An Artificial Neural Network for High Precision Eye Movement Tracking

Marc Pomplun[a], Helge Ritter[a], Boris Velichkovsky[a,b]

[a] Department of Neuroinformatics, University of Bielefeld, Germany
[b] Unit of Applied Cognitive Research, Dresden University of Technology, Germany
email: impomplu@techfak.uni-bielefeld.de

Abstract. Research of visual cognition often suffers from very inexact methods of eye movement recording. A so-called *eye tracker*, fastened to the test person's head, yields information about pupil position and facing direction related to a computer monitor in front of the subject. It is now a software task to calculate the coordinates of the screen point the person is looking at. Conventional algorithms are not able to realize the required non-linear projection very precisely. Especially if the test person is wearing spectacles, the deviation may exceed 3 degrees of visual angle. In this paper a new approach is presented, solving the problem with a *parametrized self-organizing map* (PSOM). After a short calibration it reduces the average error to approximately 30 percent of its initial value. Due to its high efficiency (less than 150 μs per computation on a PC with a 486DX2-66 processor) it is perfectly suited for real-time application.

1 Introduction

High precision recording of eye movements in real time is of considerable importance in many fields, ranging from research in visual cognition, over commercial studies, to the creation of future, more powerful man-machine interfaces that use eye movement information to control the interaction with a computer or with virtual reality.

A sufficiently flexible approach for achieving this goal is to rely on the evaluation of camera images of the human eye, together with additional sensors from which the head position can be inferred. A recent system of that kind has been developed by Stampe and Reingold at the University of Toronto [7] and is in use in our lab for visual cognition research.

In this system two tiny cameras, fastened to a head set, monitor the eye of the test person and four reference marks in his or her view field (see figure 1). Suitable image processing software then extracts the required coordinate information from the two camera images (the first providing the gaze direction in head centered coordinates, the second yielding the orientation of the head relative to the scene). While this approach yields satisfactory results for normal-sighted persons, the accuracy decreases with gaze eccentricity and drops significantly when subjects wear spectacles.

This motivated the research reported in the present paper, namely the use of adaptive neural networks that can be trained to compensate (at least part of) the (systematic) measurement errors made by the current system. Since the characteristics of these errors vary with the test person, it is essential to use an adaptive network that can be retrained for each different test person. Furthermore, practical considerations dictate that such retraining be fast and possible on the basis of a small number of training examples (for which the test person has to fixate known positions on a computer screen).

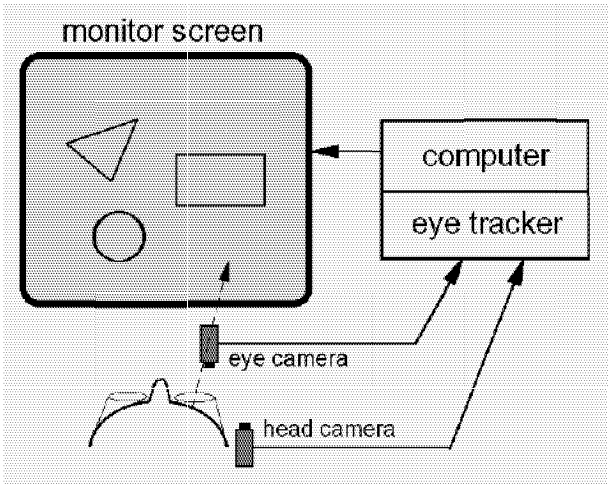

Figure 1: Scheme of the system developed by Stampe and Reingold

In the present paper we report an approach that solves this problem with a *parametrized self-organizing map* ("PSOM") [4, 5, 6], a recently proposed variant of Kohonen's self-organizing map [2, 3]. An attractive property of the PSOM is that it can be trained with an extremely small number of training examples (this does, in fact, lead to restrictions on the representable mappings, but they are of no consequence in the present context). We have implemented and tested this approach on a PC with a 486DX2-66 processor and applied it to correcting data subsequently to an experiment. However, the computation of a single position takes less than 150 μs, which is negligible considering the system's temporal resolution. The method can therefore be applied in real time as well. We found that by this approach the average error of the system can be reduced to approximately 30 percent of its previous value. This improvement is achieved with only 16 training examples, gathered during a short calibration session where the subject has to fixate known points of a 4×4 grid. As a result, we can now track eye movements for subjects wearing spectacles at a higher precision than the system could achieve for normal sighted subjects before. For normal sighted subjects very high precision eye tracking (error $< 0.4^o$) has become possible.

