Appearance-based Gaze Estimation with Online Calibration from Mouse Operations

Yusuke Sugano, Yasuyuki Matsushita, Yoichi Sato, and Hideki Koike

***Abstract*—This paper presents an unconstrained gaze estimation method using an online learning algorithm. We focus on a desktop scenario where a user operates a personal computer, and use the mouse-clicked positions to infer where on the screen the user is looking at. Our method continuously captures the user's head pose and eye images with a monocular camera, and each mouse click triggers learning sample acquisition. In order to handle head pose variations, the samples are adaptively clustered according to the estimated head pose. Then local reconstruction-based gaze estimation models are incrementally updated in each cluster. We conducted a prototype evaluation in real-world environments, and our method achieved an estimation accuracy of 2.9 degrees.**

***Index Terms*—Eye movement, tracking, Computer Vision, Human-computer interface**

I. Introduction

Gaze estimation is the process of detecting at what the eyes are looking. Many applications have been proposed in the field of human-computer interaction, including attentive user interfaces [1], [2] that use user attention for the goal of natural interaction. Contrary to traditional gaze-pointing applications, gaze and attention are intended to play a supplemental role to assist user interaction [3].

Current techniques of gaze estimation suffer from technical limitations such as a lengthy calibration process, extensive hardware requirements, and difficulty in handing head pose variations. Creating a calibration-free gaze estimator that uses simple and low-cost equipment while allowing users to freely move their heads is an open challenge.

One limitation of existing gaze estimation techniques is that users actively participate in the calibration task. Users are typically asked to look at several reference points to acquire the ground-truth data. Since such an active task interrupts the user interaction, even a short calibration step can be a critical limitation in application scenarios that assume a natural state of attention.

Our goal is to make a completely passive, non-contact, single-camera system for estimating gaze direction that does not require an explicit calibration stage yet still allows head pose movement. To achieve this goal, we develop a new appearance-based system of estimating gaze direction based on an online-learning approach.

Y. Sugano is with Perceptual User Interfaces Group, Max Planck Institute for Informatics, Campus E1 4, Saarbrücken, 66123, Germany. e-mail: sugano@mpi-inf.mpg.de

Y. Sato is with the Institute of Industrial Science, The University of Tokyo.

Y. Matsushita is with Microsoft Research Asia.

H. Koike is with the Graduate School of Information Science and Engineering, Tokyo Institute of Technology.

Our system incorporates advances of the single-camera three-dimensional (3-D) estimation of head poses to continuously capture users' head poses and eye images. We limit our scenario to a desktop environment with a camera mounted on the monitor. During the operation of a personal computer (PC), the user looks at the mouse cursor when he or she clicks the mouse button. Using the clicked coordinates as gaze labels, the system automatically collects learning samples while users are operating the PC. Our method continuously and adaptively learns the mapping between eye appearance and gaze direction without the need for lengthy calibrations.

Although the idea of using mouse clicks to explicitly recalibrate an eye tracker has been presented in [4], in this work we show that mouse clicks can serve as training data without the user's intention for eye tracker calibration. Using our method, the user's natural behavior can serve as a calibration process and the gaze estimation process can be integrated into desktop user interfaces. This leads to a scenario where users can install a software-based gaze estimation system which gradually learns to predict gaze positions on her/his own PC. Although it does not directly enable the quick use of the system, the mapping function is often device- and user-dependent and the system can be used by the target user once the function is learned.

In prior work we utilized visual saliency of a displayed video to infer focus of attention [5], [6]. Chen *et al*. [7] applied a similar idea to the case of model-based gaze estimation, and Alnajar *et al*. [8] proposed to directly use actual human gaze patterns collected from other viewers for the calibration-free gaze estimation task. These approaches have complementary scopes of application. Although the saliency-based technique [5]–[7] can be applied to completely passive systems without active user interaction, it is quite difficult to compute accurate visual saliency maps in desktop environments. Further, reusing gaze patterns that are obtained from other users as in [8] is difficult in our interactive setting, where the gaze behavior is heavily person-dependent. Here we demonstrate the practical advantage of the proposed method that achieves comparable accuracy while running in real time as a component of a interactive system.

This paper extends our prior work [9] by considering: 1) Online refinement of gaze labels, 2) an improved approach for discarding inappropriate training samples, and 3) subpixel eye-image alignment and blink detection. We also present results using a web browsing scenario.

The rest of the paper is organized as follows. Section II presents related work, and Section III describes the architecture of our system. Section IV explains our gaze esti-

mation method based on an incremental-learning algorithm. The details of implementing the head tracking and eye-image cropping are in Section V. Proof of concept evaluations appear in Section VI. Section VII concludes the paper.

II. Related work

A. Model-based methods

Model-based approaches use an explicit geometric model of an eye and estimate the eye's gaze direction using geometric eye features [10], [11]. While model-based approaches tend to produce more accurate results than appearance-based methods, they typically require a high-resolution camera for accurately locating the geometric features in the eyes.

In addition, model-based methods often require additional hardware such as multiple cameras or calibrated light sources to handle head movements [12]–[20]. Such requirements result in large systems with special equipment that are not readily available to end-users. Also, the algorithms are specialized to their own hardware configuration; therefore, it is difficult to implement a similar approach with only a single web camera.

There have been methods proposed to remove such restrictions in model-based approaches. Ishikawa *et al*.'s method [21] uses an active appearance model [22] to extract eye features and head poses only with a monocular camera. Yamazoe *et al*. 's method [23] estimates gaze direction by fitting a 3-D eye model to 2-D eye images. These approaches work in the real-world, and ordinary low-resolution cameras are used in both methods. Unfortunately, their methods are limited to only computing coarse features, such as the edge of the iris and the corners of the eyes, due to the low resolution of the camera, and result in lower accuracy in comparison with other model-based methods.

B. Appearance-based methods

Appearance-based approaches directly compute features from the appearance of eye images and estimate the gaze points by learning the mapping between eye image features and gaze points [24]–[28]. Compared to model-based methods, appearance-based methods have an advantage of simpler and less restrictive systems and have robustness against outliers even when implemented with relatively low-resolution cameras. The downsides are 1) typically more data are needed in comparison with the model-based methods, and 2) the estimation accuracy is in general not as high as with model-based methods.

With appearance-based approaches, it is difficult to deal with changes in head pose and head pose variation introduces the requirement for additional training samples. Baluja *et al*.'s method [24] allows for some head movements among the appearance-based methods. Their method collects training samples for each different head pose while the range of head pose change is limited. They describe two major difficulties: 1) the appearance of an eye looking at the same point drastically varies with the head pose. Therefore, additional information about the head pose is necessary and 2) the training samples have to be collected across the pose space to account for head movements. This results in a large number of training samples and an unrealistically lengthy calibration stage.

To address head pose changes. Lu *et al*.'s method [29] utilizes additional training data, *i.e*., a video of the target person rotating her/his head while fixating on a calibration target, to learn an error compensation function caused by head movements. They also proposed an approach to synthesize eye images for unknown head poses using additional reference images to estimate pixel flows [30]. Although it relies on an additional RGB-D input, Funes *et al*.'s method [31] uses front-facing eye images that are warped using estimated 3D facial shapes. Valenti *et al*. [32] proposed a pose-retargeted gaze estimation method that adaptively maps the calibration plane and gaze displacement vectors to target planes according to 3D head poses. However, they require specially-designed calibration processes at reference head poses, and cannot be applied to our problem setting which results in an uncontrolled stream of training samples.

III. Architechture

The process flow of our approach is illustrated in Figure 1. The input to the system is a continuous video stream from the camera as well as the display coordinates of the clicked points. The 3-D model-based head tracker [33] keeps running during the entire process to capture the head pose $\boldsymbol{p}$ and to crop the eye image $\boldsymbol{x}$.

Our approach assumes that the user's gaze is directed at the mouse cursor on the monitor when the user clicks a mouse button. With this assumption, we collect learning samples by capturing mouse cursor positions when clicking as well as eye images and head poses. We create a training sample at each mouse click using the screen coordinates of the mouse position as the gaze label, $\boldsymbol{g}$, associated with the appearance features (head pose $\boldsymbol{p}$ and eye image $\boldsymbol{x}$). More training samples are obtained the more the user clicks. Our system incrementally updates the mapping function between appearance features and gaze positions using the labeled samples.

In the learning stage, incremental learning is performed in a reduced principal components analysis (PCA) [34], [35] subspace to decrease the computational cost of dealing with multi-dimensional image features. The samples are adaptively clustered according to their head poses, and the local appearance manifold is updated in each sample cluster.

