Realtime 3D Eye Gaze Animation Using a Single RGB Camera

Congyi Wang*# Fuhao Shi† Shihong Xia* Jinxiang Chai†
* Institute of Computing Technology (Chinese Academy of Sciences)
# University of Chinese Academy of Sciences †Texas A&M University

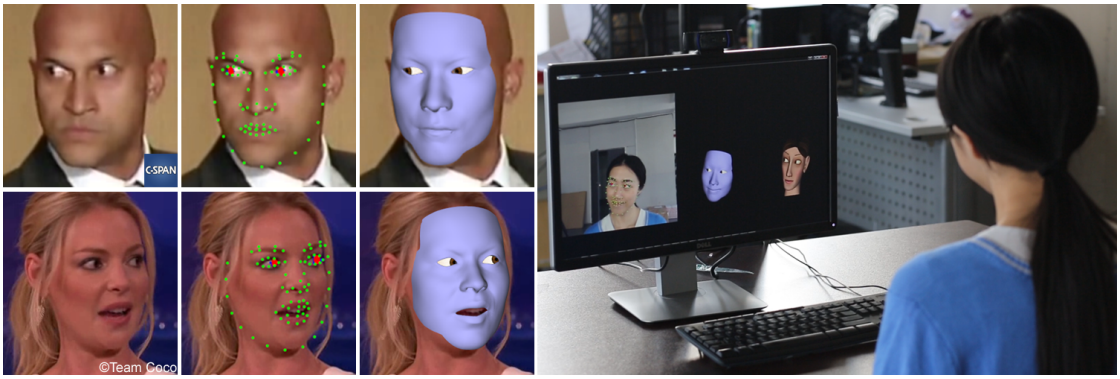


Figure 1: *Our realtime system automatically captures 3D facial and eye gaze performances using monocular video sequences: (left) the input videos downloaded from the Internet (column 1), the detected 2D facial features (green dots) and the classified iris and pupil pixels (red pixels) (column 2) and the captured 3D head poses, facial expression and eye gaze (column 3); (right) our system is running on a live stream.*

Abstract

This paper presents the first realtime 3D eye gaze capture method that simultaneously captures the coordinated movement of 3D eye gaze, head poses and facial expression deformation using a single RGB camera. Our key idea is to complement a realtime 3D facial performance capture system with an efficient 3D eye gaze tracker. We start the process by automatically detecting important 2D facial features for each frame. The detected facial features are then used to reconstruct 3D head poses and large-scale facial deformation using multi-linear expression deformation models. Next, we introduce a novel user-independent classification method for extracting iris and pupil pixels in each frame. We formulate the 3D eye gaze tracker in the Maximum A Posterior (MAP) framework, which sequentially infers the most probable state of 3D eye gaze at each frame. The eye gaze tracker could fail when eye blinking occurs. We further introduce an efficient eye close detector to improve the robustness and accuracy of the eye gaze tracker. We have tested our system on both live video streams and the Internet videos, demonstrating its accuracy and robustness under a variety of uncontrolled lighting conditions and overcoming significant differences of races, genders, shapes, poses and expressions across individuals.

Keywords: 3D eye gaze tracking, facial performance capture, facial animation and control

Concepts: •**Computing methodologies** → **Animation; Virtual reality;** •**Human-centered computing** → **Gestural input;**

SIGGRAPH '16 Technical Paper,, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925947

1 Introduction

Facial animation is an essential component of many applications, such as movies, video games and virtual environments. Thus far, one of the most popular and successful approaches for creating virtual faces often involves capturing facial performances of real people. An ideal solution to the problem of facial performance capture is to use a standard video camera to capture live performances in 3D. The minimal requirement of a single video camera is particularly appealing, as it offers the lowest cost, a simplified setup, and the potential use of legacy sources and uncontrolled videos (e.g., Internet videos).

Recent advancements in computer graphics and vision have permitted the development of an impressive series of 3D facial performance capture methods, including both online [Cao et al. 2014a; Cao et al. 2015] and offline systems [Garrido et al. 2013; Shi et al. 2014], using a single RGB camera. Notably, Cao and his colleagues [2015] presented the first realtime facial performance capture system for capturing 3D head poses, large-scale facial deformation and medium scale facial details such as expression wrinkles. However, all the previous facial capture systems lack the capability to capture an indispensable component of facial performance: *eye gaze*. As noted in Ruhland et al. [2014], the Latin proverb states: "The face is the portrait of the mind; the eyes, its informers." Eyes are central in conveying emotional information because we are able to interpret the intentions and feelings of other humans by observing their eyes. Animation and control of realistic 3D virtual eyes remains challenging because eye gaze is so subtle that any unnatural gazing will be discerned easily by human eyes and will probably imply wrong intentions to the observers.

This paper introduces the first realtime 3D eye gaze capture method that simultaneously tracks 3D eye gaze, head poses and large-scale facial deformation using a single RGB camera (See Fig. 1). We start the process by automatically detecting important facial features such as the nose tip for each input frame. The detected facial features are then used to reconstruct 3D head poses and large-scale facial deformation using multi-linear expression deformation models. Next, we train a user-independent iris and pupil pixel classifier

based on random forests and use it to extract the iris and pupil pixels in each frame. We formulate the eye gaze tracker in the MAP framework, which sequentially infers the most probable 3D eye gaze at each frame using the reconstructed 3D head poses and the classified iris and pupil pixels of the current frame, as well as the estimated eye gaze state from the previous frame. The eye gaze tracker often fails when eye blinking occurs. This challenge motivates us to develop a novel eye close detector to further improve the robustness and accuracy of our eye gaze tracker.

The final facial performance capture system is robust and fully automatic, allowing for simultaneous capture of 3D head poses, large-scale facial expression deformation and 3D eye gaze using a single RGB camera. We have tested our system on both live video streams and the Internet videos, demonstrating its accuracy and robustness under a variety of uncontrolled lighting conditions and overcoming significant differences of races, genders, shapes, poses and expressions across individuals. We assess the quality of captured eye gaze by comparing with ground truth data annotated by human subjects. In addition, we evaluate the importance of key components of our 3D gaze tracker and show our system achieves the state-of-the-art accuracy by comparing it against alternative systems. Finally, we show the applications of our performance capture system in performance-based facial animation, realtime gaze data capture and eye gaze visualization.

1.1 Contributions

Our system is made possible by the following technical contributions:

- First and foremost, the first realtime 3D eye gaze capture system that complements with any facial performance capture system using RGB images.
- A novel user-independent iris and pupil pixel classifier based on random forests.
- An efficient eye gaze tracker that applies importance sampling to infer the most probable eye gaze state in the MAP framework.
- A new eye close detector that significantly improves the robustness and accuracy of our eye gaze tracker.

2 Background

Our realtime facial performance system automatically tracks 3D eye gaze, 3D head poses and facial expression deformation using a monocular RGB camera. Therefore, we focus our discussion on methods and systems developed for acquiring 3D facial performances and gaze motion.

2.1 Facial Performance Capture

Facial performance capture has a long history in computer graphics and vision. Various methods have been proposed in film and game production, such as marker-based motion capture systems [Bickel et al. 2007; Huang et al. 2011], marker-less facial capture that uses depth and/or color data obtained from structured light systems [Zhang et al. 2004; Ma et al. 2008; Weise et al. 2009] and multi-view stereo reconstruction systems using RGB images obtained by multiple cameras [Bradley et al. 2010; Beeler et al. 2010; Beeler et al. 2011; Valgaerts et al. 2012]. Recent advancement in 3D depth sensing has enabled a number of facial performance capture techniques using RGBD cameras [Weise et al. 2011; Chen et al. 2013; Bouaziz et al. 2013; Li et al. 2013; Li et al. 2015; Hsieh et al. 2015; Liu et al. 2015].

A more appealing solution for facial capture is to use a monocular RGB camera, as it offers the lowest cost and a simplified setup. These methods [Chai et al. 2003] first locate facial landmarks such as the nose tip and then use them to drive 3D facial animation. Recent advances for locating/tracking facial landmarks include constrained local model [Saragih et al. 2011; Baltrušaitis et al. 2012] and boosted regression [Cao et al. 2012; Xiong and De la Torre 2013; Ren et al. 2014]. In particular, Cao and his colleagues [2012] proposed an explicit two-level cascaded shape regressor for facial feature detection and Ren and his colleagues [2014] further refined the accuracy and efficiency of facial feature detector by utilizing the locality principle and a local binary feature based shape regressor.

Recently, Cao and colleagues [2013; 2014a; 2015] extended the idea of cascaded shape regression [Cao et al. 2012] for 3D facial capture. Their first system trained a user-specific 3D shape regressor and used it to directly track 3D facial expression deformation from 2D image sequences at runtime. Next, Cao and colleagues [2014a] proposed a user-independent displacement dynamic expression regression that adaptively refines the camera matrix and user identity during tracking. Recently, Cao and colleagues [2015] further extended the idea to realtime high-fidelity facial capture by adding a local user-specific detail regressor. Besides these online techniques, there are also offline systems that combine large-scale expression deformation tracking with shape-from-shading to capture detailed and dynamic 3D facial geometry [Garrido et al. 2013; Shi et al. 2014].

