Latent Space Gaussian Process Gaze-Tracking

Nicolai Wojke, Jens Hedrich and Detlev Droege[1]

Abstract— **Commercial gaze-tracking devices provide accurate measurements of the visual gaze and are applied to a broad range of problems in marketing, human-computer interaction, and health care technology. In some applications commercial systems are either unavailable or unaffordable. Therefore, developing low cost solutions using *off the shelf* components is worthwhile. In the paper at hand, we apply a hierarchy of Gaussian processes, a class of probabilistic function regressors, to the problem of visual gaze-tracking for consumer grade cameras. Gaussian process latent variable models lead to a lower dimensional manifold which represents the gaze space. Finally, a Gaussian process mapping from screen coordinates to gaze manifold enables us to seek for the users visual gaze point given a previously unseen eye-patch. In our experiments, we achieve mean errors of approximately 2 cm for a consumer grade webcam that is positioned 30–40 cm in front of the user.**

I. INTRODUCTION

The quality and accuracy of common eye and gaze-tracking devices provides a very solid base for gaze based interaction systems. However, they are usually not precise enough to select tiny screen elements of common graphical user interfaces, leading to the development of specific user interfaces for gaze interaction. Such interfaces are designed to be used with a much lower need for accuracy, not only due to the technical limitations, but also to account for the users capabilities [8]. Gaze interaction systems are often used by disabled users, who, depending on the type and severeness of their handicaps, might also not be able to position their gaze as exact as would be required by conventional user interfaces.

Depending on the service capability of the health care system in different countries, affected persons might often not be funded to obtain a gaze-tracking system. Therefore, several research groups work on systems using inexpensive *off the shelf* devices to compile simple gaze-tracking systems like e.g. [16]. Such systems will perform at a significantly lower accuracy than the established systems, but still good enough for being used with gaze interaction systems. While the goal of using cheap (US$ 20-30) *web cams* could not yet be met, system costs of less than US$ 300 are currently realistic.

A number of algorithms has been published to determine the pupil center in eye tracking systems like [3], [7], [9], and others. These algorithms were shown to work good on medium to high resolution images, for low resolution input however they leave room for improvement. Given the position of the pupil center in an image, the visual gaze can be computed from the known geometry of camera setup. These computations are prone to inaccuracies in pupil center detection. We therefore circumvent pupil center detection and geometric computations by learning (1) a low dimensional latent manifold of eye-patches, and (2) the transformation from screen coordinates to latent space using Gaussian process regression. Given such a mapping, we seek for the unknown visual gaze by maximizing the normalized cross correlation between expected and observed eye patches.

The remainder of the paper is structured as follows: In section II we give an overview of related work in the field of gaze-tracking. In section III we then introduce Gaussian processes as the main framework for our gaze-tracking system. In section IV we describe the details of our methodology and section V describes the experiments we used to evaluate our approach. In section VI we summarize the paper and conclude from our results.

II. RELATED WORK

Numerous approaches to gaze tracking from video images use the pupil center as primary feature. The visual gaze is then recovered using a geometric model. A thorough overview of methods for pupil center detection is given by Hansen and Ji [4]. Most of these algorithms were developed with specific setups and conditions in mind, as the usage scenarios for eye detection and tracking are quite different.

In the context of iris recognition, close-up high resolution images of the subjects eye are self-evident. Appropriate oculars, specific illumination and appropriate cameras deliver almost perfect images of the iris, however accept a distraction of the eye using extra illumination. Such images were the basis for Daugmans algorithm described in [2]. He defines the integrodifferential operator, based on a circular integral around the currently estimated pupil/iris center. Increasing the radius for this measure gives notable changes at the pupil and iris border, providing new estimates for the radii. These can then be used to find new center estimates.

