6

From door to door – principles and applications of computer vision for driver assistant systems

Uwe Franke, Dariu Gavrila, Axel Gern, Steffen Görzig, Reinhard Janssen, Frank Paetzold and Christian Wöhler,
DaimlerChrysler AG

6.1 Introduction

Modern cars will not only recover information about their internal driving state (e.g. speed or yaw rate) but will also extract information from their surroundings. Radar-based advanced cruise control was commercialized by DaimlerChrysler (DC) in 1999 in their premium class vehicles. A vision-based lane departure warning system for heavy trucks was introduced by DC in 2000.

This will only be the beginning for a variety of vision systems for driver information, warning and active assistance. We are convinced that future cars will have their own eyes, since no other sensor can deliver comparably rich information about the car's local environment. Rapidly falling costs for the sensors and processors combined with increasing image resolution provide the basis for a continuous growth of the vehicle's intelligence. At least two cameras will look in front of the car. Working in stereo, they will be able to recognize the current situation in 3D. They can be accompanied by other cameras looking backwards and to the side of the vehicle.

In this chapter, we describe the achievements in vision-based driver assistance at DaimlerChrysler. We present systems that have been developed for highways as well as for urban traffic and describe principles that have proven robustness and efficiency for image understanding in traffic scenes.

6.1.1 Vision in cars: why?

Three main reasons promote the development of computer vision systems for cars.

1. Safety

The constant improvement of vehicle safety led to a gradual decrease of injured traffic participants all over the world. A further considerable progress will be possible with sensor systems that perceive the environment around the car and are able to recognize dangerous situations. For example, they will alert the driver if he is leaving the lane, disregarding traffic signs and lights or overlooking a possible collision.

The important advantage of vision-based systems is their potential to understand the current traffic situation, a prerequisite for driver warning or interventions in complex situations, in particular to avoid false alarms. Today's radar-based systems, for example, suppress reflections of still objects since they cannot distinguish between a small pole and a standing car.

Stereo vision allows obstacle detection by three-dimensional scene analysis, whereas fast classification techniques are able to recognize the potential collision partner and to distinguish between cars, motorcycles and pedestrians. So, computer vision offers increased safety not only for the people inside the vehicle but also for those outside.

2. Convenience

Vision-based driver assistance systems allow an unprecedented increase in driving convenience. Speed limit signs can be recognized by the computer and taken into account in an adaptive cruise control (ACC) system. Tedious tasks like driving in stop-and-go traffic can be taken over by the system as well as distance or lateral control on highways.

3. Efficiency

It is obvious that less traffic accidents mean less traffic jams and less economical loss. In addition, computer vision can be used to automate traffic on special roads or to improve the efficiency of goods transport by coupling trucks by means of an electronic tow-bar system. The American Advanced Highway System (AHS) programme aimed at a throughput optimization on existing highways by reducing the vehicle spacing and lateral width of the lanes.

Another important aspect is that in the future drivers can do other jobs like administrative work, if the truck or the car is in autonomous mode. This is of interest to all drivers who use their car for business purposes.

6.1.2 One decade of research at DaimlerChrysler

The progress over the past ten years of vision research for vehicle applications is reflected in our demonstrator vehicles. The first experimental car, VITA I, was a 7.5 ton van built 1989 as a platform for experiments within the Prometheus project. It was equipped with a transputer system for lateral guidance and obstacle detection on highways and offered full access to the vehicle's actuators.

This vehicle was replaced by the well-known VITA II demonstrator for the final Prometheus demonstration in Paris 1994. VITA II was a Mercedes-Benz S-class and showed fully autonomous driving on public highways including lane changes (Ulmer

1994). It was equipped with 18 cameras looking in front, to the rear and to the sides in order to allow a 360° view. Built in cooperation with Ernst Dickmanns (University of the Armed Forces Munich) and Werner V. Seelen (University of Bochum), VITA II was able to recognize other traffic participants, the road course as well as the relevant traffic signs. In addition, it was provided with a behavioural module that was able to plan and perform overtaking manoeuvres in order to keep the desired speed. The side-looking and rear-looking cameras were used to ensure safety of these manoeuvres.

In parallel to the Prometheus demonstrators, the Mercedes-Benz T-model OSCAR was built to investigate vision algorithms and control schemes for robust and comfortable lateral guidance on highways at high speeds. The algorithms were based on the standard lane tracking approach developed by Dickmanns in a joint project. Based on the transputer technology of the early 1990s, OSCAR drove about 10000 km on public highways with maximum speeds of 180 km/h using conventional as well as neural controllers (Neußer *et al.*, 1993). OSCAR tracked not only the markings, but looked also for structures parallel to the lane. From the algorithms used in this car the lane departure warning system mentioned in the introduction has been derived (Ziegler *et al.*, 1995).

In 1995, Daimler-Benz finished the work on the OTTO-truck (Franke *et al.*, 1995). With the AHS-idea in mind, this truck was designed to follow a specific leader with minimum distance. To accomplish this task, OTTO measured the distance to the vehicle in front by looking at known markers. An infrared light-pattern was used as well as two checker-board markings. In order to reach minimum distance and to manage emergency braking of the leader, the acceleration of the leader was transferred to the follower by means of a communication link. Recently, OTTO has been replaced by a heavy duty truck (40 ton) within the European Chauffeur project. Investigations revealed an increased throughput on highways and a reduced fuel consumption of 10–20 per cent depending on the mass of the trucks.

The UTA project (Urban Traffic Assistant) aims at an intelligent stop-and-go system for inner-city traffic. At the Intelligent Vehicles Conference 1998, our UTA I (Mercedes-Benz S-class) demonstrator performed vision-based vehicle following through narrow roads in Stuttgart (Franke *et al.*, 1998). Recently, this car has been replaced by UTA II (Mercedes-Benz E-class), which uses standard Pentium III processors instead of PowerPCs and has increased image understanding capabilities. Details on this project and the developed techniques are given in Section 6.5.2.

6.1.3 A comprehensive driver assistance approach

A vision-based system should be able to assist the driver not only on the highway, but in all traffic situations. It is our goal to realize such a comprehensive system. Here is our vision:

Imagine you are driving to an unknown city to meet a business partner. From the beginning of your trip the vision system acts as an attentive co-driver. You will be warned of the bicyclist from the right, that you have failed to recognize. At the next intersection, it will save you from a possible rear-end collision if you are distracted.

On the highway, the car is able to take over control. Steering is based on the reliable recognition of lanes, longitudinal control exploits stereo vision to improve the radar system and takes into account the speed limit signs. If you prefer driving yourself, you still get this information as a reminder.

Near your destination you get stuck in a slowly moving tailback. The car offers you automated stop-and-go driving. This means that it is able to follow the vehicle in front of you longitudinally as well as laterally. This behaviour is not purely reactive. Traffic lights and signs are additionally taken into account by your intelligent stop-and-go system. Driving manually, the system is able to warn you if you have overlooked a red traffic light or a stop sign. Crosswalks are detected and pedestrians that intend to cross the road are recognized. Finally you reach your destination. The small parking lot is no problem, since you can leave your car and let it park itself.

6.1.4 Outline of the chapter

This chapter describes our work at DaimlerChrysler Research towards such an integrated vision system. It is outlined as follows: Section 6.2 describes the capabilities for the highway scenario developed within the early 1990s including lane and traffic sign recognition and presents improvements by sensor fusion. Section 6.3 concentrates on the understanding of the urban environment. Stereo vision as a key to three-dimensional vision is described. A generic framework for shape-based object recognition is presented. Section 6.4 regards object recognition as a classification problem. Various methods for the recognition of the infrastructure as well as the recognition of cars and pedestrians are presented. All modules for the urban scenario have been integrated in the UTA II demonstrator. A multi-agent software system controls the perception modules as described in Section 6.5.

6.2 Driver assistance on highways

In 1986, when Ernst Dickmanns demonstrated autonomous driving on a closed German highway with a maximum speed of 96 km/h, a revolution in real-time image sequence analysis took place. Whereas other researchers analysed single images and drove some metres blindly before stopping again for the next picture, he exploited the power of Kalman filtering to achieve a continuous processing. Only a small number of 8086 processors were sufficient to extract the information necessary to steer the car from the image sequence delivered by a standard camera. With this new idea adopted from radar object tracking, he influenced the field of image sequence analysis strongly. Today, Kalman filters are considered as a basic tool in image sequence analysis.

This first successfully demonstrated application of computer vision for vehicles was the starting shot for a quickly increasing number of research activities. During the following ten years, numerous vision systems for lateral and longitudinal vehicle guidance, lane-departure warning and collision avoidance have been developed all over the world.

This chapter presents systems for highways, which have already left their infancy and have been tested on many thousands of kilometres on European highways. It starts with classical lane recognition and explains the basic idea of Kalman filter based parameter estimation. Possible extensions are described that are under investigation for higher robustness. Important for future advanced cruise control and driver information systems is the knowledge of the current speed limit. A robust traffic sign recognition system is described in Section 6.2.2. It can easily be extended to other scenarios like urban traffic.

6.2.1 Lane recognition

Principles

The estimation of the road course and the position of the car within the lane is the basis for many applications, which range from a relatively simple lane departure warning system for drowsy drivers to a fully autonomously driving car. For such systems the relevant parameters are the same as for a human driver: the curvature of the road ahead and the position of the car within the lane, expressed by the lateral position and the yaw angle.

The idea of most realized vision-based lane recognition systems is to find road features such as lane markings or road surface textures that are matched against a specific geometrical model of the road (e.g. Kluge and Thorpe, 1992; Dickmanns, 1986). Using these, the parameters of the chosen model and the position of the car in the lane are determined, for example using least-square fitting. However, processing every single image independently is not very smart. A much better way is to take the history of the already driven road and the dynamic and kinematic restrictions of vehicles into account, especially when driving at higher speeds.

According to the recommendations for highway construction, highways are built under the constraint of slowly changing curvatures. Therefore, most lane recognition systems are based on a clothoidal lane model, that is given by the following equation:

$$c(L) = c_0 + c_1 \times L \tag{6.1}$$

$c(L)$ describes the curvature at the length L of the clothoid, c_0 is the initial curvature and c_1 the curvature-rate, which is called the clothoidal parameter. The curvature is defined as $c = 1/R$, where R denotes the radius of the curve. As already mentioned, the vehicle's position within the lane can be expressed by the lateral position x_{off} in the lane and the yaw angle $\Delta\psi$ relative to the lane axis.

Assuming the pinhole-camera model and knowing the camera parameter's focal length f, tilt angle α and height-over-ground H, the relation between a point on a marking and its image point $P(x, y)$ can be described by the following equations:

$$x_p = \frac{f}{L}\left(a \times w - x_{\text{off}} - \Delta\psi \times L + \tfrac{1}{2} \times c_0 \times L^2 + \tfrac{1}{6} \times c_1 \times L^3\right)$$

$$L = \frac{H}{\alpha + (y/f)} \tag{6.2}$$

w is the lane width and $a = \pm 0.5$ is used for the left or the right marking, respectively. Hence, every measurement is projected onto a virtual measurement directly on the centre-line of the lane. In all equations, the trigonometrical functions are approximated by the argument ($\sin x = x$, $\tan x = x$), because we consider only small angles. These equations allow the relevant road course and vehicle position parameters to be determined.

Driving at higher speeds, dynamic and kinematic restrictions have to be taken into account. These constraints can be expressed by the following differential equations:

$$\begin{aligned} \dot{x}_{\text{off}} &= v \times \Delta\psi + v_x \\ \dot{\Delta\psi} &= \dot{\psi}_{\text{veh}} - c_0 \times v \\ \dot{c}_0 &= c_1 \times v \\ \dot{c}_1 &= 0 \end{aligned} \tag{6.3}$$

In these equations, v denotes the longitudinal speed of the vehicle, v_y the lateral speed caused by a possible side slip angle and $\dot{\psi}_{veh}$ the yaw rate. v_x and $\dot{\psi}_{veh}$ are measured by inertial sensors.

The integration of the above described models in the lane recognition system is done by means of a Kalman filter as first proposed by Dickmanns (1996). With this optimal linear estimation scheme, it is possible to estimate the state vector, i.e. the relevant model parameters. The Kalman filter is a recursive observer that uses the actual measurements to correct the predicted state (see e.g. Bar-Shalom and Fortmann, 1988).

Each cycle of the estimation process consists of two phases:

1. Prediction phase. Using the model of the system behaviour (in this case described by the differential equations (6.3), the state-vector estimated at time n is propagated to the next time step $n + 1$. With the above given measurement equations (6.2), one can estimate the positions of the markings in the next time step.
2. Update phase. Depending on the predicted state and the actual measurements a new state of the system is calculated such that the estimation error is minimized.

It is common to search for marking positions inside small parallelogram shaped regions only. They are placed in the image according to the predicted positions. 1D-signals are obtained by integrating the intensity within these windows parallel to the predicted direction. The marking positions are found by analysis of these signals.

By calculating the expected measurement error variance, the size of the regions in which to search for markings can be minimized. The so-called 3σ-area describes where to find about 99 per cent of all measurements, if a gaussian noise process is assumed. As can be seen in Figure 6.1, the 3σ-area (the horizontal lines) significantly reduces the search range. This leads to a fast and robust lane recognition system because false-positive markings are not analysed.

The system described above is based on the assumption that the road in front of the car is flat. Finding markings on both sides of the car, it is possible to estimate the tilt

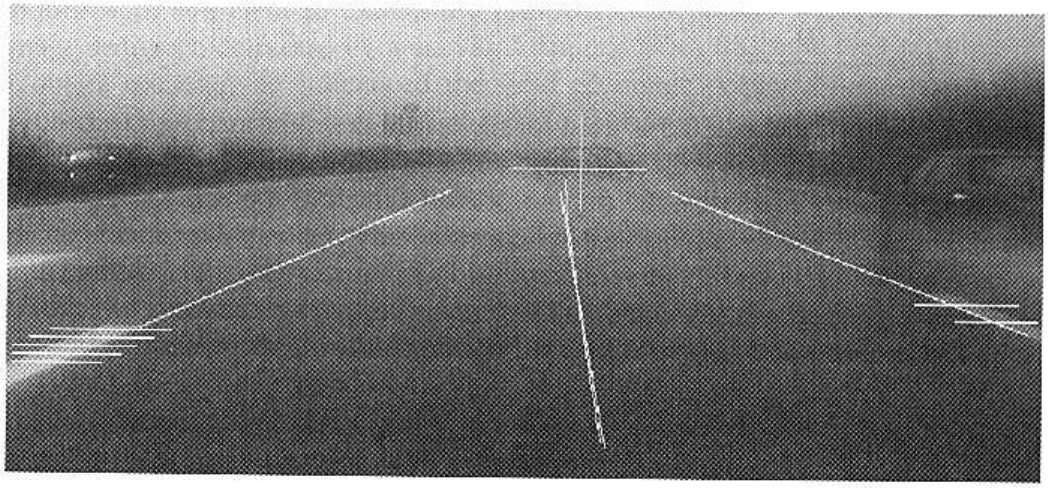


Fig. 6.1 The lane recognition system under rainy conditions, showing the tracked markings with found measurements, the predicted centreline of the lane and one tracked radar obstacle in front.

angle α and the lane width w in addition to the other parameters. Mysliwetz (1990) even estimates the vertical curvature assuming a constant lane width.

Sometimes problems occur because 'markings' are falsely found on cars cutting in or crash barriers within the 3σ-area, because they cannot be separated by using only a monocular camera. This violates the system, causing a wrong state estimation.

These problems can be solved using stereo information, delivering three-dimensional information for each point on the markings. It allows to estimate the vertical curvature c_v besides the already mentioned lane width w and the tilt angle α without further geometrical constraints. German highways are designed according to a parabola vertical curvature. The horizontal and vertical curvature models are separated. The parabola curvature is approximated using a clothoid as described in Mysliwetz (1990):

$$c_v(L) = c_{v,0} + c_{v,1} \times L \tag{6.4}$$

Besides a higher accuracy in all measurements, stereo vision allows the discard of all measurements of non-road objects, that lie above ground. This leads to a more reliable system.

Applied lane recognition

Lane keeping A couple of years ago, our test vehicle OSCAR required about ten transputers for monocular lane recognition at a cycle rate of 12.5 Hz. Today, the job is done with improved performance in less than 2 milliseconds on a 400 MHz Pentium II. An optimized nonlinear controller allows comfortable driving at high speeds. Since the car is often driven by non-experts, velocity is limited to 160 km/h at the moment.

Field tests revealed a high degree of acceptance for two reasons. First, the handling is very simple. When the car signals its readiness, you have just to push a button to give the control to the car. Whenever you like, you can gain the control back by pushing the button again or just steering. Second, autonomous lateral guidance is surprisingly precise. Figure 6.2 shows a comparison of manual and autonomous driving at the same speed on the same section of a winding highway. Although the human driver was told to stay in the middle of the lane as precisely as possible, he needed about 40 centimetres lateral space to both sides, which is typical. As can be seen, the autonomous vehicle performed significantly better. The stronger oscillations between 50 and 90 seconds

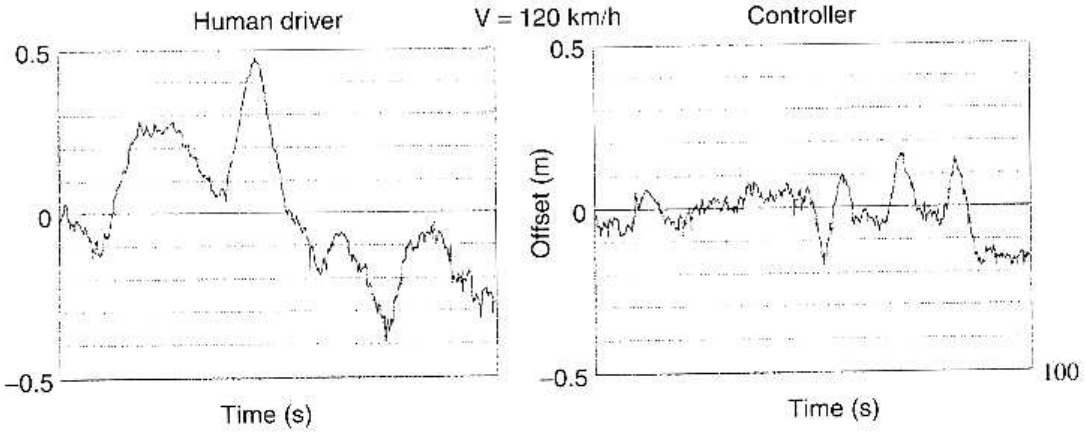


Fig. 6.2 Comparison of human lateral guidance and controller performance on a windy highway.

of driving stem from the tendency of the controller to cut the curves, which have a radius of about 500 metres. In accordance with human behaviour this deviation to the centreline is accepted.

Lane departure warning Many accidents occur due to inattentiveness or drowsiness of the driver, particularly at night. Two types of accidents dominate: rear-end collisions and leaving the lane.

A reliable lane departure warning would therefore lead to a significant increase in traffic safety and is of special interest for trucks and busses. Based on the above described methodology, we have realized an optical system that has been commercially available since May 2000. Leaving the lane without indicating the lane-change, it warns the driver acoustically by means of a rumble strip like sound from the left or right loudspeaker. Naturally, the direction depends on the marking the vehicle is crossing. Camera and processor are integrated in a small box that fits into a palm.

In order to achieve maximum availability and a minimum number of false alarms, the camera is mounted 2–3 metres above ground with a large tilt angle. This maximizes the robustness of the image processing since glare due to a low sun or reflections due to a wet road are avoided. Since the warning system has to be operational on all roads outside built-up areas and only the lateral position of the vehicle is of interest, the road is assumed to be straight. Thus only offset and yaw angle are determined in the estimation process. The error introduced by this assumption is negligible since the maximum look-ahead distance is smaller than 10 metres to guarantee optimal performance at night.

Advanced lane recognition

The traditional lane recognition system described above runs robustly and reliably under fair weather conditions. Problems occur when driving in adverse weather conditions such as rain or snow. Often, the contrast between the markings and the pavement is poor, sometimes the colours of the markings look negated. The range of sight is reduced enormously, causing a bad prediction of the lane parameters, especially the curvature.

A significant improvement of the reliability of the lane recognition system is possible by integration of other sensors that offer a better availability in darkness, rain and snow.

We are investigating two different systems using:

- radar information
- a GPS-based map information system.

a) Radar information DISTRONIC is a radar-based adaptive cruise control system which was introduced in the Mercedes-Benz S-class in May 1999. It measures the following three parameters for every radar obstacle i:

1. The distance d_{obj_i}.
2. The relative speed $v_{\mathrm{rel,\ obj}_i}$.
3. The angle φ_{obj_i}.

Our approach for improved lane recognition using radar information, as first described in Zomotor and Franke (1997), is motivated by the human strategy when driving in bad weather conditions. Human drivers use the cars in front in order to estimate the road

course, assuming that these cars stay in their lanes without significant lateral motion. Such a situation is shown in Figure 6.1.