When the new training samples are not given to the system, the system runs in a prediction loop. In the prediction loop, the inputs to the system are only the head pose $\boldsymbol{p}$ and eye image $\boldsymbol{x}$, and the system produces gaze estimates $\hat{\boldsymbol{g}}$. The gaze estimate $\hat{\boldsymbol{g}}$ is produced by local linear interpolation of the accumulated training samples. As more samples are accumulated in the learning loop, the sample clusters and local manifolds are updated; therefore, it produces more reliable gaze estimates.

IV. Algorithm

The heart of our gaze estimator is to learn the mapping between appearance features $\{\boldsymbol{x}, \boldsymbol{p}\}$ and the gaze label $\boldsymbol{g}$. Once the mapping is established, our method predicts the unknown label $\hat{\boldsymbol{g}}$ from the unlabeled features $\{\boldsymbol{x}, \boldsymbol{p}\}$. Our method uses a

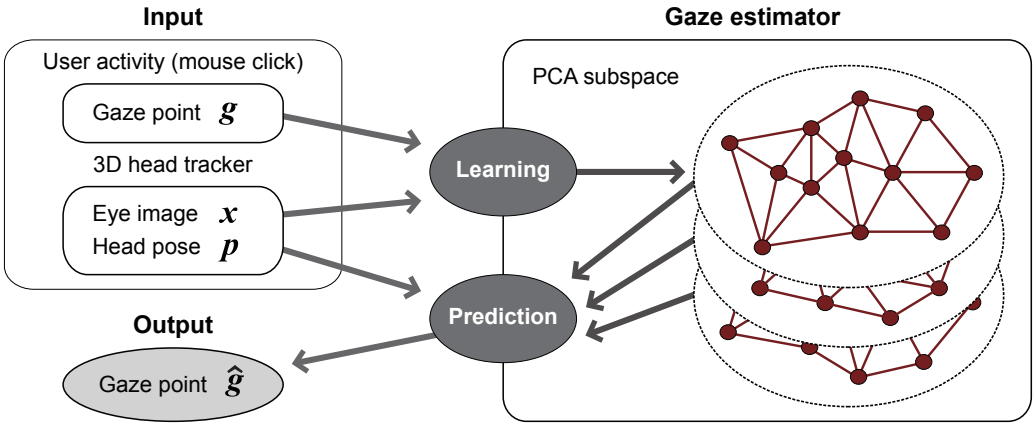

Fig. 1. Learning and prediction flow for the proposed framework. Our method continuously takes gaze points $\boldsymbol{g}$, eye images $\boldsymbol{x}$, and head poses $\boldsymbol{p}$ from the mouse clicks and synchronously captured images of the user for learning. For the prediction stage, the method takes only eye images $\boldsymbol{x}$ and head poses $\boldsymbol{p}$ as input to produce gaze estimates $\hat{\boldsymbol{g}}$.

local linear interpolation method that is similar to [26], [36], *i.e.*, we predict the unknown label $\hat{\boldsymbol{g}}$ by choosing k nearest neighbors from the labeled samples and interpolating their labels using distance-based weights.

For the accurate interpolation, it is critical to choose the correct neighbors from the appearance manifold, which models appearance changes of different gaze directions. Tan *et al*. [26] use 2-D topological information about the coordinates of the gaze labels as a constraint. Two eye images are assumed to be neighbors on the manifold in their method when they have similar gaze directions instead of simply evaluating by the similarity of their appearances. This assumption, however, does not always hold if head pose changes are considered. With the head pose variations, two different gaze directions yield similar appearances, or conversely, similar gaze directions lead to very different appearances.

To overcome this problem, we compute the sample clusters with similar head poses and create a local manifold for each sample cluster. This model is inspired by the locally weighted projection regression (LWPR) algorithm [37]. The local linear regressors are adaptively created and learned in LWPR according to the distance of input features. We employ similar adaptive architecture to create pose-dependent clusters of eye images.

In our method, the similarity measure of the cluster, *i.e.*, the distance between the head pose and the sample cluster, is defined as a product of the Gaussian functions of head translation and rotation. Given a pose $\boldsymbol{p}$, specified by translation $\boldsymbol{t}$ and rotation $\boldsymbol{r}$ in 3-D, the distance s_k between the head pose $\boldsymbol{p}$ to a certain cluster (say, the k-th cluster) is computed as

$$\begin{aligned} s_k(\boldsymbol{p}) \quad = \quad & \frac{1}{\sqrt{2\pi\kappa_t\sigma_t^2}} \exp\left(-\frac{||\boldsymbol{t}-\bar{\boldsymbol{t}}||^2}{2\kappa_t\sigma_t^2}\right) \\ & \frac{1}{\sqrt{2\pi\kappa_r\sigma_r^2}} \exp\left(-\frac{||\boldsymbol{r}-\bar{\boldsymbol{r}}||^2}{2\kappa_r\sigma_r^2}\right), \end{aligned} \tag{1}$$

where $\bar{\boldsymbol{t}}$ and σ_t^2 are the average and variance of head translation calculated from the samples in the cluster. Likewise, $\bar{\boldsymbol{r}}$ and σ_r^2 are the average and variance of head rotation. The constant weights κ_t and κ_r are empirically set.

In Equation (1), the Euclidean distance measure is used for both translation and rotation vectors. In our method, the rotation vector is represented by quaternions. With the quaternion representation, the distance can be measured by an angular distance ω_d, *i.e.*, the angle of rotation from one quaternion to the other. However, we approximate it using the Euclidean distance. Because we incrementally and continuously update the clusters, calculating the average $\bar{\boldsymbol{r}}$ is computationally expensive in the angular-distance measure. However, the average orientation in the Euclidean-distance measure can easily be obtained as an arithmetic average of the quaternions [38]. The Euclidean distance $||\boldsymbol{r}-\bar{\boldsymbol{r}}||^2 = ||\boldsymbol{I}-\bar{\boldsymbol{r}}\boldsymbol{r}^{-1}||^2 = 4\sin^2(\omega_d/4)$ can also be a good approximation of the angular distance when the two rotations are close.

Given a labeled sample $\{\boldsymbol{x},\boldsymbol{p},\boldsymbol{g}\}$, the eye image $\boldsymbol{x}$ is first used to update the PCA subspace of the eye images. The subspace that we use is described with N eigenvectors as

$$\boldsymbol{x} \approx \bar{\boldsymbol{x}} + \boldsymbol{U}\boldsymbol{a}, \tag{2}$$

where $\bar{\boldsymbol{x}}$ is the mean eye image, $\boldsymbol{U}$ is the matrix whose columns are composed of the first N eigenvectors, and a is an N-dimensional vector of PCA coefficients. After updating the subspace, the sample is added to all clusters whose similarity $s_k(\boldsymbol{p}_t))$ is greater than the predefined threshold τ_x. If no suitable clusters are found, a new cluster is created to only contain the new sample. In the prediction stage, given an unlabeled feature $\{\boldsymbol{x},\boldsymbol{p}\}$, the output gaze $\hat{\boldsymbol{g}}$ is computed as a weighted average of the candidate predictions obtained from multiple sample clusters. The following sections IV-A and IV-B describe further details of prediction and learning methods. The prediction and learning algorithms are outlined in Algorithm 1.

A. Prediction

When unlabeled data $\{\boldsymbol{x},\boldsymbol{p}\}$ are given, the system predicts the gaze estimate $\hat{\boldsymbol{g}}$ from the learnt data. First, the eye image $\boldsymbol{x}$ is projected onto the current PCA subspace computed from all the training samples as

$$\boldsymbol{a} = {}^t\boldsymbol{U}(\boldsymbol{x}-\bar{\boldsymbol{x}}), \tag{3}$$

Algorithm 1 Adaptive clustering framework

Prediction: Given input features $\{\boldsymbol{x}, \boldsymbol{p}\}$

Project image $\boldsymbol{x}$ into the current subspace: $\boldsymbol{a} = {}^t\boldsymbol{U}(\boldsymbol{x} - \bar{\boldsymbol{x}})$
for $k = 1$ to K (the number of clusters) **do**
Calculate the interpolated gaze $\hat{\boldsymbol{g}}_k$ and prediction confidence c_k (Section IV-A).
end for
Compute final prediction as a weighted average: $\hat{\boldsymbol{g}} = \sum_k c_k \hat{\boldsymbol{g}}_k / \sum_k c_k$.