In [3] two methods to determine the center of a pupil are presented. In both cases the first step is to find the transition from dark luminance values in the pupil to brighter values in the iris. This transition is considered to be monotonic when approximated by a polynomial to determine the pupil rim with sub-pixel accuracy. The *coordinates averaging* approach then forms horizontal and vertical scan lines between corresponding rim points. The middle of such lines should lie on the vertical resp. horizontal center axis of the pupil. The pupil center is estimated by averaging the horizontal mid points for the x-component and the vertical mid points for the

[1]Active Vision Group, Institute for Computational Visualistics, University of Koblenz-Landau, 56070 Koblenz, Germany {nwojke, jenshedrich, droege}@uni-koblenz.de

y-component. For images without artifacts from e.g. glints in the pupil this approach gives rather accurate results. For their *circle approximation* method the rim points are used to estimate a circle. Obvious outliers to the circle fitting are weighted down to find a good approximation after a few iterations.

An approach named *Starburst* is presented in [7]. After preprocessing the image, which includes the removal of glints, an initial rough guess for the pupil center is made. From here a number of radial rays is followed and observed for a significant jump in intensity, determined by a difference threshold. These pupil rim points are the origin for a number of secondary rays which are sent like a fan in the opposite direction in a range of $\pm 50^\circ$. These rays provide additional points on the rim.

Peréz et al. [11] describe a similar technique, starting from an initial pupil center guess calculated from the center of gravity of those pixels which are considered pupil pixels by using a threshold. However, only primary rays are used for the pupil rim detection, which is done by employing a Laplace filter. If the detected points are not equidistant from the estimated center, a new center is chosen by using the mid points of the diagonals (where only diagonals of reasonable length are considered). This calculation is repeated until the points are equidistant or some iteration limit is reached.

The algorithm described in [9] is used in the FreeGaze system and called *double ellipse fitting*. A first guess for pupils is done by segmenting the image and looking for dark, round regions. From their center circular rays are followed and rim points are detected by a sudden raise of intensity, similar to Starburst. An ellipse is fitted to these points and its center is used as starting point for a second run. Here, the number of rays is doubled and points having significantly different distance from the center than can be expected are discarded. The remaining points are used for a second ellipse fitting.

This is not a comprehensive list of published algorithms on pupil center detection, several other approaches do exist, e.g. [12], [14], [10], but throughout follow similar principles.

Other methods involve appearance based approaches where machine learning techniques are applied to recover the visual gaze based on a set of image features. Zhang et al. [19] extract a set of color and gradient features from eye-patches taken from a head-mounted camera and learn the mapping to screen coordinates using a 2-layer regression neural network. Similarily, [18] learn a neural network based on the relative position of the pupil, cornea, and an equalized histogram of the image patch surrounding the eye. Williams et al. [17] learn a relevance vector machine on preprocessed eye-patches to obtain probabilstic estimates of visual gaze. The so obtained visual gaze is filtered over time using a linear Kalman filter with stationary motion model. Prisacariu and Reid [13] learn a shared space Gaussian process latent variable model, a variant of non-linear factor analysis, on segmented binary images of the eye. Preprocessing and feature extraction in all of these methods are determined by an expert in the field.

III. GAUSSIAN PROCESSES

Since most of our work is based on Gaussian processes (GPs), we first introduce GPs following the function-space view described in [15]. Given data $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ consisting of inputs $\boldsymbol{x}_i$ and observations y_i, drawn from a noisy process

$$y_i = f_i + \epsilon_i = f(\boldsymbol{x}_i) + \epsilon_i$$

with noise $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, a Gaussian process estimates the posterior distribution of function f. Therefore, Gaussian processes are distributions over functions. Formally, in the GP framework the stochastic properties of function f are characterized by a mean and a covariance function

$$m(\boldsymbol{x}) = \mathrm{E}[f(\boldsymbol{x})] \qquad k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{Cov}[f(\boldsymbol{x}_i, \boldsymbol{x}_j)]$$

such that for any given subset of deterministic inputs $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ random variables $(f_1, \ldots, f_N)$ have N-dimensional joint normal distribution

$$p(f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N) \,|\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N) = \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}). \qquad (1)$$

