ELSEVIER

Image and Vision Computing 19 (2001) 593–618

image AND vision COMPUTING

www.elsevier.com/locate/imavis

Real-time object recognition on image sequences with the adaptable time delay neural network algorithm — applications for autonomous vehicles

C. Wöhler[a,*], J.K. Anlauf[b]

[a]DaimlerChrysler Research and Technology, Image Understanding Systems, (FT3/AB), P.O. Box 2360, D-89013 Ulm, Germany
[b]Rheinische Friedrich-Wilhelms-Universität Bonn, Institut für Informatik II, Römerstr. 164, D-53117 Bonn, Germany

Received 5 April 2000; accepted 18 December 2000

Abstract

Within the framework of the vision-based "Intelligent Stop&Go" driver assistance system for both the motorway and the inner city environment, we present a system for segmentation-free detection of overtaking vehicles and estimation of ego-position on motorways as well as a system for the recognition of pedestrians in the inner city traffic scenario. Both systems are running in real-time in the test vehicle UTA of the DaimlerChrysler computer vision lab, relying on the adaptable time delay neural network (ATDNN) algorithm. For object recognition, this neural network processes complete image sequences at a time instead of single images, as it is the case in most conventional neural algorithms. The results are promising in that using the ATDNN algorithm, we are able to perform the described recognition tasks in a large variety of real-world scenarios in a computationally highly efficient and rather robust and reliable manner.

Keywords: Autonomous driving; Vehicle detection; Pedestrian recognition; Image sequence; Time delay neural network

1. Introduction

During the last 10 years, various vision systems for vehicle guidance, lane departure warning, and collision avoidance have been developed ([16,39,48]). An example is the Daimler–Benz demonstration vehicle VITA II [45] that is able to drive autonomously on motorways and perform overtaking manoeuvres without interaction with the driver. The more advanced driver assistance system Urban Traffic Assistant (UTA) for both the motorway and the inner city environment is outlined in Ref. [15]. While UTA works on the motorway at speeds of up to 130 km/h, it is especially useful in the inner city at a high traffic density and in stop&go traffic. For Intelligent Stop&Go, a stereo vision algorithm [17] detects an appropriate leading vehicle and signals the driver that it is ready for autonomous following. Another module recognizes arrows on the road surface and extracts the course of the lane even if it is badly marked and does not show clothoidal geometry. It is furthermore possible to recognize traffic signs and traffic lights to eventually terminate the autonomous following process. First ideas within the UTA framework on the recognition of pedestrians, which are the most vulnerable traffic participants in the inner city environment, are presented in Refs. [15,20,52].

In this paper, we will describe methods for object detection and recognition in the traffic environment based on the adaptable time delay neural network (ATDNN) approach [51], which have been developed within the UTA project. These methods are applied to the detection of overtaking vehicles and estimation of ego-position on motorways as well as to the recognition of pedestrians. For the scenario of pedestrian recognition, the ATDNN algorithm is compared to standard polynomial support vector machine classifiers applied to the same training and test data with respect to recognition errors, computational complexity, and memory demand.

Our contribution is structured as follows: In Section 2 we summarize neural methods which are related to the ATDNN algorithm and suitable for processing time signals and give a brief introduction to the ATDNN algorithm. We then describe in Section 3 our ATDNN-based system for segmentation-free-detection of overtaking vehicles and estimation of ego-position on motorways. In Section 4 we describe the ATDNN-based system for pedestrian recognition. An overview about the state-of-the-art systems for vehicle detection and pedestrian recognition is as well

* Corresponding author.
E-mail addresses: christian.woehler@daimlerchrysler.com (C. Wöhler), anlauf@informatik.uni-bonn.de (J.K. Anlauf).

0262-8856/01/$ - see front matter
PII: S0262-8856(01)00040-3

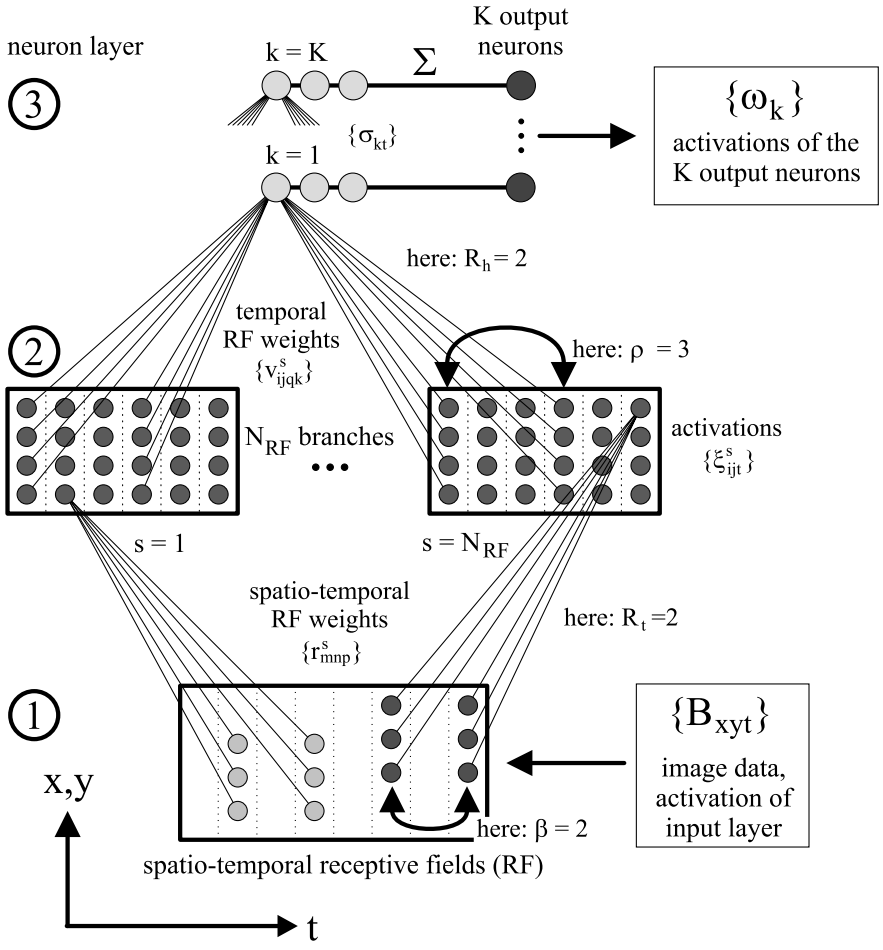

Fig. 1. Architecture of the ATDNN with spatio-temporal receptive fields. In this figure, the time delay parameters are set to $\beta = 2$ and $\rho = 3$. For clarity, the two spatial directions x and y are compressed into one in this figure.

given in Sections 3 and 4, respectively. In Section 5, we summarize our results and give a brief discussion.

2. The ATDNN algorithm for image sequence analysis

2.1. Neural concepts related to the ATDNN algorithm

The general TDNN concept is well known from applications in the domain of speech recognition [46]. An important training algorithm for the TDNN, named temporal backpropagation, is presented in Ref. [47]. A training algorithm for adaptable time delay parameters is developed in Ref. [8] for continuous time signals. A related algorithm is described in Ref. [5] for input data defined at discrete time steps, as it is the case for video image sequences. The shape of the temporal receptive fields of the input layer, however, is predetermined to be Gaussian, and the task of adapting the time delay parameters reduces to determining optimal values for the centre and the width of the input window. Neural networks with spatial receptive fields have been applied in the field of character recognition [18,32]. Here, the spatial receptive fields induce a certain tolerance of the network with respect to variations in scale and position of the objects to be classified.

A TDNN structure with spatio-temporal receptive fields and fixed time delay parameters for image sequence analysis is presented in Ref. [50]. The ATDNN algorithm [51] extends this concept toward adaptable time delay parameters such that the network not only learns its weights but also the temporal extensions of its receptive fields. This algorithm will be described in brief in this section.

2.2. Architecture of the ATDNN

The ATDNN algorithm used for the analysis of image sequences throughout this paper is based on a time delay neural network with spatio-temporal receptive fields and adaptable time delay parameters. Its architecture is shown in Fig. 1. The three-dimensional input layer receives a sequence of $S_t^{(1)}$ images of size $S_x^{(1)} \times S_y^{(1)}$ pixels — the index (1) refers to the first neuron layer-acquired at a constant rate, as it is the case, e.g. for video image sequences. The activation B_{xyt} of the input neuron at position

(xyt) corresponds to the pixel intensity at position (xy) on the t-th image of the input sequence. In the ATDNN architecture, a neuron of a higher layer does not receive input signals from all neurons of the underlying layer, as it is the case, e.g. for multi layer perceptrons (MLPs), but only from a limited region of it, called the *spatio-temporal receptive field* of the corresponding neuron. Such a spatio-temporal receptive field covers a region of $R_x \times R_y \times T_{\mathrm{eff}}^{(1)}$ pixels in the input sequence with $T_{\mathrm{eff}}^{(1)} = 1 + (R_t - 1)\beta$ and R_t as the number of weight sets that belong to the same time slot, respectively. We call β the *time delay parameter*. The distance of the centres of two neighbouring spatio-temporal receptive fields in x, y and t direction is given by D_x, D_y and D_t; we constantly take $D_t = 1$ as in our applications, the temporal extension of the image sequences to be analysed is significantly lower than the spatial extension. In contrast to a standard MLP architecture, which is invariant with respect to permutations of the input neurons, the concept of spatio-temporal receptive fields takes into account local relations between the pixels, a property that will turn out to be very efficient for image sequence analysis concerning computational complexity and memory requirements.

The ATDNN is composed of N_{RF} different branches, each of which consists of a three-dimensional layer of neurons (layer 2 in Fig. 1). As we follow the *shared weights* principle inside each branch, the same set of weight factors $\{r_{mnp}^s\}$ is assigned to each layer 2 neuron of the s-th branch, with m and n as the spatial and p as the temporal coordinate inside the weight configuration of the receptive field. We thus have $1 \le s \le N_{\mathrm{RF}}$, $1 \le m \le R_x$, $1 \le n \le R_y$, and $1 \le p \le R_t$. Effectively, this configuration of spatio-temporal receptive fields produces activation patterns in neuron layer 2 representing one spatio-temporally filtered version of the original input image sequence per network branch.

Neuron layer 2 and 3 are fully connected in the spatial directions; in the temporal direction, we implemented a structure of temporal receptive fields and shared weights with time delay parameter ρ. The extension of the temporal receptive fields between neuron layer 2 and 3 amounts to $T_{\mathrm{eff}}^{(2)} = 1 + (R_h - 1)\rho$, the spatial extensions of neuron layer 2 itself are given by $S_x^{(2)}$ and $S_y^{(2)}$. To each branch s and each output class k one temporal receptive field is assigned. The weight factors are denoted by $\{v_{ijqk}^s\}$ with $1 \le s \le N_{\mathrm{RF}}$, $1 \le i \le S_x^{(2)}$, $1 \le j \le S_y^{(2)}$, $1 \le q \le R_h$ and $1 \le k \le K$.

The ATDNN as shown in Fig. 1 is only defined for integer-valued time delay parameters β and ρ. As we will show later on how to extend the concept to real-valued time delay parameters, we will first define the activations for the example integer-valued combination $(\lceil\beta\rceil, \lceil\rho\rceil)$. In the following, $\lfloor x\rfloor$ denotes the largest integer that is smaller than or equal to the real number x, $\lceil x\rceil$ the smallest integer that is larger than or equal to x. Analogous expressions of the activations depending on the combinations $(\lfloor\rho\rfloor, \lceil\beta\rceil)$, $(\lceil\rho\rceil, \lfloor\beta\rfloor)$, and $(\lfloor\rho\rfloor, \lfloor\beta\rfloor)$ are obtained in a straightforward manner.

The activation $\xi_{ijt}^s(\lceil\beta\rceil)$ of the layer 2 neuron at position (i, j, t) in branch s for the integer-valued time delay parameter $\lceil\beta\rceil$ is given by

$$\xi_{ijt}^s(\lceil\beta\rceil) = g_2\left(\sum_{p=1}^{R_t}\sum_{n=1}^{R_y}\sum_{m=1}^{R_x} r_{mnp}^s B_{D_x(i-1)+m, D_y(j-1)+n, t+(p-1)\lceil\beta\rceil} - \theta^s\right) \tag{1}$$

with $g_2(x) = \tanh(x)$ as a sigmoidal activation function and θ^s as the respective threshold value. The dependence on the input data $\{B_{xyt}\}$ is omitted in Eq. (1) and the following equations. For the activations $\sigma_{kt}(\lceil\rho\rceil, \lceil\beta\rceil)$ of neuron layer 3 we have

$$\sigma_{kt}(\lceil\beta\rceil, \lceil\rho\rceil) = g_3\left(\sum_{s=1}^{N_{\mathrm{RF}}}\sum_{q=1}^{R_h}\sum_{j=1}^{S_y^{(2)}}\sum_{i=1}^{S_x^{(2)}} v_{ijqk}^s \xi_{i,j,t+(q-1)\lceil\rho\rceil}^s(\lceil\beta\rceil)\right) \tag{2}$$

with $g_3(x) = \tanh(x)$. The K output neurons of the ATDNN perform a class-wise temporal integration of the activations of neuron layer 3, yielding the output activations

$$\omega_k(\lceil\beta\rceil, \lceil\rho\rceil) = \frac{1}{S_t^{(3)}(\lceil\beta\rceil, \lceil\rho\rceil)} \sum_{t=1}^{S_t^{(3)}(\lceil\beta\rceil\lceil\rho\rceil)} \sigma_{kt}(\lceil\beta\rceil, \lceil\rho\rceil). \tag{3}$$

Here, $S_t^{(3)}(\lceil\beta\rceil, \lceil\rho\rceil) = S_t^{(1)} - (R_t - 1)\lceil\beta\rceil - (R_h - 1)\lceil\rho\rceil$ denotes the number of layer 3 neurons per class. The value $S_t^{(3)}(\lceil\beta\rceil, \lceil\rho\rceil)$ may change under variation of β or ρ such that the sum in Eq. (3) has to be normalized correspondingly. The usual way to evaluate the ATDNN output (3) is that the output neuron with the highest activation determines the class to which the input pattern belongs. In the following sections, however, we will mostly regard classification problems with $K = 2$ classes. In this special case, the output is evaluated such that if $q\omega_1 > \omega_2$, the input pattern belongs to training class $k = 1$, otherwise to class $k = 2$. By varying the parameter q, regarding a large test set, we then obtain the receiver operating characteristics (ROC) curve that denotes the trade-off between the classification errors of the two training classes.

As we employ a gradient descent based method to adapt the time delay parameters β and ρ, it is necessary to define an ATDNN output for real-valued β and ρ. A straightforward way is bilinear interpolation:

$$\begin{aligned}\omega_k(\beta, \rho) = {} & \mathrm{frac}(\beta)\mathrm{frac}(\rho)\omega_k(\lceil\beta\rceil, \lceil\rho\rceil) \\ & + \mathrm{frac}(\beta)(1 - \mathrm{frac}(\rho))\omega_k(\lceil\beta\rceil, \lfloor\rho\rfloor) \\ & + (1 - \mathrm{frac}(\beta))\mathrm{frac}(\rho)\omega_k(\lfloor\beta\rfloor, \lceil\rho\rceil) \\ & + (1 - \mathrm{frac}(\beta))(1 - \mathrm{frac}(\rho))\omega_k(\lfloor\beta\rfloor, \lfloor\rho\rfloor) \\ & \text{with } \mathrm{frac}(x) = x - \lfloor x\rfloor. \end{aligned} \tag{4}$$

The computational cost of calculating $\omega_k(\beta,\rho)$ is not four times higher than the one required for one of the configuraions $\omega_k(\lceil\beta\rceil, \lceil\rho\rceil)$, $\omega_k(\lceil\beta\rceil, \lfloor\rho\rfloor)$, $\omega_k(\lfloor\beta\rfloor, \lceil\rho\rceil)$, and $\omega_k(\lfloor\beta\rfloor, \lfloor\rho\rfloor)$, as

one might think regarding Eq. (4), but only about twice as high. The reason is that under the kind of setting of the ATDNN architecture parameters regarded here, more than 95% of the CPU time is needed to compute the activiations $\{\xi^s_{ijt}\}$ of neuron layer 2, which has to be done twice, i.e. for $\lceil\beta\rceil$ and $\lfloor\beta\rfloor$, respectively (Eq. (1)). In comparison, the computational cost of calculating the activations of the higher neuron layers is irrelevant.