Our work enhances facial performance capture by adding a realtime 3D eye gaze tracker into facial capture. This enhancement allows us to capture coordinated movements between 3D head poses, facial expression and eye gaze, a capability that has not been demonstrated in any previous work. It is worth mentioning that our framework is flexible and our eye gaze tracker can be integrated with any 3D facial capture system using RGB images (*e.g.*, [Beeler et al. 2011; Valgaerts et al. 2012; Bouaziz et al. 2013; Li et al. 2013; Shi et al. 2014; Cao et al. 2015]) to capture coordinated movements between 3D head poses, facial deformation and eye gaze.

2.2 Eye Gaze Tracking

Eye gaze tracking and eye detection have been an active research topic in the field of human computer interaction and computer vision for many decades. Previous methods, which are mainly focused on 2D gaze detection and tracking, can be classified into two categories: *IR-illumination based approaches* and *image based approaches*. The active IR illumination based approaches exploit the spectral (reflective) properties called cornea reflection under IR illumination to efficiently detect iris and pupil pixels while the image-based approaches aim to detect or track eye gaze based on the shapes and/or appearances of the human eyes.

Active IR based methods (*e.g.*, [Morimoto and Flickner 2000]) is one of the most successful approach for eye gaze capture. Due to the simplicity and effectiveness of the method, almost all the commercial eye trackers (*e.g.*, [Anon, Applied science laboratories 2015; Lc Technologies 2015; Tobii Technologies 2015]) are based on this technique. However, the IR based method is intrusive because it requires users to wear a special glass or set up a dedicated IR device for gaze capture. In addition, unlike our method, it is often focused on 2D eye gaze capture alone. Therefore, they are not flexible for capturing 3D eye gaze in uncontrolled videos.

Traditional methods in image-based approaches can be further divided into three categories: template matching [Chau and Betke 2005; Corcoran et al. 2012], appearance based methods [Huang and Wechsler 1999; Huang and Mariani 2000] and feature based methods [Kawato and Ohya 2000; Tian et al. 2000]. However,

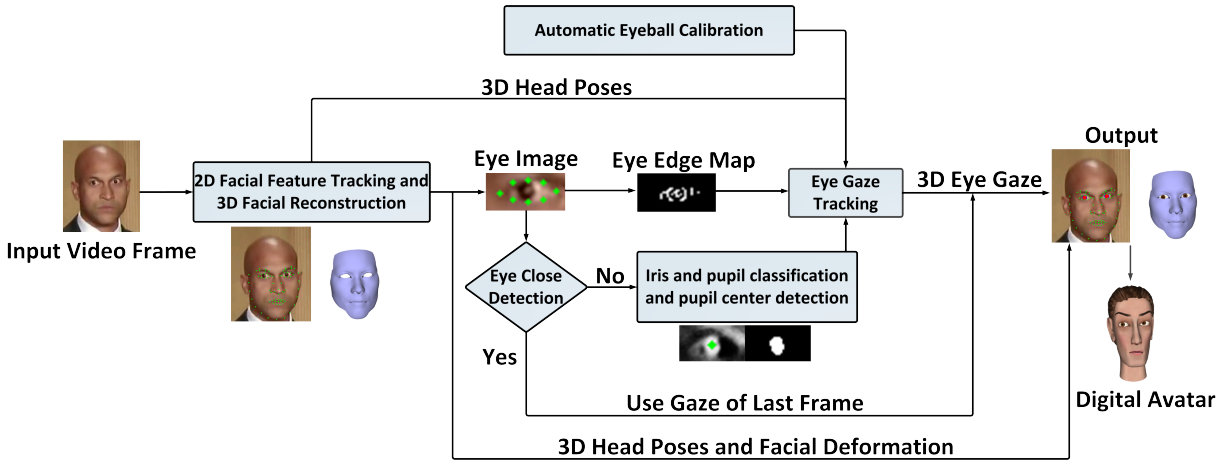


Figure 2: *An overview of our system.*

those methods are often not robust enough to handle variations in lighting, subjects, head poses and facial expressions. Recent efforts are mainly focused on applying cascade shape regression [Cao et al. 2012] or deep learning methods for pupil center detection. By adding two extra landmarks around the eye pupil center, those methods can detect both facial feature locations and the center of the pupil in a single image. Notably, the *Face++ tracker* [2015] developed by *Megvii Technology corporation* utilizes the deep neural network model for detecting and tracking 2D landmarks, achieving the state-of-the-art performance in 2D facial feature and pupil center detection. Our eye gaze detection and tracking method is different because we combine iris and pupil pixel classification, eye close detection, edge map of the iris and pupil region, and 3D head poses for 3D eye gaze tracking. Section 7.2 shows our eye gaze detection achieves much more accurate results. Our goal is also different because we focus on 3D facial performance capture with 3D eye gaze rather than 2D facial feature detection and pupil center detection.

Our work is relevant to recent efforts on appearance-based gaze estimation [Sugano et al. 2014; Wood et al. 2015; Zhang et al. 2015], which learns a regression function directly from an input eye image and a 3D head pose to eye gaze. Briefly, these systems first estimate 3D head pose using a generic 3D face model and six 2D facial features detected from input images, including the left and right corner of the mouth and the four corners of the left and right eyes, and then apply deep neural networks to learn a regression function directly from 2D eye images and 3D head poses to eye gaze. Our system is different from theirs in the following aspects. First, unlike their systems, our system is flexible and can be integrated with any existing 3D facial performance capture system because it does not require any offline camera calibration process for gaze capture. Second, we introduce an eye close detector into the system, thereby significantly improving the robustness and accuracy of our eye gaze tracker. Finally, unlike their systems, which aim to estimate eye gaze from a single image, our gaze tracking is focused on 3D facial performance capture that simultaneously tracks 3D head poses, facial deformation and eye gaze simultaneously from a monocular RGB sequence.

3 Overview

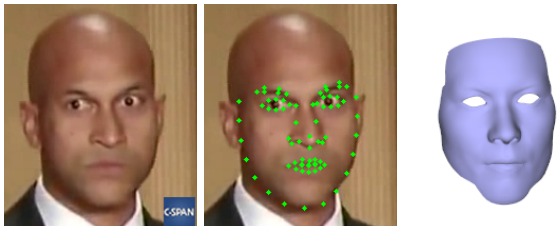
We aim to build a realtime facial performance capture system that robustly and accurately tracks 3D eye gaze, head poses and facial expression deformation using a monocular RGB camera. The problem is challenging because head poses, facial expression deformation and eye gaze motion are often coupled together. An accurate estimate of 3D eye gaze often requires an accurate estimate of 3D head poses and facial deformation around both eyes. On the other hand, accurate detection of eye gaze and eye close/open can further improve the reconstruction accuracy of facial expression deformation around both eyes. In addition, ambiguity caused by the loss of depth information in the projection from 3D to 2D and unknown camera parameters and lighting conditions further complicates the problem. To address this challenge, we propose an end-to-end facial performance system that simultaneously tracks 3D head poses, eye gaze and facial expression deformation using a single RGB camera. The whole system consists of five main components summarized as follows (see Fig. 2).

3D facial reconstruction. We start the process by automatically detecting and tracking important facial features such as the nose tip in monocular video sequences. We introduce a data-driven 3D facial reconstruction technique to reconstruct 3D head poses and large-scale expression deformation using multilinear expression deformation models.

Pixel classification for iris and pupil. We introduce a novel user-independent pixel classifier to automatically annotate iris and pupil pixels in the eye region, which is bounded by detected facial landmarks in the eye region. We further obtain the 2D location of the pupil center by applying the mean-shift algorithm [Comaniciu and Meer 2002] to the classified iris and pupil pixels. We discuss how to extract the outer contour of iris (*i.e.*, limbus) to further improve the robustness and accuracy of our gaze tracker.

Automatic eyeball calibration. Tracking 3D eye gaze across an entire video sequence requires not only modelling the geometry of 3D eyeball but also estimating the location of the eyeball and the size of the iris and pupil region. We approximate the geometry of the eyeball with a sphere of a particular radius (12.5mm), which corresponds to the average radius of adult's eyeballs. We introduce an eyeball calibration step, which is automatically done at the beginning of each capture, to estimate the 3D location of the eyeball and the size of the iris and pupil region.

Eye gaze tracking. We represent the state of 3D eye gaze based on the location of each pupil center on the surface of the eyeball sphere. We sequentially update the state of the eye gaze based on the detected 2D pupil center, the outer contour of iris and estimated 3D head poses. We formulate the problem in the Maximum A Posteriori (MAP) framework and apply importance sampling to infer the most probable state of eye gaze.



(a) (b) (c)

Figure 3: *2D facial feature tracking and 3D face reconstruction: (a) the input image; (b) tracked 2D facial features; (c) reconstructed 3D pose and facial deformation without eye gaze motion.*

Eye close detection. In practice, we have observed that the eye gaze tracking algorithm often fails when eye blinks. This challenge leads us to introduce a novel eye close detector to automatically detect whether the eye is open or closed. Once the eye is closed, we turn off the iris and pupil pixel classifier and gaze tracking and predict the state of the eye gaze directly using the result from the previous frame. In addition, we discuss how to use eye gaze constraints embedded in training data to further improve the accuracy and robustness of the system.

We describe these components in detail in the following sections.