Learning: Given the i-th learning sample $\{\boldsymbol{x}, \boldsymbol{p}\}$ associated with the gaze label $\boldsymbol{g}$

Update the image subspace using incremental PCA: mean $\bar{\boldsymbol{x}}$, eigenvectors $\boldsymbol{U}$, eigenvalues $\boldsymbol{\lambda}$, coefficients $\{\boldsymbol{a}_1 \dots \boldsymbol{a}_i\}$.
The input $\boldsymbol{x}$ is approximated as $\boldsymbol{x} \approx \bar{\boldsymbol{x}} + \boldsymbol{U}\boldsymbol{a}_i$.
for $k = 1$ to K (with respect to each of all K clusters) **do**
if $s_k(\boldsymbol{p}_i) > \tau_x$ **then**
Add sample to the cluster and update its local manifold (Section IV-B).
end if
end for
if none of $s_k(\boldsymbol{p}_i)$ is greater the threshold τ_x **then**
Create new $(K+1)$-th cluster and add the sample.
$K \leftarrow K + 1$.
end if

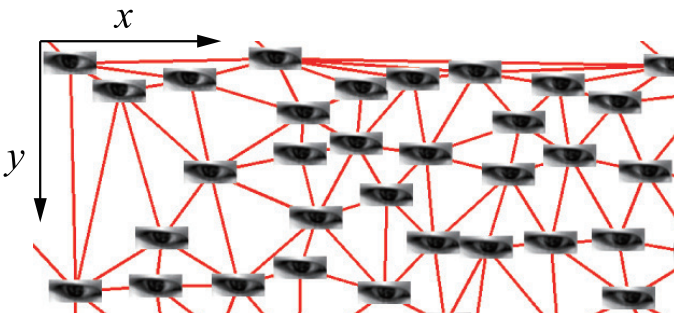

Fig. 2. Example of gaze triangulation shown in screen coordinates. Each eye image (flipped horizontally for a better visualization) is located at the corresponding gaze point, and the lines indicate Delaunay edges between the gaze points.

using the mean eye image $\bar{\boldsymbol{x}}$, the basis matrix $\boldsymbol{U}$ whose columns comprises the first N eigenvectors. $\boldsymbol{a}$ is the projected N-dimensional vector. An intermediate gaze estimate $\hat{\boldsymbol{g}}_k$ is then computed in each cluster using the projected eye image (PCA coefficients) $\boldsymbol{a}$ and the local interpolation of its neighbors. The neighboring samples of the projected eye image $\boldsymbol{a}$ are selected from the manifold, and the gaze labels of the neighbors are interpolated to determine the intermediate gaze estimate $\hat{\boldsymbol{g}}_k$ from the k-th cluster.

As in Tan *et al.* [26], we use the Delaunay triangulation of the gaze label for creating the appearance manifold. Figure 2 shows the visualization of the appearance manifold with Delaunay triangulation. Given the projected eye image $\boldsymbol{a}$, our method finds neighboring triangles in the appearance subspace. The distance from the projected eye image $\boldsymbol{a}$ to a triangle is measured by the average distance from the projected eye image $\boldsymbol{a}$ to the samples (vertices) of the triangle. The samples on the triangle as well as the samples adjacent to the triangle are regarded as neighboring samples. By selecting such neighboring samples that are located near the triangle, the sample set for interpolation is restricted to have a limited amount of gaze variations. To ensure computational efficiency, the above process of finding neighboring triangles is performed using the N_s closest samples in the cluster. If none of the N_s samples forms a triangle, N_s is increased by n_s until a triangle set is found.

Using the selected sample set $\mathcal{N}_p$, we compute the interpolation weights $\boldsymbol{w} = (w_1, w_2, \dots, w_{|\mathcal{N}_p|})$. The interpolation weights $\boldsymbol{w}$ are computed by minimizing the reconstruction error as

$$\boldsymbol{w} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \left(\boldsymbol{a} - \sum_{i \in \mathcal{N}_p} w_i \boldsymbol{a}_i \right)^2 \quad s.t. \quad \sum_{i \in \mathcal{N}_p} w_i = 1, \tag{4}$$

where w_i denotes the weight of the i-th neighbor's appearance $\boldsymbol{a}_i$. Finally, assuming the local linearity, the intermediate gaze estimate $\boldsymbol{g}_k$ from the k-th cluster is computed as

$$\hat{\boldsymbol{g}}_k = \sum_{i \in \mathcal{N}_p} w_i \boldsymbol{g}_i. \tag{5}$$

To reduce negative effects from the clusters that do not contain a sufficient number of samples, we define a reliability measure for the interpolation that represents how well the input appearance $\boldsymbol{a}$ can be described by the selected neighbors as

$$r_k(\boldsymbol{a}) = \exp\left(-\frac{(\boldsymbol{a} - \sum_{i \in \mathcal{N}_p} w_i \boldsymbol{a}_i)^2}{2\varsigma_r^2} \right). \tag{6}$$

In other words, we discard samples from the clusters where the reconstruction error of the input appearance $\boldsymbol{a}$ is significant. In Equation (6), the factor ς_r is empirically set. We define the prediction confidence c_k as a product of the reliability $r(\boldsymbol{a})$ and the pose similarity $s(\boldsymbol{p})$ as

$$c_k = s_k(\boldsymbol{p}) r_k(\boldsymbol{a}). \tag{7}$$

In this manner, the prediction confidence embeds the reliability of the k-th cluster as well as the similarity with the neighboring samples. The final gaze prediction $\hat{\boldsymbol{g}}$ is computed as a weighted average of the intermediate predictions $\hat{\boldsymbol{g}}_k$ using the prediction confidence c_k as

$$\hat{\boldsymbol{g}} = \frac{\sum_k c_k \hat{\boldsymbol{g}}_k}{\sum_k c_k}. \tag{8}$$

To assess the overall reliability of the gaze estimate $\hat{\boldsymbol{g}}$, we further compute the weighted average of $r_k(\boldsymbol{a})$ using the similarity measure s_k as weights:

$$\bar{r}(\boldsymbol{p}, \boldsymbol{a}) = \frac{\sum_k s_k(\boldsymbol{p}) r_k(\boldsymbol{a})}{\sum_k s_k(\boldsymbol{p})}. \tag{9}$$

Figure 3 shows the angular error plot of the gaze estimate $\hat{\boldsymbol{g}}$ across the prediction reliability $\bar{r}$. The plot shows that the accuracy of estimation increases as the reliability measure increases. From this observation, we stabilize the gaze estimate $\hat{\boldsymbol{g}}$ by taking a weighted temporal average obtained from consecutive frames based on the reliability $\bar{r}$. The effect of the temporal averaging is discussed in Section VI.

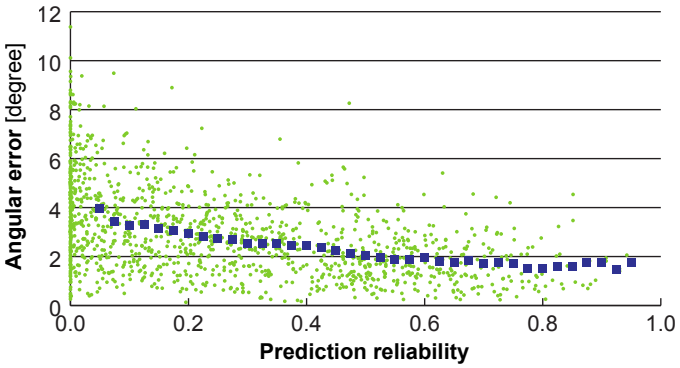

Fig. 3. Angular error of the gaze estimate $\hat{\boldsymbol{g}}$ against prediction reliability $\bar{r}$ (Eq. (9)). The bold rectangles represent local averages with a window width of 0.1 along the reliability axis.

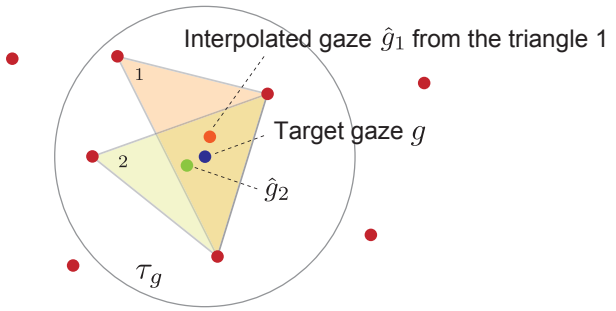

Fig. 4. The gaze label of the sample $\boldsymbol{g}$ is refined using the weighted average of interpolated labels $\boldsymbol{g}_i$ on the neighboring triangles (within the distance threshold τ_g).