2.3. The training algorithm

We use a backpropagation-like online gradient descent rule to train the ATDNN weights and time delay parameters. The error measure ϵ for an input pattern of class c is of a quadratic form and amounts to

$$\epsilon(\beta,\rho)=\frac{1}{2}\sum_{k=1}^{K}(\omega_k(\beta,\rho)-\tau_k)^2 \quad \text{with} \quad \tau_k=\begin{cases} A \text{ if } k=c \\ 0 \text{ if } k\neq c \end{cases} \tag{5}$$

where the constant A is slightly lower than 1. The learning rules for the weights and time delay parameters are now obtained by deriving the error measure (5) with respect to them. The time delay parameters are positive by definition; furthermore, their values are limited by the temporal extension of the neuron layer to which they refer. The time delay parameters are initialized such that the resulting initial temporal extension of the corresponding receptive fields amounts to between roughly 20 and 80% of the temporal extension of the neuron layer to which the time delay parameter refers, respectively. Generally, several "optimum" configurations of β and ρ can be obtained depending on their initial values. The weight parameters $\{r^s_{mnp}\}$ and $\{v^s_{ijqk}\}$ and threshold values $\{\theta^s\}$ are initialized by small positive and negative random numbers of the order $\mathcal{O}(10^{-6})$ in order to break the symmetry ([28]). Details concerning learning rates used in the ATDNN training algorithm and clipping procedures for the time delay parameters β and ρ can be found in Ref. [51].

2.4. Noise robustness

As the images used as an input for the ATDNN are eventually acquired under difficult visibility conditions, they may be highly disturbed by noise. Hence, we will comment on the noise robustness of the ATDNN algorithm. For this purpose we begin with an approximate expression for the noise of the activations of neuron layer 2 by regarding an arbitrary layer 2 neuron. The vector $\vec{b}$ denotes the part of the image sequence covered by the spatio-temporal receptive field of this neuron, the vector $\vec{r}$ the weight configuration of the receptive field. The dimension of $\vec{b}$ and $\vec{r}$, i.e. the number of pixels covered by the receptive field, is given by $N=R_xR_yR_t$.

The inner field of the layer 2 neuron thus amounts to $\vec{r}\cdot\vec{b}$. For noisy input data $\vec{b}$, the variance $\sigma^2_{\vec{r}\cdot\vec{b}}$ of the inner field amounts to

$$\sigma^2_{\vec{r}\cdot\vec{b}}=\langle(\vec{r}\cdot\vec{b}-\vec{r}\cdot\langle\vec{b}\rangle)^2\rangle=\left\langle\left(\sum_{i=1}^{N}r_ib_i\right)^2\right\rangle-\left(\sum_{i=1}^{N}r_i\langle b_i\rangle\right)^2. \tag{6}$$

Without loss of generality, we will furthermore assume zero mean of the image data vector. This leads to the subtraction of the offset $\vec{r}\cdot\langle\vec{b}\rangle$ from the inner field of the layer 2 neuron; to obtain the same activation value as with non-zero mean, only the threshold θ^s of the network branch to which the layer 2 neuron belongs has to be adjusted by the same offset value. The transformed image data vector is denoted by $\tilde{\vec{b}}=\vec{b}-\langle\vec{b}\rangle$. Replacing $\vec{b}$ by $\tilde{\vec{b}}$ in Eq. (6) then yields

$$\sigma^2_{\vec{r}\cdot\vec{b}}=\sum_{k=1}^{N}\sum_{i=1}^{N}r_ir_k\langle\tilde{b}_i\tilde{b}_k\rangle. \tag{7}$$

The averages $\langle\tilde{b}_i\tilde{b}_k\rangle$ directly correspond to the elements C_{ik} of the covariance matrix C of the stochastic process that generates the random variables $\tilde{b}_i$. With the realistic assumption of uncorrelated Gaussian noise for the image pixel values, i.e. $C_{ik}=\sigma^2_{\tilde{b}}\delta_{ik}$, we arrive at

$$\frac{\sigma^2_{\vec{r}\cdot\vec{b}}}{\sigma^2_{\tilde{b}}}=|\vec{r}|^2. \tag{8}$$

In all our experiments (cf. the following sections) we observed empirically that the training algorithm of the ATDNN adjusts the weight configurations of the spatio-temporal receptive fields such that the inner field of a layer 2 neuron typically amounts to $\vec{r}\cdot\vec{b}=\mathcal{O}(1)$. Note that $\vec{b}$ is the original, non-zero mean image data vector. We furthermore make the realistic assumption $|\vec{b}|=\mathcal{O}(\sqrt{N}I_{\max})$; we have, e.g. $I_{\max}=255$ for 8 bit greyscale images. We therefore obtain $|\vec{r}|=\mathcal{O}(1/(I_{\max}\sqrt{N}))$, leading to the expression

$$\sigma_{\vec{r}\cdot\vec{b}}\approx\frac{\sigma_{\tilde{b}}}{I_{\max}}\frac{1}{\sqrt{N}} \tag{9}$$

for the noise $\sigma_{\vec{r}\cdot\vec{b}}$ in neuron layer 2. As we are interested in approximate relations only, we neglect the effect of the non-linear activation function $g(x)=\tanh(x)$ on the noise reduction. In Eq. (9), the term $\sigma_{\tilde{b}}/I_{\max}$ can be regarded as the inverse signal-to-noise ratio of the input images. Repeating this procedure until we arrive at the output layer of the ATDNN, we obtain for the noise σ_ω of the output activations:

$$\sigma_\omega\approx\frac{\sigma_{\tilde{b}}}{I_{\max}}\frac{1}{\sqrt{R_xR_yR_tS^{(2)}_xS^{(2)}_yR_hS^{(3)}_t}}. \tag{10}$$

Results (9) and (10) mean that the noise of the input image data is suppressed inside the network such that it hardly affects the output activations-for the ATDNN configurations used in the following sections, the second factor in Eq. (10) is of the order $\mathcal{O}(10^{-2})$. Hence, the influence of the noise on the training process and thus on the final, "optimal"

configuration of weights and time delay parameters is negligible. This is in strong contrast, e.g. to the support vector machine (Ref. [40] and Section 4.4), where the decision surface in feature space is directly defined by a certain subset of the noisy training samples and thus strongly depends on the noise properties.

2.5. Computational complexity and memory demand

We will now regard the computational complexity F of the ATDNN algorithm in terms of the number of floating point operations necessary to classify an input pattern after having finished the training phase. The duration of the training process is largely irrelevant as it is not performed in the real-time system but offline in the laboratory. The expression for F is obtained in a straightforward manner:

$$F = F_{\text{flop}} + F_{\tanh}, \quad \text{where}$$

$$\begin{aligned} F_{\text{flop}} = {} & R_x R_y R_t N_{\text{RF}} S_x^{(2)} S_y^{(2)} (S_t^{(2)}(\lceil\beta\rceil) + S_t^{(2)}(\lfloor\beta\rfloor)) \\ & + S_x^{(2)} S_y^{(2)} R_h N_{\text{RF}} K (S_t^{(3)}(\lceil\beta\rceil,\lceil\rho\rceil) + S_t^{(3)}(\lceil\beta\rceil,\lfloor\rho\rfloor) \\ & + S_t^{(3)}(\lfloor\beta\rfloor,\lceil\rho\rceil) + S_t^{(3)}(\lfloor\beta\rfloor,\lfloor\rho\rfloor)) \end{aligned} \tag{11}$$

$$\begin{aligned} \frac{F_{\tanh}}{z} = {} & N_{\text{RF}} S_x^{(2)} S_y^{(2)} (S_t^{(2)}(\lceil\beta\rceil) + S_t^{(2)}(\lfloor\beta\rfloor)) + S_t^{(3)}(\lceil\beta\rceil,\lceil\rho\rceil) \\ & + S_t^{(3)}(\lceil\beta\rceil,\lfloor\rho\rfloor) + S_t^{(3)}(\lfloor\beta\rfloor,\lceil\rho\rceil) + S_t^{(3)}(\lfloor\beta\rfloor,\lfloor\rho\rfloor). \end{aligned} \tag{12}$$

Here, F_{flop} is the number of floating point multiplications and additions to be performed in the weighted sums, whereas $F_{\tanh}$ denotes the number of operations for evaluating the non-linear activation functions; z is the number of floating point operations needed to perform one evaluation of the function $\tanh(x)$. By regarding the corresponding CPU times it turned out that $F_{\tanh} \ll F_{\text{flop}}$ in all examined configurations, such that $F \approx F_{\text{flop}}$.

The number n_{w} of weights and time delay parameters determines the memory needed to store the ATDNN structure in a certain hardware. It is given by the expression

$$n_{\text{w}} = \underbrace{R_x R_y R_t N_{\text{RF}}}_{\text{spatio-temporal RF}} + \underbrace{N_{\text{RF}}}_{\text{thresholds}} + \underbrace{S_x^{(2)} S_y^{(2)} R_h K N_{\text{RF}}}_{\text{temporal RF}} + \underbrace{2}_{\beta \text{ and } \rho}. \tag{13}$$

Due to the shared weights principle and the fact that the ATDNN architecture is not fully connected, n_{w} always amounts to below 1000 in the network configurations considered in the following sections, which is significantly smaller than the number of pixel values of an input pattern, respectively. This would not be possible even with the most elementary fully connected neural network architecture, the perceptron.

The computational complexity F of the ATDNN algorithm as well as its memory demand n_{w} will be compared to that of a support vector machine classifier with and without preliminary dimensionality reduction by principal component analysis (PCA) in Section 4.5.

3. Segmentation-free detection of overtaking vehicles and rough estimation of ego-position in the motorway scenario

Within the framework of Intelligent Stop&Go, autonomous driving is controlled by following the vehicle ahead at a constant or speed-dependent distance which is determined by the real-time stereo vision algorithm described in detail in Ref. [17]. On motorways; this method allows following the leading vehicle not only in stop&go traffic but also at speeds of up to 130 km/h. To enable the system to select a new leading vehicle driving at a higher speed than the current one, it is necessary to have a system that permanently observes the left lane, assumed that the ego-vehicle is driving on the right lane. Such a surveillance of the left lane would be computationally rather expensive when performed by the stereo algorithm; only in the case of detection of an overtaking vehicle the stereo algorithm changes its attention toward this new leading vehicle to be followed at higher speed. This section describes a module based on the ATDNN algorithm presented in Section 2 for the detection of overtaking vehicles on image sequences without a preliminary segmentation step Ref. [54]. In addition to that, the system recognizes if the ego-vehicle is currently driving on the very left lane of the motorway such that it cannot be overtaken.

3.1. State-of-the-art systems and techniques for vehicle detection on motorways

Traditional approaches for vehicle detection use explicit knowledge about the appearance of vehicles. In Ref. [43], the constellation of two line pairs is detected and tracked, which serves as a recursive estimation of motion according to the four-dimensional spatio-temporal technique described in Refs. [11,12]. Other approaches regard symmetry properties of vehicle rears [10,31,56]. In Ref. [49] two-dimensional object hypotheses are generated by grouping line segments according to certain heuristics such as collinearity or symmetry. An object hypothesis is accepted only if its properties in subsequent images are sufficiently similar. In [44] object hypotheses are generated with respect to the appearance of vertical edges fulfilling certain symmetry and size limitations. Stereo vision ([17]) can be used as well for vehicle detection.

In Ref. [30] a system for detecting preceding and overtaking vehicles from inside a moving vehicle by analysis of the optical flow vector field is described. The optical flow vectors are calculated in video real time by a special hardware. Similar vectors are clustered in order to eliminate outliers and to increase computational efficiency. In a subsequent stage independently moving objects are separated

Table 1
Composition of the training and the test set for the detection of overtaking vehicles

	Overtaking vehicles	Garbage	Sum
Training set	484	2402	2886
Test set	881	4369	5250

from the background, which is moving as well, by taking into account the speed and the gear angle of the ego-vehicle. Related approaches can be found in Refs. [13,14]. In Refs. [21] the relatively high computational complexity of the optical flow approach is avoided by modelling the motion trajectory of the non-stationary observer in the scene. Given the ego-motion of the observer, the trajectories of several points in the image are predicted. Analysis of the temporal behaviour of the corresponding greyscale profiles then reveals if the scene contains independently moving objects.

In Ref. [23] a distance-independent representation of the objects is generated by position-dependent subsampling of the image. The detection step then consists of searching for structures of a certain size and shape.

In Ref. [22] a trainable system for the recognition of front and rear views of vehicles in the highway scenario is presented. The image is preprocessed by orientation coding, which assigns local features like edges and corners to certain codes. Horizontal and vertical structures are then grouped using a histogramming technique and classified with a neural network similar to a MLP.

In Ref. [3] overtaking vehicles are detected by examining the temporal behaviour of the distribution of grey-values at the left and right border of the image. The detection is performed by template matching with previously stored rear views of vehicles. This technique is extended in Ref. [4] towards more distant and hardly moving vehicles by regarding vertical and horizontal edge configurations and symmetry properties. The tracking procedure is significantly stabilized by generating the correlation mask online, then scaling it according to the vehicle distance. Using a colour camera makes it possible to detect rear lights, to distinguish between pixels belonging to the road surface and those belonging to objects, and to recognize from the sky colour if the image has been acquired by day or at night.

In Ref. [25] overtaking vehicles are detected by applying the colour blob flow technique. This involves clustering image regions according to their colour after strongly reducing the number of colours in the image by a colour segmentation technique. Regions with a similar motion behaviour are grouped into object hypotheses. To enhance the stability of the object hypotheses, the clustering procedure is extended from colour space to colour-position space (colour cluster flow [26]).

3.2. Acquisition of the training data

The image data for the ATDNN-based vehicle detection system is produced by a standard greyscale video camera fixed behind the windscreen and directed slightly to the left, connected to a PC equipped with a framegrabber that delivers half frames of a resolution of 720×288 pixels. The input image sequences for the ATDNN are obtained by cropping a region of interest (ROI) sized 350×175 pixels out of the left half of each half frame. This ROI is then downsampled to $S_x^{(1)} \times S_y^{(1)} = 32 \times 32$ pixels. As an overtaking process takes about 1 s and the grabber hardware is able to grab, crop, and scale four ROIs per second, four subsequent ROIs are ordered into an image sequence, respectively, forming now an input pattern to the ATDNN. As this operation is performed at each time step of 0.25 s, two subsequent input patterns overlap by three images.

The composition of the training and the test set is shown in Table 1. An input pattern is said to display an overtaking vehicle if the vehicle has an apparent velocity of at least 1 pixel per time step and a minimum size of 10 pixels on the first image of the sequence. The collected garbage examples contain many different views of empty motorway lanes including various kinds of slopes and curves; they have furthermore been acquired under strongly varying weather conditions, including sunny weather, snow, rain and fog. Typical representatives of the two training classes are shown in Fig. 2. The test set has been acquired completely independent of the training set, i.e. several weeks later on different motorways under different visibility conditions.