In the next section we first describe the PSOM-network in the context of the present application (for a more general description, see e.g. [5, 6]). In Sec.3 we describe a method to speed up the basic approach, Sec.4 concludes with results from experimental measurements and a discussion.

2 How to Construct a PSOM

The basic computational task to be solved by the network is to transform the output data (a two-dimensional vector specifying the gaze direction of the subject) in such a way that the systematic measurement errors become compensated. Since the errors vary with gaze direction, the required mapping $\mathbf{f}$ must be nonlinear, and it must be rapidly constructable for a new subject.

For the representation of nonlinear mappings, in particular to two-dimensional spaces,

the Kohonen *self-organizing map* (SOM) has during recent years turned out to be a very favorable method [3]. However, the basic approach is restricted to a discrete lattice, and, while showing very favorable convergence properties in many situations, would still require more training examples than would be convenient for the present purpose.

Exactly these two aspects, continuity and the fact that only a very small number of training examples are required, are among the favorable properties of PSOMs. In a PSOM, the "localist" representation of the SOM is built from a set of *basis* or *prototype manifolds*. The contribution of each basis manifold to the map is controlled by a weight vector that can be viewed as a member of a very coarse SOM with only very few nodes. As a result, smooth mappings can be constructed from very few data samples.

However, as a price to pay, the usual "bestmatch-step" of the SOM has to be replaced by an iterative scheme for the continuous manifold (see eq. (6) below). In neural terms, this iterative scheme can be interpreted as a recurrent dynamics that assigns a point on the manifold to each input vector as the associated output. Therefore, a PSOM can be viewed as a recurrent network which realizes a continuous attractor manifold that represents the graph of the desired non-linear mapping (for additional details, see [5, 6]).

In the present context, this mapping $\mathbf{f}$ shall be constructed on the basis of a set of 4×4 input-output pairs $\mathbf{w_r} \in I\!R^4$. Each reference vector is associated with a fixation point $\mathbf{r} \in \mathbf{A}$, where $\mathbf{A} = \{\mathbf{r}_{ij} \mid \mathbf{r}_{ij} = i\hat{\mathbf{e}}_x + j\hat{\mathbf{e}}_y;\ \ i,j = 0\ldots3\}$ is a two-dimensional 4×4 grid of 16 fixation points shown to the subject during the calibration phase. The 4×4 set of reference vectors $\mathbf{w_r}$ could be regarded as a (very coarse) SOM which would represent the desired transformation, however, at a very low resolution that would be unacceptable for the present purpose. The PSOM now extends this SOM by providing a smooth, parametrized manifold that passes through the points $\mathbf{w_r}$ of the coarse SOM.

The PSOM represents the desired interpolating function $\mathbf{f(s)}$ that is a superposition of a suitable number of simpler basis functions $H(.,.)$ in the following way:

$$\mathbf{f(s)} = \sum_{\mathbf{r}\in\mathbf{A}} H(\mathbf{s},\mathbf{r})\mathbf{w_r} \quad , \tag{1}$$