4 2D Facial Feature Tracking and 3D Face Reconstruction

This section discusses how to reconstruct 3D head poses and facial expression deformation from a RGB video sequence, which is critical to our 3D gaze tracker. We start with 2D facial features detection/tracking (Fig. 3 (b)), which builds on local binary feature (LBF) based regression [Ren et al. 2014]. Next, we reconstruct the 3D head poses and large-scale expression deformation (Fig. 3 (c)) using the tracked 2D features.

4.1 2D Facial Feature Detection/Tracking

This step aims to detect and track 2D facial features from a monocular video. This task is achieved by a local binary feature (LBF) based regression, which has shown to outperform the cascaded regressors [Cao et al. 2012] in both accuracy and efficiency. Local binary feature is a long 1D vector assembled by 1D binary features obtained at each landmark. Given this vector, the facial shape S (*i.e.*, a collection of 2D facial feature locations) is progressively refined by estimating a shape increment ΔS stage-by-stage. The shape increment ΔS^t at stage t is regressed using the input image and extracted local binary features using random forests (Eq. 1).

$$\Delta S^t = W^t \Phi^t(I, S^{t-1}), \tag{1}$$

where W is a linear regression matrix, I is the input image, S^{t-1} is the shape at the previous stage, and Φ^t is the mapping function (random forests) which maps (I and S^{t-1}) to the local binary features L.

Though the LBF regressor [Ren et al. 2014] is fast and robust, we found the result could degenerate significantly for cases with extreme poses and low-quality image. We have made two refinements in the training/prediction process (for details, please refer to Appendix). The training database we use for learning the LBF regressors consists of 10858 images, which are selected from labeled faces in the wild (LFW) [Huang et al. 2007] and FaceWarehouse [Cao et al. 2014b].

4.2 3D Facial Performance Reconstruction

We now describe how to reconstruct 3D facial deformations and head poses from the tracked 2D locations. Similar to Shi et al. [2014], we represent the 3D facial models using multi-linear models (Eq. 4) [Vlasic et al. 2005; Cao et al. 2013], and formulate the problem in an optimization framework.

We represent the 3D facial models using multi-linear models [Vlasic et al. 2005; Cao et al. 2013]. Specifically, we describe a 3D face using two low-dimensional vectors controlling the identity and expression of the 3D face, respectively:

$$M = R(C_r \times_2 m_{id}^T \times_3 m_{exp}^T) + T, \tag{2}$$

where M represents large-scale facial geometry of an unknown subject, R and T represent the global rotation and translation of the subject, C_r is the reduced core tensor, and m_{id} and m_{exp} are identity and expression parameters respectively. Our multi-linear model was constructed from FaceWarehouse [Cao et al. 2014b], which contains face meshes corresponding to 150 identities and 47 facial expressions. In our experiment, the numbers of dimensions for the identity and expression parameters are set to 50 and 25.

By assuming an ideal pinhole camera model, the projected 2d features at image space can be represented as:

$$\mathbf{p}_k = Q(R((C_r \times_2 m_{id}^T \times_3 m_{\text{exp}}^T)^{(k)}) + T), \tag{3}$$

where $Q = [f\ 0\ u; 0\ f\ v; 0\ 0\ 1]$ is the ideal pinhole projection matrix, (R, T) is the 3D rotation and translation, f is the focal length and (u, v) is the principal point.

The goal here is to minimize the difference between the detected 2D features and the projected 2D features from the hypothesized face model. Similar to Shi et al. [2014], extra prior and smoothness terms for expression and pose are also imposed. Note that the identity weight and focal length are only estimated at the start of the video, and then fixed for the remaining frames. We follow the same binary search process in [Cao et al. 2013] to find the optimal focal length. The principle point (u, v) is set to the center of the image. For the rest frames, the objective function is as follows.

$$\underset{m_{\text{exp}}, R, T}{\arg\min} \quad E_{feature} + w_1 E_{exp} + w_2 E_{exp}^s + w_3 E_{pose}^s, \tag{4}$$

where the first term is the $feature$ term that measures how well the reconstructed facial geometry matches the observed facial features across the entire sequence. The second term is the *prior* term used for regularizing the expression parameters, which is formulated as a multivariate Gaussian. The third and fourth terms are the *smoothness* terms that penalize sudden changes of expressions and poses over time. In all of our experiments, w_1, w_2 and w_3 are set to 0.00001, 100 and 10, respectively.

The pose smoothness term constrains large rotation and translation changes between frames:

$$E_{pose}^s = w_4 E_{rotation}^s + w_5 E_{translation}^s, \tag{5}$$

where w_4 and w_5 are set to 1 and 0.1, respectively.

5 User-independent Iris and Pupil Pixel Classifier

In this section, we describe how to train a user-independent iris and pupil pixel classifier and use it to extract the pupil centers required for 3D gaze tracking. In addition, we extract the outer contour of iris to further improve the accuracy of our gaze tracker.

5.1 Iris and Pupil Pixel Classification

The eyeball, though its small size, is capable of executing a wide range of movements, fixation, saccade and smooth pursuit [Ruhland et al. 2014]. The pattern of eyeball movement is often quite complex. It can not only smoothly shift at a small acceleration but also can shift at a very large acceleration, such as the eye saccade. Thus temporal tracking could be vulnerable to error accumulation. To address this challenge, we introduce an efficient user-independent iris and pupil pixel classifier to perform per-frame iris and pupil pixels classification, which provides the prerequisite on the agility and accuracy of our eye gaze tracker.

We propose to use randomized forest [Breiman 2001] to train the iris and pupil pixel classifier. We advocate the use of randomized trees because they are robust and fast, while remaining reasonably easy to train. A randomized forest is an ensemble of L decision trees $D_1, \ldots, D_L$. Each node in the tree contains a simple test that splits the space of data to be classified, in our case the space of image patches. Each leaf contains an estimate based on training data of the posterior distribution over the classes. A new patch is classified by dropping it down the tree and performing an elementary test at each node that sends it to one side or the other. When it reaches a leaf, it is assigned probabilities of belonging to a class depending on the distribution stored in the leaf. Once the trees $D_1, \ldots, D_L$ are built, their responses are combined during classification to achieve a better recognition rate than a single tree could. More formally, the tree leaves store posterior probabilities $Pr_{\lambda(l,W)}(c|w)$, where c is a label in the label set C and $\lambda(l, w)$ is the leaf of tree D_l reached by the patch w. Such probabilities are evaluated during training as the ratio of the number of patches of class c in the training set that reach λ and the total number of patches that reach λ. The whole forest achieves an accurate and robust classification by averaging the class distributions over the leaf nodes reached for all L trees:

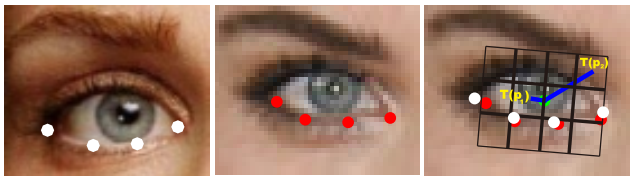
$$\tilde{c} = \arg\max_{c} \frac{1}{L} \sum_{l=1,\ldots,L} \Pr_{\lambda(l,W)}(c = Y(w)). \qquad (6)$$

Node testing. The tests performed at the nodes are simple binary tests based on simple functions of raw pixels taken in the neighborhood of the classification pixel. Our feature function calculates the difference of intensity values of a pair of pixels taken in the neighborhood of the classification pixel:

$$F = I(T(p_1)) - I(T(p_2)), \qquad (7)$$

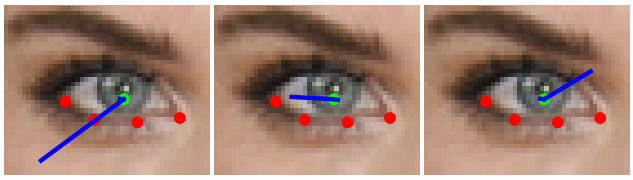
where I is the input eye image, T is the 2D similarity transformation from the reference to the current eye image, and p_1 and p_2 are the 2D locations of the feature pair.

To make the feature functions invariant to rotations, translations and scalings, we compute the similarity transformation to align the current eye image with the reference eye image in the training database. We calculate the similarity transformation based on the four lankmarks located on the lower eyelid (see Fig. 4). We normalise the offset of each pixel based on the calculated similarity transform to ensure the features are invariant to similarity transformations. If the value of a splitting function is larger than a threshold, go to left child and otherwise go to right child. In all our experiments, the patches are of size 30×30. And the optimal threshold for splitting the node is automatically determined by maximizing the information gain for particular features. We implement the randomized forest prediction on GPU for realtime performance. The average computational time for classification is about 2ms per frame for both of eyes.