3.3. Classification results

For the ATDNN, a configuration with $N_{\mathrm{RF}} = 2$ branches and spatio-temporal receptive fields of spatial size $R_x \times R_y = 15 \times 15$ pixels applied at a spatial offset of $D_x \times D_y = 8 \times 8$ pixels yields the best performance on the test set. Here, the number of temporally parallel sets of weights within the receptive fields is set to $R_t = R_h = 2$. In a first training scenario the initial values of the time delay parameters β and ρ are set to $\beta_1(0) = 1.8$ and $\rho_1(0) = 0.8$, in a second run we choose $\beta_2(0) = 0.8$ and $\rho_2(0) = 1.2$. In the first scenario, the time delay parameters converge toward $\beta_1(\alpha \to \infty) \approx 2$ and $\rho_1(\alpha \to \infty) \approx 1$, in the second scenario toward $\beta_2(\alpha \to \infty) \approx 1$ and $\rho_2(\alpha \to \infty) \approx 2$ (Fig. 3). The temporal extensions of the receptive fields of the ATDNN thus amount to $T_{\mathrm{eff},1}^{(1)} \approx 3$, $T_{\mathrm{eff},1}^{(2)} \approx 2$ in the first and to $T_{\mathrm{eff},2}^{(1)} \approx 2$, $T_{\mathrm{eff},2}^{(2)} \approx 3$ in the second training scenario. The corresponding ATDNN architectures are depicted in Fig. 4. According to Fig. 5, the performance of the ATDNN in the first training scenario is slightly higher than in the second. We thus retrained a TDNN with fixed time delays $\beta = \rho = 1$ and temporal receptive field extensions $R_t = 3$ and $R_h = 2$ chosen according to the first training scenario in order to avoid the necessity to perform interpolation (4) for each recognition procedure and to remove the temporal subsampling of the input pattern as apparent from Fig. 4a. This reduces the computational cost of processing an input

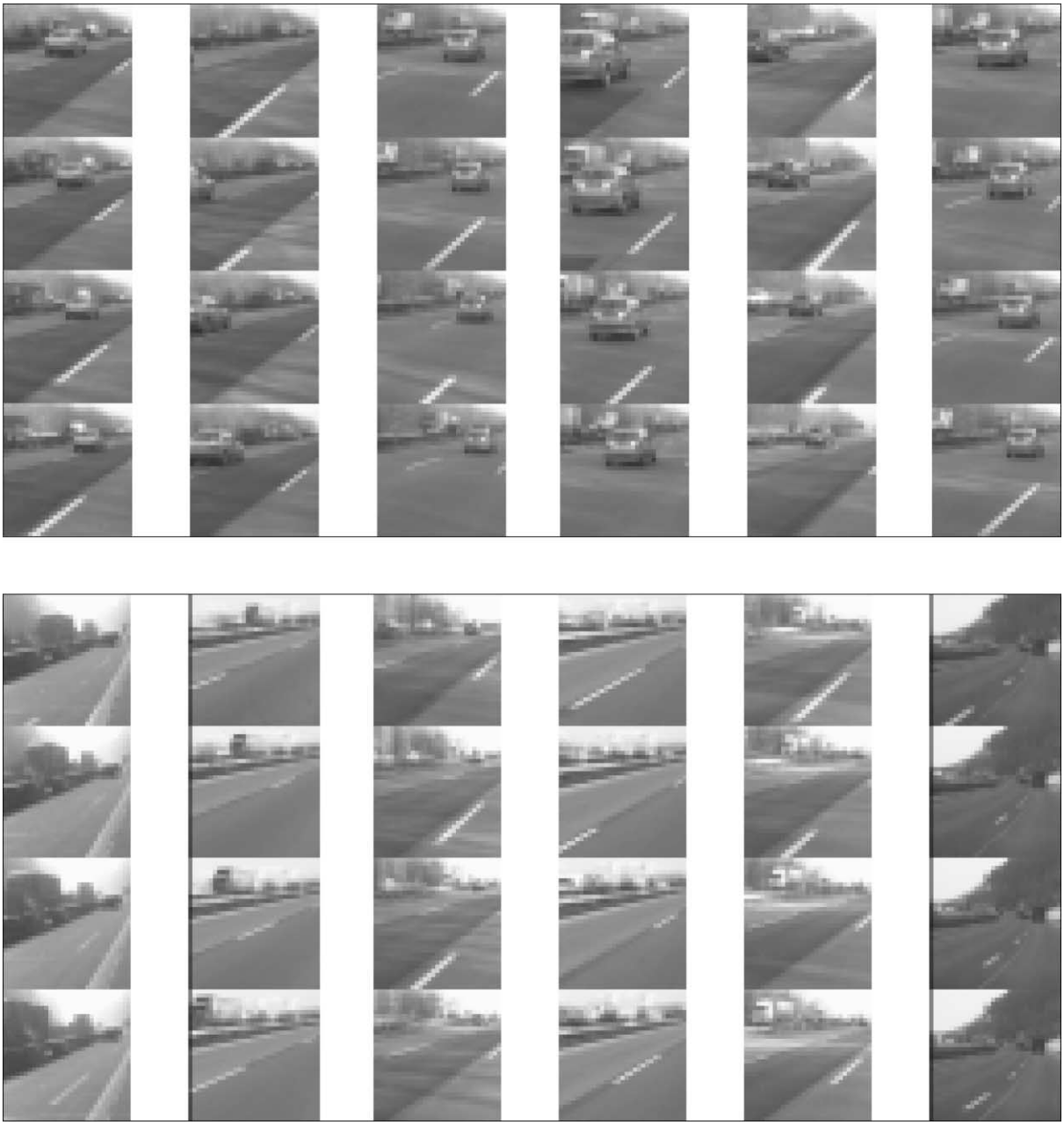

Fig. 2. Typical training sequences displaying overtaking vehicles (above) and garbage patterns (below).

pattern in recall mode by about a factor of two. The ROC curve of the corresponding TDNN is shown in Fig. 5.

3.4. Analysis of the temporal behaviour of the TDNN in decision space

At an acceptable false positive rate of 1 or 2 percent, the rate of detected overtaking vehicles per time step is rather low. We observed that the input sequences at the beginning of an overtaking process, with the vehicle just appearing and only partially visible, and at its end, with the vehicle covering only a few pixels and moving very slowly, is systematically recognized as a garbage pattern. A high rate of vehicles, however, is detected at least once during an overtaking process (Fig. 7). Using the activation patterns of the hidden neuron layer as input vectors for more complex classifiers like second-order polynomial classifiers or non-linear polynomial support vector machines yields no higher performance on the test set than that of the TDNN alone. On the contrary, we found that due to overfitting effects and systematic differences between the training set and the test set, the performance decreases with increasing complexity of the classifier. In this section we will therefore evaluate the dynamical behaviour of the TDNN in decision space to enhance its detection performance.

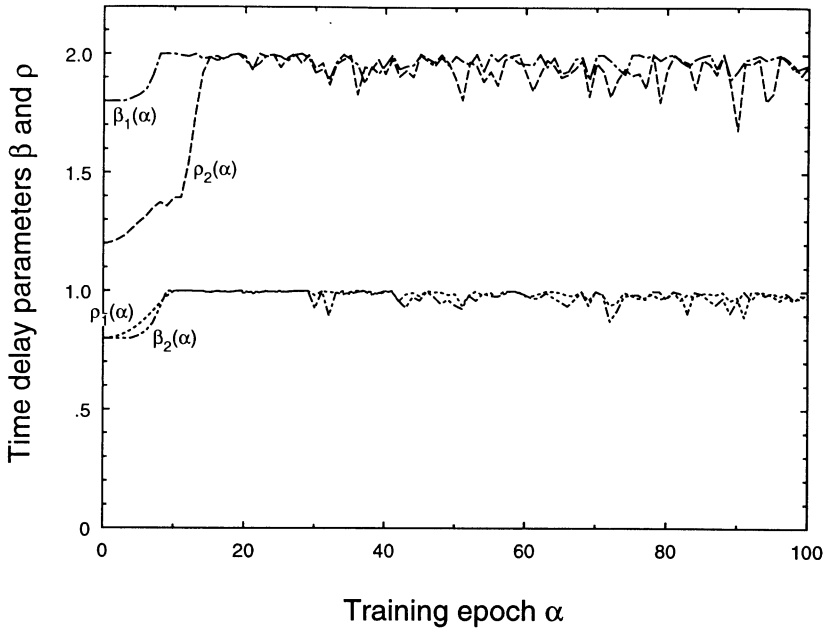

Fig. 3. Development of the time delay parameters β and ρ during the training process for both described training scenarios.

The first approach is to evaluate an appropriately averaged TDNN output. In the limit of an "ideal", i.e. infinite training set, the TDNN would act as an estimator for a posteriori probabilities and thus fulfil the condition $\omega_1 + \omega_2 = 1$ ([41]). As intermediate patterns of vehicles just entering or exiting the scene are rather rare, systematic deviations from the line $\omega_1 + \omega_2 = 1$ are most distinct in the corresponding intermediate region of decision space, leading to loops always performed in the same direction (Fig. 6b). We now neglect the two-dimensionality of the trajectory and only evaluate its projection $\omega = (1 + \omega_1 - \omega_2)/2$ on the line $\omega_1 + \omega_2 = 1$. We observed that the appearance of an impulse-like shape $\omega(t)$, $t = 1, \ldots, N_P$ more faithfully signals an overtaking vehicle than the $q\omega_1 > \omega_2$ condition, equivalent to $\omega > 1/(1 + q)$, that takes into account only one single network output; the occurrence of a loop or impulse may be evident even though the above conditions are not fulfilled. The length of the profiles was chosen to be $N_P = 8$ (Fig. 6c). The composition of the corresponding training and test set is given by Table 2. We classified the profiles with polynomial classifiers of degree 1–5 ([41]). The error rate on the test set is significantly decreasing with an increasing degree of the classifier for degrees smaller than 5, such that the fourth-order classifier was used to compute the corresponding ROC curve in Fig. 7.

We also evaluated the complete two-dimensional trajectory. In order to detect loop-like structures in output space as shown in Fig. 6b independent of their position and size which are largely variable, we regard the trajectory in output space as a sequence $c(t)$ of complex numbers at discrete time steps t with $c(t) = \omega_1(t) + i\omega_2(t)$. A Fourier transform (FFT) of an interval of N_P subsequent time steps should display strong high-frequency components in the case of a transition from the garbage toward the vehicle domain of the output space. To remove discontinuities at the borders of the interval, the sequence is mirrored according to $c(2N_P - t + 1) = c(t)$. The amplitude spectrum $A(k)$ of the sequence $c(t)$ nevertheless contains only N_P independent components as it can be shown that $A(N_P + 1) = 0$ and $A(k) = A(2N_P - k + 2)$ for $k = 2, \ldots, N_P$. It can be classified as representing loop or non-loop states. The phase spectrum turned out to contain no relevant information with respect to this classification task and was neglected. In this second approach, the performance of a linear polynomial classifier on the same training and test sequences as before could not be exceeded by higher-order polynomial classifiers or polynomial support vector machines.

Both proposed methods reduce the false positive rate by about an order of magnitude when the detection rate of complete overtaking processes is regarded (Fig. 7). The test set contains 150 complete overtaking processes. Here, the false positive rate denotes the fraction between the time during which an overtaking vehicle is erroneously detected and the time during which in fact no overtaking vehicle is present. Although the evaluation of the complete two-dimensional trajectory gives no better results than an evaluation of the one-dimensional $\omega(t)$ profiles, significantly fewer classifier coefficients have to be adapted as a linear classifier is already sufficient. We could show that no loss in performance on the test set occurs for the second method when only every 10th training pattern is used; a procedure that leads to strong overfitting effects when applied to the fourth-order classifier necessary to successfully employ the first method. Furthermore, the direction in which the loop is

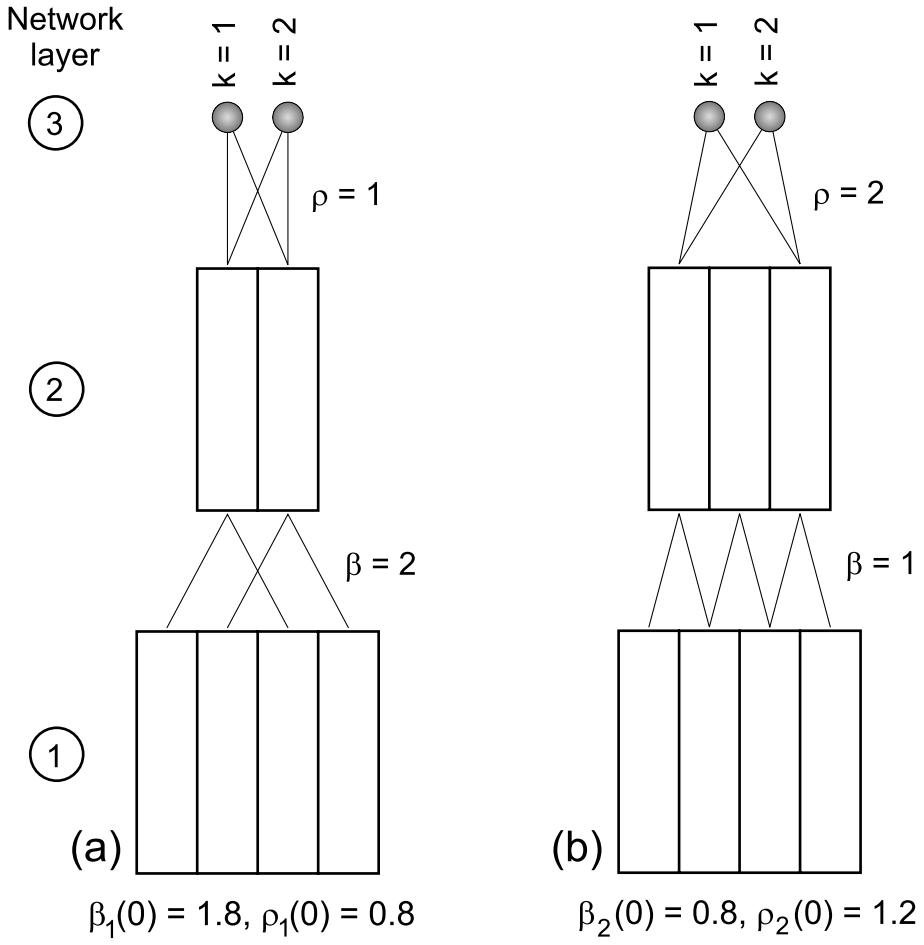

Fig. 4. ATDNN configuration after training in both described scenarios.

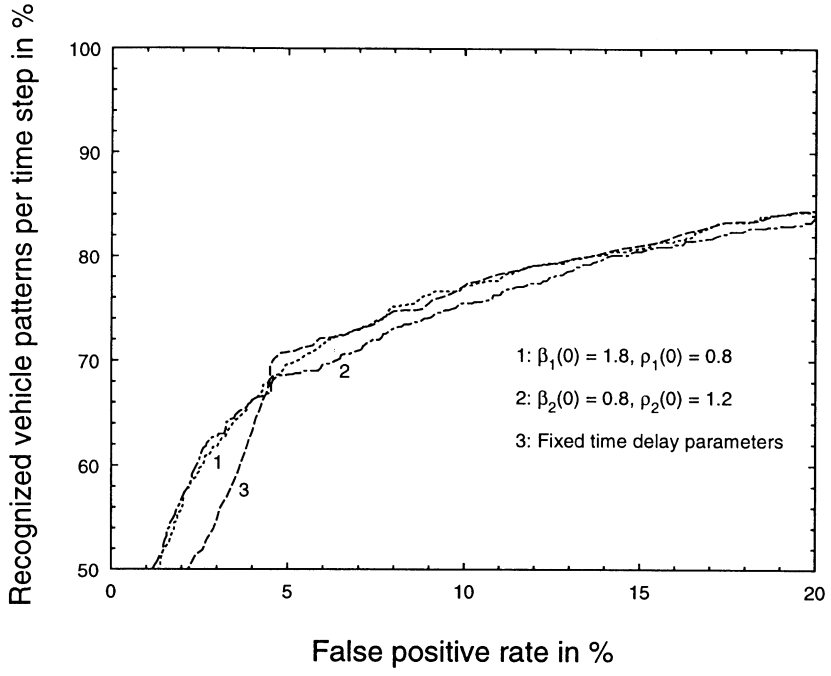

Fig. 5. ROC curves for both described training scenarios.

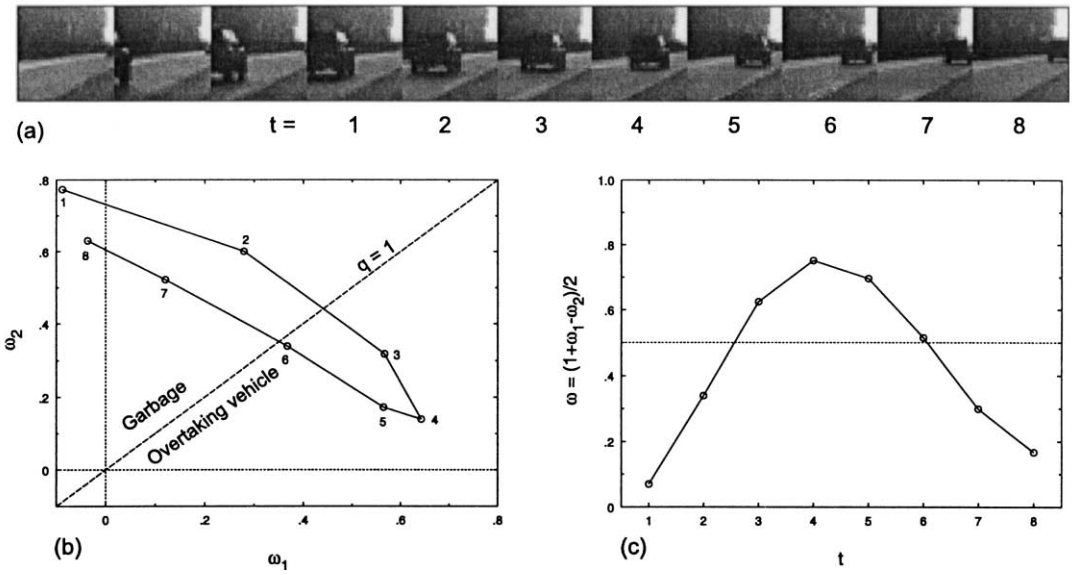

Fig. 6. (a) Image sequence displaying an overtaking vehicle. One time step corresponds to approximately 250 ms. (b) Trajectory of the TDNN output in decision space. (c) Impulse profile of the projection $\omega = (1 + \omega_1 - \omega_2)/2$ resulting from the same overtaking process.

performed might in principle be used to distinguish the direction of motion of the object in the scene.