In the following we write $f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}_i, \boldsymbol{x}_j))$. Inference for GPs is possible due to the marginalization property of the Gaussian distribution. From Equation 1 follows that for training data $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^\mathrm{T}$, $\boldsymbol{y} = (y_1, \ldots, y_N)^\mathrm{T}$ and previously unseen test point $\boldsymbol{x}_*$ with unknown, noisy function value $f_* = f(\boldsymbol{x}_*)$ the joint distribution is Gaussian. Assuming $m(\boldsymbol{x}) = 0$:

$$p(\boldsymbol{y}, f_*) = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma^2 \boldsymbol{I} & \boldsymbol{k}_* \\ \boldsymbol{k}_*{}^\mathrm{T} & k_* \end{bmatrix}\right),$$

where matrix $\boldsymbol{K}_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ contains the covariance function evaluations and similarily $\boldsymbol{k}_{i,*} = k(\boldsymbol{x}_i, \boldsymbol{x}_*)$ and $k_* = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$. Now, the posterior of unknown function value f_* can be obtained by building the conditional Gaussian distribution:

$$f_* = \boldsymbol{k}^{*\mathrm{T}} (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y},$$
$$\sigma_* = k^* - \boldsymbol{k}_*{}^\mathrm{T} (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_*.$$

There exist many problem dependent choices for the covariance function. In all experiments we used the exponentiated quadratic covariance function

$$k_{\mathrm{EQ}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha^2 \exp\left(-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^\mathrm{T} \boldsymbol{\Lambda}^{-1} (\boldsymbol{x}_i - \boldsymbol{x}_j)\right) \qquad (2)$$

with parameters α, $\boldsymbol{\Lambda}$. The exponentiated quadratic covariance function encodes *smoothness*, i.e. close points are expected to have similar function values. The parameters of the covariance function as well as noise σ^2 can be learned by maximizing the marginal likelihood

$$p(\boldsymbol{y} \,|\, \boldsymbol{X}, \sigma^2, \alpha, \boldsymbol{\Lambda}) = \int p(\boldsymbol{y} \,|\, \boldsymbol{f}, \boldsymbol{X}, \sigma^2) p(\boldsymbol{f} \,|\, \boldsymbol{X}, \alpha, \boldsymbol{\Lambda}) \,\mathrm{d}\boldsymbol{f}$$

where $p(\boldsymbol{f} \,|\, \boldsymbol{X}, \alpha, \boldsymbol{\Lambda}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$ is the Gaussian process prior on function values $\boldsymbol{f} = (f_1, \ldots, f_N)$ and

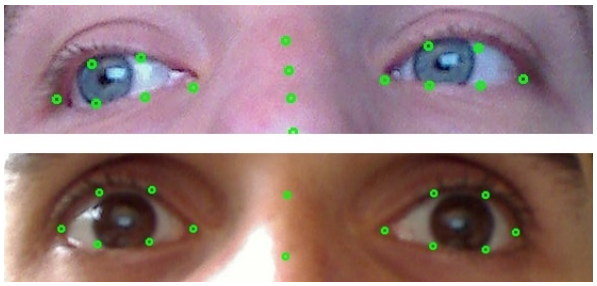

Fig. 1: Example of selected eye-patches. The green circles indicate landmarks given by the face tracking method proposed by Kazemi et al.[5].

$p(\boldsymbol{y} \mid \boldsymbol{f}, \sigma^2) = \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$ is the Gaussian observation likelihood. For completeness we write down the log marginal likelihood that is used for optimization:

$$\log p(\boldsymbol{y} \mid \boldsymbol{X}, \sigma^2, \alpha, \Lambda) = -\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\big(\boldsymbol{K} + \sigma^2 \boldsymbol{I}\big)^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}| - \frac{n}{2}\log 2\pi$$

where n is the number of columns in $\boldsymbol{X}$. A more elaborated introduction to GPs as well as a practical implementation of Gaussian process regression can be found in [15].

