Neural Computing & Applications
Draft. Final version published online Nov. 6, 2004. DOI: 10.1007/s00521-004-0444-x

Face Localization and Tracking in the Neural Abstraction Pyramid

Sven Behnke

Albert-Ludwigs-Universität Freiburg, Computer Science Institute, Georges-Köhler-Allee 52, 79110 Freiburg, Germany,
Phone: +49 761 203 7404, Fax: +49 761 203 8222,
http://www.informatik.uni-freiburg.de/∼behnke, behnke @ informatik.uni-freiburg.de

Received: 20 January 2004 / Accepted: 1 June 2004

Abstract One of the major tasks in some human-computer interface applications, such as face recognition and video telephony, is to localize a human face in an image.

In this paper, we propose to use hierarchical neural networks with local recurrent connectivity to solve this task not only in unambiguous situations, but also in the presence of complex backgrounds, difficult lighting, and noise.

The networks are trained using a database of gray-scale still images and manually determined eye coordinates. They are able to produce reliable and accurate eye coordinates for unknown images by iteratively refining initial solutions. Because the networks process entire images, there is no need for any time-consuming scanning across positions and scales.

Furthermore, the fast network updates allow for real-time face tracking. In this case, the networks are trained using still images that move in random directions. The trained networks are able to accurately track the eye positions in test image sequences.

Key words Face localization and tracking – Hierarchical network – Local connectivity – Recurrence – Iterative refinement – Continuous attractors

1 Introduction

Computers must adapt to the user's needs to make human-computer interaction more pleasant, and for adaptation it is necessary to perceive the user. One important step for many adaptive human-computer interface applications, such as face recognition, lip reading, and video telephony is the localization of the user's face in a captured image or an image sequence.

A large body of literature exists on the topic of face detection and localization. Recent surveys have been compiled, e.g., by Hjelmas and Kee Low [7] and by Yang, Kriegman, and Ahuja [17].

Many existing localization techniques rely on image motion or skin color – features that are not always available. Here, we focus on still gray-scale images.

One of the most popular neural approaches for face detection in still gray-scale images has been proposed by Rowley, Baluja, and Kanade [13]. It uses a window that scans across image positions and scales as input to a neural classifier. Such sequential search techniques are computationally expensive. Viola and Jones [15] recently introduced an intermediate representation, called integral image, to speed-up the computation of overcomplete rectangular features. Using a cascade of classifiers trained by boosting, they cut-down dramatically the computation time needed for the scanning process. Other face localization methods preprocess the data intensively to extract facial features and match them with predefined models [8, 10].

In this paper, we propose a method that uses a hierarchical neural network with local recurrent connectivity to localize a face in gray-scale still images. The network operates by iteratively refining an initial solution. It is trained with supervision to perform the task. The face localization also works well if the static images are moved in a random direction. In this case, the network is trained to track the eye positions.

The paper is organized as follows. The next section describes the database used for the experiments and the preprocessing procedure. Section 3 introduces the network architecture and details its training. We present experimental results with still images in Section 4.1 and with moving images in Section 4.2. The paper concludes with a discussion.

2 Face Database and Preprocessing

In order to validate the proposed approach to face localization and tracking, we used the BioID database [8]. It consists of 1.521 gray-scale images showing 23 individuals in complex office environments (see Fig. 1). The people differ in gender, age, and skin color. Some of them wear glasses or have a beard. The BioID data set is challenging because face size, position and view, as well as the facial expression and lighting vary considerably.

Such real-world conditions are the ones that reveal the limits of current localization techniques. While the hybrid system proposed by Jesorsky, Kirchberg, and Frischholz [8] correctly localized 98.4% of the XM2VTS database [11] (produced under controlled conditions), the same system localized only 91.8% of the BioID faces.

The BioID images have a size of 384×288 pixels. Fig. 2 illustrates the preprocessing we applied to the images before presenting them to the face localization network. We decided to lower the contrast towards the edges of the image to reduce border effects. Furthermore, we subsampled the images to 48×36, 24×18, and 12×9 pixels in order to limit the amount of data the face localization network has to process.

