The Driving School System: Learning Basic Driving Skills From a Teacher in a Real Car

Article *in* IEEE Transactions on Intelligent Transportation Systems · January 2012

DOI: 10.1109/TITS.2011.2157690 · Source: IEEE Xplore

CITATIONS
8

READS
28,864

11 authors, including:

Nikolay Chumerin
KU Leuven
37 PUBLICATIONS **625** CITATIONS
SEE PROFILE

Aušra Vidugirienė
Vytautas Magnus University
28 PUBLICATIONS **77** CITATIONS
SEE PROFILE

Minija Tamosiunaite
Universitätsmedizin Göttingen
78 PUBLICATIONS **782** CITATIONS
SEE PROFILE

Marc Van Hulle
KU Leuven
343 PUBLICATIONS **4,738** CITATIONS
SEE PROFILE

The Driving School System: Learning Basic Driving Skills From a Teacher in a Real Car

Irene Markelić, Anders Kjær-Nielsen, Karl Pauwels, Lars Baunegaard With Jensen, Nikolay Chumerin, Aušra Vidugiriene, Minija Tamosiunaite, Alexander Rotter, Marc Van Hulle, Norbert Krüger, and Florentin Wörgötter

***Abstract*—To offer increased security and comfort, advanced driver-assistance systems (ADASs) should consider individual driving styles. Here, we present a system that learns a human's basic driving behavior and demonstrate its use as ADAS by issuing alerts when detecting inconsistent driving behavior. In contrast to much other work in this area, which is based on or obtained from simulation, our system is implemented as a multithreaded parallel central processing unit (CPU)/graphics processing unit (GPU) architecture in a real car and trained with real driving data to generate steering and acceleration control for road following. It also implements a method for detecting independently moving objects (IMOs) for spotting obstacles. Both learning and IMO detection algorithms are data driven and thus improve above the limitations of model-based approaches. The system's ability to imitate the teacher's behavior is analyzed on known and unknown streets, and results suggest its use for steering assistance but limit the use of the acceleration signal to curve negotiation. We propose that this ability to adapt to the driver can lead to better acceptance of ADAS, which is an important sales argument.**

***Index Terms*—Advanced individualized driver-assistance system, driving, imitation learning, independently moving object (IMO), real-time system.**

ADVANCED driver-assistance systems (ADASs) that adapt to the individual driver have high potential in the car industry since they can reduce the risk of accidents while providing a high degree of comfort. Conventional systems are based on a general moment-to-moment assessment of road and driving parameters. To arrive at a judgment of the current situation, they use control laws, from which they derive output signals to aid the driver [1]–[5]. However, interindividual differences in driving can be large [6], [7], and it is difficult for conventional control systems to accommodate these differences, leading to suboptimal driver support. This can finally decrease the driver's safety [8], [9].

Such observations were our motivation to investigate and build a system that automatically adapts to the driving style of its users. In addition, current R&D efforts of the car industry focus on systems that explicitly take the driver and its behavior into account [10], [11]. In addition to the safety aspect, such systems will also be more easily accepted by users because they will provide more comfort, which is an important sales argument.

In the current study, we will describe a system based on imitation learning, i.e., a system that learns to interpret basic aspects of the road (lanes) in the same way as its driver, reproducing the driver's actions. In addition, we demonstrate its use as a basic driver assistance system by issuing warning signals if the driver deviates from his/her predicted default behavior. The so-called DRIVSCO[1] system is realized by a multithreaded parallel central processing unit (CPU)/graphics processing unit (GPU) architecture. It is vision based, operates in real time on real roads, and also includes a method for data-driven detection of independently moving objects (IMOs). The latter is not the focus of this paper and is therefore only briefly described. The system has been designed for use on motorways and country roads.

Before describing the details of our system and comparing it to the literature (see State of the Art), we shortly explain its structure as a guideline for the reader (see Fig. 1). Thus, this paper is organized as follows: in Section I, the overall structure of the system is presented. It is compared with the state of the art in Section II, with its realization explained in Section III and results presented in Section IV. In Section V, we conclude and discuss the presented work.