(a) (b) (c)

Figure 4: *The feature function is invariant to translations, rotations and scalings: (a): the reference eye image in the training data; (b): the current eye image; (c): the similarity transformation that aligns the reference eye image with the test eye image. Note that we calculate the similarity transformation based on the four landmarks located on the lower eyelid.*



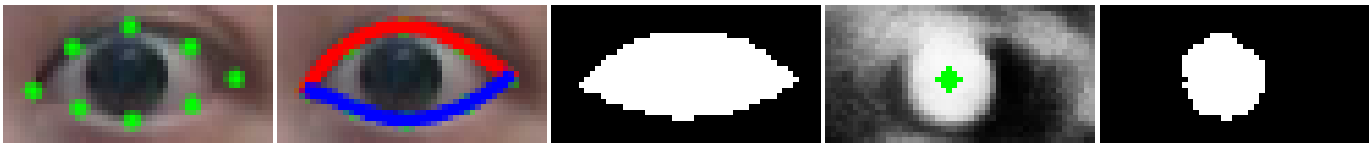
(a) (b) (c)

Figure 5: *Some selected features by randomized trees, which are intuitive and easy to understand. For example, the examples above imply the intensity values of the iris and pupil pixels are often less than the intensity values of the skin around the eyes ((a) &(c)) and the sclera (b).*

Training data. To learn a classifier for automatic iris and pupil pixel detection, we need to construct a training database containing a large set of eye images. In our experiments, we use a subset of W300 images [Huang et al. 2007] as the training image data. The training data contains large variations of head poses, facial expressions, lighting conditions and races. We selected 2941 images from the training data, and manually label the eye iris and pupil pixels and sclera ad skin pixels for each image. All our training pixels are confined in the eye region. We label the iris and pupil pixels as the positive examples and label the sclera and skin pixels as the negative examples. For all the faces with closed eyes, we regarded all the pixels in the eye regions as the sclera and skin pixels. The total number of the training pixel patches is about eight million.

Randomized trees learning. We use the randomized trees algorithm to learn binary forests. Each tree is trained separately on a small random subset of the training data. The trees are constructed in the classical, top-down manner, where the tests are chosen by a greedy algorithm to best separate the given examples. At each node, several candidates for a feature function are generated randomly, and the one that maximizes the expected gain in information about the node categories is chosen. The process of selecting a test is repeated for each nonterminal node, using only the training examples falling in that node. The recursion is stopped when the node receives too few examples, or when it reaches a given depth. Experimentally, we randomly select 512 candidate features for each node and stop the training when the tree depth exceed 13 or the number of examples in the leaf node is less than 10. We train 60 trees independently and combine them in the final decision forests.

The selected features are often intuitive and easy to understand. For example, the intensity values of the iris and pupil pixels are often less than the intensity values of the skin around the eye (Fig. 5 (a) &(c)) and sclera (Fig. 5 (b)).



(a) (b) (c) (d) (e)

Figure 6: *Classified iris and pupil pixels and extracted pupil center: (a): eye image and its corresponding eyelid landmarks (shown in green); (b): the upper (shown in red) and lower (shown in blue) spline curves defined by the eyelid landmarks; (c): the eye region mask (o_{eye}); (d): the probability map (I_{prob}) obtained by the iris and pupil pixel classifier. The intensity value of each pixel represents the probability of being an iris and pupil pixel. Brighter pixels indicate higher probabilities. The green dot is the 2D pupil center (o_{cen}) extracted by the mean-shift algorithm; (e): the silhouette map (o_{sil}) of the iris and pupil region obtained by thresholding the probability image (I_{prob}).*

5.2 Observation Extraction

We now describe how to extract the pupil center and the edge map of iris and pupil region required for 3D gaze tracking.

2D pupil center. The eye landmarks obtained by facial feature tracking defines the 2D location of the eyes in the face image. We can determine the eye image using the bounding box of the eye landmarks (Fig. 6(a)). The eye landmarks also provides the specific eye region in the eye image, including pupil center and sclera pixels. We determine the region by fitting two cubic splines to the 2D landmarks in the upper and lower eyelid (Fig. 6(b)). We fill the closed polygon to obtain the silhouette map of the eye region o_{eye} (Fig. 6(c)). With the trained iris and pupil pixel classifier, we can automatically annotate the label of each pixel in the eye image and obtain a probability map I_{prob} with the same size as the extracted eye image (Fig. 6). We apply the mean-shift algorithm to extract the 2D pupil center o_{cen}. The kernel function required for the mean-shift algorithm is defined as follows:

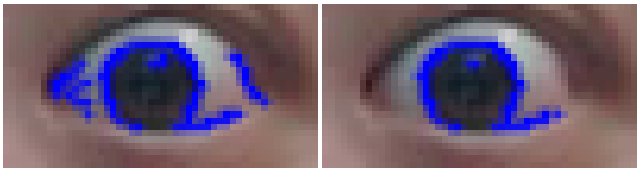
$$k(x, x_i) = (I_{prob}(x_i) * o_{sil}(x_i))exp(-\frac{\|x - x_i\|^2}{\sigma^2}), \quad (8)$$

where the bandwidth σ is chosen as the half of the eyes height and o_{sil} is the silhouette map of the iris and pupil region. We obtain the silhouette map of the iris and pupil region by first thresholding I_{prob} using a pre-defined threshold τ, which is experimentally set to 0.65, performing pixel open morphism operation to remove some noise, and extracting the strongest connected components of the remaining pixels. Fig. 6(e) shows the resulting rediris and pupil silhouette map o_{sil}.

Edge map of the iris. The edge map o_{edge} of iris region, like the detected pupil center o_{cen}, provides another strong visual cue about the true state of eye gaze. We apply Canny edge operator [Canny 1986] to extract the edge map of the eye region. However, the resulting edge maps are often very noisy and contain many outliers (Fig. 7(a)). In our experiment, we adopt two criteria to reduce noise and remove outliers in the extracted edge map. The first criterion requires that the Euclidean distance between inlier edge pixels and the extracted pupil center o_{cen} must lie within τ_1 and τ_2 times the height of eye region. The second criterion ensures that the angle between the gradient direction of the inlier pixel and the direction pointing from the pupil center o_{cen} to the pixel must be smaller than 90 degrees. Specifically, the two criteria are defined as follows:

$$\begin{aligned} &\tau_1 Height(o_{sil}) \leq \|p - o_{cen}\|_2 \leq \tau_2 Height(o_{sil}) \\ &(p - o_{cen})^T gradient(p) \geq 0, \end{aligned} \quad (9)$$

where p denotes the location of the candidate pixel in the edge map. The thresholds τ_1 and τ_2 are experimentally set to 0.1 and 0.6 respectively. Fig. 7(b) shows a side-by-side comparison for the edge map before and after noise reduction and outlier removal.



(a) (b)

Figure 7: *Edge map with/without outlier removal: (a): the original edge pixels obtained by the Canny operator; (b): the resulting edge map after outlier removal.*

6 Model Based 3D Eye Gaze Tracking

This section describes our idea on how to track the 3D eye gaze from the reconstructed 3D head poses, facial deformation (Section 4), and the extracted 2D pupil center and edge map (Section 5). We first calibrate the eyeball center and the size of iris and and pupil region which defines a user-specific eye model. We then track the pupil center frame-by-frame. We formulate the problem in an Maximum A Posteriori (MAP) framework and apply importance sampling technique to infer the most probable state of the eye gaze.

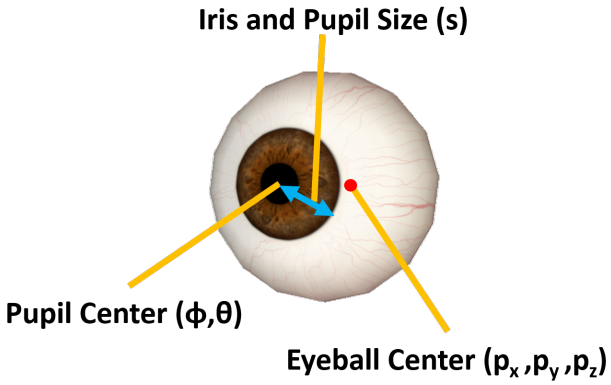


Figure 8: *The representation of the eye gaze state. A full eye gaze state can be represented as (P, s, ϕ, θ), where $P = (p_x, p_y, p_z)$ is the eyeball center at the face model space, s is the size of the iris and pupil region, and (ϕ, θ) is the pupil center, represented as the spherical coordinate on the eyeball surface.*

6.1 Representation

We represent the eye gaze state V as:

$$V = (P, s, \phi, \theta), \tag{10}$$

where $P = (p_x, p_y, p_z)$ is the eyeball center at the face model space, s is the size of the iris and pupil region (the radius), and (ϕ, θ) is the pupil center, represented as the spherical coordinate on the eyeball surface (Fig. 8). Specifically, the eyeball center and iris and pupil size defines a personalized eyeball model, and they need to be calibrated for each individual. With a given (P, s), the pupil center (ϕ, θ) then determines the 3D eye gaze movement for each frame.

6.2 Eyeball Calibration

Our first step is to calibrate the eyeball center and the size of iris and pupil region using the reconstructed 3D head pose and facial deformation, as well as the extracted edge map of iris and pupil region. Note that this step is done only once for each capture, as the eyeball center and the iris and pupil radius are constant for each individual.

To obtain the eyeball center, one can fit a sphere according to the image appearance and reconstructed 3D facial model. However, since only partial eyeball is visible, the estimate could be unreliable. For simplicity and a better robustness, we use a fixed eyeball radius (12.5mm), which is the average adult eyeball radius. The eyeball center can be then calculated as the mean of the preselected eyelid vertices on the face model plus a 3D offset that moves the radius distance toward the principle direction of 3D face (z-direction in our case). This allows us to define the eyeball model using its center and radius.

We now describe how to estimate the size of the iris and pupil region. We assume that the subject is looking at the camera with fully open eyes in the starting frames, and the first 20 valid frames are used for calibration. Based on the extracted edge map of iris and pupil region (see Section 5), we first perform Hough transform to fit a circle for each iris and pupil region. We then back project the 2D circles to the 3D eyeball model using the 3D head poses, and obtain the corresponding iris and pupil radius on the model space. Finally, we average all the radii to obtain the final size of the iris and pupil region.