In addition to the images, the BioID data set also contains manually labeled eye positions, indicated in the figure by crosses. We produced a multiresolutional Gaussian blob for each eye that serves as a target output for training the face localization network, as shown in the right part of the figure.

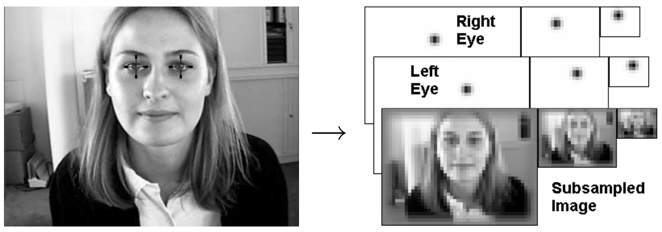

Fig. 2 Preprocessing: Framing and subsampling. Blobs indicate eye positions.

Because the BioID data set does not specify which images should be used for training and testing, we divided it randomly into 1000 training images and 521 test examples. In order to facilitate the learning of translation invariance, we also generated eight image variants for each training example by translating it by ± 3 pixels in different directions before subsampling.

For experiments with moving images, we created another version of the BioID data set by generating three image sequences of length 21 for each original image. To do so, we randomly selected a direction of motion from $(\{-1, 0, +1\}$, $\{-1, 0, +1\})$ and moved the static images in this direction with constant speed such that in the middle of a sequence the original position was obtained. As illustrated in Figure 3, this approximates the output of a moving camera capturing a static scene.

For the training of face tracking networks, it is necessary to move the blobs indicating the target eye positions accordingly.

3 Network Architecture and Training

3.1 Neural Abstraction Pyramid Architecture

The preprocessed images are presented to a Neural Abstraction Pyramid [4,3]. This hierarchical neural architecture has

Fig. 3 Moving input created by translating a still image in a random direction with constant speed.

been motivated by the structure of the human visual system. It represents images at different levels of abstraction using 2D feature arrays.

High-resolution signal-like representations are present at the bottom of the pyramid. The character of the representations changes continuously when going up in the abstraction hierarchy. The resolution of the feature maps decreases while the number and the diversity of the represented features increase. This yields symbol-like high-level features representing large image areas that are less variant to image transformations than the lower-level features.

All these representations are produced by simple processing elements, called feature cells, that interact locally. Because the connections between the feature cells form horizontal and vertical feedback loops, the network constitutes a dynamical system whose activity develops over time. The iterative computation of the network allows for a refinement of an initial solution by incorporating contextual information. While unambiguous stimuli are interpreted quickly, the interpretation of local ambiguities is deferred until further evidence arrives from already interpreted neighboring stimuli. This is an efficient way to resolve ambiguities.

As can be seen in Figure 4, the network variant used here consists of four layers. Each layer contains excitatory and inhibitory feature arrays. Each feature is computed at a 2D-grid of locations by $\sum$-units that share a common weight template.

The resolution of the feature arrays decreases from layer L_0 (48×36) to L_2 (12×9) by a factor of 2 in both dimensions. Layer L_3 has only one cell per feature array. In contrast, the number of feature arrays per layer increases when going from layer L_0 (4+2) to L_2 (16+8). Layer L_3 contains 10 excitatory and 5 inhibitory feature cells.

The network's connectivity is recurrent and local. The weights of all cells of a feature array are described by a common template. Such weight sharing has been used in the Neocognitron [6] and in convolutional neural networks [9] to reduce the number of free parameters in situations where translational invariance is desirable.

Each feature cell receives input from only a small window of cells that correspond to similar locations in the layer below (forward weights), the same layer (lateral weights), and the layer above (backward weights). Weights from excitatory cells are non-negative, weights from inhibitory cells are non-positive, and input weights can have any sign.

The excitatory feature cells of L_0-L_2 receive input from 4×4 windows of the feature arrays one layer below them. They look at 5×5 input cells and at a 3×3 neighborhood at

Fig. 1 Some images from the BioID face localization data set.