B. Learning

When the user clicks a mouse button, our system takes the clicked position (as the labeled gaze $\boldsymbol{g}$) as well as the user's head pose $\boldsymbol{p}$ and eye image $\boldsymbol{x}$ as a training sample. Given the i-th training sample $\{\boldsymbol{x}_i, \boldsymbol{p}_i, \boldsymbol{g}_i\}$, our method updates the appearance subspace using Skocaj *et al.* [35]'s incremental PCA method. The mean eye image $\bar{\boldsymbol{x}}$ and the basis matrix $\boldsymbol{U}$ in Equation (2) as well as all the previous coefficients $\{\boldsymbol{a}_1 \dots \boldsymbol{a}_{i-1}\}$ are updated at the same time.

After updating the appearance subspace, the reduced learning sample $\{\boldsymbol{a}_i, \boldsymbol{p}_i, \boldsymbol{g}_i\}$ is added to a pose cluster only when its head pose $\boldsymbol{p}_i$ is sufficiently close to the cluster's center. With this approach, all clusters are guaranteed to contain samples with similar head poses. Tan *et al.* [26] use a topological manifold model with a similar setting to ours. In their method, however, the gaze label $\boldsymbol{g}$ is treated as a static quantity without any error. In reality, humans cannot gaze at a point with a pixel-level accuracy. Even when a user is looking at a target carefully, a certain level of fixational eye movement occurs. About 1-degree of microsaccades occur during fixation [39]. In our case, since the user is not forced to look carefully at the mouse cursor when clicking, larger errors can be included in the gaze label $\boldsymbol{g}$. Moreover, there could be meaningless samples due to random mouse clicks without sufficient attention.

For these reasons, we avoid directly using the clicked coordinates $\boldsymbol{g}_{\mathrm{c}}$ as a gaze label $\boldsymbol{g}$. Instead, we estimate the probable gaze label $\boldsymbol{g}$ constrained by $\boldsymbol{g}_{\mathrm{c}}$ using a refining approach starting with $\boldsymbol{g}_{\mathrm{c}}$ as an initial value for $\boldsymbol{g}$. To refine the gaze label $\boldsymbol{g}$ of the sample, we first select all existing samples whose distance to the incoming sample's click point $\boldsymbol{g}_{\mathrm{c}}$ are under a threshold τ_g in the gaze space. Using these existing samples and incoming-sample $\boldsymbol{g}$, the set of all combinations of three samples that are nearby and enclose incoming-sample $\boldsymbol{g}$ can be computed (see Figure 4). Using the method described in the previous section, interpolated gaze label $\boldsymbol{g}_i$ can be computed from each triangle. All the interpolated labels are aggregated as a Gaussian-weighted average around $\boldsymbol{g}_{\mathrm{c}}$ as

$$\boldsymbol{g} = \frac{\boldsymbol{g}_{\mathrm{c}} + \sum_i r_i q_i \boldsymbol{g}_i}{1 + \sum_i r_i q_i}, \tag{10}$$

where the Gaussian weighting factor q_i is defined as

$$q_i = \exp\left(-\frac{||\boldsymbol{g}_i - \boldsymbol{g}_{\mathrm{c}}||^2}{2\varsigma_q^2}\right). \tag{11}$$

In the above equations, i is the triangle index, and r_i is the reliability measure calculated as in Equation (6). The factor ς_q is empirically set. The clicked point $\boldsymbol{g}_{\mathrm{c}}$ is added with the full-weight 1.

As mentioned above, there are incoming samples that are inadequate as learning samples, *e.g.*, clicks without due attention. These samples do not convey the correlation between the appearance and gaze label (clicked point). To avoid such outliers, we assess the data through cross validation. In addition to computing the interpolated gaze label $\boldsymbol{g}$, we can compute a standard interpolation $\dot{\boldsymbol{g}}$ without the constraint of $\boldsymbol{g}_{\mathrm{c}}$ as

$$\dot{\boldsymbol{g}} = \frac{\sum_i r_i \boldsymbol{g}_i}{\sum_i r_i}. \tag{12}$$

If the distance $d_g = ||\dot{\boldsymbol{g}} - \boldsymbol{g}_{\mathrm{c}}||^2$ is too large, *i.e.*, the interpolated gaze $\dot{\boldsymbol{g}}$ is too far from the clicked point $\boldsymbol{g}_{\mathrm{c}}$, the sample can be considered as an outlier. In that case, we eliminate the sample from the cluster instead of refining its gaze label.

To avoid biased distribution of the training samples in the gaze space, we further prune the learning samples to improve the quality of the training data when the density of the training samples becomes high. If there is more than one sample within radius τ_r around the position of incoming gaze label $\boldsymbol{g}$, we keep the nearest sample (with the lowest d_g) and eliminate the other samples. The threshold value τ_r should be set with respect to both the size of the display area and memory capacity. For example, Tan *et al.* [26] used one sample per 2.76 cm^2. Whenever a new incoming sample is provided, the data resampling process described above is executed for every sample in the clusters.

Once a sample is added to the cluster, we incrementally update the cluster mean $\bar{\boldsymbol{t}}_k$ and variance $\sigma_{t,k}^2$ of Equation (1) as

$$\begin{aligned} \bar{\boldsymbol{t}} &\leftarrow \frac{n\bar{\boldsymbol{t}} + \boldsymbol{t}}{n+1}, \\ \sigma_t^2 &\leftarrow \frac{n\sigma_t^2 + (\boldsymbol{t} - \bar{\boldsymbol{t}}_{\mathrm{new}})(\boldsymbol{t} - \bar{\boldsymbol{t}}_{\mathrm{old}})}{n+1}, \end{aligned} \tag{13}$$

where n denotes the number of samples in the cluster before updating, $\boldsymbol{t}$ represents the translation vector of the incoming sample, and $\bar{\boldsymbol{t}}_{\mathrm{old}}$ and $\bar{\boldsymbol{t}}_{\mathrm{new}}$ are the cluster means before and after

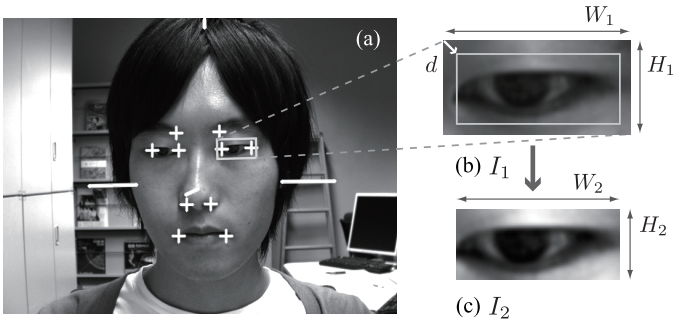

Fig. 5. Head pose tracking and eye image capturing. (a) Estimation of the head pose. The lines represent the normal directions of the bounding box of the user's head. (b) Cropped eye image. The eye image is cropped based on the predefined eye region on the face mesh (rectangle in (a)). (c) Cropped eye image after post-processing. The eye location is aligned and intensity is enhanced. This image is used as the eye appearance feature.

the update, respectively. The updating procedure is applied to the rotation component $\bar{\boldsymbol{r}}$ and its variance σ_r^2. After, the Delaunay triangulation in the gaze-label coordinates is recomputed.

V. Implementation

In this section we describe the methods of obtaining input features, *i.e.*, the head pose $\boldsymbol{p}$ and eye image $\boldsymbol{x}$ from a sequence of gray-scale input images. We also explain the method of detecting blinks for improving the accuracy of gaze estimates.

A. Head pose tracking

Our method uses the head-tracking method [33] based on a multi-linear model, which represents face shape variations by two separate factors: variations across people and facial expressions. The head data are represented as the appearance of the face and 3-D positions of 10 feature points defined in the local-coordinate system of the user's head (Figure 5 (a)). In this work, we ignore the shape variations caused by facial expressions and use a simplified linear model to adapt to the variations across people. Our method precomputes 8 eigenshapes from the database of facial shapes, and the face shape is represented by the linear combination of the basis shapes. Using the model, our system simultaneously tracks the 3-D head pose using a particle filter [40] and estimates the face shape based on bundle adjustment [41]. As a result, the tracker outputs the user's 3-D head pose $\boldsymbol{p} = \{\boldsymbol{t}, \boldsymbol{r}\}$, where $\boldsymbol{t} = {}^t(x, y, z)$ is a 3-D translation and $\boldsymbol{r} = {}^t(q_1, q_2, q_3, q_4)$ is a 4-D rotation vector defined by four quaternions. Figure 5 (a) shows an example of head pose tracking. The crosses indicate the positions of the detected feature points, and the lines represent the head pose.