IV. LATENT-SPACE GAUSSIAN PROCESS GAZE-TRACKING

In this section we describe the methodology of the proposed gaze-tracker. We show the system architecture and illustrate how we apply Gaussian process latent variable models for learning a low dimensional manifold of eye-patches, as well as how Gaussian process regression is used for learning a mapping from screen coordinates to latent space for determining the visual gaze of the user.

The presented gaze-tracking system expects eye-patches of an interacting user who is sitting in front of a computer screen. In our setup we use a standard webcam[1] to observe the scene. In order to select the eye-patches we use the head tracking method proposed by Kazemi et al. [5]. This method uses a regression tree to detect 68 distinctive face landmarks. Six landmarks are used for each eye (c.f. Figure 1). In our test setup the landmarks are stable enough to extract stable eye-patches even for small head movements.

A. System Overview

The proposed gaze tracker can be divided into four major steps. First, a small representative set of eye-patches and the corresponding gaze points in screen coordinates are captured during calibration phase. These patches are used to generate a lower dimensional gaze manifold using the Gaussian process latent variable model. In this specific application two latent dimensions were sufficient for capturing the relevant information. In order to identify the users gaze during tracking phase, we select nearest neighbor eye-patches in training data to initialize a non-linear optimization routine where we compare predicted eye-patches to the observed eye-patch. A simplified overview of the proposed method is given in Figure 2.

[1]Logitech Quickcam Pro 9000 Business

Fig. 2: System Overview of the gaze tracker. First, a small sample set of eye-patches and the corresponding gaze points in screen coordinates are captured. Second, the Gaussian process latent variable model [6] is used to compute a lower dimensional gaze manifold from eye-patches. Third, Gaussian process regression is used to establish a mapping from screen coordinates to eye-patches.

B. Eye-Patch Latent Space Representation

From the image region around the eyes valuable information for determining the visual gaze may be obtained. While there exist algorithms to localize the pupil center as a primary feature [3], [7], [9], their accuracy is highly dependent on image resolution. Further, light reflections easily decrease stability of results. We therefore opt to circumvent error prone preprocessing and apply a non-linear dimensionality reduction method instead. The resulting low-dimensional manifold of eye-patches is then used as our sole feature for determining the visual gaze. In this section we describe the dimensionality reduction method that we have applied in our experiments. In section IV-C we show how to use this representation to determine the visual gaze.

Let $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N]^{\mathrm{T}}$ be a set of N observations of dimensionality D written as design matrix. We then may assume that the variations in the data is governed in a low dimensional manifold $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^{\mathrm{T}}$ of dimension Q with $Q << D$. In the gaze-tracking scenario, under stable lighting conditions, appearance variations in eye-patches are governed by only few parameters that describe mainly the position of the pupil, the eyelid, plus some noise. This motivates our application of (non-linear) dimensionality reduction for visual gaze discovery from image data.

Here, we apply the Gaussian process latent variable model (GP-LVM) of Lawrence et al. [6]. From calibration, we are given a set of eye-patches that make up our observations $\boldsymbol{Y}$. For dimensionality reduction, we define D mappings

$$\boldsymbol{y}_j = g_j(\boldsymbol{x}_i) + \epsilon_j$$

from (unknown) latent sample $\boldsymbol{x}$ to the j-th element of observed eye-patch $\boldsymbol{y}$. If we assume independent Gaussian noise $\epsilon_j \sim \mathcal{N}(0, \sigma^2)$, the conditional of observed eye-patch $\boldsymbol{y}$ given latent sample $\boldsymbol{x}$ is