Comparing our detection results to those obtained by applying other techniques is not at all straightforward as for a relevant comparison, one would actually need one common set of test data for all algorithms as a base for the comparison. Furthermore, even without such a common database, most authors give no explicit statements about the performance of their respective algorithms (cf. the references discussed in Section 3.1). We actually found only one contribution dealing with vehicle detection on motorways in which performance results are reported explicitly ([23]). The results refer to a 20 min run on a motorway under "typical" conditions during which all 170 vehicles appearing in the scene are detected. During that time 19 false alarms occur, i.e. about one per minute.

In our system, the false positives usually occur over time intervals of a duration of 1–2 s. A false positive rate of 2% thus corresponds to about one false alarm per minute in the sense of Ref. [23]. Our system then detects 96% of all overtaking vehicles; the missed hits occur due to extremely bad weather conditions (heavy rain and fog) in the correspond-

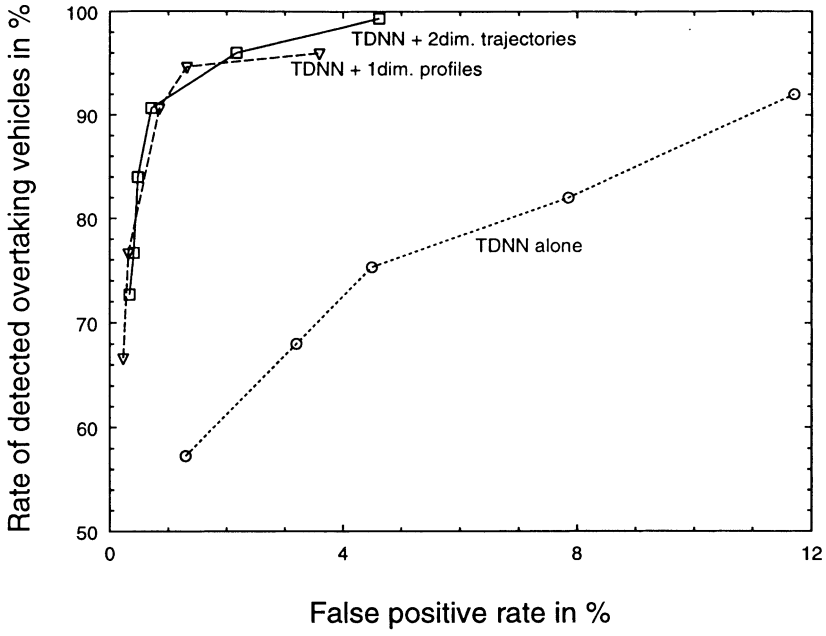

Fig. 7. Detection performance of the TDNN with and without the proposed second classification stages with respect to complete overtaking processes.

Table 2
Composition of the training and the test set for the detection of overtaking vehicles by analysis of the dynamical behaviour of the TDNN output in decision space

	Overtaking vehicles	Garbage	Sum
Training set	886	3073	3959
Test set	1283	3884	5167

ing situation. Under average weather conditions, i.e. no rain or fog, we observe a detection performance at least comparable to the one reported in Ref. [23]. This performance is achieved without any prior knowledge about the problem and especially without assigning the vehicle to a certain lane by means of a lane detection procedure, which is needed to apply the algorithm described in Ref. [23]. Furthermore, the computational complexity of our approach amounts to only about 25,000 floating point operations per classification procedure, which corresponds to a few milliseconds of CPU time on current standard PC hardware. Hence, the time needed to classify an input image sequence is nearly negligible compared to the time needed to grab, crop, and scale the input images.

3.5. Rough estimation of ego-position

To avoid false positives especially in the case of the ego-vehicle driving on the very left motorway lane such that it cannot be overtaken, we introduced a third training class "very left lane" to signal this situation. Such a module may be used independently of overtaking vehicle detection as a rough ego-position estimator; it does not aim, however, at an accurate determination of the road lanes as it would be necessary, e.g. for autonomous vehicle guidance. Such sophisticated systems are given in Ref. [15] and references therein. Our method for estimation of ego-position, though, is inspired by the kind of technique described, e.g. in Ref. [39], where a MLP receives the pixel values of a down-scaled image of the road in front of the ego-vehicle as an input and the network output determines the steering angle appropriate for following the road. This kind of approach is characterized by "low-level" input data, i.e. the raw pixel values obtained after scaling the image, and by a "high-level" output directly containing the desired information, indicating that this information has been obtained by training from examples only, without any prior knowledge about the problem.

In order to train situations with the ego-vehicle driving on the very left lane, we added 1260 corresponding training examples to the training set and 512 to the test set. Except increasing the number of training classes to $K = 3$, we did not change the TDNN architecture. The accordingly trained TDNN already recognizes rather faithfully if the ego-vehicle is driving on the left motorway lane (Fig. 9); it tends, however, to short and temporally randomly distributed drop-outs while driving on the left lane and signals every few minutes a wrong ego-position while driving on the right lane. To overcome this behaviour, we added a "memory" to the TDNN by implementing feedback loops according to Fig. 10. The network output is not fed back into neuron layer 1 or 2 as additional neurons in these layers do not fit with the blockwise processing mechanism of receptive fields. The construction shown in Fig. 10, however, corresponds to applying a recursive filter to the network output. It was not possible to train the feedback weights from examples as thousands of drop-out examples would be needed

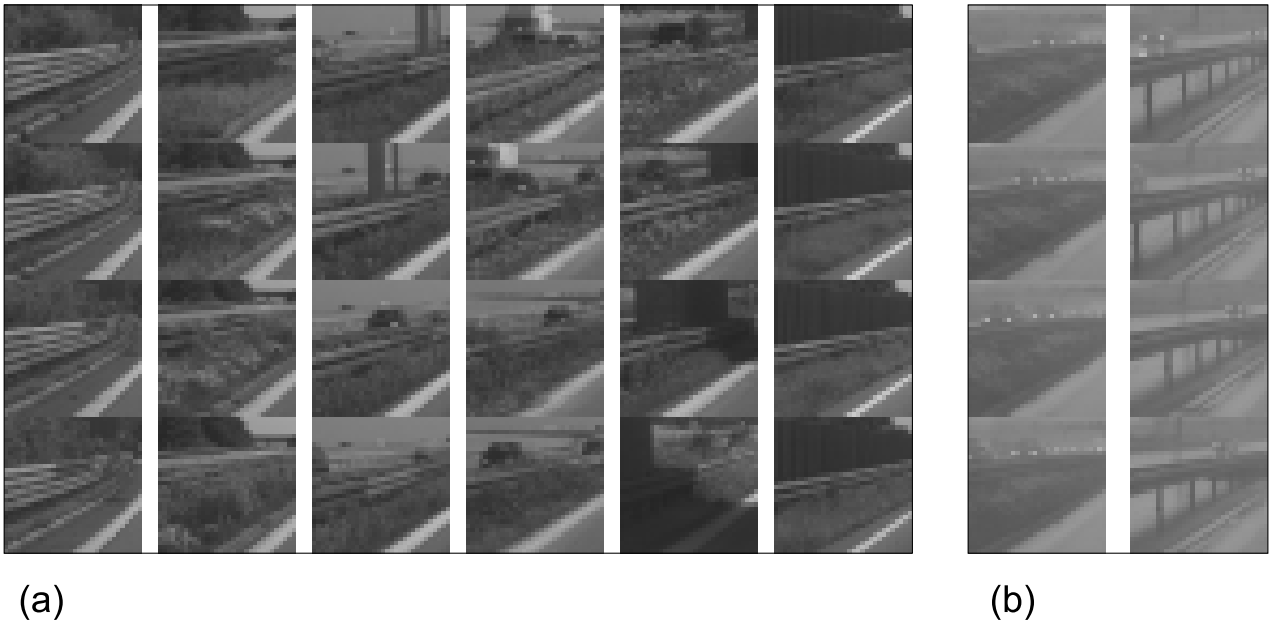

Fig. 8. (a) Typical training sequences acquired while driving on the very left motorway lane. (b) "Difficult" sequences correctly classified despite the solid white line being hardly visible.

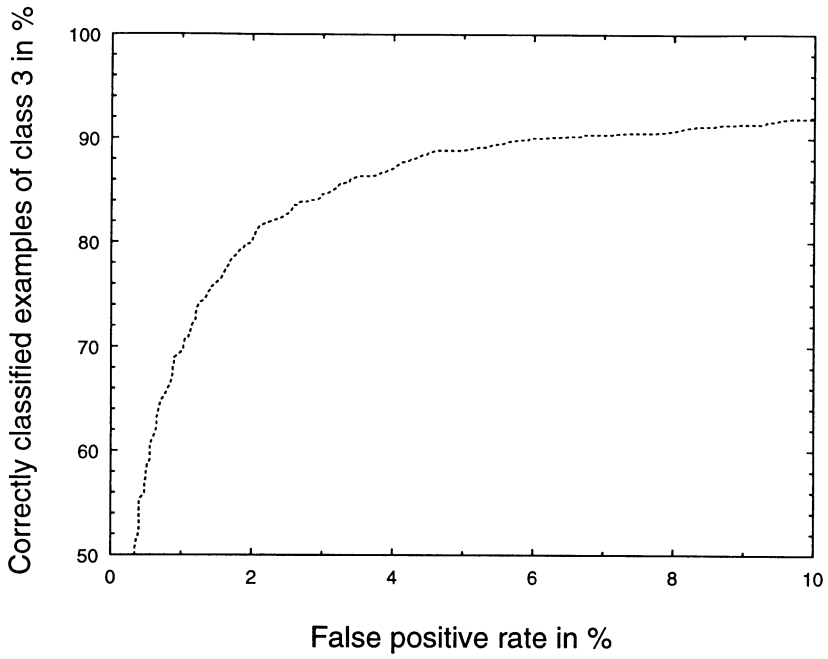

Fig. 9. ROC curve of the TDNN with $R_x = R_y = 15$, $R_t = 3$, $N_{\mathrm{RF}} = 2$, $R_h = 2$, $D_x = D_y = 8$ for the rough estimation of ego-position. A false positive denotes the TDNN erroneously signalling the ego-vehicle driving on the very left motorway lane.

which are rather difficult to obtain. Nevertheless, important properties of the feedback mechanism can be described analytically, i.e. under certain symmetry assumptions, the weights $\{f_{ij}\}$ can be chosen manually according to the recognition problem. Following Fig. 10, the recurrent network output is defined as

$$\vec{\delta}(t) = G\vec{\delta}(t-1) + A\vec{\omega}(t) \tag{14}$$

with $A_{ij} = a_i$ if $i = j$ and $A_{ij} = 0$ if $i \neq j$, $G_{ij} = f_{ij}$, $\vec{\omega}(t) = (\omega_1(t), \ldots, \omega_K(t))$, and $\vec{\delta}(t) = (\delta_1(t), \ldots, \delta_K(t))$. The output $\vec{\delta}(t)$ is evaluated in the same way as the pure feed-forward output $\vec{\omega}(t)$, as described in Section 2. Eq. (14) is a first order inhomogeneous difference equation the general solution $\vec{\delta}(t)$ of which is the sum of the general solution $\vec{\delta}^{(H)}(t)$ of the corresponding homogeneous difference equation and a special solution $\vec{\delta}^{(S)}(t)$ of the inhomogeneous Eq. (14). The solution of the homogeneous equation $\vec{\delta}(t) = G\vec{\delta}(t-1)$ is

$$\vec{\delta}^{(H)}(t) = G^t\vec{\delta}^{(H)}(0) = \sum_{k=1}^{K} b_k\lambda_k^t\vec{e}_k \tag{15}$$

with $\{\lambda_k\}$ as the eigenvalues and $\{\vec{e}_k\}$ as the eigenvectors of the matrix G of the feedback weights. The coefficients $\{b_k\}$ are determined by the initial conditions. The homogeneous solution $\vec{\delta}^{(H)}(t)$ is stable for

$$\gamma = \max_k \{|\lambda_k|\} \leq 1. \tag{16}$$

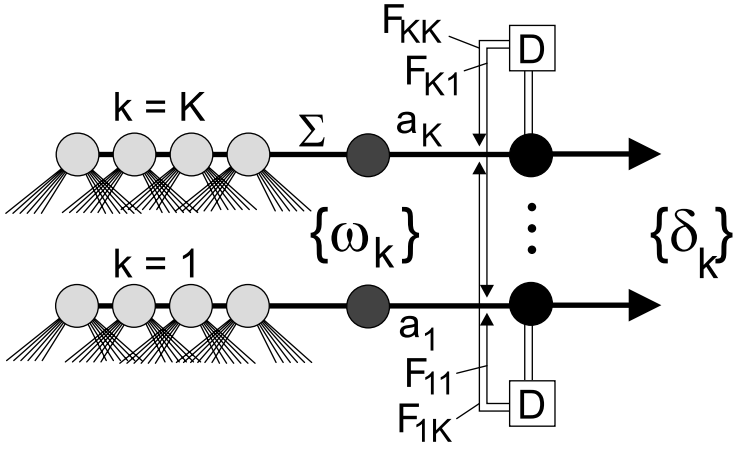

Fig. 10. Network layer 3 of the TDNN with feedback loops. The boxes marked with "D" denote a delay of one time step.

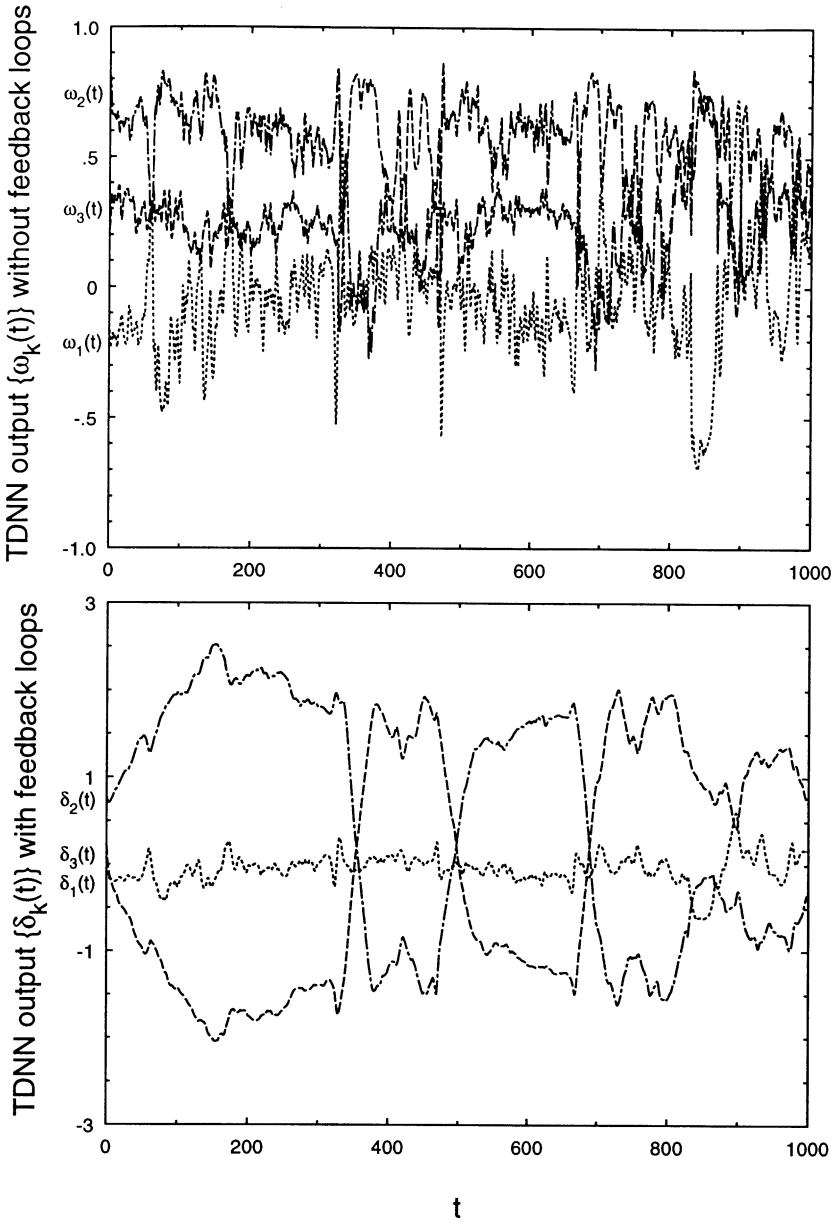

Fig. 11. Output of the TDNN for $K = 3$ training classes without feedback loops (above) and with feedback weights $g = 0.8$ and $f = -0.18$ (below). If the activation of output neuron 3 (dashed curve) is the highest one, the ego-position is on the left lane. The corresponding periods, in this example in the intervals $353 \leq t \leq 497$ and $t \geq 690$, are recognized with a significantly higher stability when feedback loops are applied.