B. Eye image cropping

Once the head pose $\boldsymbol{p}$ is estimated, the system crops the eye image $\boldsymbol{x}$. Using the estimated head position $\boldsymbol{p}$, it first extracts a rough eye region from the input image using the predefined eye location in the generic 3-D face model. Based on the distance between the two eye corners in the image coordinates, the rectangular region with a fixed aspect ratio (the rectangle in Figure 5 (a)) is cropped. The rectangle is then re-scaled to a normalized $W_1 \times H_1$ image $\boldsymbol{I}_1$ (Figure 5 (b)). We further apply histogram equalization to normalize its brightness to obtain the final eye image $\boldsymbol{I}_2$.

While head pose tracking is robust, there still remains a small error when cropping eye images. This error appears as a small amount of jittering in the eye image sequence. For an appearance-based method, accurate alignment of eye images is crucial. To improve the alignment, we apply a subspace alignment method as described below.

The eye image $\boldsymbol{I}_2$ of size $W_2 \times H_2$ (Figure 5 (c)) is cropped from the larger image $\boldsymbol{I}_1$ of size $W_1 \times H_1$ with a top left margin $\boldsymbol{d} = {}^t(x, y)$. As described in Section IV, the PCA subspace used in the learning algorithm is updated incrementally using the labeled samples. Our method tries to find the eye image $\boldsymbol{I}_2$ that maximizes the correlation with the reconstruction image $\acute{\boldsymbol{I}}_2$. created from the appearance subspace. The vector form of the reconstruction image $\acute{\boldsymbol{x}}_2$ is computed as

$$\acute{\boldsymbol{x}}_2 = \bar{\boldsymbol{x}} + \boldsymbol{U}{}^t\boldsymbol{U}(\boldsymbol{x}_2 - \bar{\boldsymbol{x}}), \tag{14}$$

where $\boldsymbol{x}_2$ is a vector form of the eye image $\boldsymbol{I}_2$, $\bar{\boldsymbol{x}}$ is the average eye image, and $\boldsymbol{U}$ is a matrix that consists of eigenvectors of the PCA subspace.

To find the optimal cropping region, a correlation map $\boldsymbol{C}$ is computed in a brute force manner in the area of $(W_1 - W_2 + 1) \times (H_1 - H_2 + 1)$. The value in the correlation map $\boldsymbol{C}(x, y)$ corresponds to the correlation between $\boldsymbol{I}_2$ and $\acute{\boldsymbol{I}}_2$ with an offset $\boldsymbol{d} = {}^t(x, y)$. The offset in the pixel level accuracy is determined by taking the point (x, y) that gives the maximal value of $\boldsymbol{C}(x, y)$. Using this solution as the initial guess, we further compute the sub-pixel alignment using a simple 2-D parabola fitting described as

$$\begin{bmatrix} \frac{\partial \boldsymbol{C}(x+\delta_x, y+\delta_y)}{\partial x} \\ \frac{\partial \boldsymbol{C}(x+\delta_x, y+\delta_y)}{\partial y} \end{bmatrix} = \boldsymbol{0}, \tag{15}$$

where δ_x and δ_y are the subpixel displacement along the x- and y-axes, respectively. Equation (15) can be approximated by a Taylor expansion around (x, y) as

$$\begin{bmatrix} \boldsymbol{C}'_x \\ \boldsymbol{C}'_y \end{bmatrix} + \begin{bmatrix} \boldsymbol{C}''_{xx} & \boldsymbol{C}''_{xy} \\ \boldsymbol{C}''_{xy} & \boldsymbol{C}''_{yy} \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} = \boldsymbol{0}, \tag{16}$$

and the subpixel displacement (δ_x, δ_y) is obtained by

$$\begin{cases} \delta_x = \frac{\boldsymbol{C}'_y \boldsymbol{C}''_{xy} - \boldsymbol{C}'_x \boldsymbol{C}''_{yy}}{\boldsymbol{C}''_{xx} \boldsymbol{C}''_{yy} - (\boldsymbol{C}''_{xy})^2} \\ \delta_y = \frac{\boldsymbol{C}'_x \boldsymbol{C}''_{xy} - \boldsymbol{C}'_y \boldsymbol{C}''_{xx}}{\boldsymbol{C}''_{xx} \boldsymbol{C}''_{yy} - (\boldsymbol{C}''_{xy})^2} \end{cases}. \tag{17}$$

Here, $\boldsymbol{C}'$ and $\boldsymbol{C}''$ are the 1st and 2nd order derivatives of $\boldsymbol{C}$ at (x, y).

Finally, $\boldsymbol{I}_2$ is cropped with the offset $(x+\delta_x, y+\delta_y)$ to create the vectorized form of the eye image $\boldsymbol{x}$. In our configuration, the size of the final image is set to $(W_2, H_2) = (70, 30)$. In most cases eye images were around $80 \sim 100$ pixels wide, and larger than the final resolution.

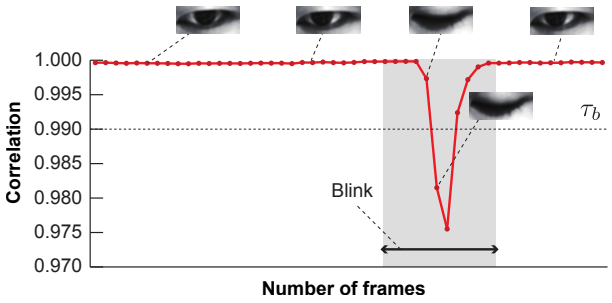

Fig. 6. Blink detection. The graph shows the correlation between the cropped eye image $\boldsymbol{I}_2$ and reconstruction image $\acute{\boldsymbol{I}}_2$. The eye images correspond to $\boldsymbol{I}_2$. The correlation drops significantly when the user blinks.

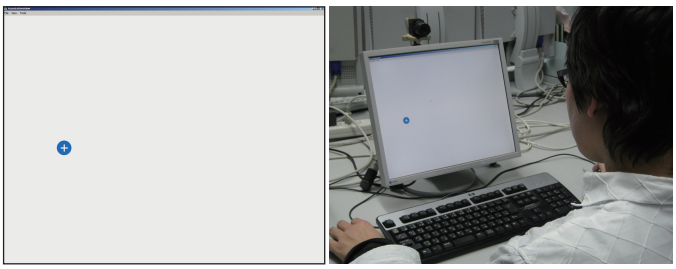

Fig. 7. System setting for the experiments. Left figure shows the screen-shot of the full-screen window shown to the user. Right figure shows the experiment setup.

C. Blink detection

If a user blinks when clicking, the data are inappropriate for training. To eliminate such samples, our method automatically detects blinks based on the correlation of the incoming eye image and the accumulated eye images. As described in the previous section, cropping of eye images is performed by finding the optimal offset where the correlation with the reconstructed image is maximized. However, if the input eye image is dissimilar to any samples that span the subspace (*e.g.*, the blinking case), the maximum correlation becomes relatively small. From this observation, our method finds the blinking eye images by evaluating the correlation as illustrated in Figure 6. If the correlation is lower than a pre-defined threshold τ_b, we treat the sample as an inappropriate sample.

A blink of an eye usually lasts for about 150 ms, which is long enough to appear in multiple video frames as illustrated in Figure 6. Therefore, we discard the neighboring frames of the detected blinks within a certain time range.

VI. Proof of Concept Evaluations

A. Apparatus

Our system consists of a VGA resolution camera (PointGrey Flea) and a Windows PC with a 2.67 GHz dual core CPU and 3 GB of RAM. The processing times are about 2 ms for the head tracking, 20 ms for eye cropping and alignment, 20 ms for gaze estimation, and 25 ms for learning. The entire estimation process including display rendering runs at about 20 fps in our research implementation. Throughout the experiments, we used the following parameters: $\kappa_t = \kappa_r = 2.0$, $\tau_x = 0.001$, $N_s = 30$, $n_s = 10$, $\varsigma_r^2 = 25000$, $\varsigma_q^2 = 2500$, $\tau_g = 100$ px, $\tau_r = 30$ px, and $\tau_b = 0.99$. We used a 17-inch display with a resolution of 1280×1024 pixels (96 dpi).

B. Evaluation with random targets

1) Participants: Ten (nine male and one female) users who did not wear glasses participated. Their ages ranged from 27 to 32, and the average age of the participants was 28.9 with a standard deviation of 1.8.