I. System Overview

The overall structure of the DRIVSCO system is shown in Fig. 1. The yellow box ("human senseact") symbolizes the human driver who senses the environment, which is denoted by the "world" box, and responds to it with adequate driving actions ("act"). At the same time, the system senses the

[1]This is short for Driving School.

Manuscript received March 12, 2010; revised November 5, 2010 and March 15, 2011; accepted April 6, 2011. Date of publication June 20, 2011; date of current version December 5, 2011. This work was supported in part by the European Commission under Project FP6-IST-FET (DRIVSCO) and in part by the Bernstein Focus Neurotechnology (BFNT) Göttingen. The Associate Editor for this paper was S. Tang.

I. Markelić and F. Wörgötter are with Georg-August-University Göttingen, 37077 Göttingen, Germany (e-mail: irene@physik3.gwdg.de; worgott@physik3.gwdg.de).

A. Kjær-Nielsen, L. Baunegaard With Jensen, and N. Krüger are with Maersk McKinney Moller Institute, University of Southern Denmark, 5230 Odense, Denmark (e-mail: akn@mmmi.sdu.dk; lbwj@mmmi.sdu.dk; norbert@mmmi.sdu.dk).

K. Pauwels, N. Chumerin, and M. Van Hulle are with Katholieke Universiteit Leuven, 3000 Leuven, Belgium (e-mail: Karl.Pauwels@med.kuleuven.be; Nikolay.Chumerin@med.kuleuven.be; Marc.VanHulle@med.kuleuven.be).

A. Rotter is with Hella KGaA Hueck & Co, 59552 Lippstadt, Germany (e-mail: Alexander.Rotter@hella.com).

A. Vidugiriene and M. Tamosiunaite are with Vytautas Magnus University, 44248 Kaunas, Lithuania (e-mail: m.tamosiunaite@if.vdu.lt; a.vidugiriene@if.vdu.lt).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2011.2157690

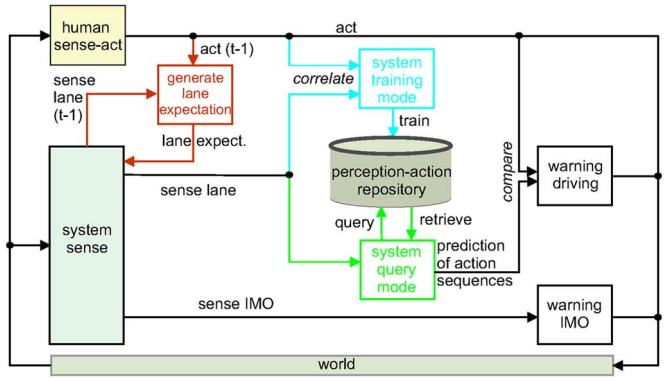


Fig. 1. Block diagram of the DRIVSCO overall system structure. The notation $t-1$ indicates an earlier time frame.

environment ("system sense") by detecting the street "sense lane" and IMOs ("senseIMO") in incoming image frames. The latter refer to objects that move with respect to the static environment. If an IMO is detected, a warning is triggered ("warning IMO") if it is considered to affect the driver (see Section III-D). Detected lanes are filtered by comparing them to a predicted expectation "generate lane expectation," based on a previous (indicated by $t-1$) detection and action (see Section III-E). Once the lane detection is initialized, it is compared with a prediction about the expected lane positions ("generate lane expectation"), which is obtained by using previously captured human action data and the previously detected lane, which is indicated by the notation $t-1$, to predict the current lane structure. This way, incorrectly detected lanes can be filtered out (see Section III-E). The learning system is realized by the perception–action repository (PAR) depicted as the gray container in the figure, which serves as the system's memory, where it stores its observations and retrieves them at a later stage (see Section III-F). The experience stored in the PAR is a state vector containing a concise lane description extracted from the current image frame and sequences of preceding and succeeding action data. Thus, there are two modi: 1) *training*, during which the system gathers experience to fill the PAR, which is denoted in blue as "system training mode" (explained in Section III-F2) and 2) *retrieval*, during which the system queries the PAR to retrieve stored knowledge, which is denoted in green as "system query mode" (see Section III-F2). Since the goal is to learn lane following, i.e., context-free driving, training the PAR requires the recorded data to be filtered to free it from context-dependent scenes (see Section III-F1). This makes the training phase a demanding procedure that is done offline, i.e., not during driving. The retrieval mode, however, can be used offline and online. Based on accumulated PAR returns, action plans for the expected human steering and acceleration behavior for lane following are predicted, i.e., "prediction of action sequences" (see Section III-F4). To demonstrate the system's use as an assistance system, predicted action sequences are compared with the driver's actions, and a warning is issued if they considerably differ, i.e., "warning driving" (see Section III-F4). In other words, the driver is informed when deviating from his or her learned default driving behavior. The warning signals are displayed on a screen close to the driver, as shown in Figs. 2(c) and 4.