In fact, every car in front contains strong information on the road curvature that can be used in the estimation process. If we track the vehicles, we can extract these parameters from the measured distance and angle over time. The basic assumption that the lateral motion of the leading vehicles relative to the lane is small can be expressed by:

$$\dot{x}_{\text{off,obj}_i} = 0 \tag{6.5}$$

Large lateral motion caused by lane changes of the tracked vehicles can be detected by appropriate tests.

The radar measurements are incorporated in the Kalman estimation by an additional measurement equation given by:

$$x_{\text{obj}_i} = x_{\text{off}} - x_{\text{off,obj}_i} - \Delta\psi \times d_{\text{obj}_i} c_0 \times d_{\text{obj}_i}^2 + \tfrac{1}{6} \times c_1 d_{\text{obj}_i}^3 \tag{6.6}$$

The lateral offset x_{obj_i} of each radar obstacle i is related to the measured angle φ_{obj_i} via

$$\varphi_{\text{obj}_i} \approx \sin \varphi_{\text{obj}_i} = \frac{x_{\text{obj}_i}}{d_{\text{obj}_i}}, \varphi_{\text{obj}_i} < 5^\circ$$

Getting raw data from the radar sensor, an adequate kinematic model of the obstacles is given by the differential equations:

$$\begin{aligned} \dot{\mathrm{d}}_{\text{obj}_i} &= v_{\text{rel,obj}_i} \\ \dot{v}_{\text{rel,obj}_i} &= 0 \end{aligned} \tag{6.7}$$

As can be seen in Figure 6.1, the range of sight can be enormously enlarged using radar obstacles. Particularly the curvature parameters c_0 and c_1 are improved significantly.

The improvements of the fusion approach can be seen best in simulations as shown in Figure 6.3. The graph shows the curvature c_0 of a simulated road as obtained from the lane recognition system under good weather conditions (range of sight about 50 m) as reference curve, the lane recognition system under bad visibility (range of sight about 10–14 m) and the sensor fusion system following one and two cars under bad visibility (range of sight again about 10–14 m). The distance to the cars in front is about 60–80 m.

The road consists of a straight segment, going into a right bend of radius 300 m and again a straight road. The curvature estimation is improved enormously taking other cars into account. Looking closer at the diagram, it can be seen that the radar-fusion estimates of the curvature run ahead of the lane recognition system. This effect can be explained by the other cars going earlier into the bend than one's own car. The same effect can be seen between 320 and 400 m, where the cars in front are already going out of the bend. The pure lane recognition system under bad visibility shows a delay in the curvature estimation due to the small range of sight. The detailed results are presented in Gern *et al.* (2000).

Assigning cars to specific lanes is an important task for ACC-systems. Since we observe the lateral position of the leading vehicles relative to our lane, this assignment is simultaneously improved by the described approach. This avoids unwanted accelerations and decelerations.

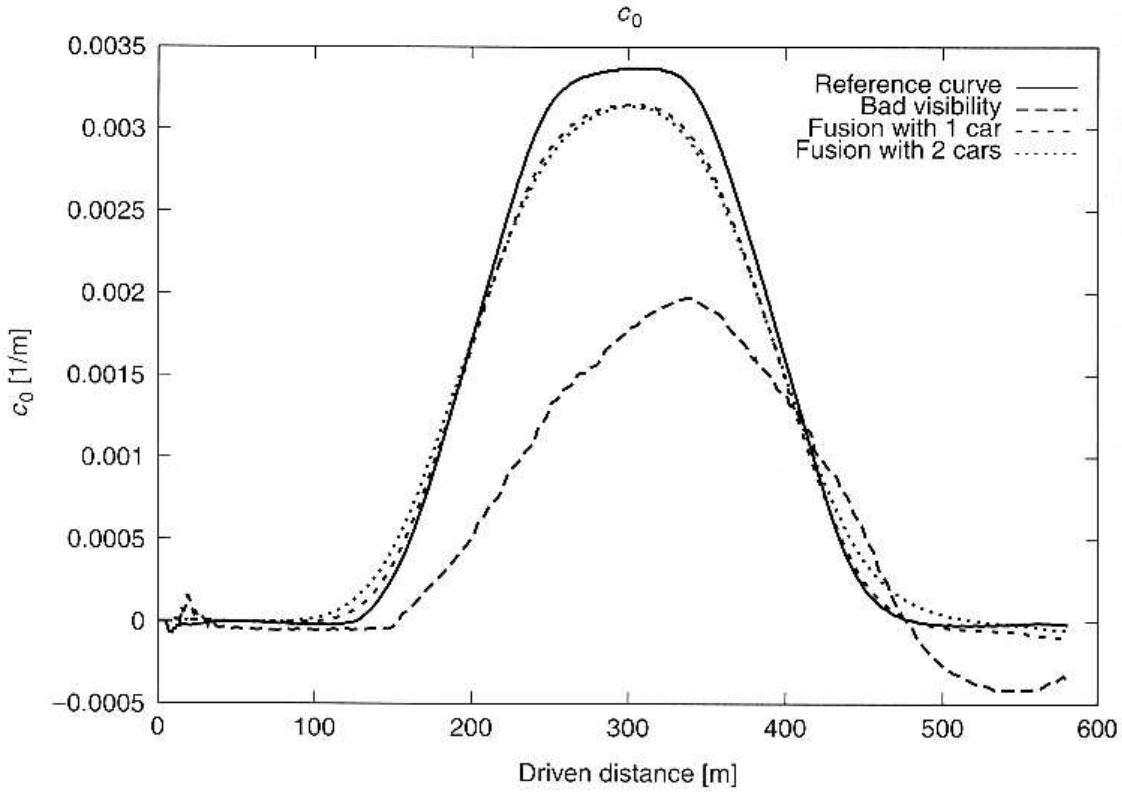


Fig. 6.3 Curvature estimated under bad weather conditions.

The fusion approach assumes that the radar measurements coincide with the centre axes of the cars. Unfortunately, the radar often detects the edges of an obstacle and not the middle axis. Sometimes it is sliding between the left and the right.

It is obvious that the run of the curve can't be estimated correctly if the radar delivers imprecise estimations of the positions of the vehicles. One consequence is that the measurement for every radar obstacle has a large variance. This weakens the strong information obtained by using radar information to increase the range of sight for the lane recognition system.

In order to solve this problem, we are developing a system that detects and tracks all radar obstacles in monocular images. A detailed description of this approach can be found in Gern *et al.*, 2000.

b) GPS-based map information system The second approach to enhance the robustness of the lane recognition system is to integrate GPS-based map information.

Since precise maps are not available at the moment, we generate our own data. During the generation phase, the road course is recorded. Using the optical lane recognition system and a highly accurate GPS-system, the centreline of the lane is determined to generate the necessary highly accurate map.

Later, when driving on this road, the lane recognition system can exploit the map information, especially in adverse weather conditions. The GPS-based map system provides the before measured centreline, describing the road course. This includes the curvature and the clothoidal parameter as well as a number of road points.

We are investigating two different fusion approaches, extending the already described Kalman filter:

1. Using the curvature c_0 and the clothoid parameter c_1 provided by map information system directly as measurements in the lane recognition system.
2. Using the world coordinates of the centreline given by the map information system directly as measurements using an adequate measurement equation.

Simulations and test drives show a higher accuracy of the curvature and yaw angle estimation for both approaches, even if the accuracy of the map information system is low. Results are comparable to those obtained by fusing radar obstacles.

6.2.2 Traffic sign recognition (TSR)

For the foreseeable future visible road signs will regulate the traffic. If vision systems are to assist the driver, they have to be able to perceive traffic signs and observe the implied rules.

Let us first characterize the nature of this object recognition task for a vision based system.

1. Traffic signs are man-made objects and standardized according to national law. Shape and colour are chosen such that they can be easily spotted by a human observer. Both facts alleviate the perception task also for a machine vision system.
2. Traffic signs mounted on sign posts or on bridges spanning the road may have high contrast against the sky. A vision sensor must cover a large dynamic range to make the sign clearly visible in the image. Poor visibility affects the system performance not less than that of a human driver.
3. A large family of traffic signs denote road attributes such as speed limits, which are valid inside a continuous interval, in a discontinuous way. One sign marks the beginning, and another sign the end of the interval. While we can apply an initialization and update process for tracking lanes or for following cars, here we have to search all the acquired images for new appearances of traffic signs in an exhaustive way. Since we do not know a *priori* where to expect traffic signs, this task will bind a considerable amount of computing effort if we do not want to miss a single sign, even driving at high speeds.

Of course there exist non-vision approaches to that problem. Road signs equipped with IR or microwave beacons signal the information regardless of visibility conditions but at high infra-structural costs. Digital maps attributed with traffic regulations supply the correct information only if they are up to date. Temporary changes due to road work, accidents, or switching electronic signs cannot be easily integrated into a map. Thus we are convinced that vision is an essential part of the solution.

The scope of traffic signs handled by the TSR module depends on the range of operation. For a highway scenario useful applications can start with a small set of signs, including speed limits, no overtaking, and danger signs. The urban scenario described in Section 6.3 requires the set to be extended by adding signs which regulate the right of way at intersections.

Detection

On highways, the TSR module is confronted with high resolution images acquired at high velocities of the vehicle. The key to a real-time traffic sign recognition system is a fast and robust detection algorithm. This algorithm detects regions which are likely to contain traffic signs. The traffic sign candidates will be tracked through the image sequence until a reliable result can be obtained. This allows the recognition process to focus on a limited number of small search areas, which speeds up the whole process significantly.

The detection stage can exploit colour and/or shape as the first cue for traffic sign hypotheses. The shape matching algorithm described in Section 6.3.2 is used for traffic sign detection when colour is not available from the camera.

For example, we will here elaborate on a detection algorithm which relies on the colour of traffic signs. The advantage of colour cues, in contrast to shape, is their invariance against scale and view and their highly discriminative power. Even partially occluded or deformed traffic signs can be detected using the colour information.

The algorithm consists of the following three principal steps (Janssen, 1993):

1. In the first step the significant traffic sign colours are filtered out from the acquired colour image. As a result of the colour segmentation the pixels of the image are labelled with the colours *red*, *blue*, *yellow*, *black*, *grey* and *white*.
2. In the second step the iconic information is transformed to a symbolic description of coloured connected components (CCC) by applying a fast connectivity analysis algorithm (Mandler and Oberländer, 1990).
3. Finally the CCC database is queried for objects with certain geometrical attributes and colour combinations. Ensembles of CCC objects extracted by those queries are called meta CCC objects. Meta CCC objects serve as hypotheses for the subsequent traffic sign recognition process.

Colour segmentation The task of colour segmentation is to mimic what the human observer does when he or she recognizes a specific *red* as 'traffic sign red'. The visual system seems to have a strong concept of colour constancy which enables recognition of this certain red although the colour description covers a wide range of different hues. These hues are influenced by the paint, the illumination conditions, and the viewing angle of the observer. Since the current knowledge is not adequate to model all facets of colour perception, a learning approach was chosen to generate colour descriptions suitable for the machine vision system.

The mapping from the colour feature vector to the colour label in the decision space is a typical classification problem which can be solved by means of statistical pattern recognition. The classification is performed by neural networks or polynomial classifiers as described in Section 6.4. The coefficients of the network have to be adapted during the learning phase with a representative set of labelled samples for every colour class. Manually labelled traffic scene images are used to adapt a polynomial classifier to the colours *red*, *blue*, *yellow*, *black*, *grey* and *white*. The traffic signs in the application scenarios can be described by these colour terms.

Colour connected components The colour labelled image is still an iconic representation of the traffic scene. Grouping all neighbouring pixels with a common colour into regions creates a symbolic representation of the image. The computation of the

so-called colour connected components (CCC) is performed by a fast, one-pass, line-scan algorithm (Mandler and Oberländer, 1990). The algorithm produces for each CCC a list of all neighbouring components, thus providing full topological information. The connected component analysis does not induce any information loss since the labelled image can be completely reconstructed from the CCC database.

Now it is easy to retrieve candidate regions with a specific topological relationship and colour combination efficiently.

Meta CCC language Traffic sign candidates are extracted from the CCC database with queries searching for instance for regions with a certain colour, inclusions of a certain colour, and geometrical attributes inside specific intervals. The ensemble of CCC objects extracted by those queries is called a meta CCC object. The meta CCC query language efficiently parses the database at run-time. The language comprises pure topological functions (adjacency, inclusion, etc.) as well as functions exploiting the shape of colour components (size, aspect ratio, coverage, eccentricity, etc.). e.g. the filter

inside of(RED,WHITE) | aspect(0.8,1.2) | larger(12) | smaller(10 000) |group +
inside of(RED,GREY) | aspect(0.8,1.2) | larger(12) | smaller(10 000) | group

searches for red objects which have white or grey inclusions, a square bounding box, and a size reasonable for the traffic sign recognition procedure. Figure 6.4 shows that the use of adequate queries helps to detect traffic signs even under non-ideal conditions.

The bounding boxes of the meta CCC objects into which the objects are grouped with their inclusions form a region of interest (ROI). This ROI is the input to the verification and classification stage of traffic sign recognition.

Recognition

The task of the recognition stage is to map the pixel values inside the ROI either to a specific traffic sign class, or to reject the hypothesis. Employing statistical pattern recognition (see Section 6.4) ensures that this mapping is insensitive to all the variations of individual traffic signs, which are due to changes in illumination, weather conditions, and picture acquisition for instance. Figure 6.5 shows a collection of red danger signs varying in colour, spatial resolution, background, viewing aspect, and pictographic symbols.

Feature extraction A main problem to be solved in building statistical pattern recognition systems is the design of a significant feature vector.

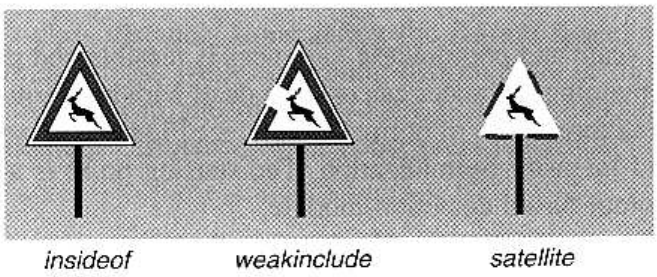


Fig. 6.4 Traffic sign detection in case of poor segmentation results.

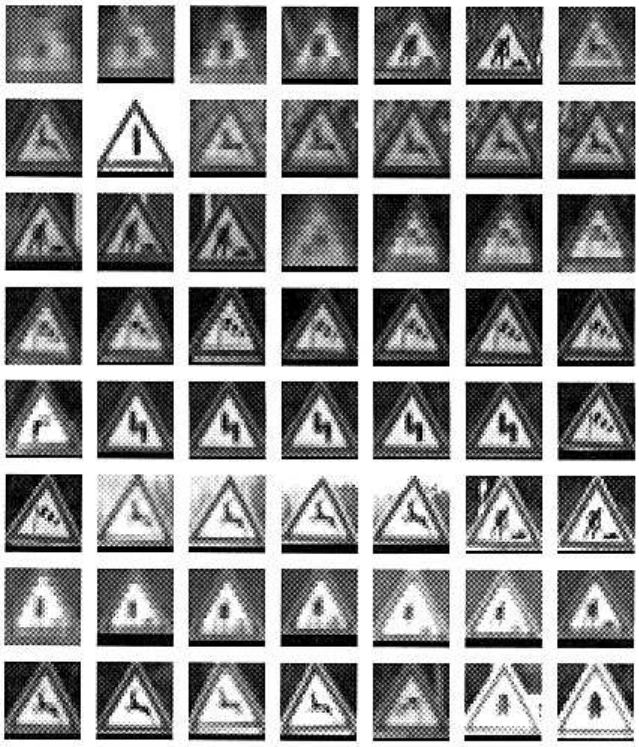


Fig. 6.5 Collection of red danger signs varying in colour, spatial resolution, background, viewing aspect, and pictographic symbols.

There is a trade-off between the expenditure on feature extraction and on classification. This means that clever normalization measures can simplify the classification process considerably. Hence, if we manage to design a well balanced system, we can do with a smaller learning set, will spend less computational effort, and increase the classification efficiency and performance.

In samples of traffic scenes we observe several kinds of variances:

1. in scale and resolution due to variable sizes of traffic signs and variable distances between camera and object,
2. in translation due to inaccurate segmentation,
3. in photometric parameters (brightness, contrast, colour) due to variable weather and lighting conditions,
4. in design of the traffic signs themselves due to different versions (fonts, layout).

A normalization of scale and photometric variances is feasible and part of our recognition system. The remaining variances can only be dealt with by learning from examples.

In fact we carry out three steps of normalization (see Figure 6.6): scale normalization, colour mapping and intensity normalization. The starting point is a region of interest delivered by our colour-based detection module.

Scale normalization is not dispensable, because the dimension of the input layer is fixed for most classifier networks. Each pixel is interpreted as one component of the feature vector.

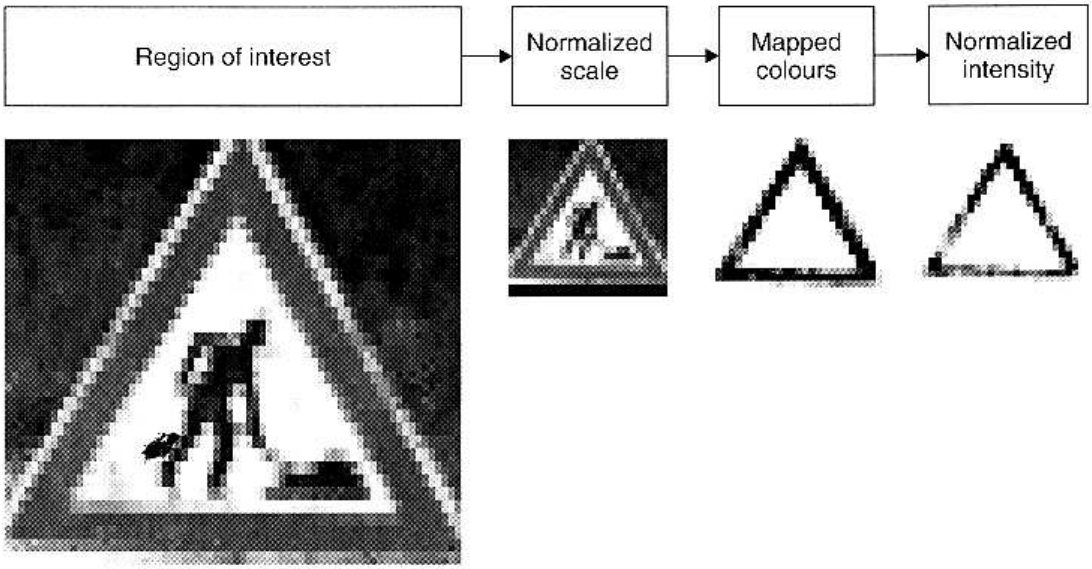


Fig. 6.6 Normalization steps.

In feature extraction for shape recognition a complementary colour mapping is applied. Chromatic pixels are mapped to high and achromatic pixels to low intensities. This mapping emphasizes shape discriminating regions and suppresses noise. Both background and pictograms are suppressed in the same manner, because these regions have low colour saturation.

There are three different mapping functions applicable according to the colour of the traffic sign, which is already known after traffic sign detection. In order to reduce the remaining variances of intensity (brightness, contrast), the mean and dispersion of the pixels' intensities are calculated and adjusted to standard values. If a pattern is entirely achromatic (e.g. end of restrictions and most pictograms) complementary colour mapping makes no sense. In this case we only normalize the intensity of the size-scaled ROI.

$$\begin{aligned} Y &= R - \frac{G+B}{2} && \text{complementary colour mapping for red signs} \\ Y &= B - \frac{R+G}{2} && \text{complementary colour mapping for blue signs} \\ Y &= \frac{R+G+B}{3} && \text{colour mapping for achromatic signs} \end{aligned} \tag{6.8}$$

We conclude that the goal of normalization is to decrease the intra-class distance while increasing the inter-class distance. Considering the feature space, we try to compress the distributions of each class while separating distributions of different classes, thus improving the ratio of spread and mean distance of the corresponding distributions. This effect supports the performance of the subsequent classification, no matter how it is implemented.

Model-based recognition A very important requirement for an autonomous TSR system is the capability of considering variable road traffic regulations due to differences in jurisdiction between countries and temporal changes of rules.

A hard coded structure of fixed classifiers with pre-programmed rules would mean a drawback in flexibility. For this reason we used a model-based approach to organize a number of shape and pictogram classifiers. In this approach we try to separate domain-dependent from domain-independent data, thus providing an interchangeable model of objects we intend to recognize and a more universal recognition machine.

In order efficiently to construct a traffic sign model we investigate significant features at first. Our system, motivated by human perception, is sketched in Figure 6.6. It starts with the dominant colour, e.g. red, white or blue, which has already been used for detection of the regions of interest. Second, the shape of the traffic sign candidate is checked before the pictogram is classified as described in Section 6.4.