2) Procedure: We first conduct the experiment using random click targets. A target for clicking is randomly shown to the user in a full-screen window. To simulate a typical target, like a button or an icon on the desktop, we use a circle with a 64-pixel diameter (Figure 7). The users were asked to click the displayed targets as usual. During the operation, the users are allowed to freely move their head poses. Experiments were conducted for about 20 minutes (until about 1200 clicks) to evaluate the performance variation across the running time and diverse head pose variations.

3) Dependent measures: Whenever a new labeled sample is given, the prediction is performed prior to the learning. The estimation error is evaluated as the distance between the clicked position $\boldsymbol{g}_t$ and the estimated gaze position $\hat{\boldsymbol{g}}_t$. The angular error θ is computed as

$$\theta_t = \tan^{-1}\left(\frac{D_m(\boldsymbol{g}_t, \hat{\boldsymbol{g}}_t)}{z_t - d_{\text{cam}}}\right), \tag{18}$$

where D_m indicates the distance between two points in the metric unit, z_t is the depth of the estimated head pose at time t in the camera coordinate system, and d_{cam} is the pre-defined distance between the camera and the display. The d_{cam} is 10 cm in our configuration.

4) Results: Table I shows the angular and pixel errors (denoted as *average* $\pm$ *standard deviation*), click count, the numbers of clusters. "Normal" error corresponds to the error of the raw output $\hat{\boldsymbol{g}}$, and "weighted" error indicates the error of the results with the weighted temporal averaging (taking the past 5 frames in this experiment) based on the weight $\bar{r}$ in Equation (9). "Used" clicks denote the number of clicks that are not discarded by the rejection process described in Section V-C. The last six columns show the ranges of head movement of each user. Translation ranges x, y, and z correspond to horizontal, vertical, and depth directions, and rotation ranges ϕ, θ, ψ correspond to angles around the z-, x-, and y-axes, respectively. The average range in the experiment was $23 \times 7 \times 35$ cm and $15 \times 32 \times 24$ degrees.

The angular error is consistently low across different users (around 3 degrees), and it demonstrates the better performance when the weighted temporal averaging is used. Figure 8 shows the evolution of the average of weighted angular error against the number of clicks. There are some variations across users, in the early stage of the learning, *e.g.*, less than 400 clicks, and the errors generally tend to be higher due to an insufficient number of learning samples. However, the errors consistently converge to a certain range after 600 clicks and do not diverge throughout the sessions.

Table II compares our method to a commercial gaze tracker

TABLE I
RESULT USING RANDOM TARGETS. "NORMAL" ERROR CORRESPONDS TO THE RAW OUTPUT $\hat{\boldsymbol{g}}$, OF THE SYSTEM, AND "WEIGHTED" ERROR CORRESPONDS TO THE TEMPORAL WEIGHTED AVERAGE BASED ON THE WEIGHT $\bar{r}$ IN EQ. (9). ϕ, θ, AND ψ CORRESPOND TO THE ROTATION ANGLES AROUND THE z-, x-, AND y-AXES, RESPECTIVELY.

Person		Angular error deg		Pixel error px		Num. clicks	Num.	Trans. cm			Rot. deg		
		Weighted	Normal	Weighted	Normal	Used/All	clusters	x	y	z	ϕ	θ	ψ
	A	2.5 ± 1.5	2.9 ± 1.8	89 ± 58	105 ± 67	1313/1313	13	16	7	31	5	18	17
	B	3.0 ± 2.2	3.7 ± 2.8	135 ± 101	165 ± 130	1293/1302	11	27	10	36	9	32	19
	C	2.4 ± 1.5	3.0 ± 1.9	102 ± 65	126 ± 82	1305/1308	7	23	3	37	6	32	12
	D	3.1 ± 1.5	3.7 ± 2.7	107 ± 73	129 ± 92	1301/1302	7	22	8	31	7	29	21
	E	3.0 ± 2.3	3.3 ± 2.4	126 ± 92	140 ± 101	1226/1248	21	27	6	52	16	33	29
	F	3.2 ± 2.0	3.6 ± 2.3	150 ± 95	170 ± 112	1318/1319	17	34	7	37	18	36	25
	G	2.8 ± 2.0	3.5 ± 3.0	120 ± 85	148 ± 130	1308/1312	11	19	6	34	17	23	28
	H	3.1 ± 2.2	3.6 ± 2.6	150 ± 105	174 ± 125	1305/1308	7	23	8	31	8	29	16
	I	2.8 ± 2.2	3.1 ± 2.4	110 ± 89	122 ± 99	1278/1309	13	21	10	34	41	43	29
	J	3.3 ± 2.6	3.7 ± 2.9	145 ± 117	164 ± 132	1267/1315	17	17	8	23	22	42	43
Average		2.9 ± 2.1	3.4 ± 2.5	123 ± 88	144 ± 107			23	7	35	15	32	24

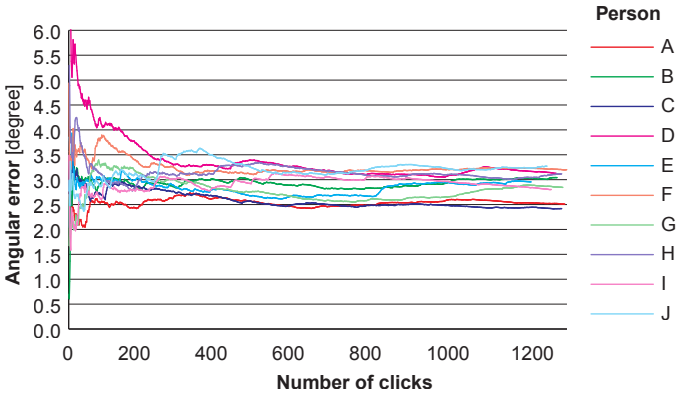

Fig. 8. Evolution of the average angular error. The graph shows the average of the weighted angular error in the random-target experiments against the number of clicks. Each line corresponds to each user in Table I.

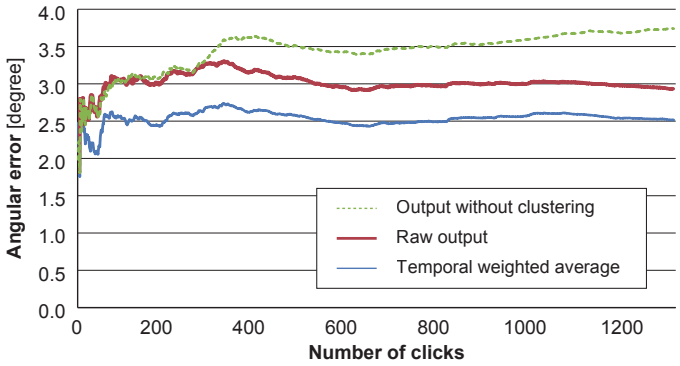

Fig. 9. Comparison of errors with and without the clustering. The plot shows the average errors of User A using random targets. The red and blue lines correspond to the results with clustering (normal and weighted output, respectively), and the green line corresponds to normal output without clustering.

TABLE II
COMPARISONS WITH A COMMERCIAL GAZE TRACKER AND CAMERA-BASED GAZE ESTIMATION METHODS THAT ALLOWS FREE HEAD MOVEMENTS [29], [30], [32].

Method	Angular error deg	Movement range cm	Missing estimation rate
Proposed	2.9 ± 2.1	23 × 7 × 35	-
TX300	1.7 ± 2.4	29 × 12 × 31	35%
Lu *et al.* [29]	2.38 [29]	-	-
Lu *et al.* [30]	2.24 [30]	-	-
Valenti *et al.* [32]	3 ∼ 5 [32]	-	-

(Tobii TX300[1]) and camera-based gaze estimation methods that allows free head movements [29], [30], [32]. Average estimation error of TX300 is evaluated with the same experimental setting, *i.e.*, users were instructed to click random targets under free head movement. The second column in Table II shows an average angular error of 10 users, where each of the users clicked about 300 times. The other columns show head movement range during the experiments and the percentage of frames with missing estimation results. While the estimation accuracy of TX300 is better than our camera-based approach, when the users are freely moving their heads, the error tends to become higher than 1 degree and the system frequently returns missing estimation results. The overall missing estimation rate of TX300 was about 35%. For the other three methods, their reported estimation errors are shown. These methods depend on explicit calibration stages, and [32] is the only method which is reported to work in real-time. Despite a lack of reliable calibration data obtained through an active calibration scheme, our method can achieve similar accuracy to these methods.