II. State of the Art

Since our work aims at generating sequences of future driving actions based on visual sensory input, it is closely related to vision-based autonomous driving. Many approaches in this field rely on control-theoretic whitebox methodologies, i.e., they are based on predefined analytical models and control laws. This has led to some well-performing systems, e.g., [12]–[15]. However, the drawbacks are the dependence on predefined knowledge and the difficulty of developing models and control laws, which restricts the design process to experts. In addition, these systems do not, or only in a limited way, provide means for individualization. By contrast, imitation learning [16] aims at extracting a policy for a given task by using examples provided by a teacher. This reduces the amount of required *a priori* knowledge and the need for explicit programming and thus facilitates human computer interaction. A famous example of imitation learning for driving is the Autonomous Land Vehicle In a Neural Network (ALVINN) system [17]–[20], where the steering actions of a human driver were associated with concurrent visual input from a camera via a neural network. Velocity control was handled by the driver. Further imitation learning work by Pasquier and Oentaryo describes the learning of several driving skills with a fuzzy neural network [21]. The algorithms for lane following were tested in simulation. Similar work with helicopter flying was reported in [22]. A novel form of inverse reinforcement learning [23] was introduced in [24] and applied to learning particular driving styles from example data obtained from simulation.

The European-Union funded project Dynamic Interactive Perception-action LEarning in Cognitive Systems (DIPLECS) [25] reports similar goals to ours. However, DIPLECS does not aim at building a complete system. Research conducted by Motorola [26] aims at building an adaptive driver support system using machine learning tools. Its architecture was partially implemented in a prototype system built upon a simulator.

In addition to lateral control (steering), new generations of driver assistance systems will contain support for longitudinal control (velocity) during curve negotiation, e.g., [27]. Current (non-imitation-based) methods usually do not take the street trajectory into account but assist with simpler aspects such as detecting an obstacle in front (e.g., Adaptive Cruise Control systems using radar or laser for obstacle detection), known speed limits (e.g., Intelligent Speed Adapters and Limiters [28]), or leading vehicles (e.g., [29]). Focusing on curve negotiation and based on imitation learning are [21], [30], and [31], which all employ fuzzy neural networks trained on human control data.

Our work, unlike similar approaches, describes the realization of a complete system implemented in a real car that learns the prediction of action plans, i.e., sequences of steering and acceleration actions, together with a method for IMO detection. Our research differs from others because we use data obtained from real car driving and not from a simulator. (Most of the presented algorithms were first tested on a robot platform, as reported in [32].) By contrast to the implicit mapping between sensory input and actions achieved with the neural network in the ALVINN project, our *lazy learning* approach [33] realized by the PAR allows the preservation of human interpretable

(a) (b) (c)

Fig. 2. System integration in the car. (a) Mounting for the stereo camera system. (b) Fixation of the computer in the trunk. (c) Installation of the monitor in the car.

information processing at all stages. Although storage intensive, it is a highly beneficial concerning error analysis.