For each image containing traffic sign candidates a so-called search tree is generated according to the structural plan of the model.

The development of each search tree node involves computational effort. For efficiency reasons the decision which node to develop next is taken in a best first manner. If there is an upper bound estimate of the future costs the even more efficient A* algorithm is applicable. Both methods guarantee to find optimal paths. If a terminal node is reached the search is finished.

Conclusions

The system described is capable of recognizing a subset of traffic signs in a robust manner under real-world conditions. The results obtained above, however, refer to single images only, i.e. relations between subsequent frames have been neglected. Tracking traffic signs over the time interval during which they are in the field of view adds to the stability of the recognition results. In test drives carried out by Daimler-Chrysler on German and French motorways the recognition rate could be increased from 72 per cent in single images to 98 per cent in image sequences. Tracking is also a means of reducing computational effort in that the detection can be focused to smaller search regions.

We can even estimate the relative size and position of the sign if we evaluate all monocular measurements of the tracked object with regard to the ego-motion of the vehicle (depth from motion). This kind of information is used for further plausibility checks and interpretation tasks.

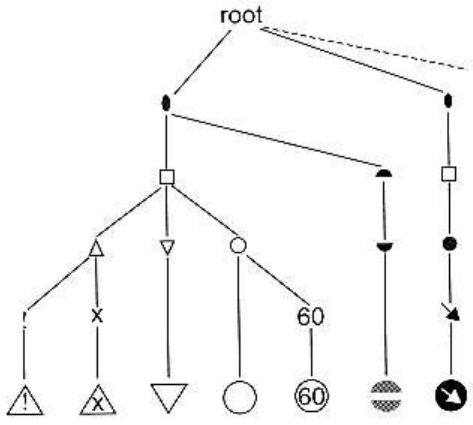


Fig. 6.7 Recognition tree.

Numerous context rules influence the validity of traffic signs, e.g. traffic signs may apply to specific lanes, at specific times of day, or not beyond the next road junction. Current vision modules cannot always gather this type of information with the required reliability. For a commercially feasible system the vision-based recognition of traffic signs and a digital map with traffic sign attributes must support each other. But an autonomous vision system will be part of the solution since even the best map is no exact mirror of the current traffic situation.

6.3 Driver assistance in urban traffic

As pointed out in the introduction, a vision-based driver assistance system would be even more attractive if it would be able to support the driver not only on the highway, but also in city traffic. Intelligent stop-and-go is our first approach to building such a system. It includes stereo vision for depth-based obstacle detection and tracking and a framework for monocular detection and recognition of relevant objects – without requiring a supercomputer in the trunk. Besides Intelligent Stop& Go, many other driver-assistance systems such as rear-end collision avoidance or red-traffic-light warning are also of interest for urban traffic. The most important perception tasks that must be performed to build such systems are to:

- detect the leading vehicle and estimate its distance, speed and acceleration;
- detect stationary obstacles that limit the available free space, such as parked cars;
- detect and classify different additional traffic participants, such as pedestrians;
- detect and recognize small traffic signs and traffic lights in a complex environment;
- extract the lane course, even if it lacks well-painted markings and does not show clothoidal geometry.

This list shows that the ability to recognize objects is essential for Intelligent Stop-&-Go. Two classes of objects pertain to this application, namely infrastructure elements and traffic participants. How do we recognize those objects? Although devising a general framework is difficult, we often find ourselves applying three steps: detection, tracking and classification.

The purpose of the detection is efficiently to obtain a region of interest (ROI) – that is, a region in the image or parameter space that could be associated with a potential object. It is obvious that not all objects can be detected by means of a unique algorithm. We have developed methods that are based on depth from stereo, shape and colour. Detected objects are tracked from frame to frame to estimate their motion and increase the robustness of the system. Once an object of interest (depending on size or shape) has been detected, the system tries to recognize it. In the considered scenario, objects have a wide variety of appearances because of shape variability, different viewing angles and illumination changes. Since explicit models are seldom available, we derive models implicitly by learning from examples. This turns object recognition to a classification problem, which is described in detail in Section 6.4.

In this section, we present our detection schemes based on stereo and shape. Object recognition tasks exploiting colour have already been described in Section 6.2.2 (traffic sign recognition) and are sketched in Section 6.4 (traffic light recognition).

6.3.1 Stereo vision

Vision systems for driver assistance require an internal 3D map of the environment in front of the car, in order to safely navigate the vehicle and avoid collisions. This map must include position and motion estimates of relevant traffic participants and potential obstacles. In contrast to the highway scenario where you can concentrate on looking for rear ends of preceding vehicles, our system has to deal with a large number of different objects, some of them non-rigid like pedestrians, some of them unknown.

Several schemes for obstacle detection in traffic scenes have been investigated in the past. Besides the 2D model based techniques that search for rectangular, symmetric shapes, inverse perspective mapping based techniques (Broggi, 1997), optical flow based approaches (Enkelmann, 1997) and correlation-based stereo systems using specialized hardware (Saneyoshi, 1994) have been tested.

The most direct method to derive 3D-information is binocular stereo vision for which correspondence analysis poses the key problem. Given a camera pair with epipolar geometry, the distance L to a point is inversely proportional to the disparity d in both images according to:

$$L = \frac{f_x \times B}{d}, \tag{6.9}$$

where B denotes the base width and f_x the scaled focal length.

We have developed two different stereo approaches, one feature-based and one area-based. Both have in common that they do not require specialized hardware but are able to run in real-time on today's standard PC processors.

Real-time stereo analysis based on local features

A fast nonlinear classification scheme is used to generate local features that are used to find corresponding points. This scheme classifies each pixel according to the grey values of its four direct neighbours (Franke and Kutzbach, 1996). It is verified whether each neighbour is significantly brighter, significantly darker or has similar brightness compared to the considered central pixel. This leads to $3^4 = 81$ different classes encoding edges and corners at different orientations. The similarity is controlled by thresholding the absolute difference of pixel pairs.

Figure 6.8 shows the left image of a stereo image pair taken from our camera system with a base width of 30 cm. The result of the structure classification is shown in the right part. Different grey values represent different structures.

The correspondence analysis works on these feature images. The search for possibly corresponding pixels is reduced to a simple test whether two pixels belong to the same class. Since our cameras are mounted horizontally, only classes containing vertical details are considered. Thanks to the epipolar constraint and the fact that the cameras are mounted with parallel optical axis, pixels with identical classes must be searched on corresponding image rows only.

It is obvious that this classification scheme cannot guarantee uniqueness of the correspondences. In case of ambiguities, the solution giving the smallest disparity, i.e. the largest distance, is chosen to overcome this problem. This prevents wrong correspondences caused by for example periodic structures to generate phantom obstacles close to the camera. In addition, measurements that violate the ordering constraint are ignored (Faugeras, 1993).

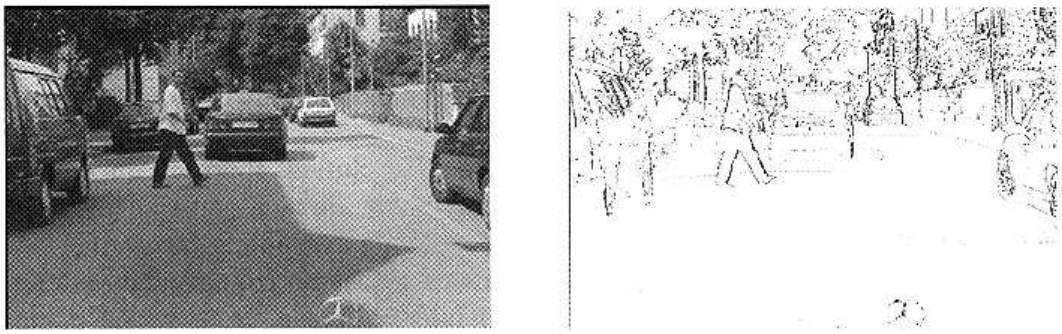


Fig. 6.8 Left image of a stereo image pair and the features derived by the sketched nonlinear operation. Each colour denotes one of the 81 structural classes, pixels in homogeneous areas are assigned to the 'white' class.

Fig. 6.9 Results of the correspondence analysis. Image (a) shows the result of the feature based approach, image (b) shows the result of the correlation-based scheme. Distance is inversely proportional to the darkness.

The outcome of the correspondence analysis is a disparity image, which is the basis for all subsequent steps. Figure 6.9 visualizes such an image in the left half. Of course, the result looks noisy due to the extreme local operation.

On a 400 MHz Pentium II processor this analysis is performed within 23 milliseconds on images of size 384 × 256 pixel.

The advantage of this approach is its speed. Two facts might be a problem in some applications. First, the disparity image is computed with pixel accuracy only. This problem can simply be overcome by post-processing. Second, the described algorithm uses a threshold to measure similarity. Although the value of this threshold turns out to be uncritical, it is responsible for mismatches of structures of low contrast.

Real-time stereo analysis based on correlation

For applications that do not need a cycle rate of 40 milliseconds but require high precision 3D information, we developed an alternative area-based approach. The maximum processing time that we can tolerate is 100 ms.

In order to reach the goal, we must use the sum-of-squared or sum-of-absolute differences criterion instead of expensive cross-correlation to find the optimal fit. Since gain and shutter of our cameras are controlled by the stereo process, this is acceptable.

However, real time is still a hard problem. Full brute-force correlation of 9×9 pixel windows requires about 9 seconds for images of size 384×256, if the maximum disparity is set to 80 pixels. With an optimized recursive implementation we achieved typical values of 1.2 seconds.

To speed up the computation, we use a multi-resolution approach in combination with an interest operator (Franke and Joos, 2000). First, a gaussian pyramid is constructed for the left and right stereo images, based on a sampling factor of 2. Areas with sufficient contrast are extracted by means of a fast horizontal edge extraction scheme. Non-maximum suppression yields an interest image, from which a binary pyramid is constructed. A pixel (i, j) at level n is marked if one of its four corresponding pixels at level $n - 1$ is set. On an average, we find about 1100 attractive points at pyramid level zero (original image level), 700 at level one, 400 at level two and about 150 at level three. Only those correlation windows with the central pixel marked in these interest images are considered during the disparity estimation procedure.

Depending on the application, the correlation process starts at level two or three of the pyramid. If D is the maximum searched disparity at level zero, it reduces to $D/2^{**}n$ at level n. At level two this corresponds to a saving of computational burden of about 90 per cent compared to a direct computation at level zero. Furthermore, smaller correlation windows can be used at higher levels which again accelerates the computation.

The result of this correlation is then transferred to the next lower level. Here, only a fine adjustment has to be performed within a small horizontal search area of ± 1 pixel. This process is repeated until the final level is reached. At this level, subpixel accuracy is achieved by fitting a parabolic curve through the computed correlation coefficients.

The price we have to pay for this fast algorithm is that mismatches in the first computed level propagate down the pyramid and lead to serious errors. In order to avoid this problem, we compute the normalized cross-correlation coefficient for the best matches at the first correlation level and eliminate bad matches from further investigations.

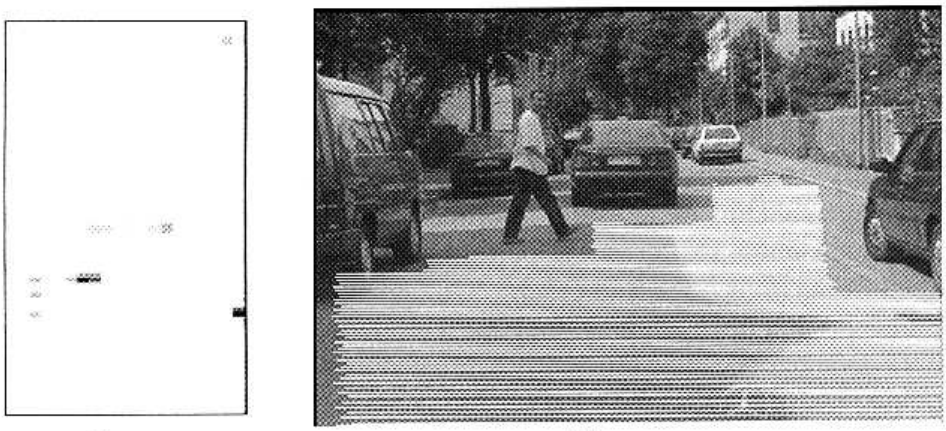
If we start at level 2 (resolution 91×64 pixels), the total analysis including pyramid construction runs at about 90 milliseconds on a 400 MHz Pentium. If we abandon the multi-resolution approach, about 450 milliseconds are necessary to yield comparable results.

A disparity image derived by this scheme is shown in Figure 6.9 in comparison to the feature-based approach. Since a large neighbourhood is taken into account during processing, the result looks less noisy. In fact, only a few mismatches remain, typically in case of periodic structures. A common left-right test applied to the results of the highest evaluation level further reduces the error rate.

Obstacle detection and tracking

The result of both algorithms is a disparity or depth image. Therefore, the further processing is independent of the method used.

Driving on roads, we regard all objects above ground level as potential obstacles. If the cameras are mounted H metres above ground and looking downwards with a tilt



(a) (b)

Fig. 6.10 From the depth map (a) the free space in front of the car is derived (b).

angle α, all image points with a disparity given by

$$d = x_l - x_r = \frac{B}{H} f_x \times \left[\frac{y}{fy} \times \cos(\alpha) + \sin(\alpha) \right] \tag{6.10}$$

lie on the road.

The projection of all features above the road plane, i.e. those with disparities larger than given by equation 6.10, yields a two-dimensional depth map. In this histogram, obstacles show up as peaks.

The map shown in Figure 6.10 covers an area of 40 m in length and 6 m in width. The hits in the histogram are clearly caused by both cars parking left and right, the car in front, the pedestrian and the white car on the right side. Although the feature-based approach looks noisy, the depth maps of both approaches are comparable. This map is used to detect objects that are tracked subsequently. In each loop, already tracked objects are deleted in this depth map prior to the detection.

The detection step delivers a rough estimate of the object width. A rectangular box is fitted to the cluster of feature points that contributed to the extracted area in the depth map. This cluster is tracked from frame to frame. For the estimation of the obstacle distance, the disparities of the object's feature points are averaged.

In the current version, an arbitrary number of objects can be considered. Sometimes the right and left part of a vehicle are initially tracked as two distinct objects. These objects are merged on a higher 'object-level' if their relative position and motion fulfil reasonable conditions.

From the position of the objects relative to the camera system their motion states, i.e. speed and acceleration in longitudinal as well as lateral direction, are estimated by means of Kalman filters. For the longitudinal state estimation we assume that the jerk, i.e. the deviation of the acceleration, of the tracked objects is small. This is expressed in the following state model with distance d, speed v and acceleration a:

$$\begin{bmatrix} d \\ v_l \\ a_l \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} d \\ v_l \\ a_l \end{bmatrix}_k - T \times \begin{bmatrix} v_e \\ 0 \\ 0 \end{bmatrix}$$

The index l denotes the states of the lead vehicle, the index e denotes the ego vehicle. T is the cycle time. The longitudinal motion parameters are the inputs for a distance controller. An example of autonomous vehicle following is given in Section 6.5.

Further analysis of the depth information

The depth image contains more useful information. The fact that we can identify structures on the road plane improves the performance of lane and crosswalk recognition as described in Sections 6.2.1 and 6.3.3.

Camera height and pitch angle are not constant during driving. Fortunately, the relevant camera parameters can be efficiently estimated themselves using the extracted road surface points. Least squares techniques or Kalman filtering can be used to minimize the sum of squared residuals between expected and found disparities (Franke *et al.*, 1998).

Active collision avoidance is the ultimate goal of driver assistance. A careful evaluation of the depth map allows extraction of free space on the road that could be used for a jink. Figure 6.10 shows the depth map derived for the situation considered here and the determined free space. Alternatively, the driving corridor can be estimated from the depth map, if no other lane boundaries are present.

6.3.2 Shape-based analysis

Another important vision cue for object detection is shape. Compared with colour, shape information tends to remain more stable with respect to illumination conditions, because of the differential nature of the edge extraction process. We developed a shape-based object detection system general enough to deal with arbitrary shapes, whether parameterized (e.g. circles, triangles) or not (e.g. pedestrian outlines). The system does not require any explicit shape-models, and instead learns from examples. A template matching technique provides robustness to missing or erroneous data; it does so without the typical high cost of template matching by means of a hierarchical technique. The resulting system is called the 'Chamfer System' (Gavrila and Philomin, 1999); it provides (near) real-time object detection on a standard PC platform for many useful applications.

The Chamfer System

At the core of the proposed system lies shape matching using distance transforms (DT), e.g. Huttenlocher *et al.* (1993). Consider the problem of detecting pedestrians in an image (Figure 6.11(a)). Various object appearances are captured with templates

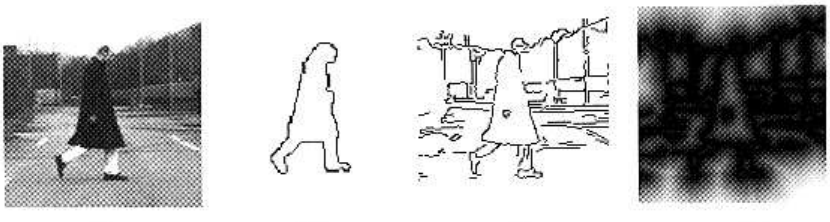


Fig. 6.11 (a) Original image (b) Template (c) Edge image (d) DT image.

such as in Figure 6.11(b). Matching template **T** and image **I** involves computing the feature image of **I**, (Figure 6.11(c)) and applying the distance transform to obtain a DT-image (Figure 6.11(d)). The template **T** is transformed (e.g. translated) and positioned over the resulting DT image of **I**; the matching measure D(**T**, **I**) is determined by the pixel values of the DT image which lie under the data pixels of the transformed template. These pixel values form a distribution of distances of the template features to the nearest features in the image. The lower these distances are, the better the match between image and template at this location. There are a number of matching measures that can be defined on the distance distribution. One possibility is to use the average distance to the nearest feature. This is the *chamfer* distance, hence the name of the system. Other more robust (and costly) measures further reduce the effect of missing features (i.e. due to occlusion or segmentation errors) by using the average truncated distance or the f-th quantile value (the *Hausdorff* distance), e.g. Huttenlocher *et al.* (1993). In applications, a template is considered matched at locations where the distance measure D(**T**,**I**) is below a user-supplied threshold.

The advantage of matching a template with the DT image rather than with the edge image is that the resulting similarity measure will be smoother as a function of the template transformation parameters. This enables the use of an efficient search algorithm to lock onto the correct solution, as will be described shortly. It also allows some degree of dissimilarity between a template and an object of interest in the image.

The main contribution of the Chamfer System is the use of a template hierarchy efficiently to match whole sets of templates. These templates can be geometrical transformations of a reference template, or, more generally, be examples capturing the set of appearances of an object of interest (e.g. pedestrian). The underlying idea is to derive a representation off-line which exploits any structure in this template distribution, so that, on-line, matching can proceed optimized. More specifically, the aim is to group similar templates together and represent them as two entities: a 'prototype' template and a distance parameter. The latter needs to capture the dissimilarity between the prototype template and the templates it represents. By matching the prototype with the images, rather than the individual templates, a typically significant speed-up can be achieved on-line. When applied recursively, this grouping leads to template hierarchy, see Figure 6.12.

The above ideas are put into practice as follows. Off-line, a template hierarchy is generated automatically from available example templates. The proposed algorithm uses a bottom-up approach and applies a K-means-like algorithm at each level of the hierarchy. The input to the algorithm is a set of templates $\mathbf{t_1}, \ldots, \mathbf{t_N}$ their dissimilarity matrix and the desired partition size K. The output is the K-partition and the prototype templates $\mathbf{p_1}, \ldots, \mathbf{p_K}$ for each of the K groups $S_1, \ldots, S_K$. The K-way clustering is achieved by iterative optimization. Starting with an initial (random) partition, templates are moved back and forth between groups while the following objective function E is minimized:

$$E = \sum_{k=1}^{K} \max_{t_i \in S_k} \mathrm{D}(\mathbf{t_i}, \mathbf{p_K^*}) \tag{6.11}$$

Here, $\mathrm{D}(\mathbf{t_i}, \mathbf{p_K^*})$ denotes the distance measure between the ith element of group k, $\mathbf{t}_i$, and the prototype for that group at the current iteration, $\mathbf{p_K^*}$. The distance measure is the same as the one used for matching (e.g. chamfer or Hausdorff distance). One way of

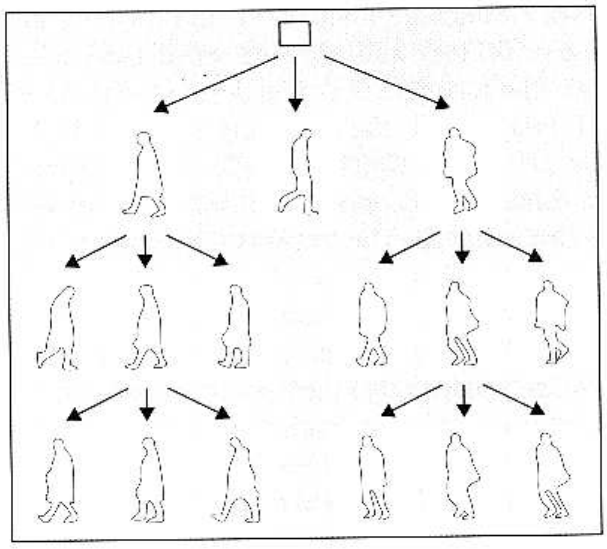


Fig. 6.12 A hierarchy for pedestrian shapes (partial view).

choosing the prototype $\mathbf{p}_K^*$ is to select the template with the smallest maximum distance to the other templates. $D(i, j)$ then represents the i-jth entry of the dissimilarity matrix, which can be computed fully before grouping or only on demand.