To assess the effectiveness of our clustering approach, we compared the performance with and without the clustering method. Figure 9 shows one of the results. The plots represent the evolution of the average angular errors of the gaze estimates for User A. The middle and the bottom lines correspond to the clustering results (normal and weighted output), and the top line corresponds to the output without clustering, *i.e.*, all samples are added to a single cluster. From the plot, the clustering approach consistently improves the performance. The error gradually increased and did not converge to a certain

[1]http://www.tobii.com/

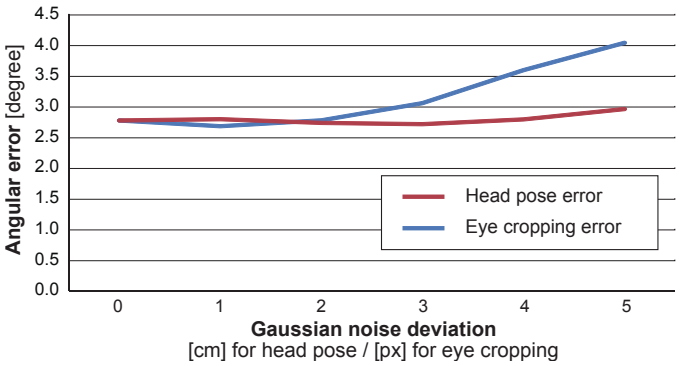

Fig. 10. Estimation error with respect to input noise. Two lines correspond to average angular errors plotted against standard deviation of the noise added to head translation and eye cropping position.

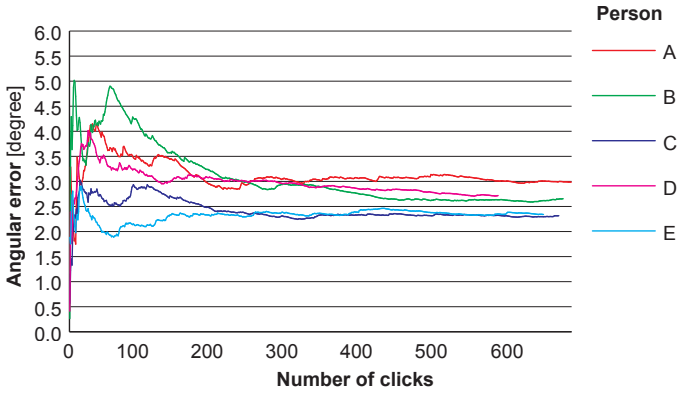

Fig. 11. Evolution of the average angular error in the real-world setting.

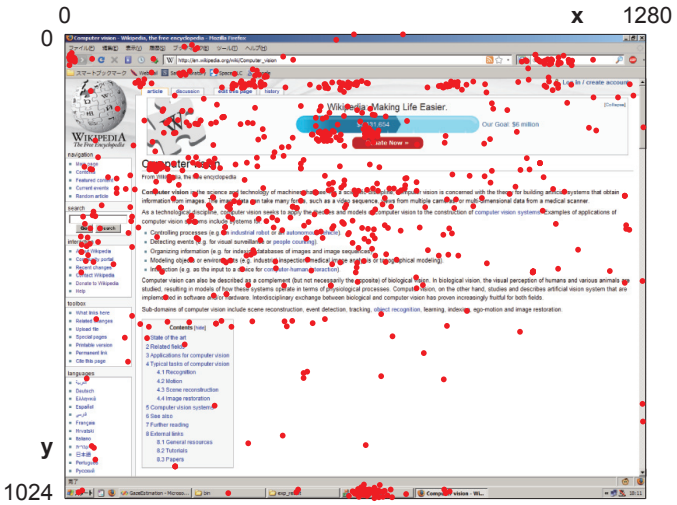

Fig. 12. Distribution of clicked points by User A in the desktop-environment scenario. Clicked points are distributed in the 1280×1024 desktop space.

error without the clustering. In addition, by comparing the middle and bottom lines, we can see that the estimation error is greatly reduced by taking the weighted temporal average. The percentage of the reduced error is about 85% on average for all users when the weighted temporal averaging is used.

We further conducted a performance comparison with artificial error to quantitatively evaluate the robustness against input noise. In Figure 10, Gaussian noise was added to two factors of the input information and average angular errors of 10 users were plotted against the standard deviation of the noise. For head pose, the noise was added to estimated head translations with the standard deviation ranging from 0 to 5 cm. The estimation error does not diverge greatly and our method can robustly handle noisy head pose. This is mainly because the head pose is indirectly used as a cue to evaluate cluster similarity in our method and it does not heavily rely on geometric information. For eye cropping, alignment error was added to the re-scaled eye image $\boldsymbol{I}_1$ with the standard deviation ranging from 0 to 5 pixels. Although the error is greater when there is larger cropping noise than, *e.g.*, 3 pixels, our method can compensate small cropping noise through the sub-pixel alignment step.

C. Evaluation with desktop environment

Five users from the prior evaluation browsed web pages for about 30 minutes. The average click count for a user is around 600. To create a natural desktop environment for testing, we implemented a global system hook that ran as a background process to capture the click events and positions. The system continuously captures the user's head poses and eye images as a background process.

Table III shows the angular and pixel errors, click count, the numbers of clusters, and the range of head pose movement. As in the previous experiment, clicked coordinates are used as the ground-truth gaze positions. Our method achieves an angular error of 2.6 degrees on average. Although the click counts and head pose variations are limited, the accuracy of gaze estimation is comparable with the result using random targets. Figure 11 shows the average of the weighted angular error with respect to the click count. We observed that the error converged in a similar manner with the previous result.

One of the most important factors in this experiment setting was the biased distribution of the click positions. Figure 12 shows the distribution of the clicked positions by User A. We can see more clicks on menu buttons and at the top of the desktop. The distribution of click points in the real-world scenario is expected to be biased, like this example. For the areas with sparse gaze labels, it is hard to achieve a good estimation. However, it is expected that the possibility of the user to steadily look at such areas is low. Another interesting observation is that the distribution of the gaze labels varies with the tasks and application scenarios. Our method can get updated with the changes in the distribution because of the incremental learning approach.

VII. Discussion

We proposed an appearance-based gaze estimation method using an incremental learning approach. The proposed method is developed for a desktop scenario, where a user clicks a mouse in PC operations. The clicked position is used as a gaze label, and the head pose and appearance of the eye are recorded with a standard desktop camera. To allow free head movement, we used a 3-D head pose tracker and proposed a clustering-based method for learning pose-dependent mapping

TABLE III
RESULT USING DESKTOP ENVIRONMENT. FROM LEFT TO RIGHT, ANGULAR AND PIXEL ERRORS, CLICK COUNT, THE NUMBER OF CLUSTERS, AND RANGE OF HEAD POSE MOVEMENT ARE SHOWN.

Person	Angular error deg		Pixel error px		Num. clicks	Num.	Trans. cm			Rot. deg		
	Weighted	Normal	Weighted	Normal	Used/All	clusters	x	y	z	ϕ	θ	ψ
 A	3.0 ± 2.8	3.4 ± 3.2	111 ± 108	127 ± 124	717/728	4	14	6	28	9	23	17
 B	2.7 ± 2.1	3.0 ± 2.4	119 ± 95	136 ± 110	706/718	6	8	5	31	26	47	20
 C	2.3 ± 1.7	2.8 ± 2.0	121 ± 90	148 ± 104	700/700	3	18	2	28	4	22	10
D	2.7 ± 1.7	3.3 ± 2.1	112 ± 73	134 ± 90	618/619	1	3	4	13	7	9	11
 E	2.3 ± 1.9	2.7 ± 2.2	92 ± 78	105 ± 88	679/692	3	4	4	10	8	21	16
Average	2.6 ± 2.1	3.0 ± 2.4	111 ± 89	130 ± 103			10	4	22	11	25	14

functions between eye appearances and gaze points. We further introduced methods of subspace eye alignment and gaze label refinement to enhance the estimation accuracy.

While a limitation of our approach is the time is takes for the gaze estimation process, the effectiveness of the proposed method is validated through proof of concept evaluations, and our method achieved an estimation accuracy of 2.9 degrees without temporal smoothing. Although less accurate than state-of-the-art commercial products, our method works with a single camera without any special hardware and does not require active participation in the calibration task. This enables a scenario where users can use a gaze estimation system that adaptively learns to predict their gaze positions through the users' daily activities on a standard PC. The performance of the proposed method is accurate enough to infer the user's area of interest and their corresponding desktop UI components.

ACKNOWLEDGMENT

This research was supported by the Microsoft Research IJARC Core Project and JST CREST.

REFERENCES

[1] E. Horvitz, C. Kadie, T. Paek, and D. Hovel, "Models of attention in computing and communication: from principles to applications," *Communications of the ACM*, vol. 46, no. 3, pp. 52–59, 2003.

