*, 20(9):657–664, 2002.

[11] A. Criminisi and J. Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. 2013.

[12] M. Dantone, J. Gall, G. Fanelli, and L. V. Gool. Real-time facial feature detection using conditional regression forests. In *CVPR*, 2012.

[13] D. DeCarlo and D. Metaxas. Optical flow constraints on deformable models with applications to face tracking. *Int. J. Comput. Vis.*, 38(2):99–127, July 2000.

[14] B. Glocker, O. Pauly, E. Konukoglu, and A. Criminisi. Joint classification-regression forests for spatially structured multi-object segmentation. In *ECCV 2012 - 12th European Conference on Computer Vision*. Springer, 2012.

[15] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[16] J. G. J. Orozco, O. Rudovic and M. Pantic. Hierarchical online appearance-based tracking for 3D head pose, eyebrows, lips, eyelids and irises. *Image and Vis. Comput.*, 31(4):322–340, 2013.

[17] V. Kazemi, C. Keskin, J. Taylor, P. Kohli, and S. Izadi. Real-time face reconstruction from a single depth image. 3DV, 2014.

[18] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. In *SIGGRAPH*, 2013.

[19] S. Z. Li, Y. Shuicheng, H. Zhang, and Q. Cheng. Multi-view face alignment using direct appearance models. In *Automatic Face and Gesture Recognition*, pages 324–329, 2002.

[20] K. Low. Linear least-squares optimization for point-to-plane ICP surface registration. Technical Report TR04-004, Department of Computer Science, University of North Carolina at Chapel Hill, 2004.

[21] I. Matthews and S. Baker. Active appearance models revisited. *Int. J. Comput. Vis.*, 60(2):135–164, 2004.

[22] C. Mutto, P. Zanuttigh, and G. Cortelazzo. Microsoft Kinect™ range camera. In *Time-of-Flight Cameras and Microsoft Kinect™*, SpringerBriefs in Electrical and Computer Engineering, pages 33–47. Springer US, 2012.

[23] H. X. Pham, C. Chen, L. N. Dao, V. Pavlovic, J. Cai, and T.-J. Cham. Robust performance-driven 3d face tracking in long range depth scenes. In *ArXiv*, 2015.

[24] H. X. Pham and V. Pavlovic. Hybrid On-line 3D Face and Facial Actions Tracking in RGBD Video Sequences. In *22nd International Conference on Pattern Recognition, ICPR*, pages 4194–4199, 2014.

[25] S. Ren, X. Cao, Y. Wei, and J. Sun. Face alignment at 3000 fps via regressing local binary features. In *CVPR*, 2014.

[26] S. Romdhani and T. Vetter. Efficient, robust and accurate fitting of a 3d morphable model. In *ICCV*, 2003.

[27] S. Romdhani and T. Vetter. Estimating 3d shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In *CVPR*, 2005.

[28] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, 91(2):200–215, 2011.

[29] B. M. Smith, J. Brandt, Z. Lin, and L. Zhang. Nonparametric context modeling of local appearance for pose-and expression-robust facial landmark localization. In *CVPR*, pages 1741–1748, 2014.

[30] F. Tarrs and A. Rama. GTAV face database. `https://gtav.upc.edu/research-areas/face-database`.

[31] D. Thomas and R.-I. Taniguchi. Augmented blendshapes for real-time simultaneous 3d head modeling and facial motion capture. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[32] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages I–511 – I–518, 2001.

[33] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. In *SIGGRAPH*, 2011.

[34] X. Xiong and F. D. la Torre. Supervised descent method and its applications to face alignment. In *CVPR*, 2013.

[35] X. Xiong and F. D. la Torre. Global supervised descent method. In *CVPR*, 2015.

[36] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale. A high-resolution 3d dynamic facial expression database. In *8th IEEE International Conference on Automatic Face Gesture Recognition*, pages 1–6. IEEE, Sept 2008.

[37] X. Yu, J. Huang, S. Zhang, W. Yan, and D. N. Metaxas. Pose-free facial landmark fitting via optimized part mixtures and cascaded deformable shape model. In *ICCV*, 2013.

[38] F. Zhou, J. Brandt, and Z. Lin. Exemplar-based graph matching for robust facial landmark localization. In *ICCV*, 2013.

[39] Y. Zhou, W. Zhang, X. Tang, and H. Shum. A bayesian mixture model for multi-view face alignment. In *CVPR*, volume 2, pages 741–746, 2005.

[40] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li. Face alignment across large poses: A 3d solution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[41] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, pages 2879–2886, 2012.