Note that a low E-value is desirable since it implies a tight grouping; this lowers the distance threshold that will be required during matching which in turn likely decreases the number of locations which one needs to consider during matching. *Simulated annealing* (Kirckpatrick *et al.*, 1993) is used to perform the minimization of the objective function E.

Online, matching can be seen as traversing the tree structure of templates. Each node corresponds to matching a (prototype) template with the image at some particular locations. For the locations where the distance measure between template and image is below a user-supplied threshold, one computes new interest locations for the children nodes (generated by sampling the local neighbourhood with a finer grid) and adds the children nodes to the list of nodes to be processed. For locations where the distance measure is above the threshold, the search does not propagate to the sub-tree; it is this pruning capability that brings large efficiency gains. Initially, the matching process starts at the root and the interest locations lie on a uniform grid over relevant regions in the image. The tree can be traversed in breadth-first or depth-first fashion. In the experiments, we use depth-first traversal, which has the advantage that one needs to maintain only $L - 1$ sets of interest locations, with L the number of levels of the tree. It is possible to derive an upper-bound on the distance threshold at each node of the hierarchy, such that one has the desirable property that using untruncated distance measures such as the chamfer distance, one can assure that the combined coarse-to-fine approach using the template hierarchy and image grid will not miss a solution (Gavrila and Philormin, 1999). In practice however, one can get away with using more restrictive thresholds to speed up detection.

A number of implementation choices further improved the performance and robustness of the Chamfer System, e.g. the use of oriented edge features, template subsampling, multi-stage edge segmentation thresholds and the incorporation of regions of interest

(e.g. ground-plane). Applying SIMD processing (MMX) to the main bottlenecks of the system, distance transform computation and correlation, resulted in a speed-up of a factor of 3–4.

Application: traffic sign detection

We tested the Chamfer System on the problem of traffic sign detection as an alternative to the colour system described above (Gavrila, 1999b). Specifically, we aimed to detect circular, triangular (up/down) and diamond-shaped traffic signs, as seen in urban traffic and on secondary roads. We used templates for circles and triangles with radii in the range of 7–18 pixels (the images are of size 360 by 288 pixels). This led to a total of 48 templates, placed at the leaf level of a three-level hierarchy. In order to optimize for speed, we chose to scale the templates (off-line), rather than scale the image (on-line).

We did extensive tests on the traffic sign detection application. Off-line, we used a database of several hundred traffic sign images, taken during both day- (sunny, rainy) and night-time conditions. Under good visibility conditions, we obtained high single-image detection rates, typically of over 95 per cent, when allowing solutions to deviate by 2 pixels and by radius 1 from the values obtained by a human. At this rate, there were a handful of detections per image which were not traffic signs, on average. These false positives were overwhelmingly rejected in a subsequent verification phase, where a RBF network (see Section 6.4) was used as pictograph classifier (the latter could distinguish about 10 pictographs); see Figure 6.13.

The traffic signs that were not detected, were either tilted or otherwise, reflected difficult visibility conditions (e.g. rain drops, partial occlusion by window wiper, direct sunlight into camera). Under the latter conditions, detection rates could decrease by as much as 30 to 65 per cent. We spent many hours testing our system on the road. The traffic sign detection (and recognition) system currently runs at 10–15 Hz on a 600 MHz Pentium processor with MMX. It is integrated in our Urban Traffic Assistant (UTA II) demonstration vehicle.

Application: pedestrian detection

The second application of the Chamfer System was the detection of pedestrians. Not surprisingly, it is more challenging than traffic sign detection since it involves a much larger variety of shapes that need to be accounted for. Furthermore, pedestrian contours are less pronounced in the images and more difficult to segment. Note that with a few exceptions much of the previous work on 'looking at people' has involved a static camera, see Gavrila (1999a) for an overview; initial segmentation was possible by background subtraction. That 'luxury' is not given to us here, because of a moving vehicle.

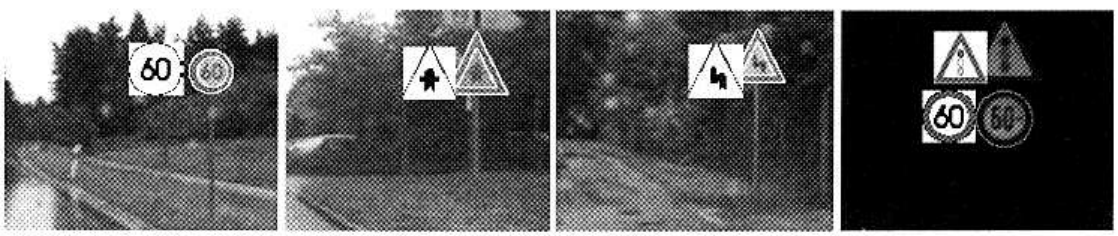


Fig. 6.13 Shape-based traffic sign detection (and recognition) with the Chamfer System.

We compiled a database of about 1250 distinct pedestrian shapes at a given scale; this number doubled when mirroring the templates across the y-axis. On this set of templates, an initial four-level pedestrian hierarchy was built, following the method described in Section 6.3.2. In order to obtain a more compact representation of the shape distribution and provide some means for generalization, the leaf level was discarded, resulting in the three-level hierarchy used for matching (e.g. Figure 6.12) with about 900 templates at the new leaf level, per scale. Five scales were used, covering a size variation of 50 per cent. Our preliminary experiments on a database of a few hundred pedestrian images (distinct from the sequences used for training) resulted in a detection rate of about 75–85 per cent per image, with a handful false-positives per image. These numbers are for un-occluded pedestrians. See Figure 6.14 for a few detection results.

Figure 6.15 shows some potential false positives; typically they are found on trees or windows. Using the flat-world assumption and knowledge about camera geometry, we

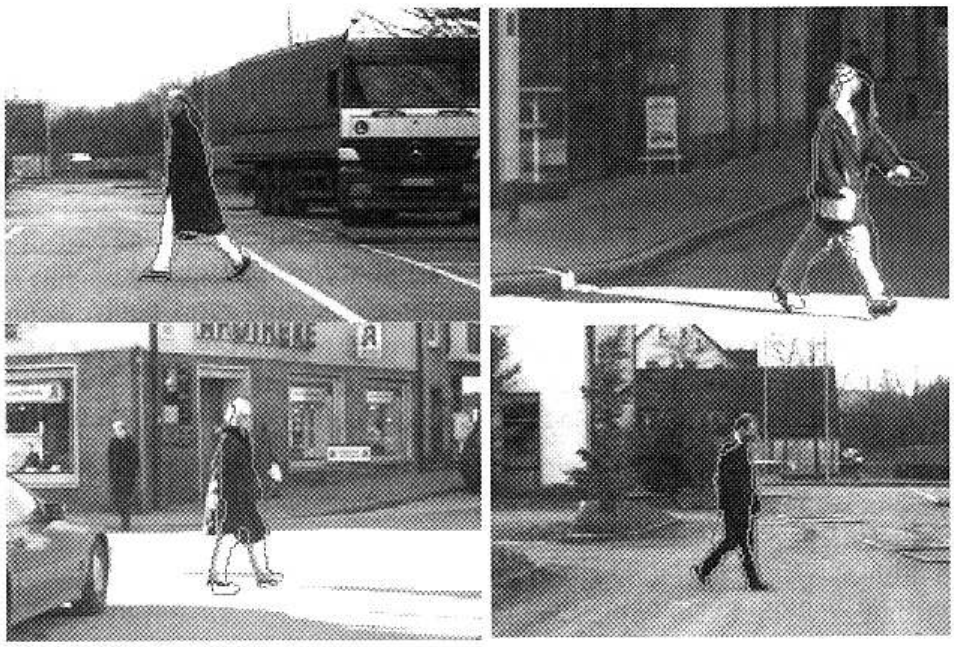


Fig. 6.14 Shape-based pedestrian detection with the Chamfer System.

Fig. 6.15 Potential false positives.

have set region of interests for the template in the hierarchy, so that many erroneous template locations can *a priori* be excluded, speeding up matching greatly. The current pedestrian system runs at 2–5 Hz on a 600 MHz Pentium processor with MMX. It is part of our pedestrian detection and recognition system that is described in Section 6.5.

6.3.3 Road recognition

The standard applications of road recognition are lane keeping and lane departure warning. In the context of a stop-and-go system, road recognition has additional relevance:

- If following a vehicle, lateral guidance can be accomplished by driving along the trajectory of the leading vehicle. If its distance and the angle between the leading vehicle and the heading direction is measured, a lateral tow bar controller can be used approximately to follow the trajectory. This controller tends to cut corners. Its performance can be improved if the position of the ego-vehicle relative to the lane is known (Gehrig and Stein, 1999).
- When the leading vehicle changes lane, a simple-minded following leads to hazardous manoeuvres. As long as the camera exclusively observes the field in front of the vehicle, collision avoidance is no longer guaranteed if the autonomous vehicle departs the lane. Thus, an intelligent stop-and-go system should be able to register lane changes of the leading vehicle and return the control to the driver.
- If the leading vehicle gets lost, but the vehicle's position in the lane is known, a lateral controller should guide the vehicle at least for a while, so that the driver has enough time to regain control.
- The response of the autonomous car to a detected object depends on the object's position in the road topology. A pedestrian on a crosswalk is clearly more critical than a pedestrian on a sidewalk.

In Section 6.2 a lane recognition system for highways is presented. The vision component of that system has a top-down architecture. A highly constrained model is matched against lane markings in image sequences. As pointed out, such model-based tracking is not only efficient since the considered image regions are small but also very reliable since the vehicle kinematics and the road geometry and its continuity are integrated in the model.

In the urban environment a global model for the broad range of possible road topographies is not available. The roads are characterized by:

- lane boundaries of which the shape and appearance are often discontinuously changing
- an arbitrary road shape
- a lack of a unique feature such as markings.

In this scenario, a pure tracking approach that simply extrapolates previously acquired information is not sufficient. The detection capability of data driven algorithms is required. Unfortunately, such bottom-up control strategies are computationally expensive since they have to process large image portions. Our urban lane recognition system combines a data driven approach with the efficiency of model-based tracking (Paetzold and Franke, 1998).

The data-driven global detection generates road structure hypotheses at a cycle rate just high enough to keep up with the dynamic environment. The model-based tracking estimates a dense description of the road structure in video real time, a requirement for comfortable lateral guidance. As common in multiple target tracking, the tracking results are referred to as tracks (Bar-Shalom and Fortmann, 1988). The detected hypothesis and the already existing tracks comprise a pool of tracks.

Both parts are fused such that the required computational power is minimized. The goal of global detection is the separation of road structures such as markings, curbs and crosswalks from heavy background clutter. Unlike highway lane boundaries, their local intensity distribution is not very distinctive. Rather, global geometric properties as length, orientation and shape are utilized. The detection schemes rely on the assumptions that:

- lane boundaries are long
- lane boundaries and crosswalks are parallel, stop lines are orthogonal to the vehicle's trajectory
- road structures have linear shape or can partially be approximated by lines
- road structures are bands of constant brightness
- road structures lie in the 3D-road surface (which is the ground-plane by default).

These global characteristics can be derived from a polygonal edge image which is well suited to describe the linear shapes of road structures, see bottom left image of Figure 6.16. For related work, see Enkelmann *et al.* (1995).

The detection of road structures is facilitated through an inverse-perspective mapping of the edge image into bird's eye view. By removing the perspective distortion, the geometrical properties of the road structures in 3D-world are restored, illustrated in top right image of Figure 6.16. In that representation, the data is scanned for subsets of

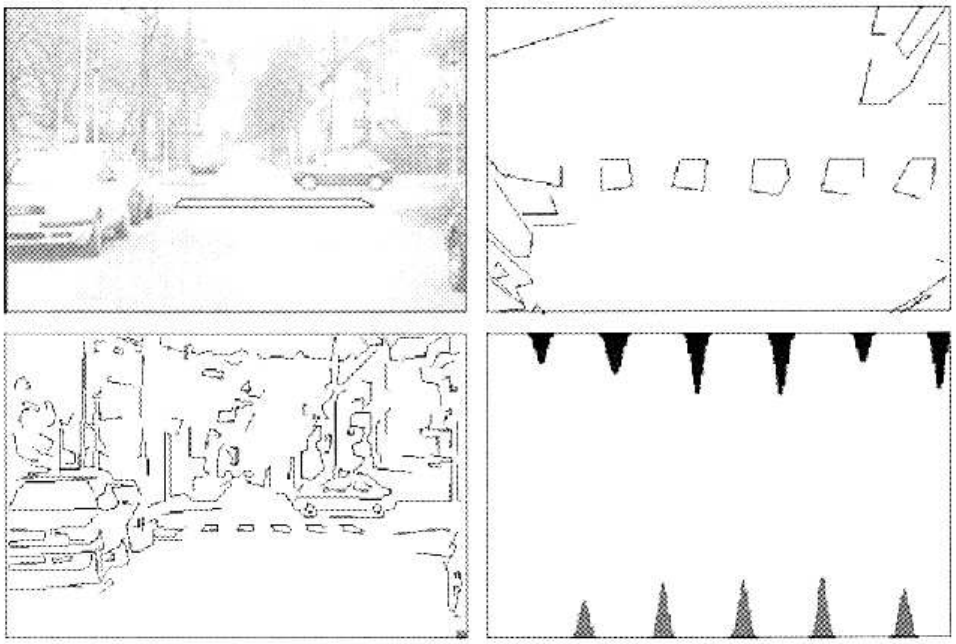


Fig. 6.16 Detection of road structures. Clockwise from left top: detected crosswalk in an urban scenario, bird's eye view on road, typical pattern of vertically projected edges in presence of a crosswalk, edge image.

features that are consistent with a simple feature model of the road structure, embodying the above listed characteristics.

To speed up this procedure, the polygon data is perceptually organized. The interesting intrinsic elements are length, orientation and position; the interesting clustering elements are parallelism and collinearity. The polygon data can be filtered for instances of these regularities in an arbitrary, iterative order to minimize the computational load. This is particularly crucial when we are interested in more than one object type that all have the same basic regularities in common.

Stereopsis helps to discard non-road measurements. Image regions where obstacles are present, detected as described in Section 6.3.1, are ruled out. Furthermore, measurements that do not lie in the 3D-road surface are suppressed. Stereopsis also enables the separation of vertical from horizontal shape and motion (pitching). Contrary to monocular vision where the pitch angle and the camera height are determined by assuming lanes of constant width, a model of the vertical geometry is recursively fitted through the stereo observations. Within the presented lane recognition system, the vertical model is linear and estimated by a first order Kalman filter.

The model underlying the tracking process must account for the arbitrary shape of urban lane boundaries. No appropriate geometric parameter model of low order exists that approximates the global road shape which can have sharp corners, discontinuities and curvature peaks. Therefore, local geometric properties such as lane tangent orientation and curvature are estimated by means of Kalman filtering.

This model definition is equivalent to a local circular fit to the lane boundary. Since a circle approximates arbitrary curves only for short ranges with negligible error, these properties are estimated not at the vehicle's position as done on highways but at a distance z_0 ahead. The system dynamics is given by

$$\begin{aligned} \dot{x}_0 &= -v\Delta\psi - \dot{\psi}_{\text{sensor}}\, z_0, \\ \Delta\dot{\psi} &= -vc_0 + \dot{\psi}_{\text{sensor}}, c < \frac{1}{80\,\text{m}} \\ \dot{c}_0 &= 0 \end{aligned} \tag{6.12}$$

The measurement equation is approximately given by

$$x_{\text{measured}}\ (z) = x_0 + (z - z_0)\Delta\psi + 1/2c_0(z - z_0)^2 \tag{6.13}$$

where z is the considered distance. Both equations turn into the standard highway equations for $z_0 = 0$ and $c_1 = 0$.

For each track in the pool the state vector is updated recursively. At each tracking cycle the track pool is subject to initiation, assessment, merging, verification, classification and deletion. The track assessment is the central operation. Its objective is to indicate whether a track is assigned to a lane boundary or to background clutter. An appropriate track quality is determined by lowpass-filtering the ratio of successful measurements and attempted measurements. This measure is used to trigger initiation and deletion.

Verification and classification are rule-based components. All tracks that are markings, parallel to already verified tracks or parallel to the vehicle's trajectory for a certain travelled distance, are labelled as verified. Verified tracks are classified into left/right

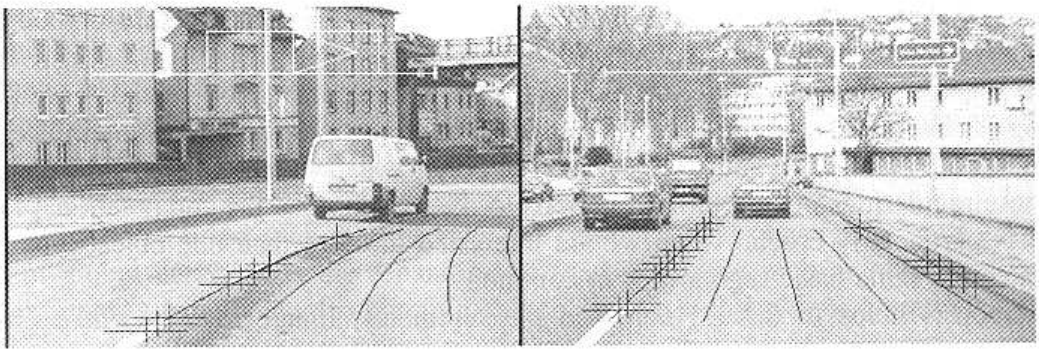


Fig. 6.17 (a) The vehicle ahead is driving in the adjacent lane. In turns, estimated lanes give valuable information about the topological traffic situation. (b) The lane is defined by tracked broken lane markings and a kerb. The leading vehicle drives in that lane.

lane boundary, other lane boundaries and clutter by evaluating the relative position of the tracks to the vehicle.

In Figure 6.17(a), the broken lane markings are tracked. The possible driving corridor takes a right turn, bypassing the light truck that is driving in another lane. In Figure 6.17(b), a kerb and broken lane markings are tracked, which define the lane of the autonomous car as well as the leading vehicle. The lane is defined by broken lane markings on the left side and a kerb on the right side. The vehicle ahead is centred in the lane.

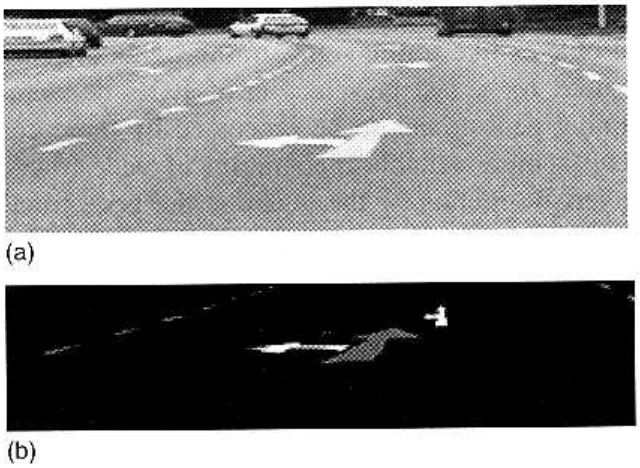
The detection of crosswalks also draws from the principles of a data-driven strategy. It consists of an early detector relying on spectrum analysis of the intensity image in horizontal direction and an edge-based verification stage.

The early detector observes the intensity function in multiple image lines evenly distributed over the interesting range of sight. Inspecting these intensity functions shows that a zebra crossing gives rise to a periodic black and white pattern of a known frequency. The power spectrum of that signal is calculated. When the spectral power at the expected frequency exceeds a threshold, the scene is subject to further investigation. By projecting the edges parallel and orthogonal to their predominant direction, marginal distributions are produced. Those marginal distributions exhibit distinctive patterns in presence of crosswalks, see Figure 6.16 where the parallel projection is displayed. These patterns are analysed by a rule-based system.