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{j=1}^{D} \mathcal{N}(\boldsymbol{y}_j \mid g(\boldsymbol{x}), \sigma^2)$$

and, assuming independence among observations $\boldsymbol{Y}$, the distribution of all eye-patches is

$$p(\boldsymbol{Y} \mid \boldsymbol{X}) = \prod_{i=1}^{N} \prod_{j=1}^{D} \mathcal{N}(\boldsymbol{Y}_{i,j} \mid g(\boldsymbol{X}_i), \sigma^2)$$

where $\boldsymbol{Y}_{i,j}$ is the j-th element of the i-th observation and $\boldsymbol{X}_i$ is the i-th latent sample. In standard principal component analysis, mapping $g_j(\cdot)$ is linear and parametrized by the j-th basis vector that is found using eigen value decomposition. In GP-LVM the mapping is not modelled in parametric form, but a common Gaussian process prior is put on all mappings $g_1, \ldots, g_D$:

$$p(\boldsymbol{Y} \mid \boldsymbol{X}) = \prod_{i=1}^{N} \prod_{j=1}^{D} \mathcal{N}(\boldsymbol{Y}_{i,j} \mid \boldsymbol{0}, \boldsymbol{K} + \sigma^2 \boldsymbol{I})$$

where $\boldsymbol{K}$ is the matrix of covariance function evaluations. Parameter learning for the covariance function in GP-LVM is equivalent to the regression case in Section III. The positions of latent samples $\mathbf{X}$ is found along with covariance function parameters by maximizing the marginal likelihood.

In our experiments we have used a two dimensional latent space and the exponentiated quadratic covariance function (2). Fig. 3 shows a visualization of the learned model as well as eye-patches that are generated at different locations. Note that eye-patches corresponding to distant gaze points are placed apart in latent space. Further, the model successfully interpolates between locations where no data has been observed. Interestingly, not only the position of the pupil, but also the opening of the eyelids contains important information for estimating the visual gaze. By generating observations from the model, one can see that the eyelid is generally much more closed when the user looks at the bottom of the screen. This information is well captured by our model.

C. Gaussian Process Gaze-Mapping

For determining the visual gaze we learn a mapping from screen coordinates to latent gaze space. For this, we use the set of gaze points $\{\boldsymbol{s}_i = (x, y)^{\mathrm{T}}\}_{i=1}^{N}$ that have been recorded during calibration phase and positions in latent gaze space $\{\boldsymbol{x}_i = (u_i, v_i)^{\mathrm{T}}\}_{i=1}^{N}$ of the corresponding eye-patches. We model the mapping with two independent GPs, one for each dimension of the latent space:

$$u = f_u(\boldsymbol{s}), \quad f_u \sim \mathcal{GP}(m(\boldsymbol{s}), k_{\mathrm{EQ}}(\boldsymbol{s}_i, \boldsymbol{s}_j)),$$
$$v = f_v(\boldsymbol{s}), \quad f_v \sim \mathcal{GP}(m(\boldsymbol{s}), k_{\mathrm{EQ}}(\boldsymbol{s}_i, \boldsymbol{s}_j)).$$

For both GPs we assume zero mean $m(\boldsymbol{s}) = 0$ and use the exponentiated quadratic covariance function (2). We then apply standard Gaussian process regression (GPR) to obtain covariance parameters by maximizing the marginal likelihood. Together with mapping

$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x})$$

from latent space to eye-patches described in section IV-B, we obtain a hierarchy of Gaussian processes

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{s}) = \boldsymbol{g}(f_u(\boldsymbol{s}), f_v(\boldsymbol{s}))$$