Here the basis functions H: $I\!R^2 \times \mathbf{A} \rightarrow I\!R$, have to comply with the requirement

$$H(\mathbf{s},\mathbf{r}) = \delta_{\mathbf{s},\mathbf{r}} \quad \forall \quad \mathbf{s},\mathbf{r} \in \mathbf{A} \quad , \tag{2}$$

where δ represents the Kronecker symbol. This ensures that $\mathbf{f(s)} = \mathbf{w_s}\ \forall\ \mathbf{s} \in \mathbf{A}$, i.e., the constructed function passes through the given points. The final question that remains is how can we choose suitable functions H, that obey (2) and that are smooth and simple to handle? One convenient choice is first to make a product ansatz

$$H(s_x\hat{\mathbf{e}}_x + s_y\hat{\mathbf{e}}_y,\ \ \mathbf{r}_{ij}) = H^{(1)}(s_x,i)\cdot H^{(1)}(s_y,j) \tag{3}$$

The new, simpler function $H^{(1)}$: $I\!R \times \{0,1,2,3\} \rightarrow I\!R$ must then have the property

$$H^{(1)}(q,n) = \delta_{q,n} \quad \forall \quad q \in I\!R,\ \ n \in \{0,1,2,3\} \tag{4}$$

to constitute a valid function H. Due to the discrete parameter n it is possible to use a set of 4 basis functions $I\!R \rightarrow I\!R$, and a convenient choice are cubic polynomials (see figure 2). (We also tried trigonometric functions, however the cubic polynomials gave slightly better results.)

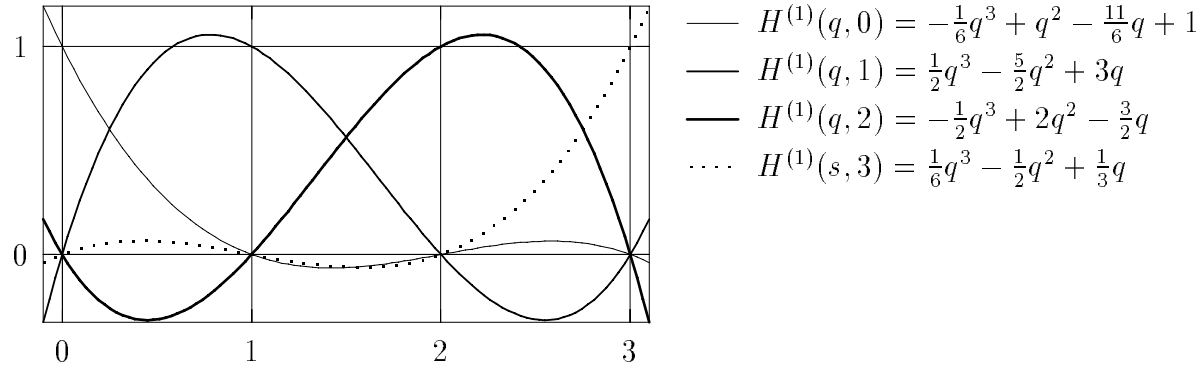

Figure 2: The four basis functions

Now we have built a function $\mathbf{f}$, but it is not what we finally desire. $\mathbf{f}$ is a projection from the "correct" orthogonal lattice to an "error lattice" distorted by the system's inaccuracy. Our intention is, however, to deduct the correct coordinates from the distorted ones! Therefore, we must calculate the inverse function $\mathbf{f}^{-1}$, and the non-linearity of $\mathbf{f}$ forces us to apply a numerical procedure. To invert $\mathbf{f}(\mathbf{s})$, we minimize the error function $E(\mathbf{s})$, given by the equation

$$E(\mathbf{s}) = \frac{1}{2}(\mathbf{f}(\mathbf{s}) - \mathbf{f}_{et})^2 \quad , \tag{5}$$

where $\mathbf{f}_{et}$ is the output vector of the eye tracker with respect to $\mathbf{s}$. Starting with an estimate value $\mathbf{s}_0 = \mathbf{s}(t=0)$, we use a gradient descent in the variables $\mathbf{s}$:

$$\mathbf{s}(t+1) = \mathbf{s}(t) - \epsilon \cdot \frac{\partial E(\mathbf{s})}{\partial \mathbf{s}} \tag{6}$$