Arrow recognition

Besides lane course, stop lines and crosswalks, arrows painted on the road are of interest. Our recognition of those arrows follows the two-step procedure, detection and classification, mentioned earlier. In contrast to the lane recognition, shape and brightness cues are used in a region-based approach. The detection steps consist of grey-value segmentation and filtering.

An adaptive grey-scale segmentation reduces the number of colours in the original image to a handful. In this application, we base this reduction on the minima and plateaux of the grey-scale histogram. Following this grey-scale segmentation the colour connected components analysis described in Section 6.2.2 is applied to the segmented image. The algorithm produces a database containing information about all regions



(a)

(b)

Fig. 6.18 (a) Street scene displaying a direction arrow. (b) The segmented and classified arrow.

in the segmented image. Arrow candidate regions are selected from the database by appropriate queries and merged to objects of interest.

The resulting set is normalized for size and given as input to a radial-basis-function classifier (see Section 6.4). It has been trained to about a thousand different arrow images taken under different viewing angles and lighting conditions. Figure 6.18 shows the original and the obtained result.

6.4 Object recognition as a classification problem

6.4.1 General aspects

Principles, pros and cons

In Section 6.3 we described methods for object detection. They yield information about the position and motion behaviour of an object, but we still have to find out about what type of object is concerned. The latter task is what we call *object recognition*.

After detecting an object that fulfils certain simple criteria concerning e.g. size and width-to-height ratio and eventually motion, the image region of interest (ROI) delivered by the detection stage is first cropped and scaled to a uniform size; eventually, a contrast normalization is performed afterwards. Our general approach is then *training instead of programming*; we thus regard the object recognition problem as a classification problem to be solved by classification techniques that all require a training procedure based on a large number of examples. The advantage of this approach is that no explicit models of the objects to be recognized have to be constructed, which would be a rather difficult, if not impossible task especially for largely variable objects, e.g. pedestrians. Robust recognition systems are obtained by performing bootstrapping procedures, i.e. beginning with a certain set of training samples, then testing the resulting system in the real-world environment and re-training the recognition errors in order to generate a new version of the system that then has to undergo the same procedure, and so on. The disadvantage of the model-free approach is of course that huge amounts of training and test data have to be acquired and labelled in order to be able to achieve a high generalization performance.

Our object recognition algorithms are based on the classification of single images when regarding rigid objects like traffic signs, traffic lights or rear views of cars; this approach is as well used for a fast preselection of ROIs delivered by the detection stage possibly containing pedestrians, to be processed by more complicated methods afterwards. To achieve a more reliable recognition of pedestrians, image sequences displaying the characteristic motion of the legs, i.e. the pedestrian's *gait pattern*, are classified as a whole by employing the novel *adaptable time delay neural network* (ATDNN) concept based on spatio-temporal receptive fields. This neural network is used both as a 'standalone' classification module and for computationally efficient dimension reduction of high-dimensional input feature vectors. The latter technique makes it possible to employ standard classification techniques like *polynomial classifiers* (Schürmann, 1996) and *support vector machines* (Schoelkopf, 1999) as well as a *radial basis function network* algorithm specially designed for the recognition of traffic signs (Kreßel *et al.*, 1999).

Description of the classification techniques

The **polynomial classifier** as described in Schürmann (1996) constructs a so-called *polynomial structure list* of multiplicative features from the original input features in the first layer as shown in Figure 6.19. These are referred to as *enhanced features*. The second layer is a linear combination of the enhanced features defined by the coefficient matrix W. While the polynomial structure list must be chosen by the designer of the classifier (e.g. complete quadratic as in Figure 6.19), the adaptation of the weight matrix W is performed using the training set. An important advantage of the polynomial classifier is that there exists an analytical solution for adapting the weight matrix W, if the *empirical risk*, i.e. the average over the training set of the sum of square differences between the actual classifier outputs and the corresponding desired values is minimized.

The **support vector machine (SVM)** (see e.g. Scholkopf, 1999) in its elementary form is a classification concept for two-class problems that constructs a hyperplane in feature space that separates the samples of the two different classes in a manner that is optimal in the sense that the euclidean distance between the samples and the separating plane is as large as possible. The underlying concept is that of *structural risk minimization*. In the context of perceptron learning, the perceptron whose weight vector defines this optimal hyperplane is called the *perceptron of maximum stability*; it is obtained by a special training procedure called the *AdaTron algorithm* (Anlauf and Biehl, 1990). The hyperplane is defined in terms of the training samples situated nearest to it only; these training samples are therefore called *support vectors*. As most

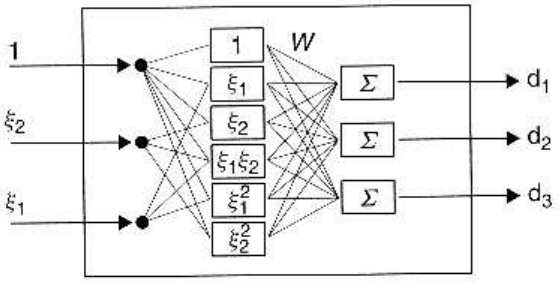


Fig. 6.19 Complete quadratic polynomial classifier for two features and three classes.

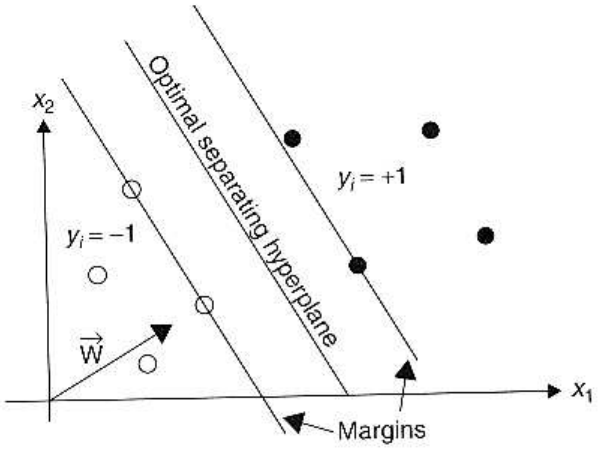


Fig. 6.20 Optimal separating hyperplane for a two-class problem.

realistic problems are not linearly separable in input space, i.e. it is impossible to find a hyperplane that perfectly separates the two classes in input space, a feature space of a usually much higher dimension is generated from the input space by a nonlinear transformation. The separating plane is constructed in this feature space; special procedures exist for handling distributions of training patterns that are still not linearly separable after transformation into feature space. Throughout this section, we will make use of the so-called *polynomial SVM* of a given order d, the feature space of which is spanned by polynomial combinations of the input features of up to dth order. Compared to other classification algorithms, the SVM concept yields a rather high generalization performance especially in the case of problems involving a relatively low number of training samples.

For the application of traffic sign recognition we developed a special **radial basis function (RBF)** classifier introduced in Kreßel *et al.* (1999). It consists of N reference vectors in feature space to each of which an object class c_i and two parameters a_i and b_i with $a_i < b_i$ are assigned. The number of object classes is denoted by K. For an unknown sample fed into the classifier, all N euclidean distances d_i in feature space between the sample and the reference vectors are calculated. The decision to which class the sample belongs is based on the value $R(d_i)$ which we define as:

$$R(d_i) = \begin{cases} 1 & \text{if} \quad d_i \leq a_i \\ \dfrac{b_i - d_i}{b_i - a_i} & \text{if} \quad a_i < d_i < b_i \\ 0 & \text{if} \quad d_i \geq b_i \end{cases} \tag{6.14}$$

The ramp function $R(d_i)$ is a radial basis function as it only depends on the euclidean distance d_i. In classical RBF classifiers (e.g. Poggio and Girosi, 1990) gaussians are used as radial basis function but the described ramp functions are more suitable for real-time applications due to their high computational efficiency. To be able to decide to which class the input sample has to be assigned we set:

$$\tilde{P}_k = \sum_{i=1, c_i=k}^{N} R(d_i), \quad S_{\tilde{P}} = \sum_{k=1}^{K} \tilde{P}_k \tag{6.15}$$

As a measure for the probability that the sample belongs to class k we then define:

$$P_k = \begin{cases} \tilde{P}_k / S_{\tilde{P}} & \text{if} \quad S_{\tilde{P}} > 1 \\ \tilde{P}_k & \text{if} \quad S_{\tilde{P}} \leq 1 \end{cases} \tag{6.16}$$

and as a measure for the probability that the sample belongs to none of the object classes (*reject probability*):

$$P_{\text{reject}} = \begin{cases} 1 - S_{\tilde{P}} & \text{if } S_{\tilde{P}} \leq 1 \\ 0 & \text{if } S_{\tilde{P}} > 1 \end{cases} \tag{6.17}$$

The input sample is assigned to the class with the highest P_k value; if P_{reject} is larger than all P_k values, the sample is assigned to an additional reject class. The sample is as well rejected if the highest P_k value is lower than a given threshold t with $0 < t < 1$. Varying t and measuring the rate of rejected test samples yields the *receiver operating characteristics* (ROC) curve of the classifier.

The reference vectors of the RBF classifier are the centres of clusters which are derived from the training examples divided into the K training classes. They are determined by an agglomerative clustering algorithm. The ramp parameters a_i and b_i of the ith radial basis function are defined in terms of the distance to the nearest cluster centre of the same class, the distance to the nearest cluster centre of one of the other classes, the average mutual distance of all clusters within each class k and the corresponding average over all K classes. Details are given in Kreßel *et al.* (1999).

For classification of image sequences we developed the **adaptable time delay neural network (ATDNN)** algorithm presented in detail in (Wöhler and Anlauf, 1999a, b). It is based on a time delay neural network with spatio-temporal receptive fields and adaptable time delay parameters. The general time delay neural network (TDNN) concept is well known from applications in the field of speech recognition (Waibel *et al.*, 1989). An important training algorithm for the TDNN, named *temporal backpropagation*, is presented in (Wan, 1990). A training algorithm for adaptable time delay parameters is developed in (Day and Davenport, 1998) for continuous time signals (*Continuous-time temporal backpropagation*). The related *Tempo 2 algorithm* is described in (Bodenhausen and Waibel, 1991) for input data defined at discrete time steps, as is the case for image sequences; the input window, however, is restricted to gaussian shape.

The architecture of the ATDNN is shown in Figure 6.21. The three-dimensional input layer receives a sequence of images acquired at a constant rate, as is the case especially for video image sequences. The activation of the input neuron at spatio-temporal position (x, y, t) corresponds to the pixel intensity at position (x, y) on the tth 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 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 $R_x \times R_y \times T_{\text{eff}}^{(l)}$ pixels in the input sequence with $T_{\text{eff}}^{(l)} = 1 + (R_t - 1)\beta$ 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 is given by D_x, D_y, and D_t, where we constantly take $D_t = 1$. For each neuron of layer 2 and higher,

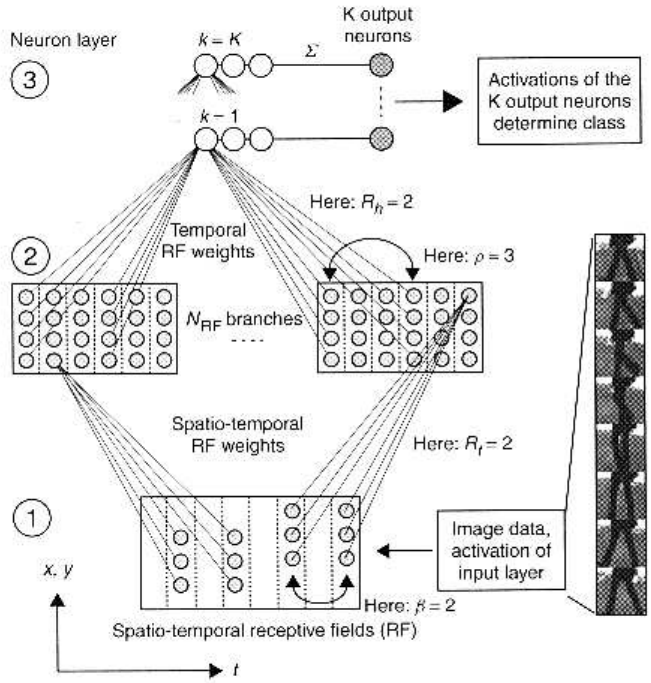


Fig. 6.21 Architecture of the adaptable time delay neural network.

we set $g(x) = \tanh(x)$ as a sigmoid activation function. The ATDNN is composed of NRF different branches, each of which consists of a three-dimensional layer of neurons (layer 2 in Figure 6.21). As we follow the *shared weights* principle inside each branch, the same set of weight factors is assigned to each layer 2 neuron of a certain branch. 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, however, 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 $T_{\text{eff}}^{(2)} = 1 + (R_h - 1)\rho$, R_h standing for the number of weight sets belonging to the same time slot, respectively. To each branch s and each output class k one shared set of temporal receptive field weights is assigned. The resulting activations of neuron layer 3 are then averaged classwise to obtain the output activations ω_k for the K output classes to which the ATDNN is trained.

The ATDNN as shown in Figure 6.21 is only defined for integer-valued time delay parameters β and ρ. The output of the ATDNN for real-valued time delay parameters is obtained by bilinear interpolation between the output values resulting from the neighbouring integer-valued time delay parameters. We use a backpropagation-like on-line gradient descent rule to train the ATDNN weights and time delay parameters, referring to a standard quadratic error measure. Details can be found in Wöhler and Anlauf (1999a, b).

Techniques for dimensionality reduction

The comparably high dimensionality of images or image sequences to be classified poses difficulties for many standard pattern recognition techniques, a problem

sometimes known as the 'curse of dimensionality'. It is therefore often necessary to reduce the dimensionality of the patterns, preferably by techniques that conserve class-specific properties of the patterns while discarding only the information that is not relevant for the classification problem.

A well-known standard method for dimensionality reduction is the principal component analysis (PCA) algorithm (for a thorough introduction, see e.g. (Diamantaras and Kung (1996), Schürmann (1996)). In a first step, the covariance matrix C of the distribution of training patterns is computed. The size of C is $N \times N$, where N denotes the dimension of the feature space in which the original patterns are defined. Then the N eigenvalues and corresponding eigenvectors of C are calculated and ordered according to the size of the eigenvalues; as C is necessarily positive semidefinite, all eigenvalues of C are non-negative. For further processing, the training patterns are then expanded with respect to the M most significant eigenvectors ('principal components') of C, i.e. the eigenvectors belonging to the M largest eigenvalues of C, neglecting the remaining $N - M$ eigenvectors. The obtained M expansion coefficients ($M < N$) are used as new features for classification. The basic property of the PCA algorithm is that it minimizes the Euclidean distance in the original N-dimensional feature space between an original pattern and its reconstructed version obtained by expansion with respect to the M principal components ('reconstruction error'). This does not necessarily mean that the information needed for classification is as well preserved in an optimal manner; in many practical applications, however, the PCA algorithm turns out to yield a very reasonable performance. Difficulties may again arise in the case of very high-dimensional patterns, i.e. large values of N, as this leads to problems concerning numerical stability when trying to diagonalize the covariance matrix C by standard numerical methods such as the Jacobi algorithm.

A further technique for dimensionality reduction that is specially adapted to process image or image sequence data consists of an extension of the previously described ATDNN algorithm. Apart from using the ATDNN 'standalone' as a classification module after training, we can as well regard the activation values of the neurons in the second layer as feature vectors to be processed, e.g. by the first three described classification techniques. For this purpose we employ a slightly simplified version of the ATDNN with integer-valued temporal extensions of the receptive fields (see Wöhler and Anlauf, 1999a). The ATDNN then serves as a preprocessing module that reduces the dimension of the input patterns (Wöhler *et al.*, 1999a). Especially, we combine the ATDNN in this manner with the RBF classifier for traffic sign recognition and with support vector machines for pedestrian recognition.

6.4.2 Traffic lights and signs

Traffic signs

We have developed traffic sign recognition systems for highways as well as urban traffic. As an example, the recognition of circular traffic signs, i.e. speed limits, passing restrictions, and related signs for ending restrictions (see Figure 6.22) is presented here. Grey-scale images of size 360×288 pixels are the basis for the investigation. The extracted regions of interest are scaled to 16×16 pixels and normalized in contrast. The general traffic sign class can be split up into 5 or 12 subclasses as shown in Figure 6.23.

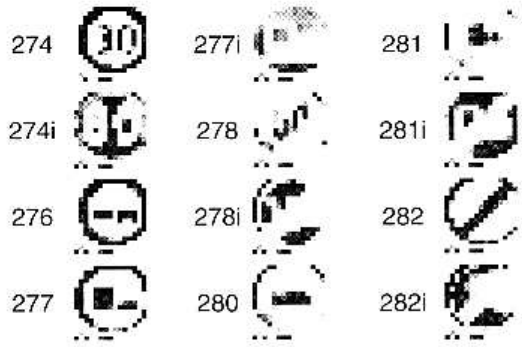


Fig. 6.22 The examined traffic sign classes with their labels according to German law.

label	#elements	13 classes	6 classes	2 classes
274	3787	class 01	class 1	class 1
274i	384	class 02		
276	1235	class 03	class 2	
277	178	class 04		
277i	34	class 05		
278	378	class 06	class 3	
278i	3	class 07		
280	118	class 08	class 4	
281	40	class 09		
281i	14	class 10		
282	119	class 11	class 5	
282i	5	class 12		
garb	19473	class 13	class 6	class 2
sum	25768			

Fig. 6.23 Composition of the training set.

The RBF classifier is trained to the five traffic sign subclasses and one garbage class only. For RBF networks the number of subclasses does not influence the recognition performance or the computational complexity of a classification cycle. A two-dimensional version of the ATDNN for processing single images is used to reduce the dimension of the original input patterns from 256 to a value of 64.

A combination of principal component analysis (PCA) and polynomial classifier is used as a reference technique that is well known from applications in the field of text analysis, especially handwritten digit recognition (see Franke (1997)). The polynomial classifier is applied to the 2, 6, and 13 class split up; here it turned out that the recognition performance as well as the computational complexity is rising with the number of subclasses. The dimension of the original input patterns is reduced by PCA to values of 40 and 50, respectively.

Concerning the recognition performance, the most interesting point is the trade-off between the false positive rate versus the rate of traffic signs that are rejected or explicitly classified as garbage. The corresponding results obtained from about 7000 separate test samples are shown in Figure 6.24. The errors among the various traffic sign subclasses are always smaller than 0.1 per cent. As a general result it comes out that the polynomial classifier yields a slightly higher overall recognition performance

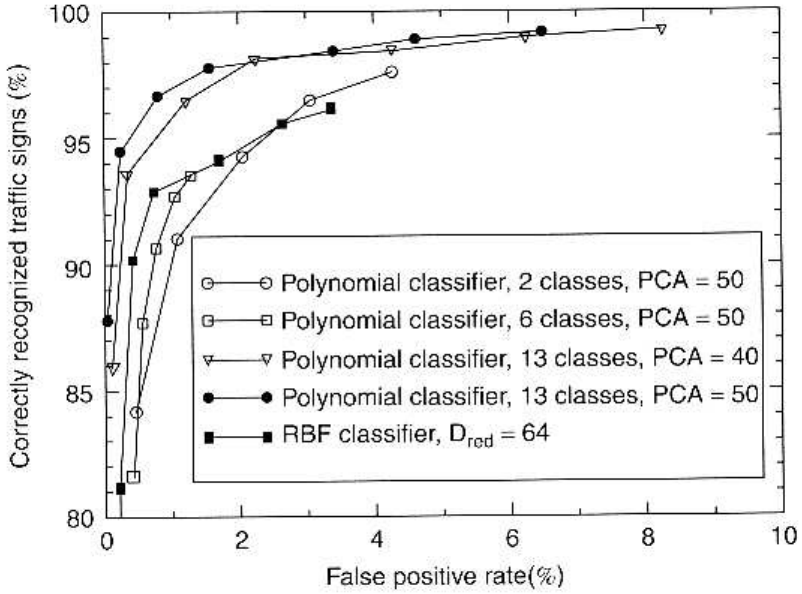


Fig. 6.24 ROC curves on the test set for several classifier settings.

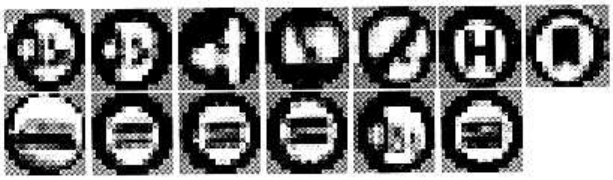


Fig. 6.25 The false positives yielded by a second degree polynomial classifier applied to 50 features obtained by PCA.

than the RBF classifier. For our real-time application, however, we choose so far the RBF network, since it is almost completely implemented by fast table look-ups such that one classification process needs only about 3 milliseconds of CPU time on the Pentium II system of our test vehicle. Moreover, it is simpler with the local RBF network method to take into account special garbage patterns which are very close to the speed limit signs in feature space. An example is the bus stop ('H') sign appearing in Figure 6.25. Adding a number of such examples to the training set creates a rather limited 'island' in feature space for this pattern with speed limit sign clusters around it.