6.3 3D Eye Gaze Tracking

With the known eyeball center and iris and pupil radius, our next step is to track the remaining gaze state, the 3D pupil center (ϕ, θ), at each frame. Eye gaze motion has complex patterns and simple temporal tracking could easily suffer from the error accumulation. On the other hand, the extracted observations (2D pupil center and edge map) from current frame, though robust, lack detailed accuracy. To address the challenge, we propose to combine the extracted 2D pupil center, the edge information and temporal coherence into a MAP framework (Eq. 11). We then solve for the most probable eye gaze state via importance sampling.

$$x_t^* = \arg\max_{x_t} Pr(x_t|o_t, H_t, x_{t-1}), \tag{11}$$

where x_t is the state of the current frame t, H_t is the head orientation in time t, and o_t is the current observation. Using Bayes' rule, we obtain

$$x_t^* = \arg\max_{x_t} \underbrace{Pr(o_t|x_t, H_t)}_{\text{observation likelihood}} \underbrace{Pr(x_t|x_{t-1})}_{\text{dynamic likelihood}}, \tag{12}$$

where the $Pr(o_t|x_t, H_t)$ is the observation likelihood that measures how well the current state fits the observation and $Pr(x_t|x_{t-1})$ is the dynamic likelihood that measures how different the current state is from the previous state.

Observation likelihood. The observation likelihood consists of two terms, namely mean-shift center term and edge term. It is formulated as follows:

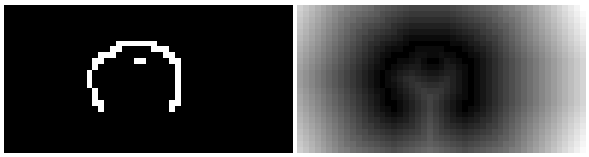
$$Pr(o_t|x_t, H_t) \propto exp(-w_{cen}E_{cen} - w_{edge}E_{edge}), \tag{13}$$

where parameters w_{cen} and w_{edge} are set to 3 and 1 respectively in the experiments.

The pupil mean-shift center term E_{cen} measures the difference between the mean-shift center of synthesized iris and pupil pixels $M(r_{sil})$, which is generated using the current head orientation H_t and the calibrated 3D eyeball model, and the observed pupil mean-shift center o_{cen}:

$$E_{cen} = \|o_{cen} - M(r_{sil})\|_2. \tag{14}$$

Intuitively, the extracted 2D mean-shift center is very likely to stay close to the true 2D pupil center, thus this term effectively constrains the search range of candidate pupil centers.



(a) (b)

Figure 9: *The extracted edge map of the iris and pupil region and its distance transform: (a) the observed edge map; (b) the distance transform of the edge map.*

The edge term measures the discrepancies of edge maps between the rendered and observed images, which also provides a strong cue to locate the 3D eye pupil center. Note that the edge map is still a partial observation of the true edge map, though we have removed many outliers during the extraction. Thus, we propose to use the trimmed chamfer distance measure for a better robustness:

$$E_{edge} = \frac{1}{K}\sum_{i=1}^{K} I_d^0(i) \cdot I_r(i), \tag{15}$$

where $I_d^0(i)$ is the distance transform of the observed edge map (Fig. 9), and $I_r(i)$ is the rendered binary edge map. This term sums the K smallest distances amongst the rendered edge pixels. K is determined by a certain ratio α (0.6 in our experiments) of the total rendered pixels:

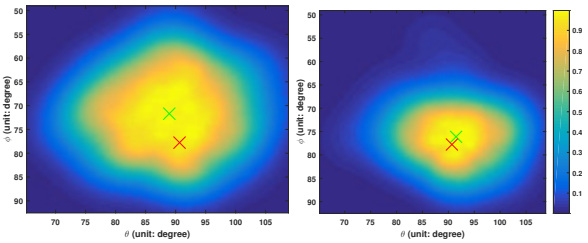
$$K = round(\sum_i I_r(i) * \alpha). \tag{16}$$

Note that we do not perform distance transform on the rendered edge map due to the limited time budget. In addition, we will set the weight of edge term to 0 if the edge map is unreliable, which is considered true when the number of the observed edge pixels are below a lowerbound (15 pixels).

Dynamic likelihood. Due to the complex eye movement patterns, the commonly used second order constraints $\|x_t - 2x_{t-1} + x_{t-2}\|$ is not suitable. We, therefore, propose a novel dynamic likelihood term which automatically degenerate (becoming flat) when the state change is large:

$$Pr(x_t|x_{t-1}) \propto exp(-\frac{min(d_{sphere}(x_t, x_{t-1}), \tau)}{\sigma^2}), \qquad (17)$$

where $d_{sphere}(x_t, x_{t-1})$ is defined as the minimum distance on the unit sphere between the two spherical coordinates, the threshold τ is set to 0.01 rad or 8 degree experimentally and σ is set to 0.1 in our experiments. The dynamic likelihood effectively penalties the deviation when the state change is small, such as fixation, and automatically degenerate when the state change is large, such as saccade. This term effectively ensures the smoothness of the eye gaze motion, while allowing for large sudden changes.



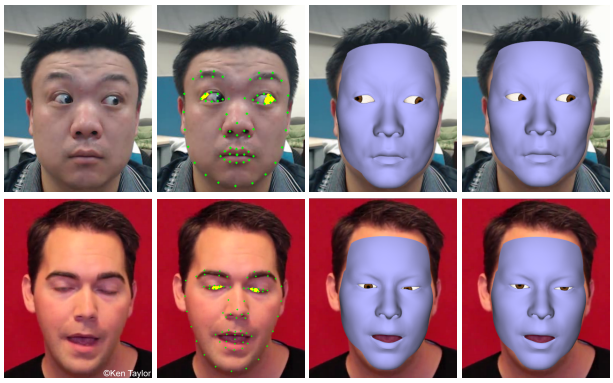
(a) (b)

Figure 10: *Effectiveness of the importance sampling (red and green cross represent the ground truth and expected eye state respectively): (a) prior Gaussian distribution fitted using the back-projected mean-shift pupil center; (b) posterior distribution after the resampling, which variance is effectively reduced and the expectation is clearly improved toward the ground truth.*

Optimization. Since it is nontrivial to evaluate the derivatives of Eq. 12, we propose to solve the problem through importance sampling. First, we form a Gaussian distribution using the back-projected 2D mean-shift center as mean, and then use it to sample the initial candidate states. The standard deviation is set to 0.2. We then re-evaluate the importance of each candidate using both observation and dynamic likelihood (Eq. 12). The weighted candidates are actually an estimate of the true posterior distribution in the MAP framework. Next, we perform another round of sampling using this posterior distribution, and the weighted average of the re-sampled candidate states is then used as the current 3D eye gaze state. Fig. 10 visualizes the initial Gaussian and candidate weight distribution after the resampling. The variance is effectively reduced and the expected state is clearly refined toward the ground truth, which demonstrate the effectiveness of the resampling method. In our implementation, the number of candidates is chosen as 200. Parallel implementation on multi-core CPU is used for realtime performance.

6.4 Eye Tracking Failure Detection and Handling

Eye blinking and eyelid occlusion are natural and frequent actions during the facial capture. When eye blinking and eyelid occlusion occurs, results of 2D pupil center detection and edge map extraction become unstable. In addition, 2D feature detection in the eye region could also become noisy and inaccurate. In both cases, the performance of our eye gaze tracker will degrade. To handle these failure cases, we introduce two novel ideas on failure detection, which include a eye close detector and a double eye gaze constraint that utilizes the movement dependency of both eyes. Once the failure is detected, we will directly use the previous eye gaze state to predict the output for the current frame.



(a) (b) (c) (d)

Figure 11: *The importance of failure detection and handling: (a): original input video frames; (b): tracked facial landmarks (shown in green) and the classified pupil pixels (shown in yellow); (c): results without failure detection and handling; (d): results with failure detection and handling.*

Eye close detector. We propose a novel eye close detector for the eye blinking event detection. It takes the eye image patch as input, and returns a binary true/false as output. When the eye is detected as closed, the system will output the previous gaze state for current frame. Like the eye iris and pupil classifier, we use the randomized forest for training and detection. Because the current training image dataset contains very few closed eye examples, we have downloaded additional sample face images with closed eyes. We further augment the database by performing random 2D similarity transformations 10 times to the selected landmarks of each sample. The augmentation makes the learnt trees more robust to possible inaccurate facial landmarks. We use a forest with 30 trees and depth 10, and it takes about 20mins for the tree training. The classification error rate is very low in both training and testing dataset, indicating that the eye close detection is much easier compared with eye iris and pupil pixel detection. We have found this strategy works well in practice. This component is also used to refine the 3D facial geometry. Once eye close event is detected and the distance between 2D landmarks of higher eyelid and lower eyelid is small enough, we will deform the pre-selected upper eyelid vertices of the facial deformation to the corresponding lower eyelid vertices using Laplacian deformation [Sorkine et al. 2004], so that the eyes are correctly closed.