If a solution obeys condition (16) and the eigenvalue with the largest absolute value γ is less than zero it is stable but shows a damped oscillatory behaviour. For time-independent $\vec{\omega}(t) = \vec{\omega}(0)$, the general solution of Eq. (14) is $\vec{\delta}(t) = G^t\vec{\delta}(0) + (1 - G)^{-1}(A\vec{\omega}(0))$; the system relaxes exponentially with the relaxation time $\nu = |1/\ln \gamma|$ toward the fixed point $(1 - G)^{-1}(A\vec{\omega}(0))$. The stability criterion (16), however, is valid independent of the form of the inhomogenity $A\vec{\omega}(t)$, i.e. our model has the important property that the feedback weights $\{f_{ij}\}$ have always to be chosen according to Eq. (16). Furthermore, the resulting relaxation time ν should be of the same order of magnitude as the typical duration of the drop-outs to be removed.

For our application, the feedback weights $\{f_{ij}\}$ are chosen such that the system tends to preserve a high activation of an output neuron. This corresponds to positive diagonal elements of the weight matrix G assumed to be identical for symmetry reasons. They are denoted by g. Furthermore, a high activation ω_3 of the output neuron signalling the ego-vehicle driving on the left lane is supposed to reduce the activation ω_2 of the neuron signalling a free left lane. The corresponding off-diagonal elements of G are thus set to a negative value f. The detection of an overtaking vehicle has to remain unaffected

by these interactions, such that all other elements of G are set to zero. We thus have

$$G = \begin{pmatrix} g & 0 & 0 \\ 0 & g & f \\ 0 & f & g \end{pmatrix} \qquad \text{with } g > 0 \text{ and } f < 0. \tag{17}$$

This implies

$$\gamma = \max_k \{|\lambda_k|\} = g - f \qquad \text{and } \nu = \frac{1}{|\ln(g-f)|}. \tag{18}$$

We usually set $g < 1$ and assume for the diagonal elements a_i of the matrix A the value $a_i = 1 - g$ similar to a first-order recursive low-pass filter. In the described recognition module we set $g = 0.8$ and $f = -0.18$, resulting in a relaxation time of $\nu = 49.5$ time steps. With these parameters no false positive occurs on the first 5250 examples of the test set, where the ego-position is always on the right lane; the false positive rate has thus dropped to well below 0.1%. Due to the rather long relaxation time ν the system needs up to some seconds to remark that the ego-vehicle has changed from the right to the left lane; after this has been achieved, however, there occur practically no more drop-outs. A typical example of this behaviour is presented in Fig. 11. In contrast to model-based methods ([10,11]) that require a solid white line to limit the left motorway lane, our method recognizes the very left lane correctly even if this line is not or hardly visible (Fig. 8).

The correspondingly configured recurrent TDNN is no more capable of faithfully detecting overtaking vehicles due to the short duration of an overtaking process, which amounts to only about one tenth of the relaxation time ν. In our test vehicle, we thus implemented a two-stage system in which the original TDNN with $K = 2$ training classes is continuously running, but a detection of an overtaking vehicle is only accepted if the recurrent TDNN trained to $K = 3$ classes signals the ego-position not to be on the very left motorway lane — this detection result is then transferred to the module that controls the autonomous driving procedure.

4. Recognition of pedestrians in the inner city scenario

In the context of vision-based driver assistance systems for the inner city environment the recognition of pedestrians is essential to avoid dangerous traffic situations. Pedestrians are non-rigid objects with a strong variability concerning colour, texture, shape, and size. Our idea is to classify walking pedestrians based on the characteristic criss-cross motion of their legs, thus taking into account shape and motion features simultaneously by analysing image sequences displaying the corresponding gait pattern using the ATDNN algorithm presented in Section 2.

4.1. State-of-the-art systems and techniques for pedestrian recognition

According to the detailed survey [19] most approaches can be divided into two-dimensional models with and without explicit shape models and three-dimensional models. In the context of the traffic scenario we will neglect the three-dimensional approaches as they mostly aim at recognizing not only the person as a whole but also body parts or gestures.

In Ref. [20] the chamfer matching algorithm based on template matching in distance-transformed binarized edge images is applied to pedestrian detection. The computational complexity is kept moderate by generating a template hierarchy during a training stage and by a coarse-to-fine approach.

Other algorithms make use of statistical shape models to detect and track persons. In Ref. [7] shape models defined by positions of feature points are obtained during a training stage. The set of parameters is then reduced by a PCA [9,41]. This helps to avoid parameter settings which are inconsistent with the training set. Occlusions, however, are not permitted as all features have to be available at all times.

In Ref. [2] the shape representation is given by B-Splines. Assuming a stationary camera, the person is separated from the background by analysing the difference image of two subsequent images.

In Ref. [15] the outline of about 1000 pedestrians on example images normalized in size is determined manually and subsequently smoothed by applying a two-dimensional average filter. A PCA allows a compact representation of this training set. Only the first eigenvectors and the mean represent the characteristic features of the pedestrian distribution. In order to find pedestrians in a test image, a gradient operator is applied to the image. The resulting normalized gradient image is then correlated with the first few eigenvectors of the pedestrian data set. One of the most significant advantages of pixel-based approaches such as that given in Refs. [15,20,36] is that they behave in a rather tolerant manner with respect to partial occlusions.

In Ref. [42] the average two-dimensional speed of pedestrians in a traffic scenario is estimated. Taking into account correlations between subsequent images of the sequence, a field of motion vectors is obtained that allows to group areas with similar properties into object hypotheses.

In Refs. [25–27] the colour cluster flow technique already mentioned in Section 3 is used for detection and tracking of persons with a moving camera. This method is rather robust with respect to partial occlusions.

In this paper, a stereo vision algorithm according to Ref. [17] will be used as a preliminary detection stage to extract potential pedestrians from the scene. In Ref. [36] a classification-based pedestrian detection algorithm is described. The image is divided into overlapping windows which undergo a Haar wavelet transformation. The corresponding wavelet coefficients are obtained by applying differential operators of different scale and orientation to different positions of the image window. From this eventually rather large set of coefficients a small number of "relevant" coefficients

is selected manually based on their absolute value and their local distribution in the image window. This reduced set of features is classified with a support vector machine (SVM, cf. e.g. Ref. [40]). In Ref. [38] temporal sequences of two-dimensional Haar wavelet features are grouped into feature vectors of a dimension of up to 6000, which yields a better recognition performance compared to the single image approach. Both approaches suffer from a rather high computational complexity as the SVM classifier has to be applied up to several 10^4 times per image.

Another class of techniques for the recognition of pedestrians are model-based ones, which require explicit knowledge about what a person looks like. As problems arise with respect to occlusions of body parts many systems require additional knowledge about the kind of motion to be detected and/or the camera position. Generally, the persons are segmented from the background by subtraction, which assumes a stationary camera and a constant or slowly varying background. Many body models consist of stick figures with the body parts being approximated by ellipsoids. Most contributions focus on pedestrians walking parallel to the image plane.

In Ref. [34] the image sequence is sliced parallel to the *xt* plane; when a walking pedestrian occurs, the intensity profiles in this plane show characteristic wave-like structures. A parameterized deformable contour model is fitted to these structures, and a sufficiently high correspondence with the model implies the appearance of a walking pedestrian. This concept is extended in Ref. [35] toward the complete three-dimensional *xyt* space by fitting a parameterized deformable surface model to the three-dimensional patterns.

In Ref. [24] the body parts of the pedestrian are approximated by a stick model. The mutual orientation of the parts of the model are used to recognize certain kinds of motion. In Ref. [6] the arms and legs of a person are detected as pairs of antiparallel lines.

In Ref. [1] it is assumed that the motion of the person is given by a set of representative stick model configurations. In the original image, the person appears dark in front of a bright background.

To recognize different kinds of motion without a-priori knowledge the arms and legs of a person are detected and tracked as antiparallel line pairs in Ref. [33]. Corresponding techniques to deal with occlusions are as well presented.

A region-based approach is presented in Ref. [55]. The body parts are described by a statistical model of gaussian distributions in a five-dimensional *xyYUV* colour-position space. For the background a statistical model in colour space is assumed. In an initialization step all pixels which are inconsistent with the distribution of background pixels are assumed to belong to the person. The models for the colour regions out of which the person consists are constructed by means of a two-dimensional contour analysis based on heuristic criteria that aims at identifying the body parts. In a subsequent tracking stage, the appearance of the person is predicted for the next image. Each pixel is then assigned to one of the region models or to the background, and the statistical models are updated.

In the multiple-cue approach described in Ref. [29] the features intensity, edges, distance, and motion are used to detect persons which are standing or walking parallel to the image plane. This approach is "object oriented" in the sense that for a certain application generic objects like, e.g. person, background, floor as well as corresponding methods are defined to detect these objects in the image. The objects are instantiated if some of their properties have been extracted from the image such that further, more specialized methods can be applied.

Most of the described techniques have in common that they actually do not try to recognize an unknown detected object as a person or to reject it as a non-person, but they tend to regard an arbitrary detected object immediately as a person just after checking some rather general properties like, e.g. motion behaviour (exceptions are the systems for pedestrian recognition described in Refs. [20,36,38]). Body models are fitted directly to the detected objects. In the traffic scenario, however, it is not necessary to determine body pose or gesture; only a rough determination of the motion behaviour may be useful with respect to collision avoidance. It is more important to distinguish between pedestrians and arbitrary obstacles in order to activate corresponding safety systems. The ATDNN-based system for pedestrian recognition described in Section 4 has specially been designed to deal with this problem.

4.2. Acquisition of the training data

In the ATDNN-based pedestrian recognition system, the objects in the scene are segmented by a real-time stereo vision algorithm described in detail in Refs. [17,52] which also works in the presence of a moving background. After detection of an object spatially emerging from the background, which is assumed to be flat, we crop the lower half of the ROI delivered by the stereo vision algorithm, which contains the pedestrian's legs, and normalize it to a size of 24×24 pixels. The stereo vision algorithm is capable of evaluating every third acquired pair of PAL/CCIR video frames, which corresponds to a time step of 120 ms between two subsequent frames of the sequence. Eight subsequent normalized ROIs are ordered into an image sequence covering a temporal range thus approximately corresponding to one walking step. After each stereo detection procedure, the batch of images is shifted backward by one image, discarding the "oldest" image while placing the new image at the first position, resulting in an overlap of seven images between two sequences acquired at subsequent time steps. By a tracking algorithm based on a Kalman filter framework which is combined with the stereo vision algorithm it is guaranteed that each image of an input sequence displays the same object. Our aim is to distinguish between pedestrian and non-pedestrian ("garbage")

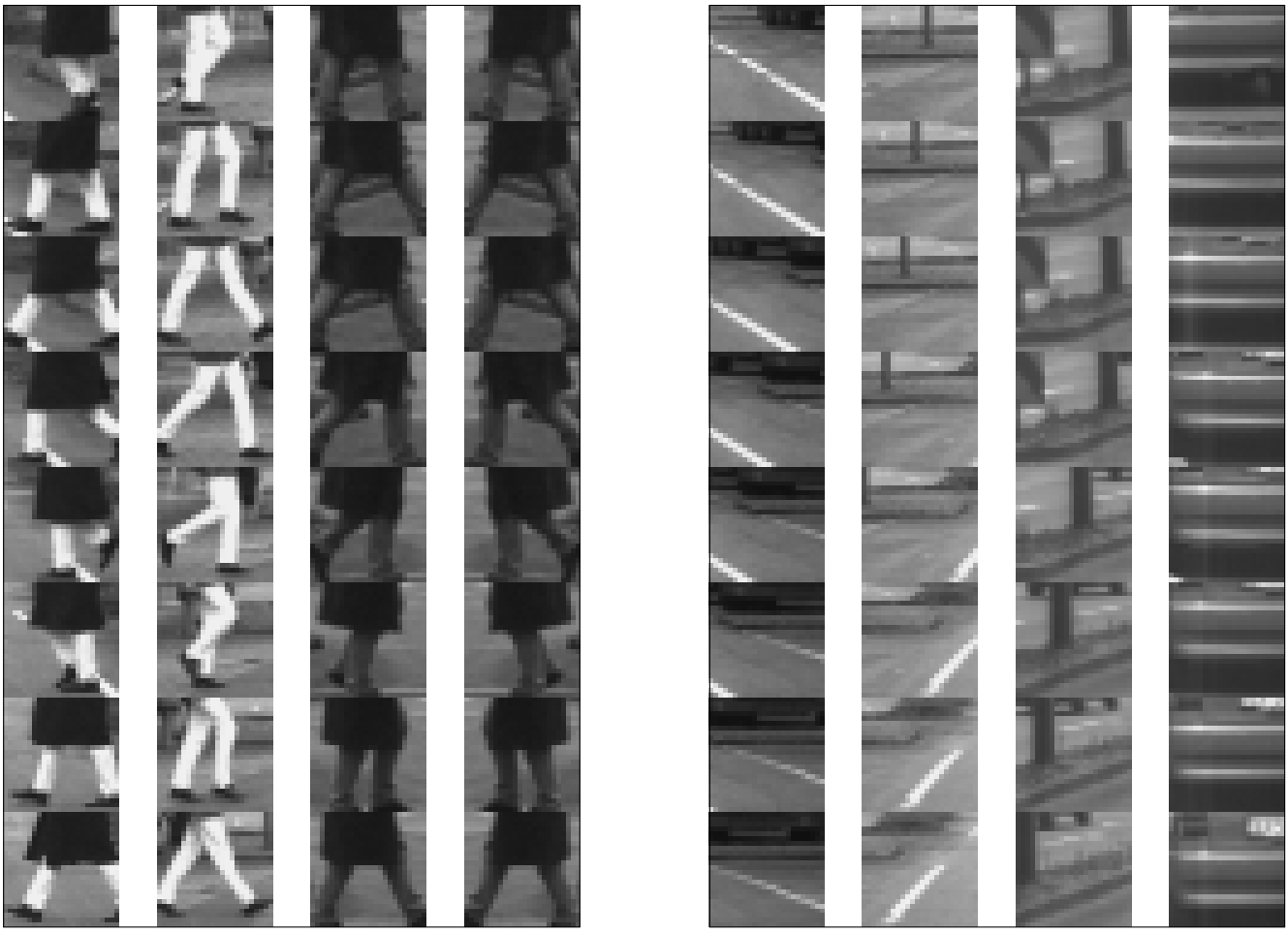

Fig. 12. Typical pedestrian (left) and garbage patterns (right).

patterns, resulting in $K = 2$ training classes typical representatives of which are shown in Fig. 12.

4.3. Classification results

The composition of the training set is shown in Table 3. It turned out that the performance on the test set was best for $N_{\mathrm{RF}} = 2$ network branches and spatio-temporal fields of a spatial size of $R_x = R_y = 9$ pixels, applied at an offset of $D_x = D_y = 5$ pixels. To achieve acceptable computation times, we first set the number of temporally parallel sets of weights within the receptive fields of the ATDNN, R_t and R_h, to the minimum required value of $R_t = R_h = 2$, performing four training runs with this setting of architecture parameters but different initial configurations of time delay parameters β and ρ. In addition to that, we performed three training runs with $R_t = 3$ and $R_h = 2$. Larger values of R_t or R_h are not tractable as the computation time of a training process would then exceed 500 h on standard hardware, like e.g. a Pentium II processor. The temporal behaviour of the resulting temporal extensions $T_{\mathrm{eff}}^{(1)}$ and $T_{\mathrm{eff}}^{(2)}$ of the receptive fields of the ATDNN is shown in Fig. 13. It becomes clear that during training, $T_{\mathrm{eff}}^{(1)}$ converges either toward 1 or to a value between 4.5 and 5.5, while $T_{\mathrm{eff}}^{(2)}$ obtains the approximate values 1, 3 or 5. We essentially have four "optimal" combinations of $T_{\mathrm{eff}}^{(1)}$ and $T_{\mathrm{eff}}^{(2)}$; the resulting ATDNN architectures are depicted in Fig. 14.