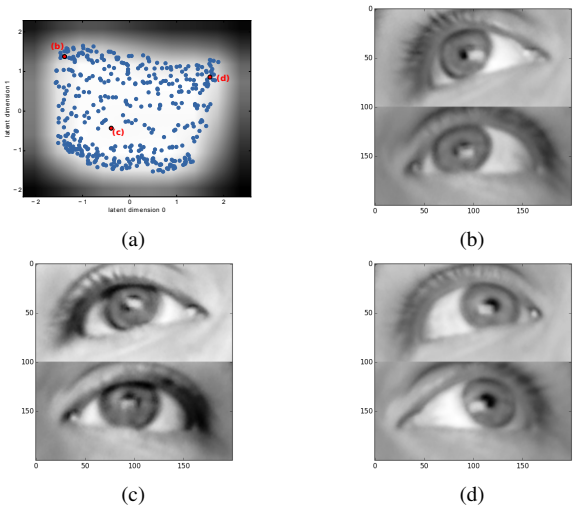

Fig. 3: (a) Visualization of the two dimensional latent gaze space. Training samples are shown as blue dots, uncertainty is gray-shaded (low uncertainty is shown in white, high uncertainty in black). Images (b)–(d) show eye-patches generated from the model at the annotated positions.

which outputs an *expected*[2] eye-patch for each location in screen coordinates. Further, we can compare the predicted eye-patch $\boldsymbol{y}_*$ of our model with any currently observed eye-patch $\boldsymbol{y}$ using normalized cross-correlation:

$$d(\boldsymbol{y}, \boldsymbol{y}_*) = \frac{\sum_i^N (\boldsymbol{y}_i \boldsymbol{y}_{*,i})}{\sqrt{\sum_i (\boldsymbol{y}_i)^2 \sum_i (\boldsymbol{y}_{*,i})^2}}. \tag{3}$$

So far we have explained how the mapping between screen coordinates and eye-patches is established. Further, we have defined normalized cross-correlation as a distance measure for computing similarity between prediceted eye-patches and the currently observed eye-patch. In order to find the visual gaze in screen coordinates during tracking, we solve a non-linear optimization problem in the following way:

1) Find M nearest neighbors of newly acquired eye-patch $\boldsymbol{y}$ in training data that was collected during calibration. In our experiments we have used $M = 3$.
2) Initialize an estimate of the current gaze point in screen coordinates $\boldsymbol{s}_*$ as mean of the screen coordinates of the M nearest neighbors in the training data.
3) Iteratively maximize the normalized cross-correlation (3) between predicted eye-patch $\boldsymbol{y}_* = \boldsymbol{h}(\boldsymbol{s}_*)$ and observed eye-patch $\boldsymbol{y}$ until convergence or a maximum number of iterations has been reached.

In our experiments we have used the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method for solving this optimization problem. Gradients have been computed numerically.

[2]Note that we do not propagate uncertainty from the GPR mapping through the GP-LVM. Therefore, the resulting eye-patch is not an expectation in the maximum likelihood sense.

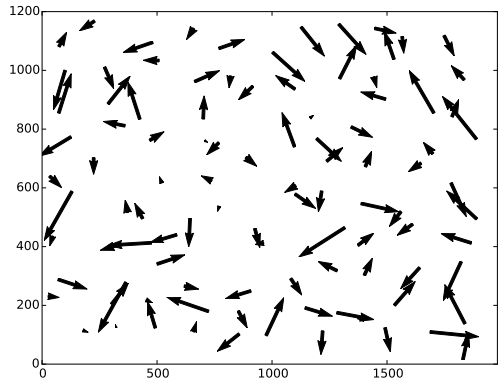

Fig. 4: Displacement field between ground truth and predicted position (in screen coordinates).

	mean	std	max
ϵ_x	12.865 mm	8.881 mm	38.903 mm
ϵ_y	13.405 mm	9.405 mm	39.930 mm
ϵ_{diag}	20.169 mm	10.137 mm	45.494 mm

TABLE I: Mean, standard deviation and maximal error along the horizontal axis ϵ_x, vertical axis ϵ_y and diagonal axis ϵ_{diag}.