with the positive step size parameter ϵ. Equation (6) has to be iterated until $E(\mathbf{s}(t))$ falls below a prespecified threshold value, which we should set according to the screen solution. In neural terms, the iteration of (6) can be viewed as a recurrent network dynamics driving the "state vectors" towards an optimum value that is constrained to lie on the (here) two-dimensional PSOM-manifold embedded in the four-dimensional "feature space" of input-output pairs. The resulting final value $\mathbf{s}(t)$ represents the output vector $\mathbf{s}_{out}$ of the PSOM. As a result of the polynomial structure of $\mathbf{f}$, the implementation of this iteration is not problematic at all. Figure 3 demonstrates the very natural and topology-conserving interpolation capabilities of our just completed PSOM by illustrating the distortion effects of $\mathbf{f}$ and its inverse $\mathbf{f}^{-1}$ for a typical situation.

3 Speeding up the Neural Net

The calculation speed of the PSOM mainly depends on the efficiency of its iteration procedure. For that reason, the parameters $\mathbf{s}_0$ and ϵ need a careful investigation. A good choice for $\mathbf{s}_0$ seems to be the input vector $\mathbf{f}_{et}$, because it will be in the range of the result $\mathbf{s}_{out}$, unless we use a strange distortion like mirroring the whole vector plane. The average amount of iterations needed by this straight forward method is shown in figure 4, yielding best results at $\epsilon \approx 0.79$. At this point it must be explained that all measured values in this section vary slightly with the underlying function $\mathbf{f}$ and the test person's eye movement

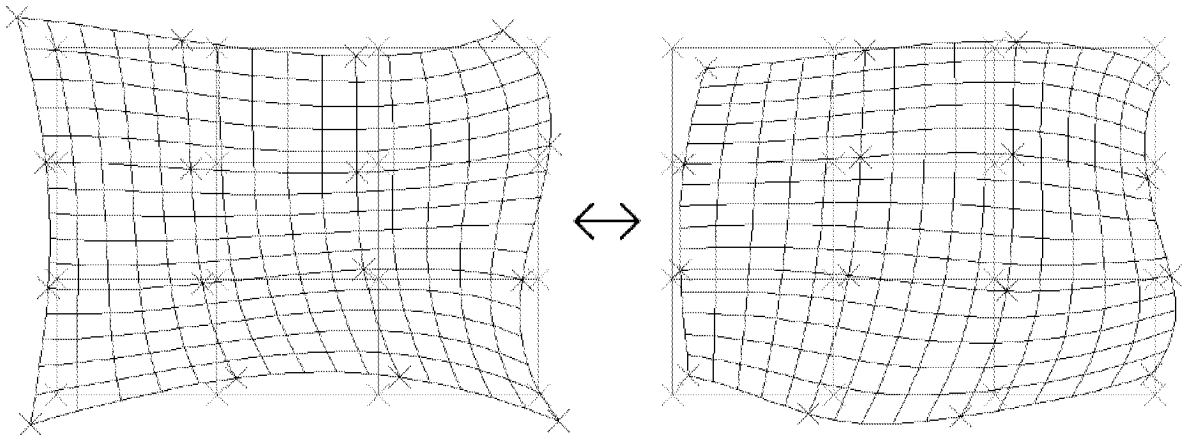

Figure 3: Left: PSOM-Interpolation $\mathbf{f}(\mathbf{s})$ for eye tracker errors, constructed from 16 measurements at the positions of the underlying 4×4 "calibration grid". Right: Inverse mapping $\mathbf{f}^{-1}(\mathbf{s})$ that, when applied to the eye tracker output, will compensate the errors. This mapping is constructed via equation (6).

characteristics. But the differences do not exceed a few percent and have no qualitative effects.