Table 3
Composition of the training and the test set for the recognition of pedestrians

	Pedestrian patterns	Garbage	Sum
Training set	3926	4426	8352
Test set	1000	1200	2200

The recognition performance of these ATDNNs more significantly depends on the value of R_t than on the combination of $T_{\mathrm{eff}}^{(1)}$ and $T_{\mathrm{eff}}^{(2)}$, respectively. For $R_t = 2$ we have a much stronger temporal subsampling in the input layer of the ATDNN than for $R_t = 3$, resulting in a slightly higher recognition performance in the latter case. Obviously, for the configuration $T_{\mathrm{eff}}^{(1)} = T_{\mathrm{eff}}^{(2)} = 1$ (training run 2) the lack of a temporal receptive field structure significantly reduces the recognition performance, compared to the other configurations where a temporal receptive field structure is present in the ATDNN.

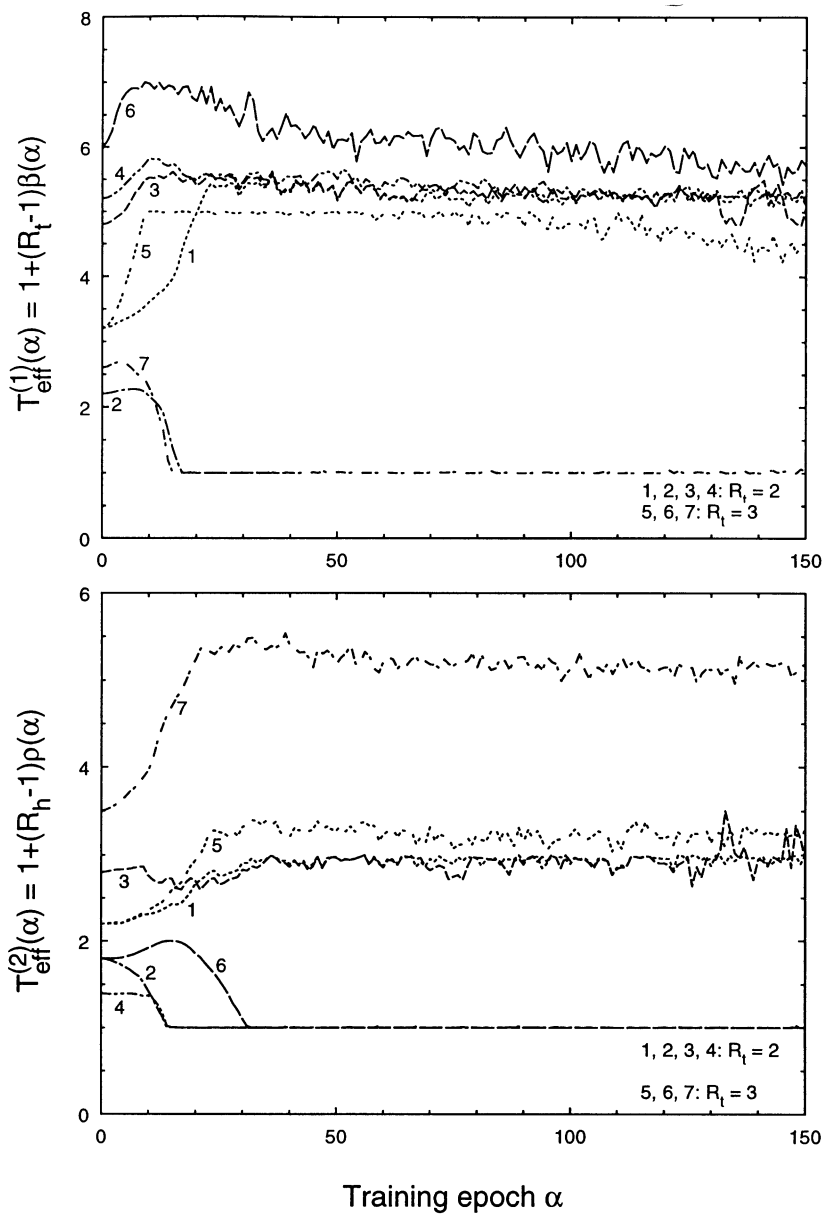

Fig. 13. Development of the effective temporal extensions $T^{(1)}_{\mathrm{eff}}$ and $T^{(2)}_{\mathrm{eff}}$ of the receptive fields of the ATDNN during training. The curves are denoted by the number of the respective training run.

From the practical point of view, 1% false positives are tolerable in the recognition module. The corresponding rate of correctly recognized pedestrian patterns is highest for training runs 5, 6 and 7, where we have chosen $R_t = 3$. Note that for our data, false positive rates of less than 1% are statistically rather meaningless as the absolute number of false positive examples in the test set is then smaller than 12.

The computation time needed for the classification of an input pattern amounts to about 20 ms on our Pentium II processor. Especially when the stereo vision algorithm detects several objects in the scene, there is a need to reduce computation time in order not to exceed 120 ms per cycle for the detection and classification procedure, as otherwise, reliably tracking the objects will become more and more difficult. We thus trained a TDNN with fixed time delay parameters $\beta = \rho = 1$ and temporal receptive field extensions $T^{(1)}_{\mathrm{eff}} = 5$ and $T^{(2)}_{\mathrm{eff}} = 3$, equivalent to $R_t = 5$ and $R_h = 3$, derived from the configuration on the upper right in Fig. 14. In practice, the resulting small loss in recognition performance compared to the full ATDNN architecture (Fig. 15) is irrelevant, taking into account that the computation time can be reduced by about a factor of 2.

The recognition results mean in practice that in scenes of a length of some seconds, pedestrians are recognized with a rather high stability. Typical scenes are shown in Fig. 16.

However, there may be drop-outs if the contrast between the legs of the pedestrian and the ground is too low or the

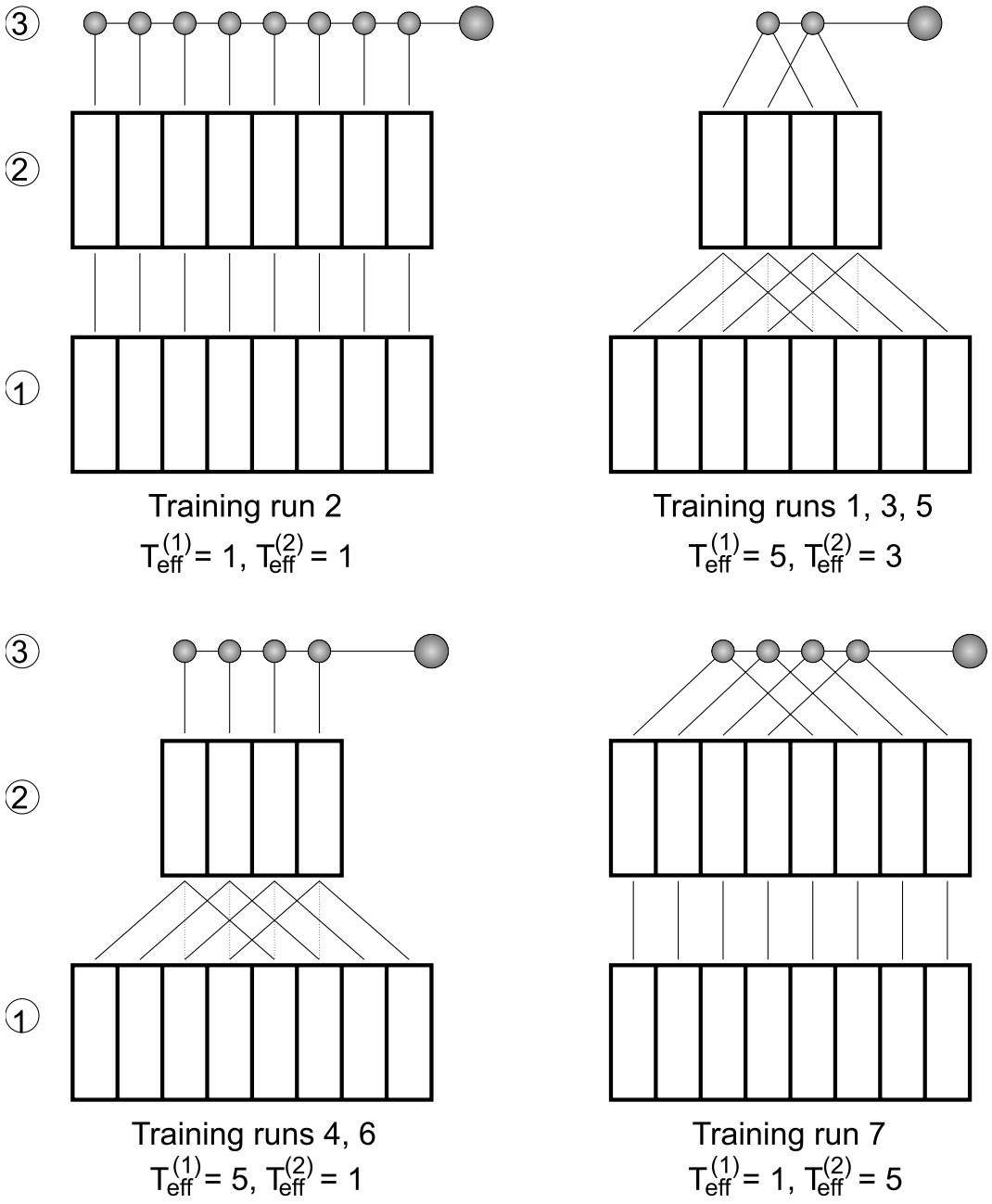

Fig. 14. ATDNN architectures corresponding to the approximate values of $T_{\mathrm{eff}}^{(1)}$ and $T_{\mathrm{eff}}^{(2)}$, resulting from the seven performed training runs. The connections drawn as dashed lines only exist for $R_t = 3$.

pedestrian is imprecisely segmented by the stereo vision algorithm. The recognition result is then strongly confirmed if a certain object detected by the stereo vision algorithm has been determined to represent a pedestrian at several subsequent time steps. In many cases, such drop-outs can be removed by applying feedback loops to the TDNN as described in Section 3.5 (Fig. 17). For the implemented relaxation time of $\nu = 4.48$ time steps of the recurrent system, however, the pedestrian is recognized some 200 ms later than without feedback loops. Our database contains six such "difficult" sequences in which either tracking is imprecise due to a sudden rotation of the ego-vehicle or the pedestrian changing its direction of motion, or the contrast between pedestrian and background tends to be rather low. Without feedback loops, the pedestrian is correctly recognized during 79.0% of the period in which it is apparent on these sequences. With feedback loops, this fraction rises to 88.3% despite the recognition constantly setting in some 200 ms later — essentially, the drop-outs after inset of recognition could be removed.

The ATDNN-based system for pedestrian recognition is specially designed for laterally walking pedestrians as they

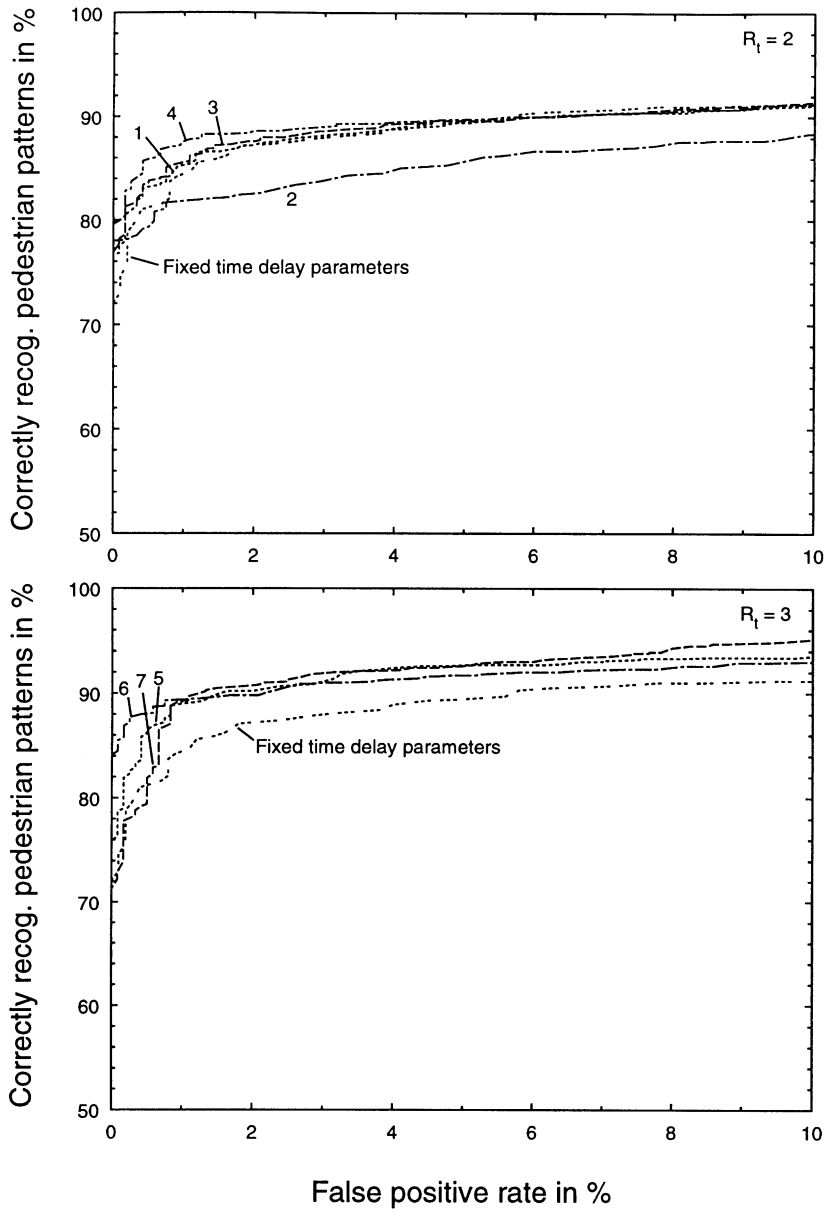

Fig. 15. ROC curves on the test set for the ATDNN, with $R_t = 2$ (above) and $R_t = 3$ (below). The ROC curve for the TDNN with $R_t = 5$ and $R_h = 3$ and fixed time delay parameters $\beta = \rho = 1$ is shown for comparison.

may cause the most dangerous traffic situations when, e.g. crossing the street in front of the ego-vehicle (Fig. 16). We observed that the system behaves quite robust with respect to different walking speeds and directions. It correctly recognizes running pedestrians and allows for walking directions from the interval between 45 and 135 degrees with respect to the line of sight. Possibilities to recognize standing pedestrians, e.g. waiting at a zebra crossing or pedestrians directly approaching the camera such that no gait pattern is visible are to train the ATDNN with image sequences displaying such essentially stationary examples or to combine it with a classifier for single images or a shape-based technique like, e.g. the Chamfer matching algorithm [20].