The recognition of speed limits has to be further extended as generally one wants to know not only that there is a speed limit but also what is the maximum allowed speed. We thus added a second classification stage to read the inlays, i.e. the numbers on the traffic signs, which is activated when the first classification stage described above has recognized a speed limit sign. The dimension of the input patterns is again reduced by the two-dimensional ATDNN version for single images. One classification process needs only about 0.5 milliseconds of CPU time on the Pentium II system of our test vehicle. For a rate of 0.4 per cent of incorrectly classified inlays, 92 per cent of the inlays are correctly classified, with the remaining samples being rejected.

Traffic lights

Traffic lights are detected using the colour segmentation techniques described in Section 6.5. The detection stage is extracting blobs in the traffic light colours red, yellow and green; around each blob a region of interest (ROI) is cropped that contains not only the blob itself but also, if a traffic light has been detected, the dark box around it. The size of the ROI is thus related to the size of the detected blob. As the colour of the traffic light candidate is already known by the segmentation procedure, classification is performed based on the grey-scale version of the ROI only.

The ROI is first scaled to a size of 32×32 pixels. In order to enhance the contrast between the box of the traffic light and the background, which is often very weak, we perform a local contrast normalization by means of a simulated Mahowald retina (see Mead (1985)). We have three training and test sets, one for red, one for yellow, and one for red-yellow traffic lights, as shown in Table 6.1.

As our colour camera displays most green traffic lights as white blobs we could not generate a large set of well-segmented green traffic lights; on the detection of a green blob, we thus flip the corresponding ROI at its horizontal axis and classify this flipped ROI with the module designed for red traffic lights. This workaround will of course become obsolete with a high dynamic range colour camera of a sufficiently high resolution.

The recognition performance of the three classification modules for the different traffic light types is very similar, such that in this summary we only present the ROC curve of the classification module for red traffic lights. We compare the performance of the two-dimensional version of the ATDNN with spatial receptive fields of size $R_x = R_y = 13$ pixels, applied at an offset of $D_x = D_y = 6$ pixels, to the performance of a first and second order polynomial SVM and a linear polynomial classifier that have all been applied directly to the preprocessed ROIs of size 32×32 pixels (Figure 6.26). It becomes very obvious that with respect to the recognition performance, the 'local' ATDNN concept of spatial receptive fields is largely superior to 'global' approaches not explicitly taking into account the neighbourhood of the respective pixel. Both the first and the second order polynomial SVM separate the training set with only one error; we obviously observe overfitting effects due to systematic differences between training and test set which illustrate a very low generalization capability of the global approaches in this special application.

6.4.3 Pedestrian recognition

Single images

Possible pedestrians in the scene are detected by applying the stereo vision algorithm described in Section 3.1. Around each detected object an image area (ROI) of 1 m

Table 6.1

	Training set			Test set		
	Traffic lights	garbage	Sum	Traffic lights	garbage	Sum
Red	540	1925	2465	172	400	572
Yellow	292	1292	1584	80	400	480
Red-yellow	395	1292	1687	80	400	480

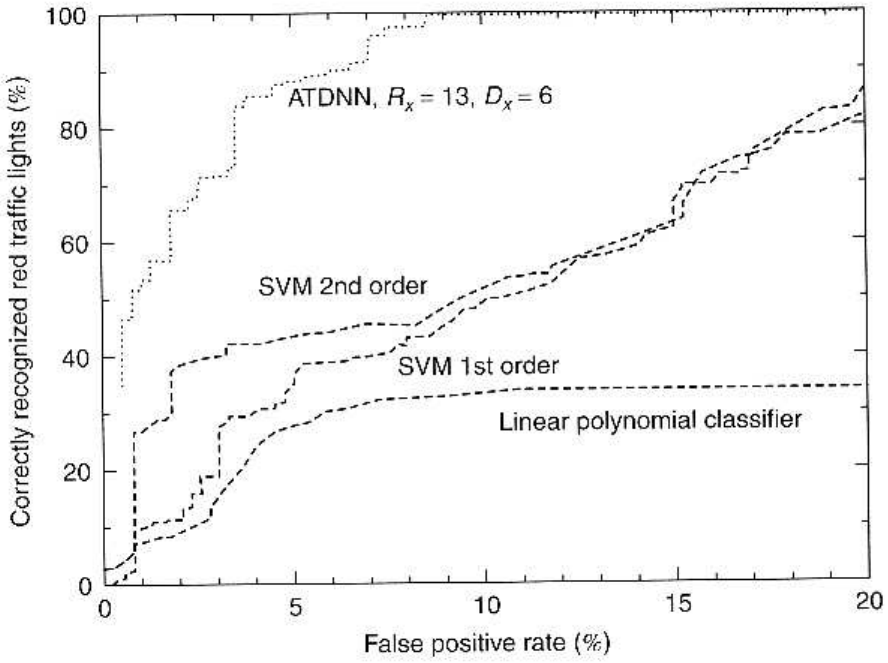


Fig. 6.26 Recognition performance of several classification modules for red traffic lights on the test set.

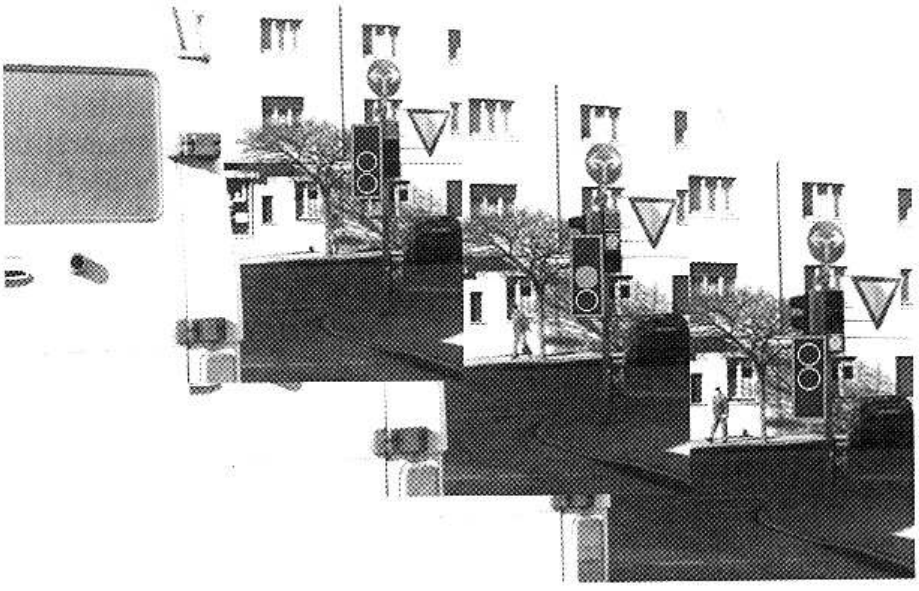


Fig. 6.27 Observing a traffic light at a crossing.

width and 2 m height, taking into account the object distance, is cropped, such that if the object is a pedestrian the resulting bounding box circumscribes it. These ROIs are scaled to a size of 24×48 pixels but not further preprocessed. Typical training samples are shown in Figure 6.28. The training set consists of 1942 pedestrian and 2084 garbage patterns, the test set of 600 pedestrian and 907 garbage patterns. The two-dimensional version of the ATDNN is again used for classification. Combining the

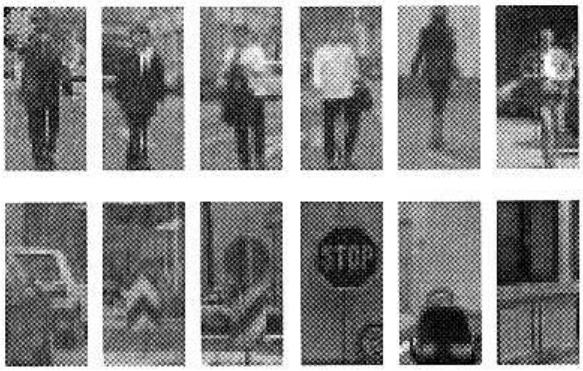


Fig. 6.28 Typical training samples for pedestrian recognition on single images.

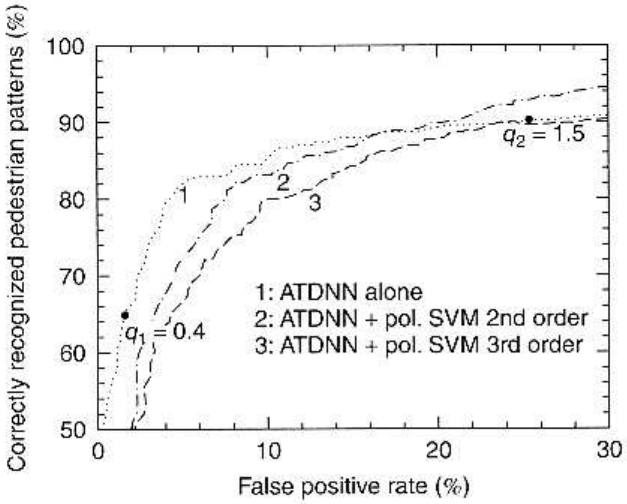


Fig. 6.29 ROC curves of the single image ATDNN on the test set.

ATDNN with a polynomial SVM of order 2 and 3 does not increase the recognition performance (see Figure 6.29).

The single image ATDNN tends to incorrectly classify vertically elongated shapes such as pedestrians to the extent that it should be combined with more complex approaches. The recognition result is thus only accepted if the network is rather 'sure' to have made a correct decision (see Figure 6.30), i.e. if the network output is lying well inside the pedestrian or the garbage region in decision space. Intermediate network outputs are regarded as a 'don't know' state. If the stereo vision algorithm detects no lateral motion of the object, the chamfer matching algorithm is activated, otherwise the full image sequence version of the ATDNN is used to more reliably classify the object based on combined shape and motion features. The single image ATDNN acts as a very fast preselection stage (the CPU time per classification process is about 1 ms) that eventually triggers computationally more complex classification modules. This decision structure is illustrated in Figure 6.30.

Image sequences

After detection of an object spatially emerging from the background by the stereo vision algorithm, we crop the lower half of the ROI delivered by the stereo vision

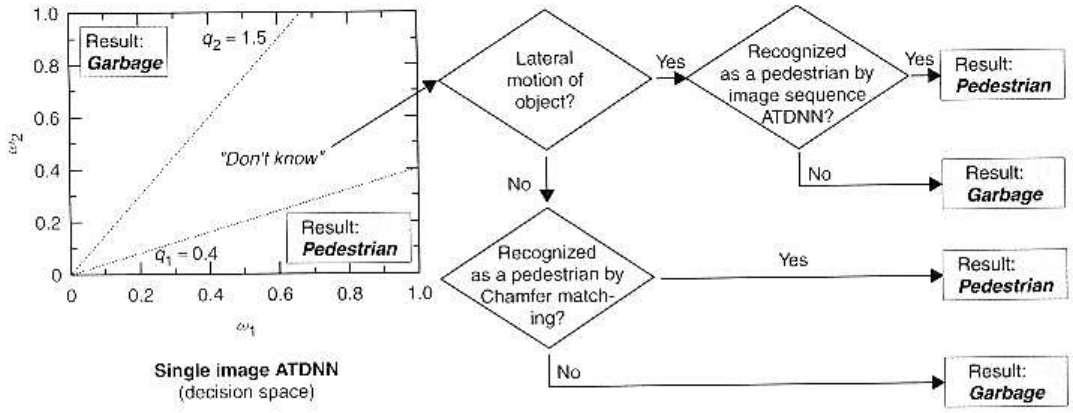


Fig. 6.30 Decision structure of the system for pedestrian recognition.

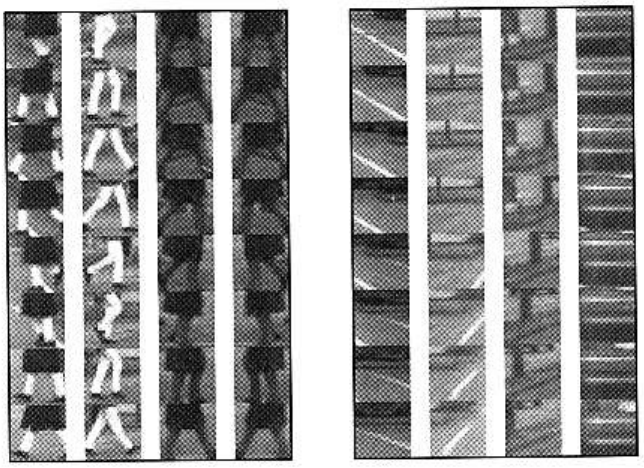


Fig. 6.31 Typical pedestrian (left) and garbage patterns (right).

algorithm, which will contain the pedestrian's legs, and normalize it to a size of 24×24 pixels. The time step between two subsequent frames of the sequence is 80 ms. 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 (see also Wöhler *et al.*, 1998). Our aim is again to distinguish between pedestrian and garbage patterns, resulting in two training classes typical representatives of which are shown in Figure 6.31. Our training set consists of 3926 pedestrian and 4426 garbage patterns, the test set of 1000 pedestrian and 1200 garbage patterns. It turned out that the performance on the test set is best for $N_{RF} = 2$ network branches and spatio-temporal receptive fields of a spatial size of $R_x = R_y = 9$ pixels, applied at an offset of $D_x = D_y = 5$ pixels. We performed seven training runs with different initial configurations of the time delay parameters, resulting in four 'optimal' ATDNN

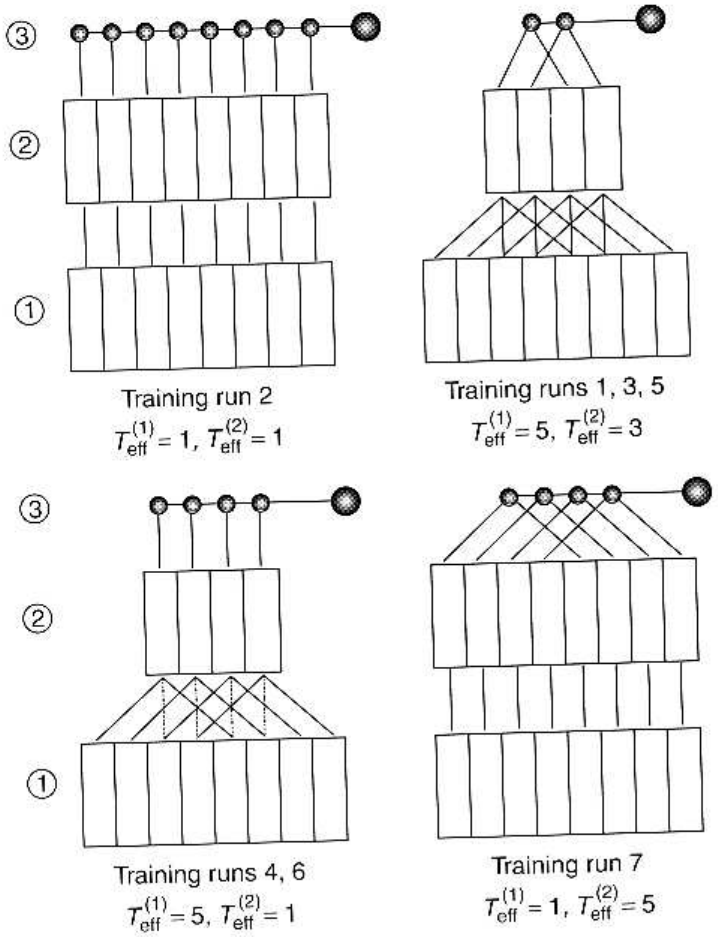


Fig. 6.32 ATDNN architectures corresponding to the approximate values of the learned temporal extensions of the respective fields, resulting from the seven performed training runs. The connections drawn as dashed lines only exist for $R_t = 3$.

architectures as depicted in Figure 6.32, the ROC curves of which are shown in Figure 6.33. Obviously, for the configuration $T_{\text{eff}}^{(1)} = T_{\text{eff}}^{(2)} = 1$ (training run 2) the lack of a temporal receptive field structure significantly reduces the recognition performance.

According to Wöhler and Anlauf (1999a), we trained a TDNN with fixed time delay parameters derived from the configuration on the upper right in Figure 6.32, the ROC curve of which is also shown in Figure 6.33. The performance could be slightly enhanced by combining this TDNN with a second and third order polynomial SVM (see Figure 6.35); the length of the feature vector processed by the SVM is reduced to $D_{\text{red}} = 128$.

The recognition rates as given by Figures 6.33 and 6.35 refer to single input patterns such that by integration of the results over time our system recognizes pedestrians in a very stable and robust manner. In Figure 6.34, typical scenes are shown. 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.

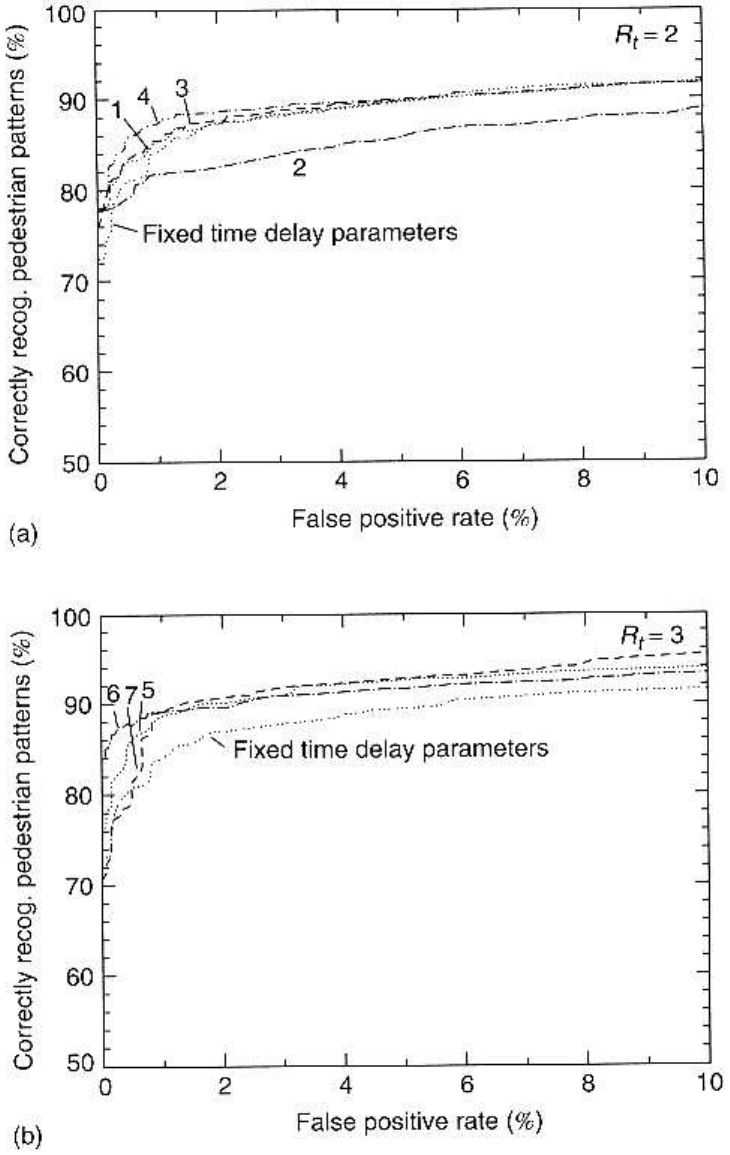


Fig. 6.33 ROC curves on the test set for the ATDNN, with $R_t = 2$ (above) and $R_t = 3$ (below). The ROC curve for the TDNN with $R_t = 5$ and $R_h = 3$ and fixed time delay parameters $\beta = \rho = 1$ is shown for comparison.

Global approaches versus local spatio-temporal processing

We will now compare the ATDNN approach based on local spatio-temporal feature extraction by receptive fields and its combination with polynomial SVMs to standard 'global' classification approaches, i.e. polynomial SVMs applied directly to the image sequences and after dimension reduction by principal component analysis (PCA).