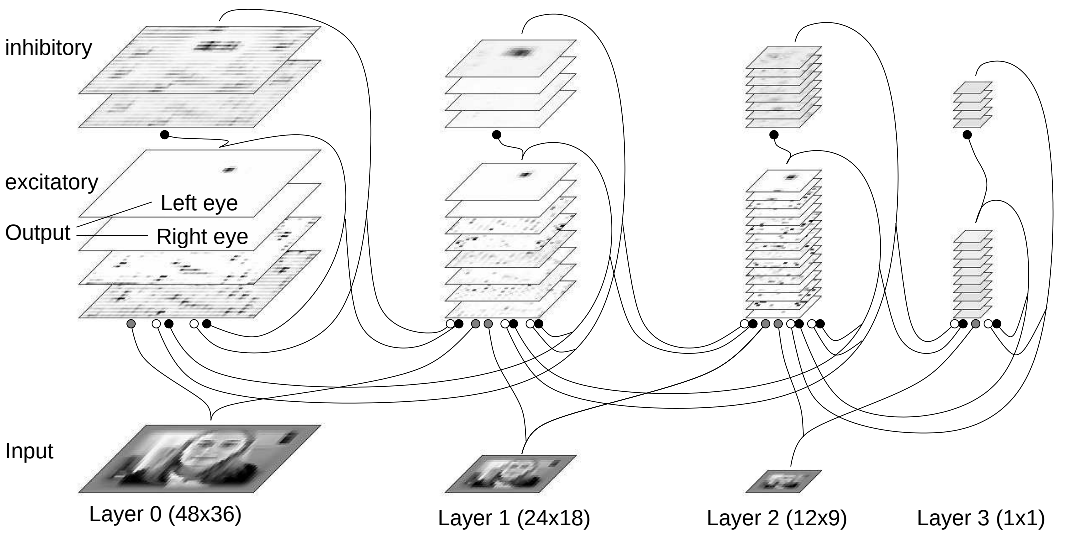

Fig. 4 Sketch of the Neural Abstraction Pyramid network used for face localization.

the same layer. Their backward weights have a window size of 2×2.

Connections between L_2 and the topmost layer L_3 are different, because the resolution drops from 12×9 to 1×1. Both, forward- and backward weights have a 12×9 window size.

The input to the inhibitory feature cells is less complex. They look only at 5×5 windows of all excitatory feature arrays at the same layer. Of course, in layer L_3 the size of this input window reduces to 1×1.

The activities of the feature cells are updated for each iteration of the network. The update step $(t+1)$ of value $v_{x,y,z,q}$ at position (x,y) in layer L_z for feature array q is done as follows:

$$v^{t+1}_{x,y,z,q} = \sigma\left[\sum_{j\in\mathcal{L}(i)} \mathcal{W}(j)\, v^t_{\mathcal{X}(j,x),\mathcal{Y}(j,y),\mathcal{Z}(j,z),\mathcal{Q}(j)} + \mathcal{B}(i)\right].$$

The template $i = \mathcal{T}(z,q)$ is associated with feature array q at layer z. Its set of input links is represented by $\mathcal{L}(i)$, and $\mathcal{B}(i)$ denotes the template bias. The index $(\mathcal{X}(j,x), \mathcal{Y}(j,y), \mathcal{Z}(j,z), \mathcal{Q}(j))$ describes location and feature array of the input for link j. The link weight is denoted by $\mathcal{W}(j)$.

The output function $\sigma(x) = \ln(1+e^{\beta x})/\beta$ used here is a smooth approximation to the rectifying function $\max(0,x)$. It limits the activity of a feature cell to be positive and provides a nonlinearity which is essential to network stability.

For the uniform initialization at $t=0$, the activity of all feature cells associated with template i is set to a start value $\mathcal{V}^0(i)$.