Double eye gaze constraints. Up to now, the two eyes are separately tracked and can move independently. Thus, the result could be bad when one of the two gaze trackers outputs an invalid eye gaze state (Fig. 11). To ensure the movements of both eyes are valid and natural, we propose a data-driven eye gaze constraints to statistically ensure the double eye dependency. Specifically, we represent the motion of both eyes as a 4-value vector $(\phi_{left}, \theta_{left}, \phi_{right}, \theta_{right})$, and use the k-means algorithm to cluster the gaze dataset and get $k = 60$ data centers $(c_1, c_2, \cdots, c_k)$. We consider the current double eye gaze states u as invalid, if

$$d = \min_i \|u - c_i\|_2 \geq \gamma, \qquad (18)$$

Figure 12: *Reconstructed 3D eye gaze results with varying view directions and eye colors. Our system detects the iris and pupil region (red) and estimates the 3D eye gaze (yellow).*

where γ is chosen by the training data, which is 0.2 rad or 11.45 degree in our experiments. Once the current eye gaze states are regarded as outliers, the previous eye gaze states are used to predict the current state.

7 Evaluation and Results

We have demonstrated the power of our system on a large number of video sequences, including live video streams captured by a web camera and monocular video sequences downloaded from the Internet. Additionally, we have evaluated the effectiveness and accuracy of our system by comparing it against alternative methods. We also evaluate the importance of key components of our system.

Our system achieves real-time performance and runs at a frame rate of about 27 frames per second (fps). Table 1 reports computational times for each key component of our eye gaze tracker. All our experiments were tested on an Intel(R) Xeon(R) CPU 3.3GHz, 16GB RAM with NVIDIA GTX 780 graphics cards. Our results are best seen in the accompanying video.

Components	Timings(ms)
Expression reconstruction	23
Observation extraction	5
Eye gaze tracking	8.3
Failure detection	0.3
Total	36.6

Table 1: *Computational cost of each key component of our system.*

7.1 Test on the Real Data

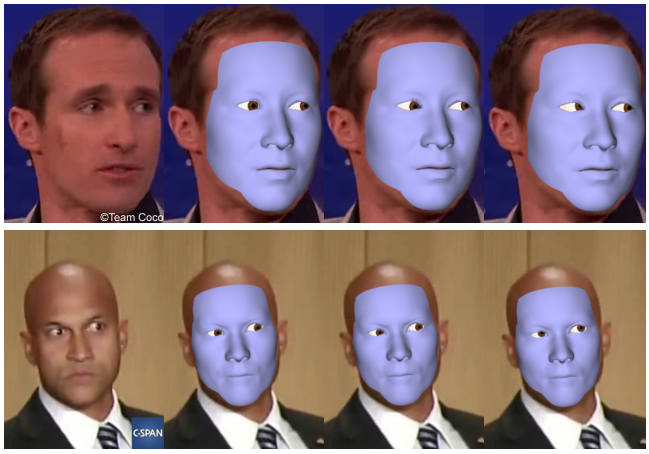
We have tested the effectiveness of our eye gaze tracker on different subjects (Fig. 12) . In the online testing, we use a web camera with resolution 800 $\times$ 600. The accompanying video shows that the system can track the eye gaze very robustly and accurately, even under fast eye gaze rolling, large head pose and extreme facial expressions. In addition, our system is also robust to large lighting changes as well as possible camera blur (Fig. 13). Besides live video streams, the accompanying video shows that our system can successfully track the eye gaze and facial expressions from eight video clips, in which six are from the Internet and two are recorded by a common RGB web camera.

7.2 Comparison against Face++

In this section, we evaluate the effectiveness and accuracy of our system by comparing against the state-of-the-art 2D facial feature detection/tracking system: Face++ [2015]. Face++ is a commercial facial landmark tracker developed by Megvii Technology. The company provides free API of facial landmark detection for comparison. The comparison is evaluated on eight video clips, which

Figure 13: *Live demos of our system. Our system is robust and accurate even under significant lighting variations and extreme pose changes.*

contain subjects of different races under various poses and lighting conditions.



(a) (b) (c) (d)

Figure 14: *Comparison against Face++ landmark detector: (a): the input images; (b): the ground truth; (c): the eye gaze results of our method; (d): the eye gaze motion fitted by using the 2D pupil centers outputted by Face++ and our reconstructed 3D head pose and facial deformation.*

Fig. 14 shows some sample results for comparisons. Note that Face++ is focused on 2D facial landmark detection/tracking rather than 3D eye gaze tracking. For the purpose of visualization, we project the 2D pupil centers estimated by Face++ into 3D using camera parameters obtained in Section 4, as well as 3D eye ball location from our calibration process. Similarly, 3D head poses and facial expression deformation of Face++ are based on those reconstructed by our system.

The image frames with closed eyes were excluded for evaluation because ground truth data for those frames are hard to obtain. We manually labeled the 2D pupil centers for the remaining image frames to get ground truth data for comparison. As suggested by [BioID 2015], the error metric is defined as follows:

$$E = \frac{max(\|p_l - {p'}_l\|, \|p_r - {p'}_r\|)}{\|{p'}_l - {p'}_r\|}, \tag{19}$$

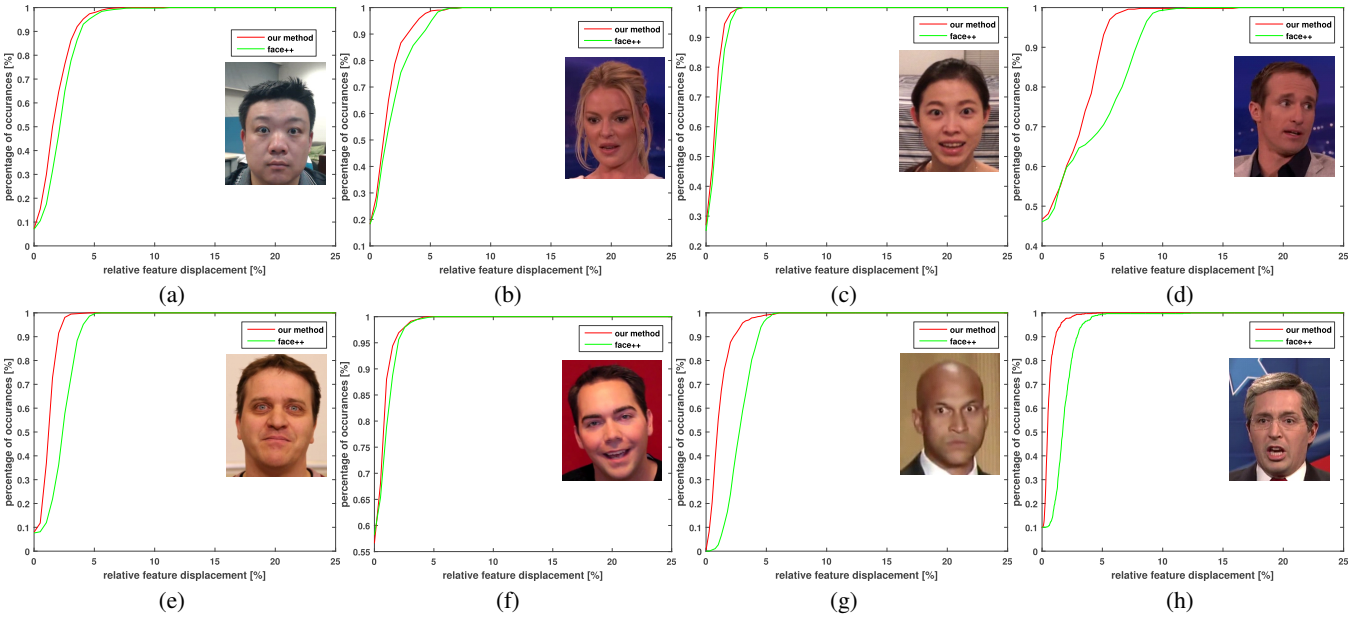


Figure 15: *The quantitative evaluation of our algorithm and comparison against Face++, where (a)-(h) correspond to results obtained from eight different video clips A - H. Note that only 2D pupil center locations are compared as Face++ does not reconstruct 3D eye gaze motion.*

where p_l and p_r are the centers of the left and right pupils obtained by the algorithm, respectively. And p'_l and p'_r are corresponding ground truth data. Fig. 15 shows that our method obtains steeper error distribution curve than face++ in all the video clips, which demonstrates that our method is more robust and accurate than face++.

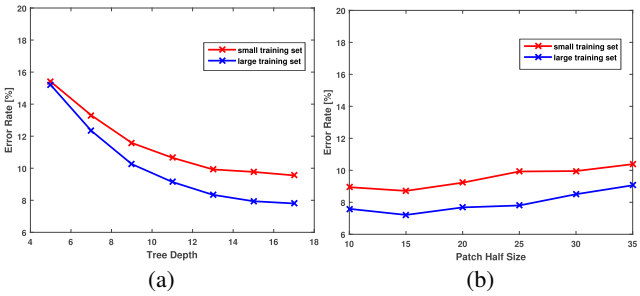


Figure 16: *Evaluation on the randomized forest learning: (a): evaluation on the tree depth; (b): evaluation on the patch size.*

7.3 More Evaluations

We now evaluate the key components of our 3D facial and eye gaze tracker.