For comparison to state-of-the-art techniques for pedestrian recognition again the difficulty occurs that hardly any explicit performance values are reported in the literature (cf. references discussed in Section 4.1). The authors of Refs. [36–38] show ROC curves of their pedestrian recognition system based on classification by SVMs. When comparing their ROC curves to the ones shown in this contribution, one should keep in mind that the test sets used in this paper for measuring the classification performance consist of image regions that contain an object that has been previously segmented from the scene by means of stereo vision. In contrast to this, the classification performance is evaluated in Refs. [36–38] with respect to

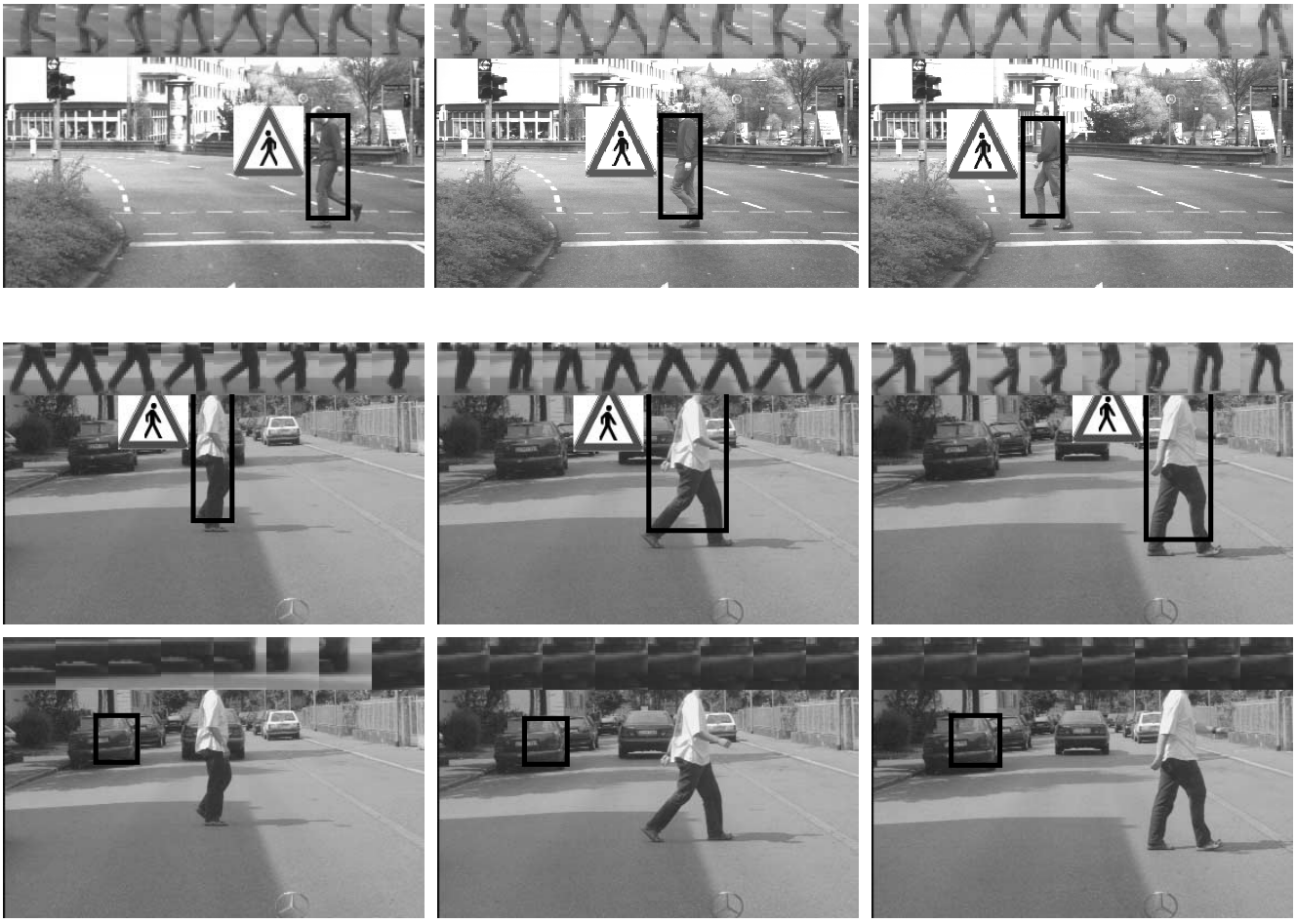

Fig. 16. Recognition of pedestrians in an urban traffic environment. The black bounding boxes have been determined by the stereo vision algorithm for the left stereo image, respectively. In the upper part of the image, the input of the ATDNN, i.e. the scaled ROI on the current and on the seven preceding images is shown. On the second sequence, a pedestrian and a garbage pattern are detected simultaneously.

arbitrary image parts which may also contain essentially unstructured regions such as road surfaces or the sky. In these regions the danger of confusing them with a pedestrian is very low. If the classifier is now applied to a grid uniformly covering all image parts, as it is the case in Refs. [36–38], the rate of such "simple" test samples is of course much higher than for a test set preselected by a preliminary detection procedure. This explains the fact that the false positive rates stated in Refs. [36–38] are lower by one order of magnitude when compared to our values, at a similar rate of correctly recognized pedestrians. We thus expect the false positive rate for the systems described in Refs. [36–38] to be at least of the same order of magnitude as our values when confronted with "real" object candidates selected by a preliminary detection stage such as stereo vision. Furthermore, especially the SVM-based technique for pedestrian recognition on image sequences described in Ref. [38] implies a high computational complexity resulting from the high dimension of the feature vectors (cf. also Section 4.5).

4.4. Combination of the TDNN with polynomial support vector machines

In this section, we will combine the previously trained TDNN with a polynomial support vector machine (SVM, cf. e.g. Ref. [40]). According to Ref. [53], we use the activation pattern of the second network layer of the TDNN, which is of dimension 128 in the configuration described above, as an input for the SVM. This technique is an alternative to the well-known PCA [9,41] algorithm for dimensionality reduction, which scales like $\mathcal{O}(D^3)$ and therefore leads to problems concerning computational complexity, given the dimension $D = 24 \times 24 \times 8 = 4608$ of the feature space regarded in this application. For comparison, we will carry out a direct classification of the image sequences by polynomial SVMs with and without dimensionality reduction by PCA in Section 4.5.

Effectively, the TDNN performs in this configuration a computationally very efficient dimensionality reduction — ten times faster than standard PCA — of the input image sequences, taking into account their spatio-temporal structure by means of its spatio-temporal receptive fields. We trained a

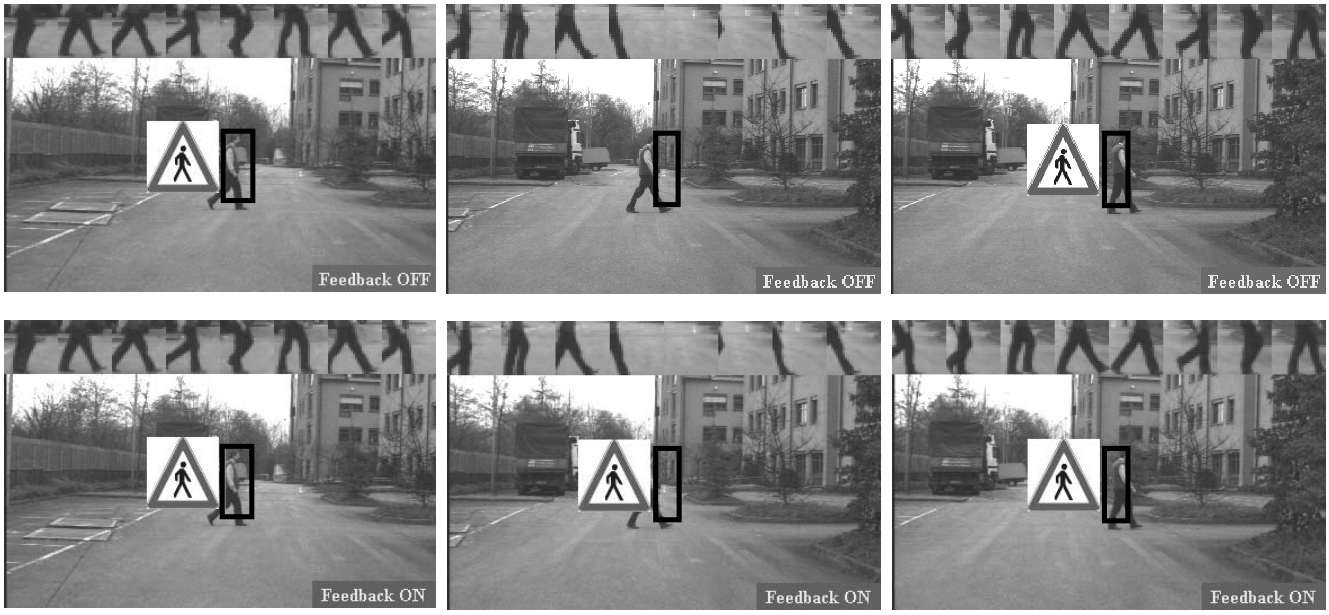

Fig. 17. Above: Without feedback loops, the pedestrian is not recognized on the second image as the prediction of the Kalman filter differs significantly from the true position of the pedestrian due to a sudden rotation of the ego-vehicle. Below: The bounding box is still the same in the second image, but the drop-out of the recognition due to its misplacement can be removed by applying feedback loops to the network output that include the previous recognition results into the current network output.

second, third, fourth, and fifth order polynomial SVM as well as a linear polynomial classifier for comparison on the 128-dimensional feature vectors. Due to the still quite large overlap between the two classes it was of no use adapting a linear SVM as in this case, the large majority of support vectors turned out to be slack variables. The SVMs of order three and higher, however, separate the two training classes perfectly on the training set. Fig. 18 shows that the performance of the TDNN alone is still better than that of the linear polynomial classifier due to the temporal receptive fields between the second and the third TDNN layer offering additional degrees of freedom. The performance is then rising with increasing order of the SVM, converging for orders higher than about three. At 1% false positive rate, the highest obtained rate of correctly recognized pedestrian patterns is 92% for the combined TDNN + SVM classifier, compared to 89%

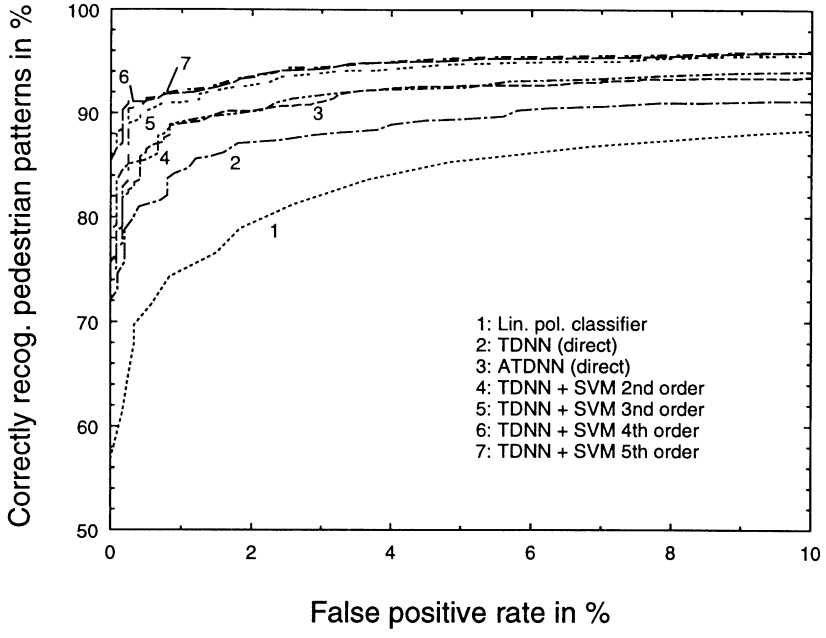

Fig. 18. ROC curves for several classifiers trained on the features obtained by dimensionality reduction with the TDNN.

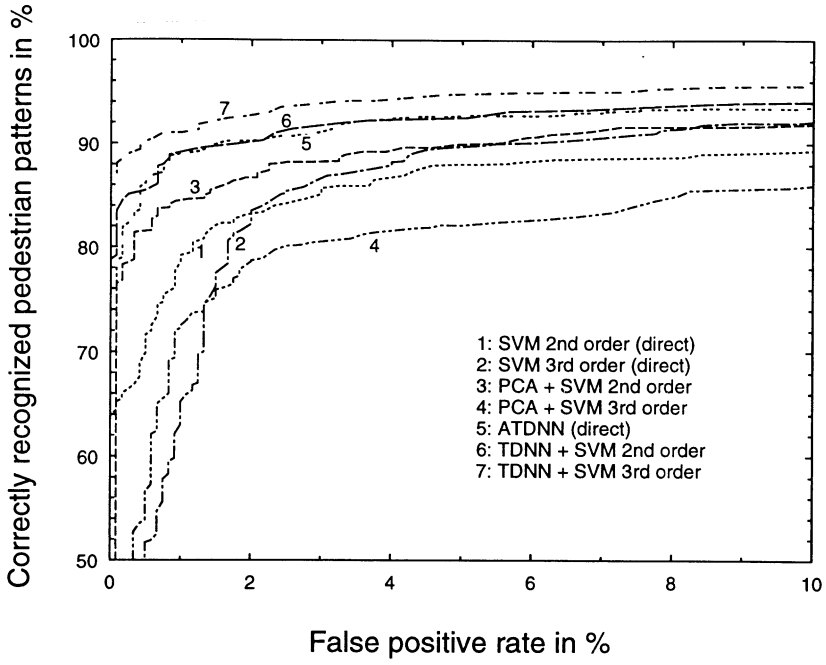

Fig. 19. ROC curves of the described classification modules on the test set for pedestrian recognition.

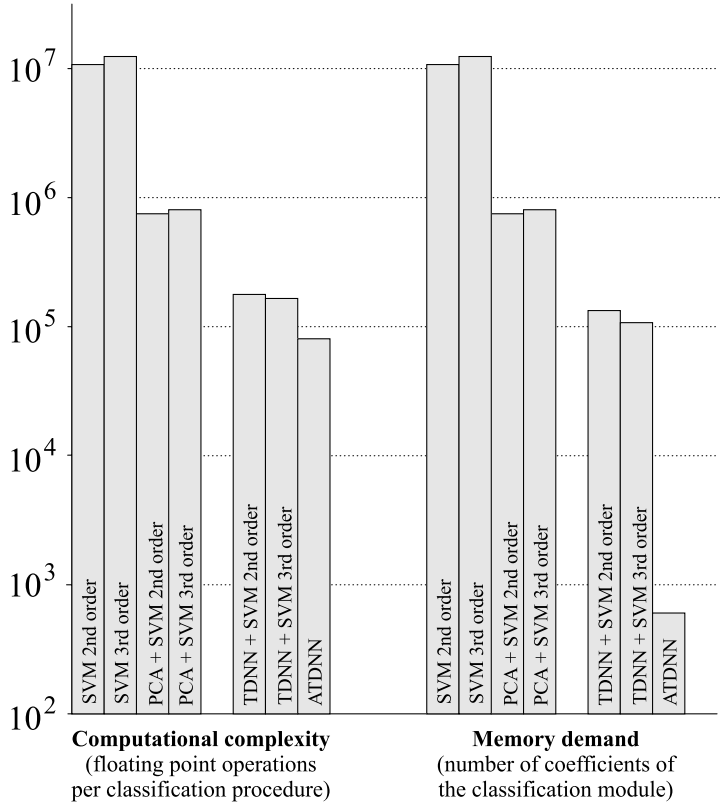

Fig. 20. Computational complexity (left) and memory demand (right) of the classification modules for pedestrian recognition. Note that the y axis is logarithmic.

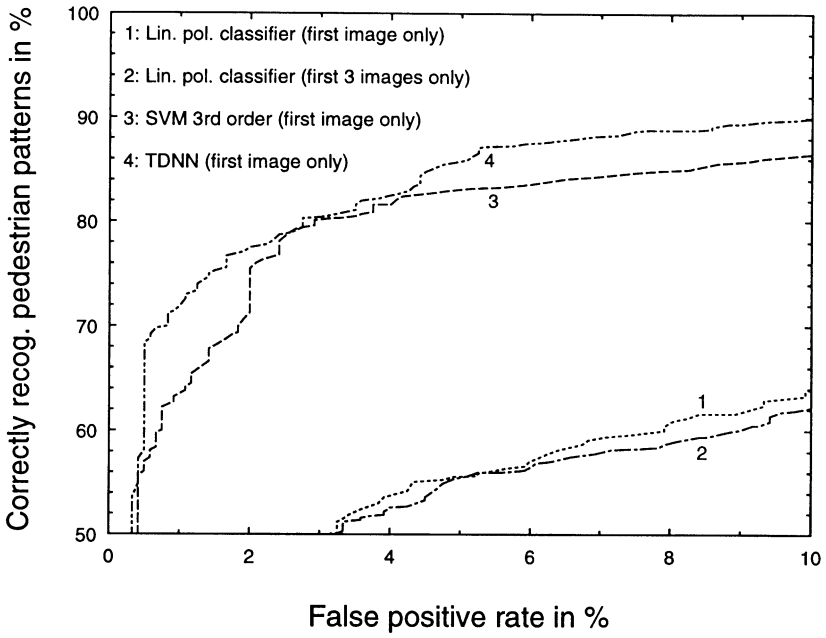

Fig. 21. ROC curves on the test set for several classifiers trained on the first image or the first three images of the original sequences, respectively. The TDNN (curve 4) was applied in the parameter setting $R_x = R_y = 9$, $R_t = 1$, $N_{RF} = 2$, $R_h = 1$, $D_x = D_y = 5$.

obtained in Section 4.3 with the ATDNN alone. A further advantage especially for applying the classification method in practice is that the combined classifier shows a more robust behaviour with respect to garbage patterns looking similar to gait patterns, e.g. displaying horizontally moving, vertically elongated structures such as traffic sign posts.