During all time steps, the value of an input node at position (x,y,z,q) is set to a copy of the corresponding component $\mathcal{I}(x,y,z,q)$ of the input vector $\mathbf{x}_k$ of the current example k:

$$v^t_{x,y,z,q} = x^t_{k,\mathcal{I}(x,y,z,q)}, \text{ if } i = \mathcal{T}(z,q) \text{ is an input template.}$$

3.2 Supervised Training

The network described above is trained in a supervised manner to solve the face localization and tracking tasks. Due to the non-linear dynamics of such recurrent neural networks, it is difficult to train them. One possible training method, backpropagation through time (BPTT) [16], unfolds the recurrent network in time and applies backpropagation to compute the gradient of an error function. This can be done efficiently if the number of time steps is limited.

For face localization, we present a static input $\mathbf{x}_k$ to the network and train it to quickly produce the desired output $\mathbf{y}_k$, where $k = 1\ldots K$ is the example index. The network is updated for a fixed number $T = 10$ of iterations. The output error $\delta^t_k = \mathbf{y}_k - \mathbf{v}^t_k$ at time t is defined as the difference between the activity of the output cells $\mathbf{v}^t_k$ and the desired output $\mathbf{y}_k$. It is not only computed at the end of the sequence, but after every update step. The squared differences are weighted progressively in the error function E, as the number of iterations

t increases:

$$E = \sum_{k=1}^{K} \sum_{t=1}^{T} t \, \|\delta_k^t\|^2.$$

For the experiments with moving inputs, the network was updated for $T = 21$ steps. The increasing error weight was saturated at the level reached after 10 iterations to give the following time steps equal importance.

Minimizing the error function with gradient descent faces the problem that the gradient in recurrent networks either vanishes or grows exponentially in time, depending on the magnitude of gains in loops [5]. Hence, it is very difficult to determine a learning constant that allows for both stability and fast convergence.

For that reason, we decided to employ the resilient backpropagation (RPROP) algorithm [12] which maintains a separate learning rate for each weight and that uses only the sign of the gradient to determine the weight change. The combination of BPTT and RPROP proved to be stable. Not only the template weights $\mathcal{W}$ are modified in this way, but their biases $\mathcal{B}$ and start values $\mathcal{V}^0$ are adapted as well.

To accelerate the training, we initially trained with small randomly chosen subsets of the training set that increased in size over time [2].

4 Experimental Results

4.1 Still Input

After the training, the network is able to localize the BioID faces. Figure 5 shows the development of the trained network's output when the test image from Fig. 2 is presented as input. One can see that the blobs indicating the eye positions develop in a top-down fashion. After the first iteration they appear only in the lowest resolution. This coarse localization is used to bias the development of blobs in lower layers. After five iterations, the network's output is close to the desired blob pattern. The output does not change significantly during the next iterations. Each iteration takes only 16ms on an AMD Athlon XP 2.08GHz PC.

The generation of stable blobs is the typical behavior of the network. This behavior is similar to the emergence of stable activity blobs in neural fields, investigated by Amari [1]. Usually, exactly one blob per eye rises. In a few cases, blobs are unstable, no blobs are produced, or multiple blobs represent multiple localization hypothesis.

To evaluate the localization performance quantitatively, one has to estimate eye coordinates from the blobs and to compare them to the target coordinates.

Each eye position is estimated separately by finding the feature cell with the highest activity in the corresponding high-resolution output array. In a 7×7 window around the most active cell the feature cells are segmented as belonging to the blob or not by comparing their activity with a threshold. This threshold depends on the activity of the most active cell and increases with greater distance from it. The weighted mean of the cells segmented as belonging to the blob is used as an estimate for the eye position.

After transforming the estimated eye positions into the original coordinate system, a scale-independent distance measure d_{eye} is computed as suggested by the creators of the data set [8]:

$$d_{eye} = \mathsf{max}(d_l, d_r)/||C_l - C_r||.$$

The distances of the estimated eye positions to the target coordinates C_l and C_r are denoted by d_l and d_r, respectively. A distance $d_{eye} < 0.25$ is considered to be a successful localization, since $d_{eye} = 0.25$ corresponds approximately to the half-width of an eye.