Evaluation on iris and pupil pixel classifier. The accuracy of our system relies heavily on the performance of iris and pupil pixel classifier. For our application, the maximum depth of the trees and the patch size are two important parameters that influence the final performance of classification. Therefore, we evaluate on the tree depth and patch size of the randomized forest classifier. We randomly split our data sets into training data (2683 images) and testing data (298 images). A smaller set of training images (804 images) are further selected for evaluation. Fig. 16 shows the evaluation on the tree depth (left) and patch size (right), respectively. We experimentally set the patch size to 15 and the maximum tree depth to 13, achieving a good trade-off between performance and accuracy.

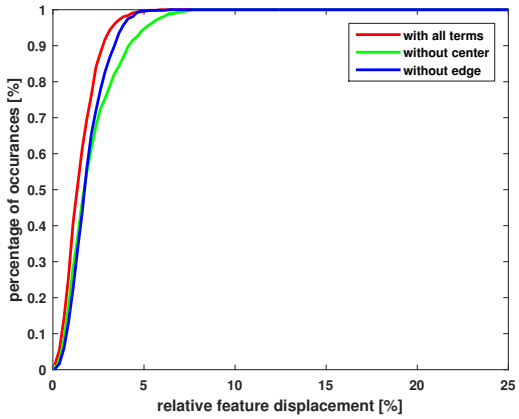


Figure 17: *Evaluation of the pupil center term and the edge terms. The x-axis represents the relative 2D pupil center displacement to the ground truth (Eq. 19), and the y-axis shows the accumulative error distributions. The combination of the pupil center and edge terms achieves the best performance.*

Importance of pupil center term and edge terms. The objective function for eye gaze tracking consists of two image terms, including the pupil center term and the edge term. We have evaluated the importance of each term to the eye gaze tracking process. Specifically, we drop off each term described in Eq. 13 and track the eye gaze motion using the same video streams. Fig. 17 shows the comparison results for three methods. It clearly shows both terms are necessary and the combination achieves the lowest error. In addition, it also shows the pupil center term seems to be more important than the edge term in the current tracking framework.

Importance of failure detection. To demonstrate the importance of failure detection, we compare the results of tracking eye gaze with or without failure detection on the same video sequences (Fig. 11). The results clearly show that the failure detection component enhances the robustness of the system.

7.4 Applications

This section discusses the applications of our performance capture system in performance-based facial animation, realtime gaze capture and visualization.

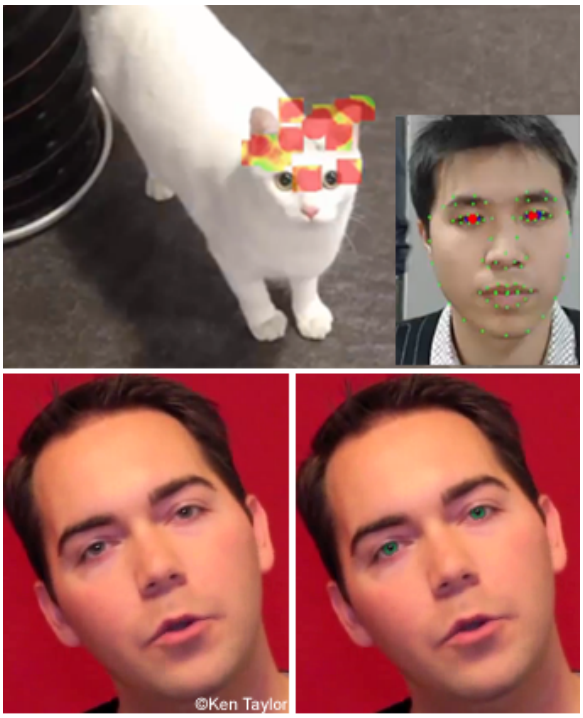


Figure 18: *Realtime eye gaze capture (top row) and eye gaze visualization (bottom row). Top row: the user is instructed to watch a video ("a running cat") on the screen and the focus points on the screen are visualized using a heatmap; bottom row: we edit the iris and pupil appearance by applying a new iris and pupil texture, projecting it to image plane and blending it with the original image.*

Facial and eye gaze retargeting. Given the captured facial performances as well as the eye gaze states of both eyes, we can easily retarget the tracked motion to a virtual avatar. To achieve this goal, we first estimate the identity parameters of a target avatar. We then estimate the eyeball center and the size of iris and pupil region for the animated avatar using our calibration process. Note that the eyeball information can also be directly from avatar modeling process. With the estimated identity parameters and eyeball model of the target avatar, we can directly transfer the expression parameters and the pupil centers (ϕ, θ), as well as 3D head poses, from the source to the target and drive the avatar to perform similar expressions, poses and eye gaze motions as the source's in real time.

Realtime eye gaze capture. With the reconstructed 3D head poses and tracked eye gaze states, we can locate eye gaze points (fixations) in screen space at runtime and visualize them using a heatmap. To start the process, we compute a linear transformation between the camera image plane and the screen plane by instructing the user to focus on a small set of pre-defined target points on the screen (*e.g.*, left top, middle top, right top, etc.) and formulating a least square problem. The calibration process allows us to map the eye gaze points from the image plane to the screen space. We visualize the focus points on the screen with a headmap generated by a gaussian kernel. We also set a timer for each pixel to cool down the previous focus points. Fig. 18(top) shows a sample heatmap generated by the use following and looking at a running cat in a video sequence.

Eye gaze visualization. One nice property of our system is that it not only captures eye gaze motion, but also calibrates the eyeball center and the size of iris and pupil region. Thus, we are able to obtain the exact iris and pupil regions in each video frame by projecting the eyeball onto the image plane. In this application, the iris and pupil appearance is edited by applying a new iris and pupil texture, projecting it to image plane and blending it with the original image (Fig. 18(bottom)). Note that this process is fully automatic and no manual refinement is needed. Though the performance has large pose and eye gaze variations as well as frequent eye blinkings, the results are continuous and natural. This clearly demonstrates the accuracy and robustness of our system.

(a) (b) (c)

Figure 19: *Limitation of our system on peculiar eye gaze motions, such as rolling white eyes: (a): input video frame; (b): result without failure detection; (c): result with failure detection. When failure is detected, the gaze state of the previous frame is used to predict the current state.*

8 Conclusion

In this paper, we demonstrate an end-to-end realtime system that captures the coordinated movements of 3D head poses, facial expression deformation and eye gaze using a monocular video camera. The key idea of our paper is to enhance a realtime facial tracker by adding a 3D eye gaze tracker that automatically and robustly track the 3D eye gaze in the Maximum A Posterior (MAP) framework. Our system is appealing for facial and eye gaze capture because it is fully automatic, runs in realtime and offers the lowest cost and a simplified setup. We have tested our system on both live video streams and the Internet videos, demonstrating its accuracy and robustness under a variety of uncontrolled lighting conditions and overcoming significant differences of races, genders, shapes, poses and expressions across individuals.

The current system has a few limitations. First, since the eye gaze tracking is based on classification of the iris and pupil pixels, the system could fail to detect the 2D pupil center locations when the subject performs some peculiar eye gaze motion, such as eye rolling. Fig. 19 illustrates an example of such eye gaze. Note that with our failure detection handling component, our system successfully detects the invalid gaze and directly uses the previous gaze state to predict the current state, thereby avoiding visually unnatural eye gaze motion. In addition, similar to other video-based facial

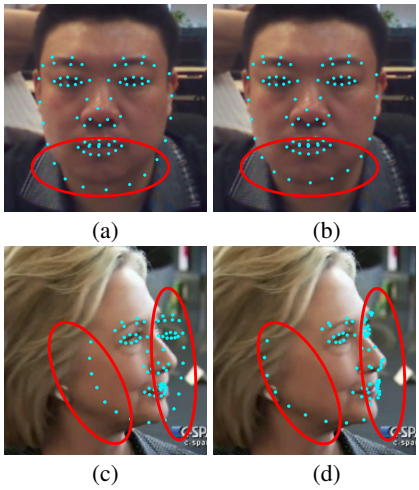


Figure 20: *Comparison against Ren et al. [2014]: (a) & (c) are results from Ren et al. [2014], and (b) & (d) are our results. The top row shows the results on an image with a low quality, and the bottom show shows the case with an extreme pose.*

performance systems, our system often fails to accurately track facial expressions and eye gaze when many of the facial features are occluded in the video frames. Another limitation of the system is that captured facial performances do not contain fine geometric details. One possible solution is to combine our eye gaze tracker with the system proposed by Cao et al. [2015] for realtime high-fidelity facial and eye gaze animation.

APPENDIX

Though the LBF regressor [Ren et al. 2014] is fast and robust, we found the result could degenerate significantly for cases with extreme poses and low-quality image (see Fig. 20 (a) & and (c)). We have made two effective refinements in the training/prediction process, and achieved a much more robust tracker (Fig. 20 (b) & (d)) than the original method described in Ren et al. [2014] .

We introduce a convex-hull based feature selection scheme that effectively avoids selecting pixels at background in the random forests training process. To train a random forest for a particular landmark, we need to select most discriminative feature pairs (giving rise to maximum variance reduction) for each node from a set of candidate feature locations. These candidates are firstly sampled at a reference shape, and then transferred to each of the tracing sample through a similarity transform. Ren et al. [2014] proposed to select features at the local region of a landmark. However, this method might select locations that are totally outside the facial region which could be random background. This is the case especially for the contour landmarks. As a result, the landmark locations can be inaccurate when the input images contain random noise and/or a background which is different from those of the training images. To address this issue, we refine the feature selection process by further constraining the sample locations to be inside the convex hull of the 2d feature points (Fig. 21). Unreliable feature locations at background can be effectively ignored. Note that this scheme is only performed on the reference shape at the training stage. Once the good feature pairs are selected, they will be directly used in the prediction. No further convex-hull check is needed at the prediction stage.