V. EXPERIMENTS

The setup of our experiments was based on a consumer grade webcam which was positioned between the computer screen and the user. The distance from the camera to the user was approximate 30–40 cm. The user faced a 24 inch computer screen with a resolution of 1980×1200 pixels. The total length of the visible horizontal axis was 51.84 cm and the total length of the visible vertical axis was 32.40 cm. Further, the distance between the user and screen was approximate 60 cm. During data acquisition, the user focused specific points on the computer screen that were highlighted by our evaluation software. The screen positions were saved together with the corresponding eye-patches taken from webcam images. We then partitioned the data into 400 samples used for training and 100 samples for evaluation.

The learned latent-space is shown in Fig. 3. Note that eye-patches corresponding to distant gaze points are placed apart in latent space. Further, the model successfully interpolates between locations where no data was observed. Interestingly, not only the position of the pupil, but also the opening of the eyelids contains important information for estimating the visual gaze. By generating observations from the model, one can see that the eyelid is generally much more closed when the user looks at the bottom of the screen. This information is well captured by our model.

Table 1 summarizes the results of our experiment. The mean error along the horizontal ϵ_x and the vertical axis ϵ_x are similar. In total we achieve errors between 12 and 45 mm. Fig. 4 depicts the displacement field between ground truth and predicted gaze points. While the error vectors in the lower right corner of the screen appear slightly longer than average, it can be seen that the model does not favor any specific region of the screen.

Fig. 5: (a), (c): Eye-patches of a previously unseen user. (b), (d): Predicted eye-patch of our model at the ground truth screen location.

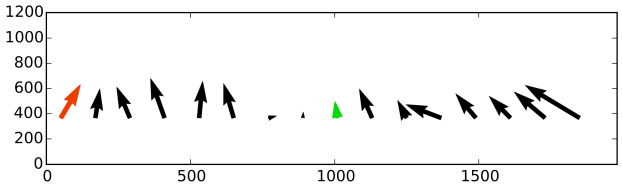

Fig. 6: Displacement field for a previously unseen user. The red arrow corresponds to the eye-patch shown in Fig. 5a, the green arrow corresponds to the eye-patch in Fig. 5c.

In an additional experiment we have tested how the trained model reacts to eye-patches of a previously unseen person with different appearance, i.e. different eye color, eye shape, and illumination when no recalibration is performed. Two eye-patches of this second experiment are shown in Fig. 5 together with the predicted eye-patches of our model at the corresponding ground truth screen locations. Note that the input images are notably darker and that eye sizes differ. This is because the second dataset was taken at a different time of day with slightly changed camera setup. As can be seen from the displacement field in Fig. 6, the model consistently predicts positions lower than ground truth. The average error was 36.590 mm along the horizontal and 55.270 mm along the vertical axis. It is not surprising that the model cannot be directly applied to estimate the visual gaze as the camera setup and therefore the mapping to screen coordinates has been altered. However, predictions are still within 6 cm of ground truth. Therefore, keeping the learned latent gaze space and only updating the mapping from screen coordinates may be possible when a new user is presented to the system. This requires further evaluation and is an open question for future work.

VI. CONCLUSION

We have proposed a hierarchy of Gaussian processes for gaze-tracking. Using a head tracking method we extract stable eye-patches of the user from a consumer grade webcam. We then apply a GP-LVM to learn a two dimensional feature space that contains the relevant information for gaze-tracking. Using standard Gaussian process regression we establish a mapping between screen coordinates and the learned latent space of the GP-LVM to generate eye-

patches for each gaze point hypothesis in screen coordinates. During tracking, we use non-linear optimization to find the visual gaze point by comparing predicted eye-patches of our model with the observed eye-patches using normalized cross-correlation. The system was evaluated on a dataset of 500 samples out of which 100 were used solely for testing where we achieved errors between 10 and 45 mm. While this is not as accurate as professional gaze-tracking hardware, the method provides reasonable results for consumer grade hardware.

There is ample opportunity for future work. First, the proposed hierarchy of Gaussian processes could be jointly optimized in a variational framework that accounts for uncertainty propagation to obtain full posterior distributions of eye-patches [1]. Using this posterior, one can seek the maximum a posteriori estimate of the gaze point instead of using normalized cross correlation as distance measure. Second, our non-parametric machine-learning approach to gaze-tracking allows for easy intergration of further data that has been disregarded in our experiments. For example, locations of face landmarks could be used for head pose estimation which then could be integrated into the tracker to make the system more robust against head movements and perspective distortion.