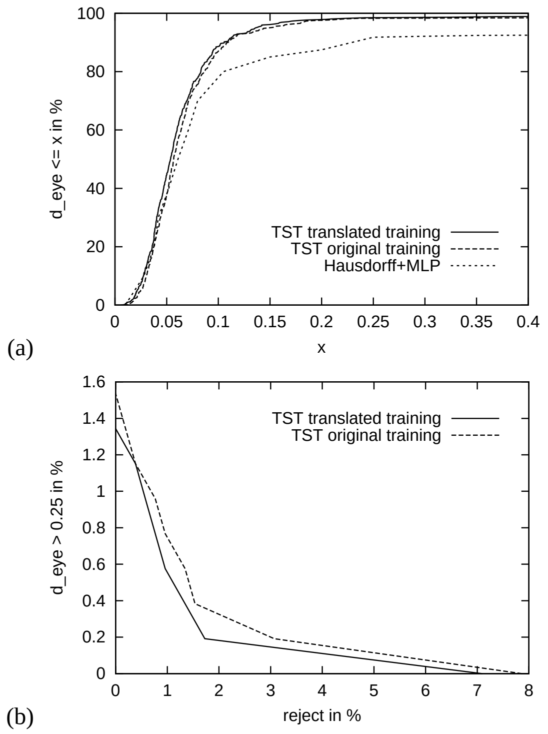

Fig. 6 Localization performance: (a) test examples with small d_{eye} for pyramid trained with the { original | translated } images and for a hybrid system (Hausdorff+MLP) [8]; (b) rejecting the least confident examples lowers the number of mislocalizations.

Not surprisingly, all training examples have been localized successfully by the trained network. The localization performance on the test set is good as well. Figure 6(a) shows the network's test set localization performance when trained with the { original | translated } images in comparison to the performance from the hybrid reference system [8] (Hausdorff+MLP). Only { 1.5% | 1.3% } of the test examples have not been localized accurately enough. Compare this to the 8.2% mislocalizations of the reference system.

A detailed analysis of the network's output for the mislocalizations showed that in these cases the output deviates from the one-blob-per-eye pattern. It can happen that no blob or that several blobs are present for an eye. By comparing the

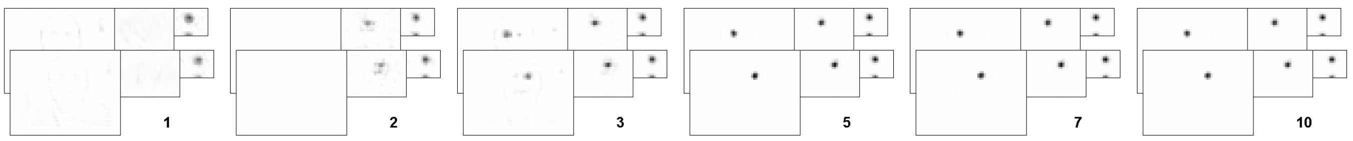

Fig. 5 Recall. The activities of the network's output over time.

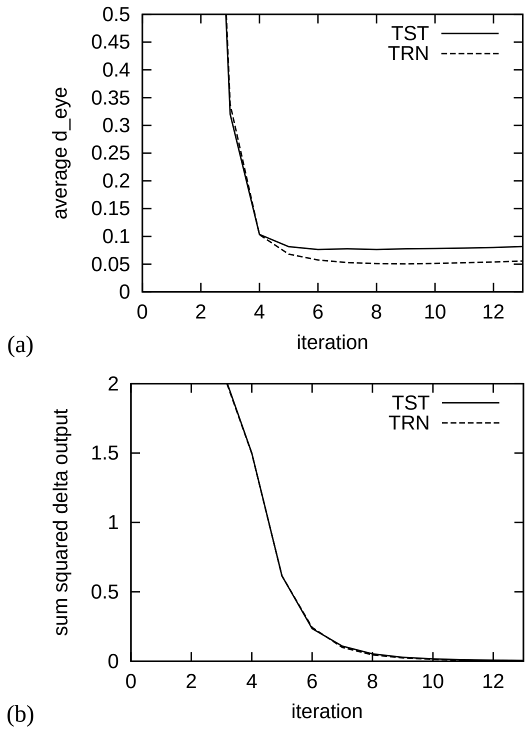

Fig. 7 Performance over time: (a) average distance d_{eye}; (b) output changes.