In addition, we refine the initialization step at the prediction stage for a better robustness to pose variations. In the tracking process

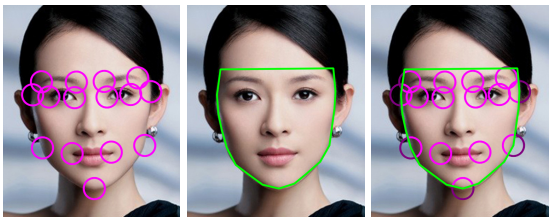


Figure 21: *Convex-hull based feature selection for random forests training: (a) the original feature sampling process selects a local region within a certain radius around each landmark; (b) convex-hull of the facial features which is used to constrain the local region; (c) our feature sampling process only selects the local regions inside the convex hull, thus the unreliable regions at background are effectively excluded.*

of Ren et al. [2014], the initial shape S^0 is set as the mean-shape aligned to the previous landmark locations with a similarity transform. However, we found in experiments that this strategy might output inaccurate result for faces with extreme poses. An alternative is to use the k nearest neighbors to the previous result as the initializations, and take the mean/median of the results as the output. Though this strategy does address the issue effectively, it might output less stable results as the nearest neighbors are more sensitive to the shape changes. As a compromise, we combine these two strategies to draw benefits from both ends. Specifically, given the landmark locations from the previous frame, we find the nearest neighbors and the aligned mean-shape. We then count the number of feature differences between them which are larger than a threshold (ϵ_1). If this number is no larger than a given upper bound (ϵ_2), we use the aligned mean-shape as the initialization. Otherwise, the k nearest neighbors ($k = 3$) are used. Our experiment shows that this idea is quite effective to obtain both frame-frame smoothness and robustness to large pose variations. ϵ_1 and ϵ_2 are experimentally set to 15 pixels and 7, respectively.

Acknowledgements

This work was supported in part by the National Science Foundation under Grants No. IIS-1065384 and IIS-1055046 and National Natural Science Foundation of China under Grants No.61173055 and No.60970086. Congyi Wang and Shihong Xia are affiliated with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology of Chinese Academy of Sciences (CAS).

References

ANON, APPLIED SCIENCE LABORATORIES, 2015. http://www.a-s-l.com.com.

BALTRUŠAITIS, T., ROBINSON, P., AND MORENCY, L.-P. 2012. 3d constrained local model for rigid and non-rigid facial tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2610–2617.

BEELER, T., BICKEL, B., BEARDSLEY, P., SUMNER, B., AND GROSS, M. 2010. High-quality single-shot capture of facial geometry. *ACM Trans. Graph. 29*, 4, 40:1–40:9.

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.

BICKEL, B., BOTSCH, M., ANGST, R., MATUSIK, W., OTADUY, M., PFISTER, H., AND GROSS, M. 2007. Multi-scale capture of facial geometry and motion. *ACM Trans. Graph. 26*, 3, 33:1–33:10.

BIOID, 2015. https://www.bioid.com/about/bioid-face-database.

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.

BREIMAN, L. 2001. Random forests. *Machine learning 45*, 1, 5–32.

CANNY, J. 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6, 679–698.

CAO, X., WEI, Y., WEN, F., AND SUN, J. 2012. Face alignment by explicit shape regression. In *IEEE Conference on Computer Vision and Pattern Recognition (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., HOU, Q., AND ZHOU, K. 2014. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Transactions on Graphics (TOG) 33*, 4, 43.

CAO, C., WENG, Y., ZHOU, S., TONG, Y., AND ZHOU, K. 2014. Facewarehouse: a 3d facial expression database for visual computing. *IEEE Transations on Visualizationa and Computer Graphics (TVCG) 20*, 3, 413–425.

CAO, C., BRADLEY, D., ZHOU, K., AND BEELER, T. 2015. Real-time high-fidelity facial performance capture. *ACM Transactions on Graphics (TOG) 34*, 4, 46.

CHAI, J., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3D facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 193-206.

CHAU, M., AND BETKE, M. 2005. Real time eye tracking and blink detection with usb cameras. Tech. rep., Boston University Computer Science Department.

CHEN, Y.-L., WU, H.-T., SHI, F., TONG, X., AND CHAI, J. 2013. Accurate and robust 3d facial capture using a single rgbd camera. In *IEEE International Conference on Computer Vision (ICCV)*, 3615–3622.

COMANICIU, D., AND MEER, P. 2002. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 5, 603–619.

CORCORAN, P. M., NANU, F., PETRESCU, S., AND BIGIOI, P. 2012. Real-time eye gaze tracking for gaming design and consumer electronics systems. *IEEE Transactions on Consumer Electronics 58*, 2, 347–355.

GARRIDO, P., VALGAERTS, L., WU, C., AND THEOBALT, C. 2013. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph. 32*, 6, 158.

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

HSIEH, P.-L., MA, C., YU, J., AND LI, H. 2015. Unconstrained realtime facial performance capture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1675–1683.

HUANG, W., AND MARIANI, R. 2000. Face detection and precise eyes location. In *International Conference on Pattern Recognition*, vol. 4, 722–727.

HUANG, J., AND WECHSLER, H. 1999. Eye detection using optimal wavelet packets and radial basis functions (rbfs). *International Journal of Pattern Recognition and Artificial Intelligence 13*, 07, 1009–1025.

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., Technical Report 07-49, University of Massachusetts, Amherst.

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.

KAWATO, S., AND OHYA, J. 2000. Real-time detection of nodding and head-shaking by directly detecting and tracking the between-eyes. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 40–45.

LC TECHNOLOGIES, 2015. http://www.eyegaze.com.

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.

LI, H., TRUTOIU, L., OLSZEWSKI, K., WEI, L., TRUTNA, T., HSIEH, P.-L., NICHOLLS, A., AND MA, C. 2015. Facial performance sensing head-mounted display. *ACM Transactions on Graphics (TOG) 34*, 4, 47.

LIU, Y., XU, F., CHAI, J., TONG, X., WANG, L., AND HUO, Q. 2015. Video-audio driven real-time facial animation. *ACM Trans. Graph. 34*, 6 (Oct.), 182:1–182:10.

MA, W.-C., JONES, A., CHIANG, J.-Y., HAWKINS, T., FREDERIKSEN, S., PEERS, P., VUKOVIC, M., OUHYOUNG, M., AND DEBEVEC, P. 2008. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph. 27*, 5, 121:1–121:10.

MEGVII TECHNOLOGY, 2015. http://www.faceplusplus.com.cn.

MORIMOTO, C. H., AND FLICKNER, M. 2000. Real-time multiple face detection using active illumination. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 8–13.

REN, S., CAO, X., WEI, Y., AND SUN, J. 2014. Face alignment at 3000 fps via regressing local binary features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 1685–1692.

RUHLAND, K., ANDRIST, S., BADLER, J., PETERS, C., BADLER, N., GLEICHER, M., MUTLU, B., AND MCDONNELL, R. 2014. Look me in the eyes: A survey of eye and gaze animation for virtual agents and artificial systems. In *Eurographics State-of-the-Art Report*, 69–91.

SAGONAS, C., TZIMIROPOULOS, G., ZAFEIRIOU, S., AND PANTIC, M. 2013. 300 faces in-the-wild challenge: The first facial

landmark localization challenge. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 397–403.

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 (FG 2011)*, IEEE, 117–124.

SHI, F., WU, H.-T., TONG, X., AND CHAI, J. 2014. Automatic acquisition of high-fidelity facial performances using monocular videos. *ACM Transactions on Graphics (TOG) 33*, 6, 222.

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. 2004. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ACM, 175–184.

SUGANO, Y., MATSUSHITA, Y., AND SATO, Y. 2014. Learning-by-synthesis for appearance-based 3d gaze estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1821–1828.

TIAN, Y.-L., KANADE, T., AND COHN, J. F. 2000. Dual-state parametric eye tracking. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 110–115.

TOBII TECHNOLOGIES, 2015. http://www.tobii.com.

VALGAERTS, L., WU, C., BRUHN, A., SEIDEL, H.-P., AND THEOBALT, C. 2012. Lightweight binocular facial performance capture under uncontrolled lighting. *ACM Trans. Graph. 31*, 6 (Nov.), 187:1–187:11.

VLASIC, D., BRAND, M., PFISTER, H., AND POPOVIĆ, J. 2005. Face transfer with multilinear models. In *ACM Transactions on Graphics (TOG)*, vol. 24, ACM, 426–433.

WEISE, T., LI, H., VAN GOOL, L., AND PAULY, M. 2009. Face/off: live facial puppetry. In *Symposium on Computer Animation*, 7–16.

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

WOOD, E., BALTRUSAITIS, T., ZHANG, X., SUGANO, Y., ROBINSON, P., AND BULLING, A. 2015. Rendering of eyes for eye-shape registration and gaze estimation. *arXiv preprint arXiv:1505.05916*.

XIONG, X., AND DE LA TORRE, F. 2013. Supervised descent method and its applications to face alignment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 532–539.

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

ZHANG, X., SUGANO, Y., FRITZ, M., AND BULLING, A. 2015. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4511–4520.