How can we increase the PSOM's calculation speed further? The following idea is based on the fact that we are interested in correcting many coordinates in succession, and not in dealing with single data. Successive fixations are very likely to be located in the same area of the screen. Therefore, the *local deviation* $\mathbf{f}_{et} - \mathbf{s}_{out}$ does not vary essentially between successive PSOM computations. For the first calculation we set $\mathbf{s}_0 := \mathbf{f}_{et}$ as we did before. But this time we store the vectors $\mathbf{f}_{et}$ and $\mathbf{s}_{out}$ as $\mathbf{f}_{et,stored}$ and $\mathbf{s}_{out,stored}$ to get a better estimate value $\mathbf{s}_0$ for the next iteration:

$$\mathbf{s}_0 = \mathbf{s}_{out,stored} + \mathbf{f}_{et} - \mathbf{f}_{et,stored} \tag{7}$$

The improvement of efficiency by using this estimating method is shown in figure 4. Its effect increases with the data frequency, because the shift between successive input vectors becomes smaller on the average. At 60 Hz use, the number n of iteration steps per computation remains under the value 1, because in most cases the estimate $\mathbf{s}_0$ already constitutes a valid output $\mathbf{s}_{out}$! Here we achieve an improvement factor of approximately 10, since the efficiency of the standard method remains invariable at different frequencies.

4 Results and Discussion

We implemented the PSOM on a PC with a 486DX2-66 processor, using a screen solution of 640×480 pixels. The tolerance of $\mathbf{f}^{-1}$ was set to 0.5 pixels to get optimum accuracy. Table 1 shows the average delay caused by a single calculation of the neural net in various applications.

In our experiments the eye tracker system already used a built in conventional algorithm (see [7]), based on a calibration procedure involving 9 lattice points. Our goal was to improve the outputs of this system further by doing a suitable post-processing of the data

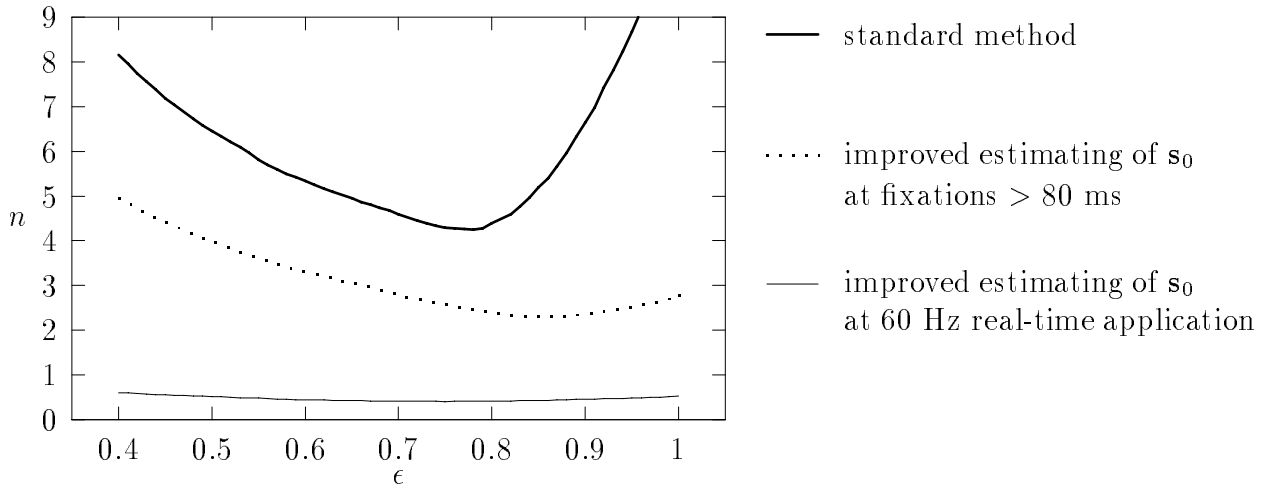

Figure 4: The average number n of performed iterations as a function of the step size parameter ϵ

	single data	fixations > 80 ms	60 Hz use
straight forward method	1480 μs	1480 μs	1480 μs
improved method	1480 μs	725 μs	143 μs

Table 1: Average delay

with the PSOM. For this purpose an additional calibration matrix was presented to get information for the subsequent data processing. After the experiments the neural net was used to correct a list of all fixations that took longer than 80 ms, received as the eye tracking result. But how is it possible to compare the accuracy of the original coordinates and the corrected ones?