For direct classification by a polynomial SVM the image sequence is regarded as a pixel vector of length $24 \times 24 \times 8 = 4608$. This approach is related to the one described in Papageorgiou and Poggio (1999), where SVM classifiers are applied to vectors consisting of temporal sequences of two-dimensional spatial Haar wavelet features. We adapted a second and third order polynomial SVM to the training set which could be perfectly separated in both cases. To reduce the very high dimension of the image sequences by PCA we took into account only the subspace of the

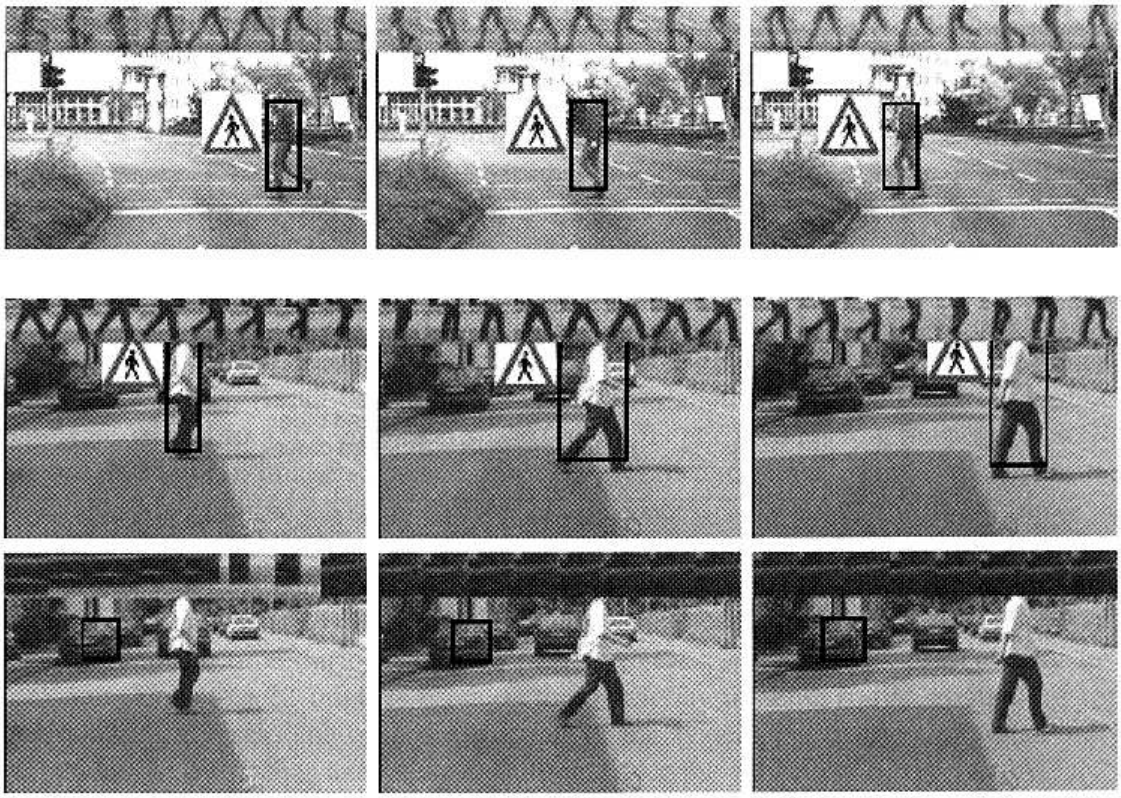


Fig. 6.34 Recognition of pedestrians in an urban traffic environment.

$D_{red} = 128$ most significant eigenvectors, which we obtained by using the well-known unsupervised perceptron learning technique known as 'Oja's rule' (Diamantaras and Kung, 1996). We then adapted a second and third order polynomial SVM to the correspondingly transformed training set, which again led to perfect separation in both cases. The local ATDNN-based classification approaches are somewhat superior to the global approaches with respect to their recognition performance on the test set (Figure 6.35). 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 local approaches are largely more efficient, as becomes obvious in Figure 6.36.

6.4.4 Further examples

Recognition of rear views of cars

The two-dimensional version of the ATDNN used for object recognition on single images as described in Sections 6.4.2 and 6.4.3 about traffic sign and traffic light recognition and pedestrian recognition is furthermore used for the recognition of the rear views of cars. This is an important application for intelligent stop-and-go, as before focusing on an object to follow that has been detected by the stereo vision algorithm, the system should be able to verify that it is indeed looking at the rear of a car. At a false positive rate of 3% (6%), a fraction of 80% (90%) of rear views of cars are correctly classified. The recognition errors occur in a temporally uncorrelated manner.

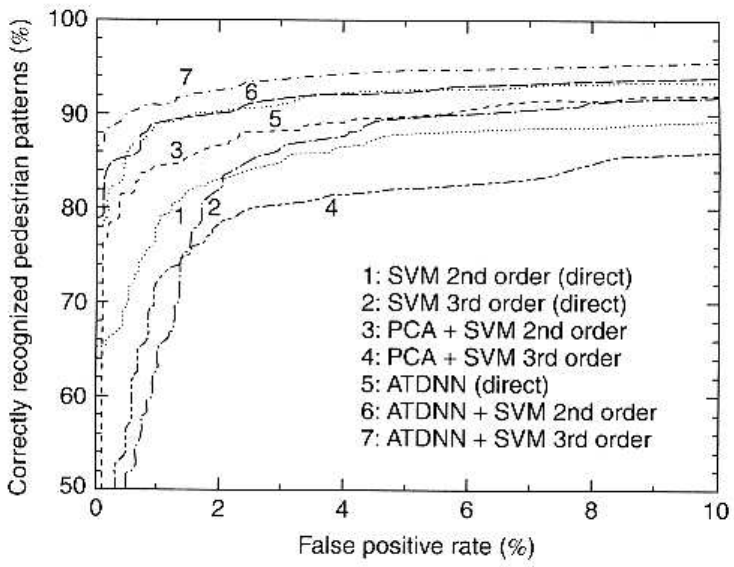


Fig. 6.35 ROC curves of the described classification modules for pedestrian recognition.

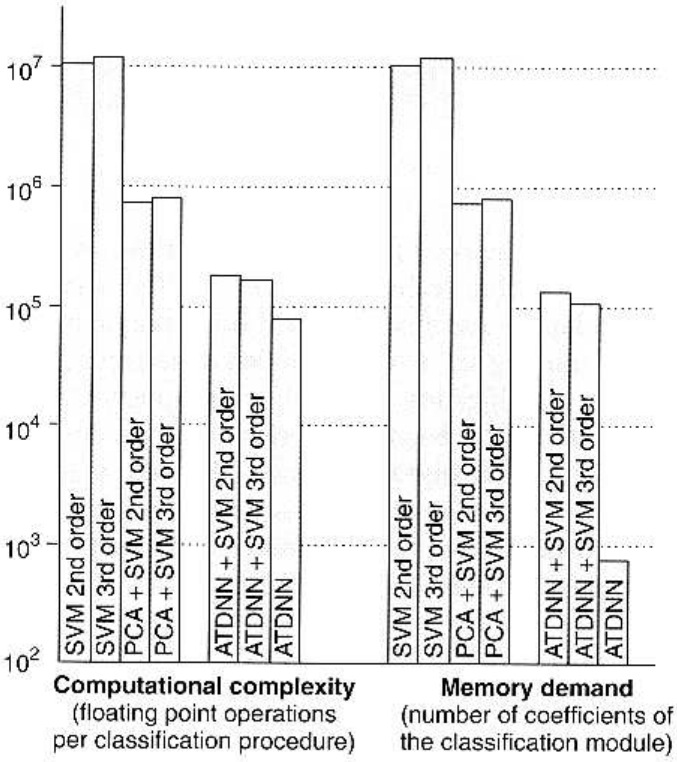


Fig. 6.36 Computational complexity (left) and memory demand (right) of the classification modules for pedestrian recognition. Note that the *y* axis is logarithmic.

As during autonomous car following the object to be followed is detected and tracked for a rather long time, integration of the single image recognition results over time yields a very stable behaviour of the system, recognizing all well-segmented cars and producing a hardly noticeable rate of false alarms. The module is again very fast as it requires less than 1 ms of CPU time per classification process.

Segmentation-free detection of overtaking vehicles

On motorways, our autonomous driving algorithm allows following the leading vehicle not only in stop-and-go traffic but also at speeds of up to about 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, assuming that the ego-vehicle is driving on the right lane. A single wide-angle camera is sufficient to perform this task; it is not necessary to employ stereo vision.

The input image sequences for the ATDNN are obtained by cropping a region of interest (ROI) sized 350 × 175 pixels at a fixed position out of the left half of each half frame, i.e. *no segmentation stage is involved*. This ROI is then downsampled to a size of 32 × 32 pixels. As an overtaking process takes about one second 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. An example of such an overtaking process as well as the ROC curve of our system is given in Figure 6.37. Here, the rate of correctly detected vehicles does not refer to single patterns but to complete overtaking processes; 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. Our test set corresponds to a 22 minutes drive on the motorway in dense traffic, containing 150 overtaking processes. The false positive rate of the system can be reduced by an order of magnitude by further analysing over time either the trajectory of the two ATDNN output values in decision space or the temporal behaviour of an appropriately averaged single output value by means of a simple second classification stage. This procedure is described in detail in Wöhler *et al.* (1999b).

6.5 Building intelligent systems

Driver assistance systems are challenging not only from the algorithmic but also from the software architecture point of view. The architectures of most driver assistant systems are usually tailored to a specific application, e.g. lane keeping on highways. The functionality is achieved by a few computer vision and vehicle control modules, which are connected in a hard-wired fashion. Although this kind of architecture is suitable for a lot of applications, it also has some disadvantages:

- The architecture is not scalable for a larger number of modules.
- There is no uniform concept for the cooperation of modules (e.g. for sensor fusion).
- New applications usually require extensive re-implementations.
- Reuse of old modules can be difficult due to missing interfaces.
- Hard-wired modules cause great efforts in maintenance since the dependencies of the modules are high. This is especially true for large systems.

The growing complexity of autonomous systems and our aim to realize a comprehensive assistance system hence reinforces the development of software architectures, which can deal with the following requirements:

- integration and cooperation of various computer vision algorithms
- different abstraction levels of perception and action

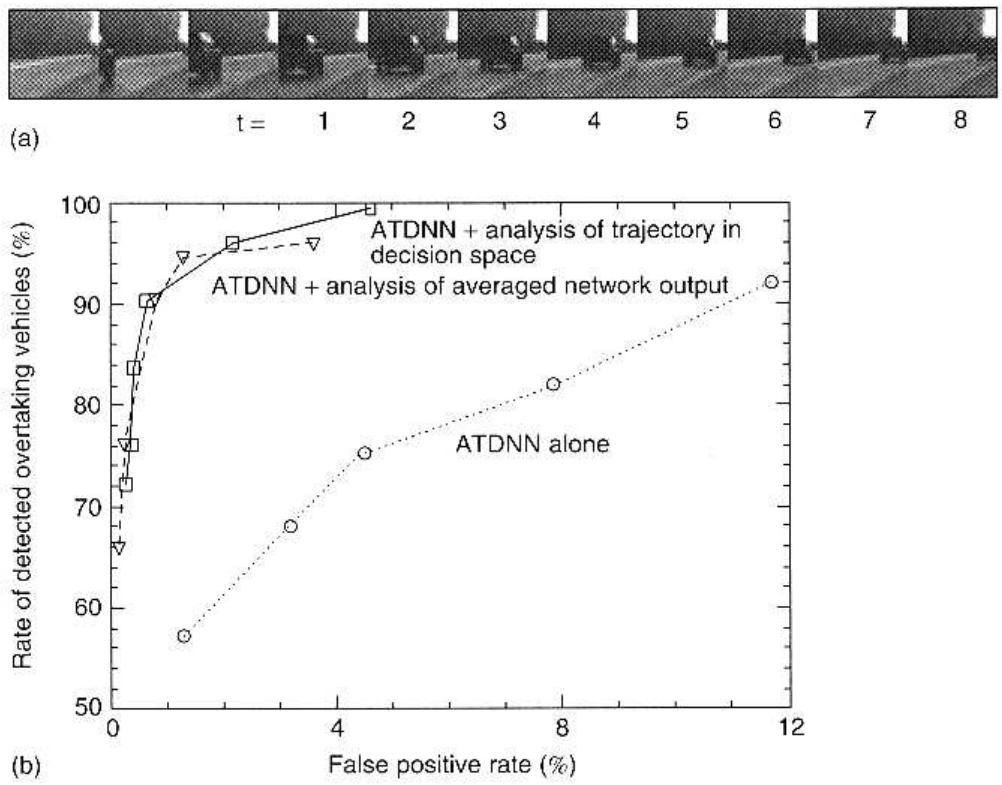


Fig. 6.37 (a) Typical example of an overtaking process. (b) Detection performance of the ATDNN with and without a second classification stage.

- sensor fusion
- economical use of resources
- integration of new algorithms without a complete redesign of the system
- simple enhancement to new computer vision applications
- distributed computing.

To meet these requirements, a multi-agent system was developed. In our demonstrator UTA II, the 'Agent NeTwork System' (ANTS) administrates computer vision, vehicle control and driver interface processes (Görzig and Franke, 1998). It selects and controls these algorithms and focuses the computational resources on relevant tasks for specific situations. For example, there is no need to look for new traffic signs or continuously determine the lane position, while the car is slowing down in front of a red traffic light.

6.5.1 ANTS: a multi-agent system

Agent software is a rapidly developing area of research. Since heterogeneous research is summarized under this term there is no consensus definition for 'agent' or 'multi-agent' system. A working definition of a multi-agent system (MAS) can be defined as 'a loosely-coupled network of problem solvers that work together to solve problems that are beyond their individual capabilities' (O'Hare and Jennings, 1996). The smallest entity of a MAS is an agent. An agent can be described as a computational entity, which provides services and has certain degrees of autonomy, cooperation and communication.

With these definitions it is not difficult to see how a MAS can be applied for autonomous vehicle guidance. Each computer vision or vehicle control module contains some functionality which can be useful for driver assistant systems. The combination of these modules allows more complex applications like autonomous stop-and-go driving in urban environments. The idea is to add the missing autonomy, cooperation and communication to the modules to create a MAS.

The main components of ANTS are a distributed data base, the administrators and the modules. Figure 6.38 visualizes the architecture.

The modules are the computational entities of the system, whereas the administrators contain the autonomy and the cooperation of the software agents. Combined with the communication ability of the distributed data base they build the MAS as described above. This distinction between a computational component and the 'intelligent' component of an agent has several advantages:

- The components can be executed in parallel.
- Existing software modules can be reused.
- The modification of one component does not necessarily require a modification of the other components.

Although ANTS is a generic MAS for various applications, we will focus in the following on driver assistance systems on our UTA II demonstrator and explain the main components in more detail.

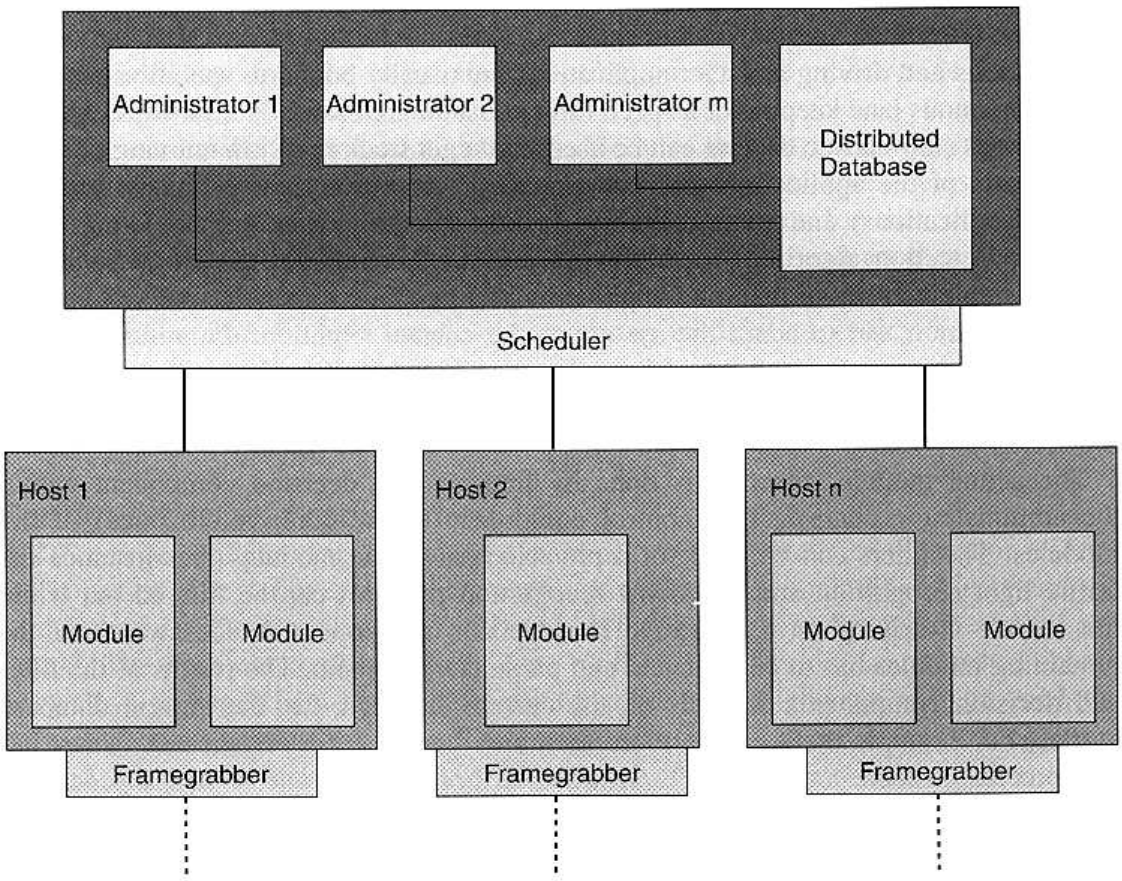


Fig. 6.38 The components of ANTS.

Distributed database

The central component of ANTS is the distributed database. It contains all incoming and outgoing data of the modules and distributes it to the computational nodes. This database typically contains information on the symbolic level and is used, e.g. for driver information or for the cooperation of modules.

The database has several access methods for its data such as exclusive write and concurrent read. For example, the stereo obstacle module can access its symbolically represented results in the exclusive write mode to track old and detect new obstacles, whereas the visualization module can access the results of all computer vision modules concurrently to submit important information to the driver at the same time.

Modules

Each module (like the vehicle control module or the stereo obstacle detection) represents a computational entity. They can be distributed transparently on the available computational nodes. The interface to a module encapsulates the in/out data, the configuration parameters and the module function calls. This is useful in many ways:

- Reuse of existing algorithms.
- Independence for the algorithm developers. They don't have to care about ANTS components like administrators or the distributed database.
- Application developers can reuse the existing modules to perform new applications without having detailed knowledge about the algorithms.

Administrator

Autonomous and driving assistant applications are usually bound to specific situations, e.g. autonomous lane keeping or a speed limit assistant are useful on highways, whereas autonomous stop-and-go driving can be used if you get stuck in a slow-moving tailback. Some parts of the applications are common (e.g. the obstacle detection is useful in several applications) and some are very specific for the current situation (e.g. there is a lane detection algorithm for highways and another one for the city). So if you want to implement more than one application you need a component to determine the current situation and to adapt the system to the current situation: the administrators. An administrator controls a set of modules (see Figure 6.39). He has to choose the modules, that have to be executed in the current situation, and to give them to the scheduler.

The actual module selection is done by filters and a decision component within the administrator control. This control component is the core 'intelligence' of the modules. The filters can be used for a pre-selection of the modules. For instance the traffic light recognition and the arrow recognition modules can be filtered out while the vehicle is driving on a highway. The decision component decides which of the remaining modules has to be executed and parameterizes them. The results of the filter and decision components depends on the current situation, i.e. the current database entries. For example there can be more than one module for a certain task: a fast but less precise obstacle tracking and a slower but more precise one. When a pedestrian enters the scene, the fast variant is used to focus the computational resources on the pedestrian detection. Otherwise, the slower one is more likely to be called.

The chosen modules are submitted to the scheduler. The scheduler communicates with the modules and executes the module tasks on the specified computational nodes.

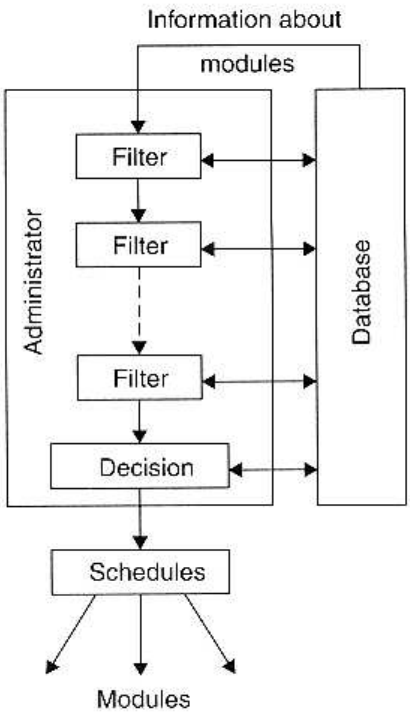


Fig. 6.39 Administrator.

Due to the real-time constraint, all processes have already started waiting for messages from the administrator. The initial static distribution of these processes is done in the start-up phase of ANTS using script files.

ANTS can handle several administrators. Modules that do not belong together are assigned to different administrators. This is useful to avoid complex decision components. In UTA II there is (among others) one administrator for the control of the computer vision modules, one to observe the system status, and another one for the driver interface modules.

An administrator can also handle exceptions in modules. A critical error within a module is submitted to the administrator. The administrator can now decide to stop the entire system, or just disable the affected module, in the case that it is not needed for safe vehicle guidance or if multiple modules are available for the same task.