4.5. Comparison to direct classification by polynomial SVMs

A relevant comparison between different algorithms can only be performed on a common set of test data. Based on our previously used training and test data, we thus compare the ATDNN approach to polynomial SVMs applied directly and after dimensionality reduction by PCA. In this context of "global" classification not explicitly taking into account the spatio-temporal structure of the data, an input image sequence is simply regarded as a vector of pixel intensities.

For direct classification by a polynomial SVM, the input vector has a dimension of $24 \times 24 \times 8 = 4608$. This approach is related to the one described in Ref. [38], that combines temporal sequences of two-dimensional wavelet features into a single high-dimensional vector which is then classified by an SVM, not explicitly taking into account temporal relationships between the features. We adapted a second and third order polynomial SVM to the training set which could be perfectly separated in both cases. The results on the test set are shown in Figs. 19 and 20, therein marked as "SVM".

To reduce the dimension of the input patterns to be classified, we furthermore employ the technique of standard PCA. As the covariance matrix involved turned out to be too large to be diagonalized by usual numerical methods like, e.g. the Jacobi algorithm, we use the well-known unsupervised perceptron learning technique known as "Oja's rule" [9] to compute the normalized eigenvectors of our training set of image sequences. Oja's rule yields the eigenvector of the covariance matrix of a distribution of training examples that belongs to the largest eigenvalue without the necessity to explicitly compute the matrix elements. Once this most significant eigenvector of the covariance matrix has been determined, all training patterns are projected onto a subspace that is orthogonal to this eigenvector, in which the next, less significant eigenvector is determined by again applying Oja's rule, and so on. According to Refs. [9,41], the feature vector of a new, unknown input vector is obtained by transforming it into the space spanned by the eigenvectors of the covariance matrix, omitting the dimensions represented by the least significant eigenvectors. We took into account only the subspace of the $D_{red} = 128$ most significant eigenvectors and adapted a second and third order polynomial SVM to the correspondingly transformed training set, which again led to perfect separation in both cases. For the second order SVM, the performance even rises compared to direct classification although the number of features is reduced by 97.2% — we found, however, that further reducing the number D_{red} of features to below 128 immediately results in a loss of performance. The results with $D_{red} = 128$ are marked as "PCA + SVM" in Figs. 19 and 20. The ATDNN-based classification approaches are somewhat superior to the global SVM approaches with respect to their recognition performance on the test set (Fig. 19). Regarding computational complexity and memory demand, values which are of high interest when it is

intended to integrate the classification modules into real-time vision systems running on hardware with limited resources, the ATDNN-based approaches are largely more efficient, up to several orders of magnitude, than the global ones. In this context, the term "computational complexity" denotes the number of floating point operations necessary to classify a single input pattern after training. For the SVMs, its value approximately corresponds to the dimension of the input vectors times the number of support vectors, as classifying an input vector involves a scalar multiplication of the input vector with each support vector. For the ATDNN, the computational complexity has been derived in detail in Section 2.5. On a Pentium II 450 MHz processor, we have corresponding approximate computation times of 6 ms for the ATDNN, 10 ms for the combination TDNN + SVM, 60 ms for the combination PCA + SVM with $D_{\mathrm{red}} = 128$ features, and 900 ms for direct classification by a polynomial SVM. The duration of the training process is not taken into account as it is performed offline. The number of coefficients that need to be stored is a measure for the memory demand of a classification module. The relations between the corresponding values are illustrated for the individual classification modules in Fig. 20; note that the y axis is logarithmic. The advantages of the ATDNN-based classification approaches are obvious.

In Fig. 21, the performance of several classifiers on the first image or the first three images of each original image sequence, respectively, is shown. It becomes clear that the third order polynomial SVM already has difficulties to cope with its input images of a size of 24×24 pixels, as the two-dimensional version of the ATDNN yields a slightly higher performance on the same training and test set despite its rather simple architecture. Furthermore, this comparison shows that the performance is significantly lower for classification of single images than for classification of image sequences. This clearly illustrates the advantage of image sequence analysis for recognition of walking pedestrians compared to the analysis of single images.

5. Summary and conclusion

In this paper, we have presented applications of the ATDNN algorithm to the recognition of objects in image sequences, based on simultaneous spatio-temporal evaluation of shape and motion features, within the framework of the vision-based driver assistance system UTA. Our first application has been the segmentation-free detection of overtaking vehicles, where we achieved detection rates of more than 96% even under especially difficult visibility conditions. We also achieved to successfully estimate the approximate ego-position of the vehicle, i.e. to recognize if the ego-vehicle is driving on the very left motorway lane and cannot be overtaken by others. Here, the recognition results were stabilized over time by means of feedback loops. Furthermore, we developed a system for real-time recognition of walking pedestrians in the inner city scenario. The objects in the scene are detected by a stereo vision algorithm. The subsequent classification into walking pedestrians and garbage patterns performed by the ATDNN algorithm relies on an analysis of sequences of image regions in which an object has been detected. Pedestrians can be recognized with a high stability at a fairly low false positive rate even under difficult visibility conditions like strong ego-motion, low contrast between the background and the legs of the pedestrian, or sudden changes in illumination. Feedback loops again help to further stabilize the recognition results over time. The recognition rate can be enhanced by combining the ATDNN approach with polynomial support vector machines (SVM); in this configuration, the spatio-temporal receptive fields perform a computationally efficient dimensionality reduction of the input image sequence. We have compared the ATDNN-based approaches relying on local spatio-temporal processing to direct classification of the image sequence by polynomial SVMs with and without dimensionality reduction by standard PCA, with respect to recognition performance, computational complexity, and memory demand. These "global" SVM approaches do not explicitly take into account the local spatio-temporal structure of the image sequences. The recognition performance of the ATDNN-based approaches is higher than that of the global SVM-based methods, while the computational complexity of the ATDNN-based methods is between one and two orders of magnitude lower than that of the global SVM-based methods; for the memory demand, the relation even amounts to up to four orders of magnitude. These results show that local spatio-temporal processing of image sequences in classification modules yields strong advantages with respect to their integration into real-time vision systems with limited computational and memory resources.

Acknowledgements

The stereo vision algorithm was provided by Dr Uwe Franke. The polynomial classifier and SVM results were obtained by applying the ClassLab toolbox written by Dr Ulrich Kreβel. Furthermore, we thank Prof. Dr Jürgen Schürmann for fruitful discussions.

References

[1] K. Akita, Image sequence analysis of real world human motion, Pattern Recognition 17 (1) (1984) 73–83.
[2] A. Baumberg, D. Hogg, An efficient method for contour tracking using active shape models, IEEE Workshop on Motion of Non-Rigid and Articulated Objects, Austin, Texas, 1994, pp. 194–199.
[3] M. Betke, E. Haritaoglu, L.S. Davis, multiple vehicle detection and tracking in hard real time, Intelligent Vehicles Symposium, Tokio, 1996, pp. 351–356.
[4] M. Betke, H. Nguyen, Highway scene analysis from a moving vehicle

under reduced visibility conditions, IEEE International Conference on Intelligent Vehicles, Stuttgart, 1998, pp. 131–136.
[5] U. Bodenhausen, A. Waibel, The tempo 2 algorithm: adjusting time-delays by supervised learning, in: R.P. Lippmann, J.E. Moody, D.S. Touretzky (Eds.), Advances in Neural Information Processing Systems 3, Morgan Kaufmann, San Mateo, CA, 1991, pp. 155–161.
[6] I.-C. Chang, C.-L. Huang, Ribbon-based motion analysis of human body movements, International Conference on Pattern Recognition, Wien, 1996, pp. 436–440.
[7] T. Cootes, C. Taylor, D. Cooper, J. Graham, Active shape models — their training and applications, Computer Vision and Image Understanding 61 (1995) 38–59.
[8] S.P. Day, M.R. Davenport, Continuous-time temporal back-propagation with adaptable time delays, IEEE Transactions on Neural Networks 4 (2) (1993) 348–354.
[9] K. Diamantaras, S.Y. Kung, Principal Component Neural Networks, Wiley-Interscience, New York, 1996.
[10] E.D. Dickmanns, R. Behringer, C. Brüdigam, D. Dickmanns, F. Thomanek, V.v. Holt, An all-transputer visual autobahn-autopilot/copilot, IEEE International Conference on Computer Vision, Berlin, Germany, 1993, pp. 608–615.
[11] E.D. Dickmanns, B. Mysliwetz, T. Christians, An integrated spatio-temporal approach to automatic visual guidance of autonomous vehicles, IEEE Transactions on Systems, Man, and Cybernetics 20 (1990) 1273–1284.
[12] E.D. Dickmanns, A. Zapp, A curvature-based scheme for improved road vehicle guidance by computer vision, SPIE Conference on Mobile Robots 727 (1986) 161–167.
[13] W. Enkelmann, Obstacle detection by evaluation of optical flow fields from image sequences, Image and Vision Computing 9 (1991) 160–168.
[14] W. Enkelmann, V. Gengenbach, W. Krüger, S. Rössle, W. Tölle, Obstacle detection by real-time optical flow evaluation, Intelligent Vehicles Symposium, Paris, 1994, pp. 97–102.
[15] U. Franke, D. Gavrila, S. Görzig, F. Lindner, F. Paetzold, C. Wöhler, Autonomous driving goes downtown, IEEE Intelligent Systems 13 (6) (1998) 40–48.
[16] U. Franke, S. Mehring, A. Suissa, S. Hahn, The Daimler–Benz steering assistant: a spin-off from autonomous driving, Intelligent Vehicles Symposium, Paris, France, 1994, pp. 120–126.
[17] U. Franke, I. Kutzbach, Fast stereo based object detection for stop&go traffic, Intelligent Vehicles Symposium, Tokyo, Japan, 1996, pp. 339–344.
[18] K. Fukushima, S. Miyake, Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position, Biological Cybernetics 15 (1982) 455–469.
[19] D.M. Gavrila, The visual analysis of human movement: a survey, Computer Vision and Image Understanding 73 (1) (1999) 82–98.
[20] D. Gavrila, V. Philomin, Real-time object detection for "smart" vehicles, International Conference on Computer Vision, Kerkyra, Greece, 1999, pp. 87–93.
[21] W.J. Gillner, Bewegungsgekoppelte Szenensegmentierung in Bildfolgen, Proceedings in Artificial Intelligence: Aktives Sehen in technischen und biologischen Systemen, B. Mertsching (Hrsg.), Hamburg, 1996.
[22] C. Goerick, D. Noll, M. Werner, Artificial neural networks in real time car detection and tracking applications, Pattern Recognition Letters 17 (1996) 335–343.
[23] V. Graefe, W. Efenberger, A novel approach for the detection of vehicles on freeways by real-time vision, Intelligent Vehicles Symposium, 1996, pp. 363–368.
[24] Y. Guo, G. Xu, S. Tsuji, Understanding human motion patterns, International Conference on Pattern Recognition, 1994, pp. 325–329.
[25] B. Heisele, Objektdetektion in Straβenverkehrsszenen durch Auswertung von Farbbildfolgen, PhD thesis, University of Stuttgart, 1997.
[26] B. Heisele, U. Kreβel, W. Ritter, Tracking non-rigid, moving objects based on color cluster flow, Proceedings on Computer Vision and Pattern Recognition, Puerto Rico, 1997, pp. 257–260.
[27] B. Heisele, C. Wöhler, Motion-based recognition of pedestrians, International Conference on Pattern Recognition, Brisbane, 1998, pp. 1325–1330.
28 J.A. Hertz, A. Krogh, R.G. Palmer, Introduction to the Theory of Neural Computation, Addison-Wesley, Redwood City, CA, 1991.
[29] R. Kahn, M. Swain, P. Propkowicz, J. Firby, Gesture recognition using the Perseus architecture, IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, 1996, pp. 734–741.
[30] W. Krüger, S. Rössle, W. Enkelmann, Real-time estimation and tracking of optical flow vectors for obstacle detection, IEEE International Conference on Intelligent Vehicles, Detroit, 1995, pp. 304–309.
[31] A. Kuehnle, Symmetry-based recognition of vehicle rears, Pattern Recognition Letters 12 (1991) 249–258.
[32] Y. Le Cun, O. Matan, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, L. Jackel, H. Baird, Handwritten zip code recognition with multilayer networks, IEEE International Conference on Pattern Recognition, 1990.
[33] W. Long, Y. Yang, Log-Tracker, an attribute-based approach to tracking human body motion, International Journal of Pattern Recognition and Artificial Intelligence 5 (3) (1991) 439–458.
[34] S.A. Niyogi, E.H. Adelson, Analyzing and recognizing walking figures in xyt, IEEE International Conference on Computer Vision and Pattern Recognition, 1994, pp. 469–474.
[35] S.A. Niyogi, E.H. Adelson, Analyzing gait with spatiotemporal surfaces, IEEE Workshop on Motion of Non-Rigid and Articulated Objects, Austin, 1994, pp. 64–69.
[36] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio, Pedestrian detection using wavelet templates, IEEE International Conference on Computer Vision and Pattern Recognition, San Juan, 1997, pp. 193–199.
[37] C. Papageorgiou, T. Evgeniou, T. Poggio, A trainable pedestrian detection system, IEEE International Conference on Intelligent Vehicles, Stuttgart, 1998, pp. 241–246.
[38] C. Papageorgiou, T. Poggio, A pattern classification approach to dynamical object detection, International Conference on Computer Vision, Kerkyra, Greece, 1999, pp. 1223–1228.
[39] D. Pomerleau, T. Jochem, Rapidly adapting machine vision for automated vehicle steering, IEEE Expert 11 (2) (1996) 19–27.
[40] B. Schölkopf, C. Burges, A. Smola, Advances in kernel methods: support vector machines, MIT Press, Cambridge, MA, 1999.
[41] J. Schürmann, Pattern Classification, Wiley-Interscience, New York, 1996.
[42] S. Shio, J. Sklansky, Segmentation of people in motion, IEEE Workshop on Visual Motion, 1991, pp. 325–332.
[43] U. Solder, V. Graefe, Visual detection of distant objects, IEEE/RSJ International Conference on Intelligent Robots and Systems, Yokohama, 1993, pp. 1042–1049.
[44] F. Thomanek, E.D. Dickmanns, D. Dickmanns, Multiple object recognition and scene interpretation for autonomous road vehicle guidance, Intelligent Vehicles Symposium, Paris, 1994, pp. 231–236.
[45] B. Ulmer, Vita II — active collision avoidance in real traffic, Intelligent Vehicles Symposium, Paris, 1994, pp. 1–6.
[46] A. Waibel, T. Hanazawa, G. Hinton, K.J. Lang, Phoneme recognition using time-delay neural networks, IEEE Transactions on Acoustics, Speech, and Signal Processing 37 (1989) 328–339.
[47] E.A. Wan, Temporal backpropagation for FIR neural networks, IEEE International Joint Conference on Neural Networks, San Diego, CA, 1990, pp. 575–580.
[48] J. Weber, D. Koller, Q.-T. Luong, J. Malik, New results in stereo-based automatic vehicle guidance, Intelligent Vehicles Symposium, Piscataway, NJ, 1995, pp. 530–535.
[49] D. Wetzel, H. Niemann, S. Richter. A robust cognitive approach to traffic scene analysis, IEEE Workshop on Applications of Computer Vision, Sarasota, Florida, 1994, pp. 65–72.
[50] C. Wöhler, J.K. Anlauf, A. time, delay neural network algorithm for estimating image-pattern shape and motion, Image and Vision Computing Journal 17 (1999) 281–294.

[51] C. Wöhler, J.K. Anlauf, An adaptable time delay neural network algorithm for image sequence analysis, IEEE Transactions on Neural Networks 10 (6) (1999) 1531–1536.

[52] C. Wöhler, J.K. Anlauf, T. Pörtner, U. Franke, A time delay neural network algorithm for real-time pedestrian recognition, IEEE International Conference on Intelligent Vehicles, Stuttgart, Germany, 1998, pp. 247–252.

[53] C. Wöhler, U. Kreβel, J. Schürmann, J.K. Anlauf, Dimensionality reduction by local processing, European Symposium on Artificial Neural Networks, Bruges, Belgium, 1999, pp. 237–244.

[54] C. Wöhler, J. Schürmann, J.K. Anlauf, Segmentation-free detection of overtaking vehicles with a two-stage time delay neural network classifier, European Symposium on Artificial Neural Networks, Bruges, Belgium, 1999, pp. 301–306.

[55] C.R. Wren, A. Azarbayejani, T. Darrell, A.P. Pentland, Pfinder: real-time tracking of the human body, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (7) (1997) 780–785.

[56] T. Zielke, M. Brauckmann, W. von Seelen, Intensity and edge-based symmetry detection with an application to car-following, CVGIP: Image Understanding 58 (1993) 177–190.