For that purpose the test person was shown a further 6×6-lattice, which was entirely different from the calibration lattice. Then we took the standard deviations σ_{orig} and σ_{PSOM} of the original output coordinates and the PSOM-corrected output respectively. One group of test persons had to be investigated separately, namely the people wearing spectacles. Table 2 demonstrates the average improving effect of our neural net for each of these groups.

The whole calibration procedure takes less than 20 seconds, so it is an acceptable effort considering the significant increase of data accuracy. The new method allows us also to carry out investigations on subjects wearing spectacles. For normal sighted subjects, the very high precision eye tracking that has now become possible allows us to envisage a range of new experiments and ways to evaluate data from the existing experiments that would not have been meaningful before.

Indeed, this is not the first approach of this kind. Baluja and Pomerleau [1] have recently reported on an eye tracker system in which the entire processing, i.e. including the evaluation of the camera images, was achieved by a neural network. However, training

	σ_{orig}	α_{orig}	σ_{PSOM}	α_{PSOM}	remaining error
spectacles	65.5 pixels	2.92^o	19.3 pixels	0.82^o	28.1%
no spectacles	25.4 pixels	1.13^o	8.9 pixels	0.39^o	34.5%

Table 2: Effects on the cartesian standard deviation σ and its corresponding visual angle α

of this network required on the order of several thousand training examples, and the final accuracy of the system is about 1.5^o, which compares well to the accuracy of Stampe's system. While the work by Baluja and Pomerleau is a very impressive demonstration of what can be achieved on the basis of neural networks alone, it is difficult to compare with the present study, since the authors solve a much more general estimation problem. The motivation of the present study was somewhat different, namely to improve the accuracy of an existing system, and to achieve this on the basis of a number of training examples that is sufficiently small to be compatible with the daily use of the system in laboratory experiments.

In the future, we intend to integrate the neural component in the eye tracker software, so that real time use becomes possible. On a broader perspective, we view the present work as one demonstration of the many potential uses of neural networks to contribute to the creation of better and more powerful, multi-modal man-machine interfaces that can adapt to their user in real time. This is a challenging area of research, in which the vigorously developing fields of virtual reality, robotics, neural networks and vision research can converge in a potentially very fruitful manner for shaping the computers of tomorrow to the needs of humans and not vice versa, as it is still very much the case today.

Acknowledgements: We would like to thank Dave Stampe and Eyal Reingold for providing the eye tracker system, Thomas Clermont for his help with the eye tracker hardware, and Peter Munsche for carrying out the experiments for collecting the test data. This work was supported by a grant of the German Science Foundation (SFB 360).

References

[1] Baluja, S., Pomerleau, D. (1994), Non-Intrusive Gaze Tracking using Artificial Neural Networks. *Neural Information Processing Systems 6*, Morgan Kaufmann Publishers.

[2] Kohonen, T. (1984), *Self-Organization and Associative Memory*, Springer Series in Information Sciences 8, Springer, Heidelberg.

[3] Kohonen, T. (1990), *The Self-Organizing Map*, in *Proc. IEEE* 78, pp. 1464–1480.

[4] Ritter, H., Martinetz, T., Schulten, K. (1992), *Neural Computation and Self-Organizing Maps*, Addison-Wesley, Reading, MA.

[5] Ritter, H. (1993), Parametrized Self-Organizing Maps. *ICANN93-Proceedings* (S. Gielen and B. Kappen eds.), pp. 568–577, Springer Verlag, Berlin.

[6] Ritter, H. (1994), Parametrized Self-Organizing Maps for Vision Learning Tasks. *ICANN94-Proceedings*, Springer Verlag (to appear).

[7] Stampe, D. (1993), Heuristic filtering and reliable calibration methods for video-based pupil-tracking systems. *Behaviour Research Methods, Instruments, & Computers 1993*, 25 (2), pp. 137–142.