Configuration and cooperation

ANTS can be configured statically and dynamically. For the static configuration a script file is parsed. The script causes the creation of the database including objects and the administrators. The modules are distributed on the available nodes, parameterized and started. The dynamic configuration is done by the administrators, as described above. They allow to switch between modules during runtime. On highways, as mentioned, the computer vision administrator can for example switch off the traffic light recognition. If you want your vehicle to park, ANTS can activate totally different modules.

ANTS allows several kinds of cooperation. A module can depend on results from other modules as well as results from several modules can be used to achieve a higher accuracy. Figure 6.40 shows some cooperation examples of administrators and modules we are using in UTA II to perform autonomous stop-and-go driving.

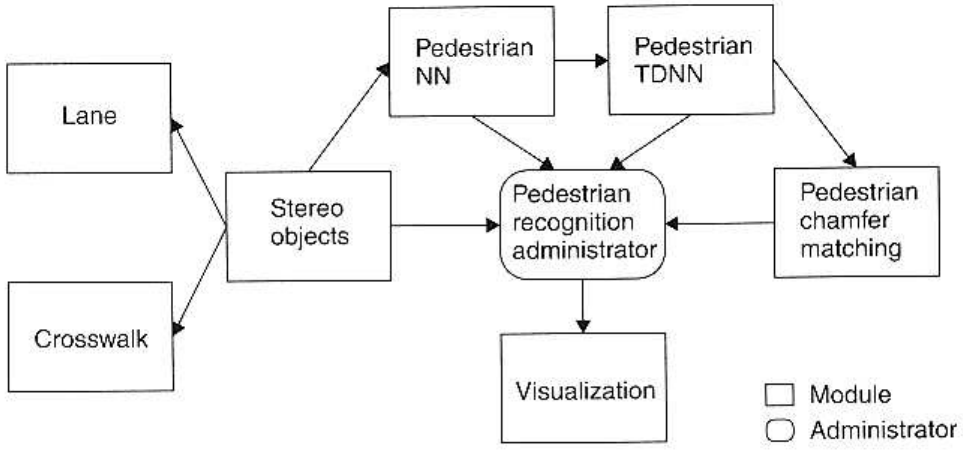


Fig. 6.40 Cooperation of modules and administrators.

The distributed database is used for data transfer (arrows). The stereo objects are transferred to the lane and to the crosswalk recognition. They mask the image to avoid wrong detections, e.g. car tail-lights as lanes. The pedestrian recognition uses more complex cooperation. Several pedestrian classifiers work together to improve the reliability of the results. The pedestrian recognition administrator receives results from the stereo module and organizes the recognition as described in Section 6.4.3.

6.5.2 UTA II on the road

The DaimlerChrysler demonstrator UTA II (Urban Traffic Assistant) was designed with special attention for information, warning and intervention systems in an inner-city environment (see Figure 6.41). UTA II is an E-class Mercedes containing sensors

Fig. 6.41 The demonstrator UTA II.

for longitudinal speed, longitudinal and lateral acceleration, yaw and pitch rate and the steering wheel angle. It is equipped with a stereo black/white camera-system as well as a colour camera. UTA II has full access to throttle, brake and steering.

The computer systems in UTA II are three 700 MHz Linux/Pentium III (SMP) PCs for the perception of the environment and one Lynx/604e PowerPC to control the sensors and actuators. So far, five administrators for computer vision, pedestrian recognition, driver interface (visualization), driving phase determination and a system status watchdog have been integrated.

Most of the integrated computer vision modules have been described above. Currently, the following modules can be activated:

- stereo-based object detection and tracking
- pedestrian recognition based on:
  - neural network (for standing pedestrians)
  - time delay neural network
  - chamfer matching
- lane detection and tracking:
  - on the highway
  - in the city
- traffic signs based on:
  - colour images (see Ritter *et al.*, 1995)
  - black/white images
- traffic light recognition
- recognition of road markings
- crosswalk recognition
- vehicle classification
- vehicle control (lateral/longitudinal)
- driver interface (2D/3D visualization).

Figure 6.46 shows a view out of UTA II. You can see the stereo camera system mounted behind the windscreen. The visualization on the monitor is enlarged in Figure 6.43.

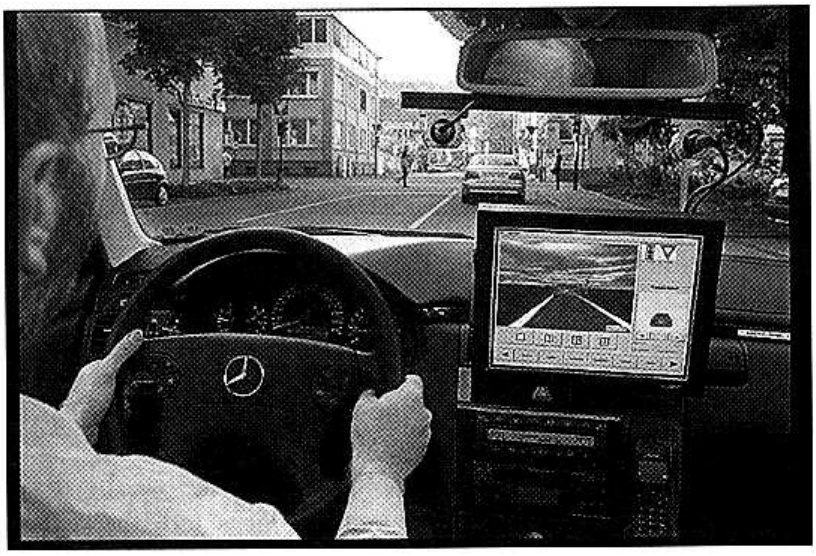


Fig. 6.42 View out of UTA II.

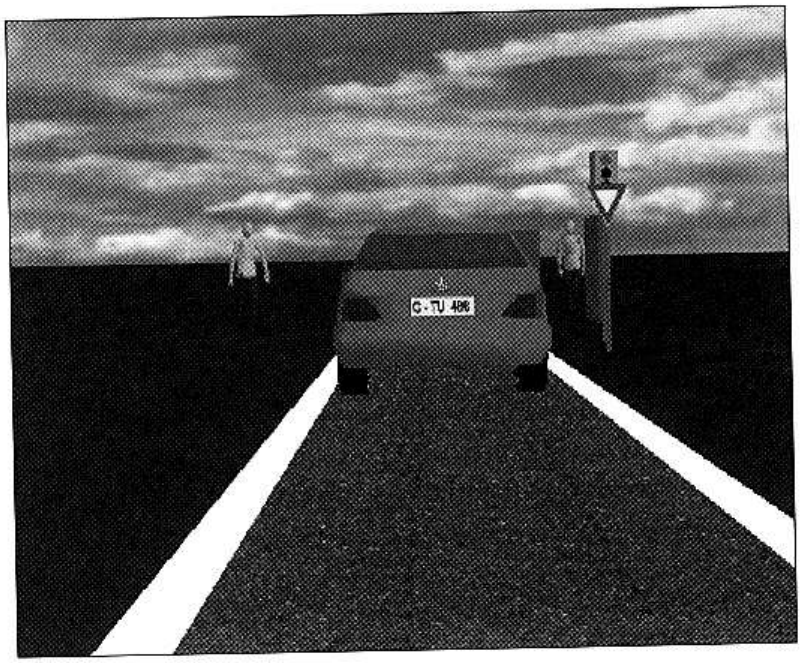


Fig. 6.43 Animated scene showing the recognized objects and their positions in the world.

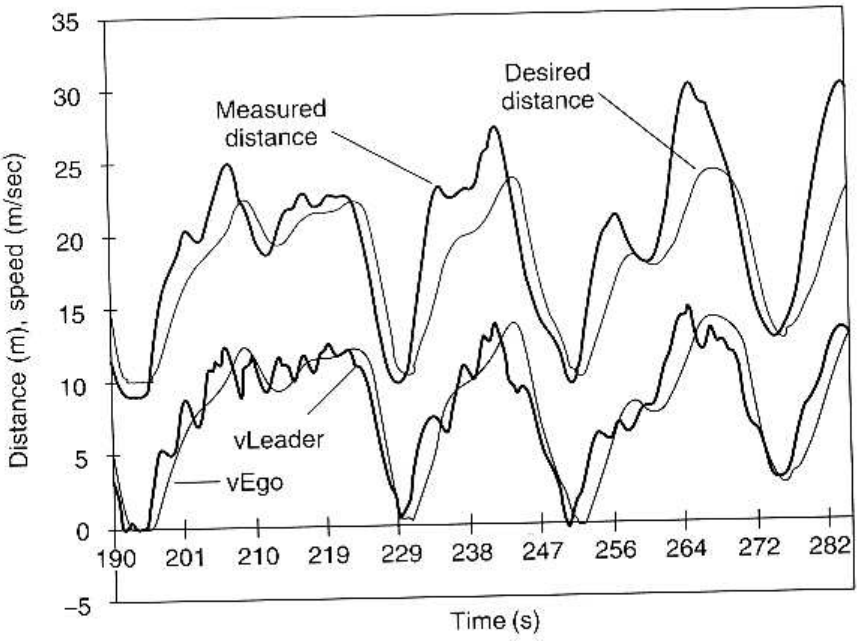


Fig. 6.44 Diagram of autonomous stop-and-go driving. Notice the small distance error when the leader stops.

It shows the objects recognized by UTA II: the detected lane, the obstacle in front classified as a car, obstacles classified as pedestrians, a traffic light and a traffic sign. Thanks to stereo vision, the scene reconstruction is geometrically correct.

The main application of UTA II is autonomous stop-and-go driving in an inner-city environment. Once the car in the visualization turns red, the driver can switch the system on. From now on, the own car follows the car in front laterally and longitudinally. Figure 6.43 shows the results of a test drive in the city of Esslingen,

Germany. The graph shows the measured and desired distance to the car in front as well as our own speed and the estimated speed of the lead vehicle. The latter distance is composed of a safety distance of 10 metres and a time headway of one second. The speed profile shows three stop-and-go cycles.

6.6 Summary

Over the past ten years, computer vision on board vehicles has evolved from rudimentary lane keeping on well structured highways to scene understanding in complex urban environments. In this chapter, we described our effort to increase the robustness on highways and to develop a next-generation cruise control called Intelligent Stop-&-Go, which takes into account relevant elements of the traffic infrastructure and other traffic participants while allowing autonomous vehicle control.

What did we learn during this time? At least three guiding principles have emerged for robust vision-based driver assistant systems:

First, vision in cars is vision over time. Kinematic and dynamic constraints applying to vehicles can be taken into account by means of Kalman filters. Obstacle candidates can be tracked over time. The repeated recognition stabilizes the decisions and allows the estimation of their motion state. A high imaging rate simplifies the establishment of object correspondences.

Second, stereo vision providing 3D information became a central component of robust vision systems. It allows the detection of arbitrary obstacles and the determination of their size and position. Monocular model based approaches as investigated in the early 1990s turned out to be less robust and reliable in the traffic scenario.

Third, object recognition can be considered as a classification problem. Powerful tools are at hand for the adaptation of generic classification schemes to a specific task, as described in Section 6.4. Developers are no longer forced to formulate heuristics but the relevant aspects of the considered objects are learned from representative examples.

In spite of the achieved success many problems related to reliability remain to be solved. Besides continuous improvement of the robustness of the image analysis modules, sensor problems have to be overcome. Standard CCD cameras lack the dynamic range that is necessary to operate in traffic under adverse lighting conditions (e.g. allowing the camera to capture structure in shadowed areas when exposed to bright light). CMOS camera technology can be of help.

As other information sources like radar, digital maps and communication become available in modern cars, their utilization will help to raise the performance of vision based environment perception. It will be a challenge to combine the power of each source in order to obtain a most reliable and complete interpretation of the current traffic situation.

Nevertheless, we are convinced that vision will be the key component of intelligent vehicles.

First vision products on board vehicles are already on the market: witness the Lane Departure Warning System available in Mercedes and Freightliner's trucks. Many more will undoubtedly follow.

Thanks to the foreseeable performance improvement, the future will see systems that assist the driver during his whole trip, from door to door.

References

Anlauf, J.K. and Biehl, M. (1990). The AdaTron: An Adaptive Perceptron Algorithm. *Europhysics Letters*, **10**, 687.

Bar-Shalom, Y. and Fortmann, T.E. (1988). *Tracking and Data Association*. Academic Press.

Bodenhausen, U. and Waibel, A. (1991). The Tempo 2 Algorithm: Adjusting Time Delays by Supervised Learning. In *Advances in Neural Processing Systems 3*, (R.P. Lippman, J.E. Moody and D.S. Touretzky, eds), pp. 155–61. Morgan Kaufmann, San Mateo, CA.

Broggi, A. (1997). Obstacle and Lane Detection on the ARGO. *IEEE Conference on Intelligent Transportation Systems ITSC'97*, Boston, 9–12, November.

Day, S.P. and Davenport, M.R. (1993). Continuous-Time Temporal Back-Propagation with Adaptable Time Delays. *IEEE Transactions on Neural Networks*, **4**, No. 2, 348–54.

Diamantaros, K. and Kung, S.Y. (1996). Principal Component Neural Networks, Wiley Interscience, New York.

Dickmanns, E.D. and Zapp, A. (1986). A curvature-based scheme for improving road vehicle guidance by computer vision. *Proceedings SPIE Conference on Mobile Robots*, Vol. 727, pp. 161–16.

Enkelmann, W. (1997). Robust Obstacle Detection and Tracking by Motion Analysis. *IEEE Conference on Intelligent Transportation Systems ITSC'97*, Boston, 9–12, November.

Enkelmann, W., Struck, G. and Geisler, J. (1995). ROMA – a system for model-based analysis of road markings. *IEEE Intelligent Vehicles Symposium*, September, pp. 356–60.

Faugeras, O. (1993). *Three-Dimensional Computer Vision*. MIT Press.

Franke, J. (1997). Isolated Handprinted Digit Recognition. In *Handbook of Character Recognition and Document Image Analysis*, (H. Bunke and P.S.P. Wang, eds), pp. 103–22, World Scientific, Singapore.

Franke, U., Böttiger, F., Zomotor, Z. and Seeberger, D. (1995). Truck platooning in mixed traffic, *Intelligent Vehicles '95*, Detroit, 25–26, September, 1995, pp. 1–6.

Franke, U. and Kutzbach, I. (2000). Fast Stereo based Object Detection for Stop & Go Traffic. *Intelligent Vehicles '96*, Tokyo, pp. 339–44.

Franke, U., Gavrila, D., Görzig, S., Lindner, F., Paetzold, F. and Wöhler, C. (1998). Autonomous Driving Goes Downtown. *IEEE Intelligent Systems*, Vol. 13, No. 6, September/October, pp. 40–48.

Franke, U. and Joos, A. (2000). Real-time Stereo Vision for Urban Traffic Scene Understanding. *IEEE Conference on Intelligent Vehicles 2000*, 3/4 October, Detroit.

Gavrila, D. (1999a). The visual analysis of human movement – a survey. *Computer Vision and Image Understanding*, Vol. 73, No. 1, 82–98.

Gavrila, D. (1999). Traffic sign recognition revisited. *Mustererkennung 1999*, W. Förstner, *et al.*, eds). Springer Verlag.

Gavrila, D. and Philomin, V. (1999). Real-time object detection for 'smart' vehicles. *Proceedings of IEEE International Conference on Computer Vision*, pp. 87–93, Kerkyra, Greece.

Gehrig, S. and Stein, F. (1999). Cartography and dead reckoning using stereo vision for an autonomous vehicle. In *SCA Eighth International Conference on Intelligent Systems*, Denver, Colorado.

Gern, A., Franke, U. and Levi, P. (2000). Advanced Lane Recognition – Fusing Vision and Radar. *Proceedings of IEEE Conference on Intelligent Vehicles*.

Görzig, S. and Franke, U. (1998). ANTS – Intelligent Vision in Urban Traffic. *Proceedings of IEEE Conference on Intelligent Transportation Systems*, October, Stuttgart, pp. 545–549.

Huttenlocher, D., Klanderman, G. and Rucklidge, W. (1993). Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**, No. 9, 850–63.

Janssen, R., Ritter, W., Stein, F. and Ott, S. (1993). Hybrid Approach for Traffic Sign Recognition. *Proceedings of Intelligent Vehicles Conference*.

Kirckpatrick, S., Gelatt, C. and Vecchi, M. (1993). Optimization by Simulated Annealing. *Science*, No. 220, pp. 671–80.

Kluge, K. and Thorpe, C. (1992). Representation and recovery of road geometry in YARF. *Proceedings IEEE Conference on Intelligent Vehicles*.

Kreßel, U., Lindner, F., Wöhler, C. and Linz, A. (1999). Hypothesis Verification Based on Classification at Unequal Error Rates. *International Conference on Artificial Neural Networks*, pp. 874–79, Edinburgh.

Mandler, E. and Oberländer, M. (1990). One-pass encoding of connected components in multivalued images. *Proceedings 10th International Conference on Pattern Recognition*, Atlantic City.

Mysliwetz, B. (1990). Parallelrechner-basierte Bildfolgen-Interpretation zur autonomen Fahrzeugsteuerung. Dissertation, *Universität der Bundeswehr München*, Fakultät für Luft- und Raumfahrttechnik.

Neußer, R., Nijhuis, J., Spaanenburg, L., Höfflinger, B., Franke, U. and Fritz, H. (1993). Neurocontrol for lateral vehicle guidance. *IEEE MICRO*, February, pp. 57–66.

O'Hare and Jennings, N. (1996). *Foundations of Distributed Artificial Intelligence*. John Wiley & Sons.

Paetzold, F. and Franke, U. (1998). Road Recognition in Urban Environment. *Proceedings of IEEE Conference on Intelligent Transportation Systems*, October, Stuttgart.

Papgeorgiou, C. and Poggio T. (1999). A Pattern Classification Approach to Dynamical Object Detection. *International Conference on Computer Vision*, pp. 1223–28, Kerkyra, Greece.

Poggio, T. and Girosi, F. (1990). Networks for Approximation and Learning. *Proceedings of the IEEE*, **78**, No. 9.

Ritter, W., Stein, F. and Janssen, R. (1995). Traffic Sign Recognition Using Colour Information. in *Mathematic Computation and Modelling*, **22**, Nos. 4–7, pp. 49–161.

Saneyoshi, K. (1994). 3-D image recognition system by means of stereoscopy combined with ordinary image processing. *Intelligent Vehicles'94*, 24–26, October, Paris, pp. 13–18.

Schölkopf, B., Burges, A. and Smola, A. (1999). *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA.

Schürmann, J. (1996). *Pattern Classification*. Wiley-Interscience, New York.

Ulmer, B. (1994). VITA II – Active collision avoidance in real traffic. *Intelligent Vehicles'94*, 24–26, October, Paris, pp. 1–6.

Waibel, A., Hanazawa, T., Hinton, G. and Lang, K.J. (1989). Phoneme Recognition Using Time-Delay Neural Networks. *IEEE Trans. Acoust., Speech, Signal Processing*, **37**, 328–339.

Wan, E.A. (1990). Temporal Backpropagation for FIR Neural Networks. *IEEE Internation Joint Conference on Neural Networks*, pp. 575–80, San Diego, CA.

Wöhler, C. and Anlauf, J.K. (1999). A Time Delay Neural Network Algorithm for Estimating Image-pattern Shape and Motion. *Image and Vision Computing Journal*, **17**, 281–94.

Wöhler, C. and Anlauf, J.K. (1999). An Adaptable Time Delay Neural Network Algorithm for Image Sequence Analysis. *IEEE Transactions on Neural Networks*, **10**, No. 6, 1531–6.

Wöhler, C., Anlauf, J.K., Pörtner, T. and Franke, U. (1998). A Time Delay Neural Network Algorithm for Real-Time Pedestrian Recognition. *IEEE International Conference on Intelligent Vehicles*, pp. 247–52, Stuttgart, Germany.

Wöhler, C., Kreßel, U., Schürmann, J. and Anlauf, J.K. (1999). Dimensionality Reduction by Local Processing. *European Symposium on Artificial Neural Networks*, pp. 237–44, Bruges, Belgium.

Wöhler, C., Schürmann, J. and Anlauf, J.K. (1999). Segmentation-Free Detection of Overtaking Vehicles with a Two-Stage Time Delay Neural Network Classifier. *European Symposium on Artificial Neural Networks*, pp. 301–6, Bruges, Belgium.

Ziegler, W., Franke, U., Renner, R. and Kühnle A. (1995). Computer Vision on the Road: A Lane Departure and Drowsy Driver Warning System. *4th Mobility Technology Conference*, SAE Brasil '95, Sao Paulo, Brasil, 2–4, October.

Zomotor, Z. and Franke, U. (1997). Sensor fusion for improved vision based lane recognition and object tracking with range-finders. *Proceedings of IEEE Conference on Intelligent Vehicles.*