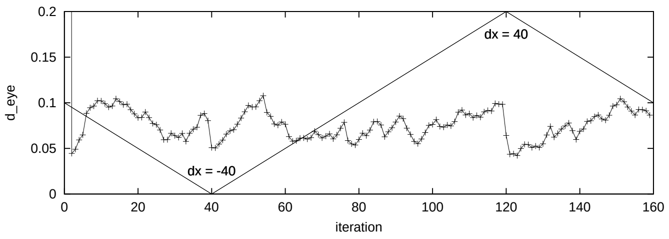

Fig. 8 Tracking of a horizontally moving input (see Fig. 2 for original).

activity of a segmented blob to a threshold and to the total activity of its feature array, a confidence measure is computed for each eye. Both eye confidences are combined to a single confidence which is compared to a rejection threshold.

In Figure 6(b) one can see that rejecting the least confident test examples lowers the number of mislocalizations rapidly. When rejecting { 3.1% | 1.7% } of the images, only one mislocalization is left. The average localization error of the accepted examples is $d_{eye} = 0.06$. That is well within the area of the iris and corresponds to the accuracy of the manually determined target coordinates.

Figure 7 illustrates the network's performance over time when trained with the original images. The average error d_{eye} drops rapidly within the first five iterations and stays low afterwards. The changes in the network's output are large during the first iterations and decrease even when updated longer than the ten steps it has been trained for.

4.2 Moving Input

Although the network has only seen still images so far, it is also able to track a moving face. Fig. 8 shows the localization error as the example from Fig. 2 is translated horizontally by ± 40 pixels with a speed of one pixel per iteration. The localization error drops quickly during the first iterations and stays low afterwards at a value of about 0.075. The blobs indicating the eye positions follow the movement of the image with a small lag. They catch up at iterations 40 and 120, when the direction of movement is reversed.

As described in Section 2, we created a version of the BioID data set to investigate the behavior of the network in the case of moving input more systematically. It contains for each still image three image sequences moving in a random direction.

Starting from the weights trained with still images, we continued the supervised training with these moving inputs and targets. The behavior of the trained network when confronted with image sequences was very similar to the still-image case. Blobs indicating the eye positions were produced in a top-down fashion and were refined over time. The main difference was that these blobs were not stationary, but moved along with the input. This is illustrated in Figure 9. In Part (c) of the figure one can see that, after the initial iterations are over, most changes in the network outputs correspond to the blob motion. The network dynamics can be viewed as being settled into a continuous attractor [14], where input changes along a manifold (in this case translations) make little difference to the energy of the system.

Figure 10 illustrates the network's test set performance over time. The average localization error d_{eye} drops rapidly within the first five iterations and stays below 0.09 afterwards. The average changes in the network's output are large during the first iterations and decrease to a floor that corresponds to the blob motion.

The test set localization performance of the network is shown in Figure 11(a) for iterations 11 and 21. Both curves are very similar. The percentages of examples that have not been localized accurately enough are 2.56% and 2.62%, respectively. While these are significantly higher mislocalization rates than have been observed for static images, this still