[2] R. Vertegaal, J. Shell, D. Chen, and A. Mamuji, "Designing for augmented attention: Towards a framework for attentive user interfaces," *Computers in Human Behavior*, vol. 22, no. 4, pp. 771–789, 2006.

[3] D. Rozado, "Mouse and keyboard cursor warping to accelerate and reduce the effort of routine hci input tasks," *Human-Machine Systems, IEEE Transactions on*, vol. 43, no. 5, pp. 487–493, 2013.

[4] R. J. K. Jacob, "Eye movement-based human-computer interaction techniques: Toward non-command interfaces," *Advances in Human-Computer Interaction*, vol. 4, pp. 151–190, 1993.

[5] Y. Sugano, Y. Matsushita, and Y. Sato, "Calibration-free gaze sensing using saliency maps," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR2010)*, 2010, pp. 2667–2674.

[6] ——, "Appearance-based gaze estimation using visual saliency," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 2, pp. 329–341, 2013.

[7] J. Chen and Q. Ji, "Probabilistic gaze estimation without active personal calibration," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 2011)*, 2011, pp. 609–616.

[8] F. Alnajar, T. Gevers, R. Valenti, and S. Ghebreab, "Calibration-free gaze estimation using human gaze patterns," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2013)*, 2013, pp. 137–144.

[9] Y. Sugano, Y. Matsushita, Y. Sato, and H. Koike, "An incremental learning method for unconstrained gaze estimation," in *Proceedings of the 10th European Conference on Computer Vision (ECCV 2008)*, 2008, pp. 656–667.

[10] T. E. Hutchinson, K. P. White Jr., W. N. Martin, K. C. Reichert, and L. A. Frey, "Human-computer interaction using eye-gaze input," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 6, pp. 1527–1534, 1989.

[11] R. J. K. Jacob, "What you look at is what you get: eye movement-based interaction techniques," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 1990, pp. 11–18.

[12] Y. Matsumoto, T. Ogasawara, and A. Zelinsky, "Behavior recognition based on head pose and gaze direction measurement," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)*, vol. 3, 2000, pp. 2127–2132.

[13] C. Morimoto, A. Amir, and M. Flickner, "Detecting eye position and gaze from a single camera and 2 light sources," in *Proceedings of the 16th International Conference on Pattern Recognition (ICPR 2002)*, 2002, pp. 314–317.

[14] J. G. Wang and E. Sung, "Study on eye gaze estimation," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 32, no. 3, pp. 332–350, 2002.

[15] D. Beymer and M. Flickner, "Eye gaze tracking using an active stereo head," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, vol. 2, 2003, pp. 451–458.

[16] S. W. Shih and J. Liu, "A novel approach to 3-d gaze tracking using stereo cameras," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 34, no. 1, pp. 234–245, 2004.

[17] Z. Zhu, Q. Ji, and K. P. Bennett, "Nonlinear eye gaze mapping function estimation via support vector regression," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR 2006)*, vol. 1, 2006, pp. 1132–1135.

[18] C. Hennessey, B. Noureddin, and P. Lawrence, "A single camera eye-gaze tracking system with free head motion," in *Proc. 2006 symposium on eye tracking research & applications*, 2006, pp. 87–94.

[19] D. H. Yoo and M. J. Chung, "A novel non-intrusive eye gaze estimation using cross-ratio under large head motion," *Computer Vision and Image Understanding*, vol. 98, no. 1, pp. 25–51, 2005.

[20] F. L. Coutinho and C. H. Morimoto, "Free head motion eye gaze tracking using a single camera and multiple light sources," in *Proceedings of the Brazilian Symposium on Computer Graphics and Image Processing*, 2006, pp. 171–178.

[21] T. Ishikawa, S. Baker, I. Matthews, and T. Kanade, "Passive driver gaze tracking with active appearance models," in *Proceedings of the 11th World Congress on Intelligent Transportation Systems*, 2004.

[22] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 681–685, 2001.

[23] H. Yamazoe, A. Utsumi, T. Yonezawa, and S. Abe, "Remote gaze estimation with a single camera based on facial-feature tracking without special calibration actions," in *Proceedings of the 2008 symposium on eye tracking research & applications*, 2008, pp. 245–250.

[24] S. Baluja and D. Pomerleau, "Non-intrusive gaze tracking using artificial neural networks," *Advances in Neural Information Processing Systems (NIPS)*, vol. 6, pp. 753–760, 1994.

[25] L. Q. Xu, D. Machin, and P. Sheppard, "A novel approach to real-time non-intrusive gaze finding," in *Proceedings of the British Machine Vision Conference*, 1998, pp. 428–437.

[26] K. H. Tan, D. J. Kriegman, and N. Ahuja, "Appearance-based eye gaze estimation," in *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision (WACV 2002)*, 2002, pp. 191–195.

[27] O. Williams, A. Blake, and R. Cipolla, "Sparse and semi-supervised visual mapping with the S^3GP," *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 230–237, 2006.
[28] F. Lu, Y. Sugano, T. Okabe, and Y. Sato, "Inferring human gaze from appearance via adaptive linear regression," in *Proc. the 13th IEEE International Conference on Computer Vision (ICCV 2011)*, 2011, pp. 153–160.
[29] F. Lu, T. Okabe, Y. Sugano, and Y. Sato, "A head pose-free approach for appearance-based gaze estimation." in *Proc. 22nd British Machine Vision Conference (BMVC2011)*, 2011, pp. 1–11.
[30] F. Lu, Y. Sugano, T. Okabe, and Y. Sato, "Head pose-free appearance-based gaze sensing via eye image synthesis," in *Proc. 21st International Conference on Pattern Recognition (ICPR2012)*, 2012, pp. 1008–1011.
[31] K. A. Funes Mora and J.-M. Odobez, "Gaze estimation from multimodal kinect data," in *Proc. CVPR 2012 Workshop on Gesture Recognition*, 2012, pp. 25–30.
[32] R. Valenti, N. Sebe, and T. Gevers, "Combining head pose and eye location information for gaze estimation," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 802–815, 2012.
[33] Y. Sugano and Y. Sato, "Person-independent monocular tracking of face and facial actions with multilinear models," in *Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures (AMFG2007)*, 2007, pp. 58–70.
[34] I. T. Jollife, *Principal Component Analysis*, ser. Springer Series in Statistics. New York: Springer-Verlag, 1986.
[35] D. Skocaj and A. Leonardis, "Weighted and robust incremental method for subspace learning," in *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003)*, 2003, pp. 1494–1501.
[36] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[37] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.
[38] M. Humbert, N. Gey, J. Muller, and C. Esling, "Determination of a mean orientation from a cloud of orientations. application to electron back-scattering pattern measurements," *Journal of Applied Crystallography*, vol. 29, pp. 662–666, 1996.
[39] S. Martinez-Conde, S. L. Macknik, and D. H. Hubel, "The role of fixational eye movements in visual perception," *Nature Reviews Neuroscience*, vol. 5, pp. 229–240, 2004.
[40] M. Isard and A. Blake, "CONDENSATION – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.
[41] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment – a modern synthesis," *Lecture Notes in Computer Science*, vol. 1883, pp. 298–372, 1999.

Yusuke Sugano is a postdoctoral researcher at the Perceptual User Interfaces Group at the Max Planck Institute for Informatics. He received his M.S. and Ph.D. degrees in information science and technology from the University of Tokyo, Japan, in 2007 and 2010 respectively. His research interests include eye tracking, computer vision and human-computer interaction.

Yasuyuki Matsushita received his B.S., M.S. and Ph.D. degrees in EECS from the University of Tokyo in 1998, 2000, and 2003, respectively. He joined Microsoft Research Asia in April 2003. He is a Senior Researcher in Visual Computing Group. His areas of research are computer vision (photometric techniques, such as radiometric calibration, photometric stereo, shape-from-shading), computer graphics (image relighting, video analysis and synthesis). He is a senior member of IEEE.

Yoichi Sato is a professor at Institute of Industrial Science, the University of Tokyo. He received his B.S. degree from the University of Tokyo in 1990, and his MS and PhD degrees in robotics from School of Computer Science, Carnegie Mellon University in 1993 and 1997 respectively. His research interests include physics-based vision, reflectance analysis, image-based modeling and rendering, and gaze and gesture analysis.

Hideki Koike is a professor of Department of Computer Science, Tokyo Institute of Technology. He received his B.E. in mechanical engineering, and M.E. and Dr. Eng. in information engineering from the University of Tokyo in 1986, 1988, and 1991, respectively. His research interests includes vision-based HCI, information visualization, and network security.