REFERENCES

[1] Andreas C. Damianou and Neil D. Lawrence. Deep gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 - May 1, 2013*, volume 31 of *JMLR Proceedings*, pages 207–215. JMLR.org, 2013.

[2] John Daugman. How iris recognition works. *Circuits and Systems for Video Technology, IEEE Transactions*, 14(1):21–30, 2004.

[3] Gintautas Daunys and Nerijus Ramanauskas. The accuracy of eye tracking using image processing. In *NordiCHI '04*, pages 377–380, New York, NY, USA, 2004. ACM.

[4] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009. (in print).

[5] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1867–1874, June 2014.

[6] Neil D. Lawrence, Matthias Seeger, and Ralf Herbrich. Fast sparse gaussian process methods: The informative vector machine. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15 [Neural Information Processing Systems, NIPS 2002, December 9-14, 2002, Vancouver, British Columbia, Canada]*, pages 609–616. MIT Press, 2002.

[7] Dongheng Li, David Winfield, and Derrick J. Parkhurst. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. In *CVPR '05*, pages 79–86, Washington, 2005. IEEE.

[8] Päivi Majaranta and Kari-Jouko Räihä. Twenty years of eye typing. In *Proceedings of Eye Tracking Research and Applications*, pages 15–22, New York, 2002. ACM.

[9] Takehiko Ohno, Naoki Mukawa, and Atsushi Yoshikawa. Freegaze: A gaze tracking system for everyday gaze interaction. In *Proceedings of the symposium on ETRA 2002: eye tracking research and applications symposium*, pages 125–132, 2002.

[10] Kun Peng, Liming Chen, Su Ruan, and Georgy Kukharev. A robust algorithm for eye detection on gray intensity face without spectacles. *Journal of computer Science and Technology*, 5(3):127–132, 2005.

[11] A. Pérez, M.L. Córdoba, A. Garcia, R. Méndez, M.L. Munoz, J.L. Pedraza, and F. Sánchez. A precise eye-gaze detection and tracking system. In *WSCG POSTERS proceedings, February 3-7, 2003*, 2003.

[12] Ahmad Poursaberi and Babak Araabi. A novel iris recognition system using morphological edge detector and wavelet phase features. *ICGST International Journal on Graphics, Vision and Image Processing*, 05, 2005.

[13] Victor Adrian Prisacariu and Ian Reid. Shared shape spaces. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc J. Van Gool, editors, *ICCV*, pages 2587–2594. IEEE, 2011.

[14] H. Proenca and L.A. Alexandre. Iris segmentation methodology for non-cooperative recognition. *IEE Proceedings-Vision Image and Signal Processing*, 153(2):199–205, 2006.

[15] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[16] Javier San Agustin, Henrik Skovsgaard, John Paulin Hansen, and Dan Witzner Hansen. Low-cost gaze interaction: ready to deliver the promises. In *CHI EA '09: 27th intl. conf. on Human factors in computing systems*, pages 4453–4458, New York, USA, 2009. ACM.

[17] Oliver Williams, Andrew Blake, and Roberto Cipolla. Sparse and semi-supervised visual mapping with the s^3p. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pages 230–237. IEEE Computer Society, 2006.

[18] Li-Qun Xu, Dave Machin, and Phil Sheppard. A novel approach to real-time non-intrusive gaze finding. In *Proceedings of the British Machine Vision Conference 1998, BMVC 1998, Southampton, UK, 1998*, pages 1–10, 1998.

[19] Yanxia Zhang, Andreas Bulling, and Hans Gellersen. Towards pervasive eye tracking using low-level image features. In *ETRA*, pages 261–264, 2012.