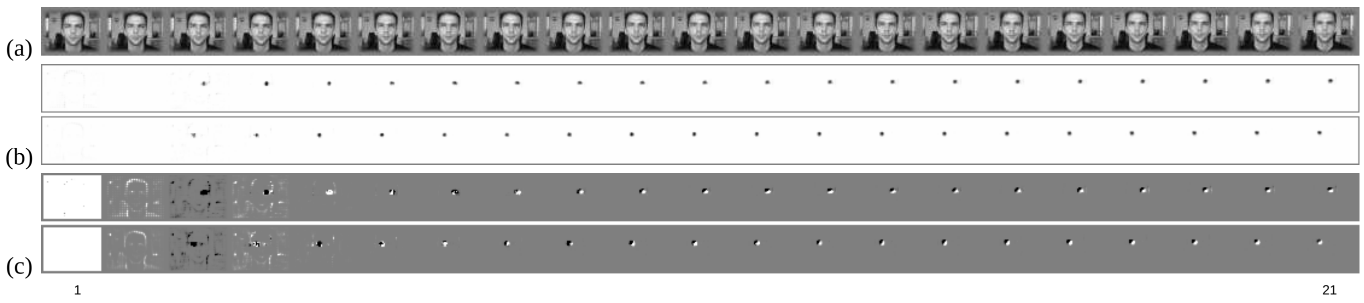

Fig. 9 Tracking the eye positions in an image sequence: (a) input moving towards NW (↖); (b) output activity in layer L_0; (c) change of output activity (medium gray indicates no change; the bright SE half together with the dark NW half of a blob indicates that it moves towards NW).

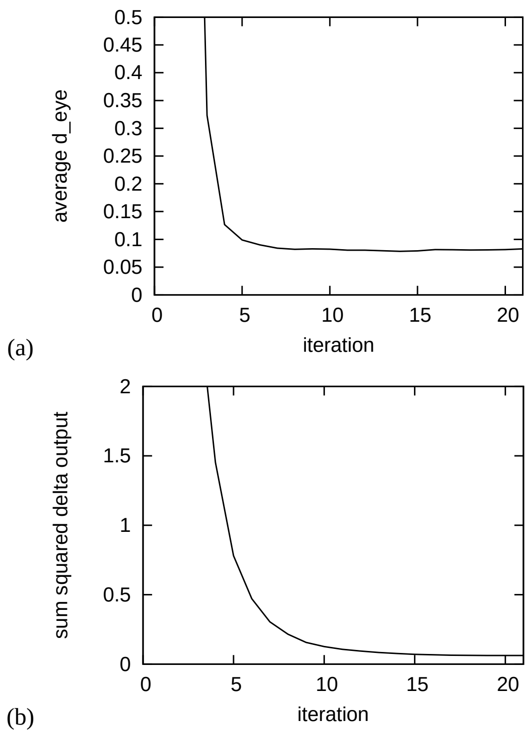

Fig. 10 Test set performance for moving input over time: (a) average distance d_{eye}; (b) output changes.

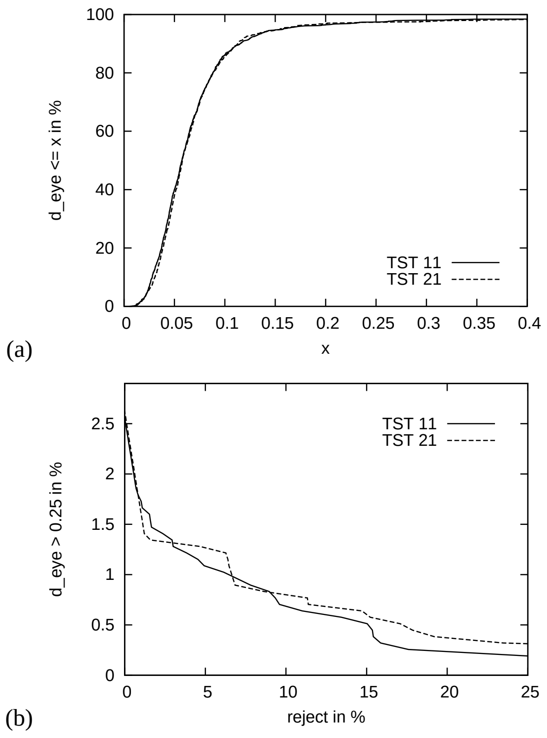

Fig. 11 Localization performance with moving input: (a) test examples with small d_{eye} after 11 and 21 iterations; (b) rejecting the least confident examples lowers the number of mislocalizations.

means that in about 97.4% of all test image sequences the eye positions can be tracked accurately enough.

Part (b) of the figure shows that one can lower the mislocalization rate by rejecting the least confident examples. Rejecting also increases the localization accuracy of the accepted image sequences. When rejecting about 2.3% of the examples, the average localization error drops to 0.068 after both 11 and 21 iterations. This shows that the vast majority of test faces can be tracked accurately.

5 Conclusions

In this paper, we presented an approach to face localization and tracking that is based on a hierarchical neural network with local recurrent connectivity. The network is trained to solve this task even in the presence of complex backgrounds, difficult lighting, and noise through iterative refinement. Since it processes an entire image in parallel, it does not need any time-consuming scanning over positions and scales.

The network's localization performance was evaluated on the BioID data set. It compares favorably to a hybrid reference system that uses a Hausdorff shape matching approach in combination with a multilayer perceptron.

Adding slightly translated variants to the training set facilitated generalization. One could extend this approach by using a larger variety of transformations, including rotation, scaling, changes in lighting, and noise.

The proposed method is not limited to gray-scale images. The extension to color is straight forward and should improve localization performance even further.

Because the network works iteratively and one iteration takes only a few milliseconds, it is also possible to use it for real-time face tracking. We trained a version of the network to localize and track faces in still images that move in a random direction. Although localization performance was not quite as good as it was in the static case, the network was still able to track the vast majority of test faces with high accuracy. Because the tracking task is more challenging than still-image localization, it would be worth investigating whether a similar network with more free parameters could approach the still-image localization performance again.

In the described tracking experiments, the image sequences have been created from still images. Hence, no motion cues could aid the tracking. For that reason, it would be desirable to train the network with annotated real video sequences.

References

1. Shun-ichi Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87, 1977.
2. Sven Behnke. Learning iterative image reconstruction in the Neural Abstraction Pyramid. *International Journal of Computational Intelligence and Applications, Special Issue on Neural Networks at IJCAI-2001*, 1(4):427–438, 2001.
3. Sven Behnke. *Hierarchical Neural Networks for Image Interpretation*, volume LNCS 2766 of *Lecture Notes in Computer Science*. Springer, 2003.
4. Sven Behnke and Raúl Rojas. Neural Abstraction Pyramid: A hierarchical image understanding architecture. In *Proceedings of International Joint Conference on Neural Networks (IJCNN'98) – Anchorage*, volume 2, pages 820–825, 1998.
5. Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.
6. Kunihiko Fukushima, Sei Miyake, and Takayuki Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:826–834, 1983.
7. Erik Hjelmas and Boon Kee Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83:236–274, 2001.
8. Oliver Jesorsky, Klaus J. Kirchberg, and Robert W. Frischholz. Robust face detection using the Hausdorff distance. In *Proceedings of Third International Conference on Audio- and Video-based Biometric Person Authentication, LNCS-2091, Halmstad, Sweden*, pages 90–95. Springer, 2001.
9. Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
10. Dario Maio and Davide Maltoni. Real-time face localization on gray-scale static images. *Pattern Recognition*, 33:1525–1539, 2000.
11. Kieron Messer, Jiri Matas, Josef Kittler, Juergen Luettin, and Gilbert Maitre. XM2VTSDB: The extended M2VTS database. In *Proceedings of Second International Conference on Audio and Video-based Biometric Person Authentication (AVBPA'99) – Washington, DC*, pages 72–77, 1999.
12. Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proceedings of International Conference on Neural Networks (ICNN'93) – San Francisco*, pages 586–591. IEEE, 1993.
13. Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. Neural network based face detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20:23–38, 1998.
14. H. Sebastian Seung. Learning continuous attractors in recurrent networks. In M.I. Jordan, M.J. Kearns, and S.A. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 654–660. MIT Press, 1998.
15. Paul Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 2(57):137–154, 2004.
16. Ronald J. Williams and Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):491–501, 1990.
17. Ming-Hsuan Yang, David Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(1):34–58, 2002.