III. Submodules and Methods

A. Hardware

The used car is a Volkswagen Passat provided by the DRIVSCO partner Hella KGaA Hueck & Co (Lippstadt, Germany). The following sensory data are accessed via the Controller-Area Network (CAN)-bus: steering angle, which describes the steering wheel's position with values between $[-360^\circ$ (left steering), 360° (right steering]; velocity in kilometers per hour; longitudinal acceleration with values between $[-10\ \mathrm{m/s^2}, 10\ \mathrm{m/s^2}]$; and curve radius measured by a gyroscope with values between [−15000 m, 15000 m]. Furthermore, a camera stereo rig is mounted behind the windscreen, as shown in Fig. 2(a). We use two color Pulnix TM1402CL cameras, which deliver 1280 × 1024 raw Bayer pattern images at a frequency of 20 Hz. A Global Positioning System (GPS) receiver is added to allow the visualization of driven routes, for which we used Google Maps (c.f. Fig. 6). All computations are carried out on a personal computer (PC) with an Intel Core i7-975 3.33-GHz quad-core processor with simultaneous multithreading enabled and 12-GB random access memory. The used graphics cards are an NVIDIA GTX295 for computation and a smaller one for displaying purposes. The PC is kept in the car's trunk, as shown in Fig. 2(b), and the system output is displayed on a screen next to the steering wheel, as shown in Fig. 2(c).

B. System Architecture

The backbone of this work is its realization as a multithreaded pipelined real-time system where shared memory is used for interprocess communication. Due to the complexity of the pipeline structure and the involvement of multiple CPU cores and multiple GPUs in its computation, we have developed our own modular architecture. By simplifying the coordinated use of the heterogeneous computational resources in modern computers, this architecture allows independent development of the individual processing stages. To our knowledge, no such CPU/GPU pipeline framework combining task and data parallelism exists, allowing a stage to share data with multiple other stages, e.g., the output of the preprocessing stage is used by both "lane detection" and "dense vision," as shown in Fig. 3. The system structure from Fig. 1 is realized by the architecture shown in Fig. 3. Each information processing entity is referred to as a stage and runs in a thread as indicated. All stages are connected through a pipeline where communication is realized

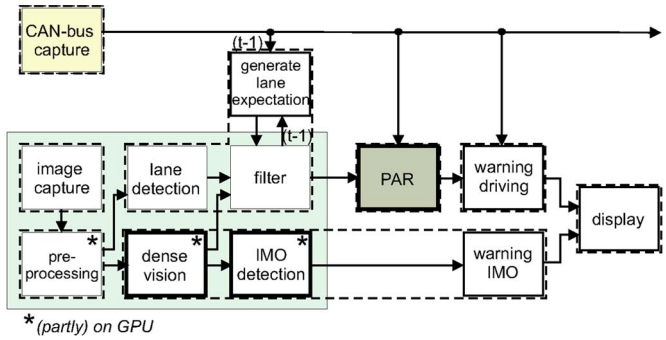


Fig. 3. Realization of the DRIVSCO system. Boxes denote processing stages, and the dashed frames indicate individual parallel threads. Arrows denote the information flow, with the exception that the display stage connects to *all* stages. The "PAR," "dense vision," and "IMO detection" are key parts of this system. The notation $t-1$ indicates an earlier time frame.

Fig. 4. One tab of the system GUI showing the detected lanes in the current image on the left and a history of prediction and steering angles of 2 s on the right, along with the computed egomotion. Thresholds are shown in gray.

by shared memory buffers, where processes can be written to and read from. The processing time of each individual stage is below 50 ms, and the entire system works at a camera rate of 20 Hz. The "preprocessing," "dense vision," and "IMO detection" stages involve massive computations but achieve a frequency of 20 Hz through the use of two GPUs, whereas all other processes run on the CPU. The "CAN-bus capture" stage is triggered every 50 ms and makes the relevant car CAN-bus data available to other processes. During the "image capture" stage, raw camera data are received, and white level calculations are performed to control the camera's shutter time. The images are then undistorted, rectified, and downsampled to 640 × 512 pixels during the "preprocessing." The boxes "lane detection," "dense vision," and "IMO detection" are explained in Sections III-C–III-E in detail. In addition, a special display unit is integrated, which connects to all buffers in the pipeline, allowing the user to view the output of *any* stage, including the generated warning signals by means of a graphical user interface (GUI). A screenshot of this GUI is shown in Fig. 4.

It shows a current image with detected lanes on the left, and a history and prediction of steering angles of 2 s on the right. The prediction is plotted in blue, and the actual human steering is plotted in red. In addition, the computed egomotion from the IMO detection stage is displayed in green. The gray dashed boundaries around the prediction indicate predefined threshold values used to issue a warning if exceeded by the

actual driving. In this case, the "unexpected driving" button below the displayed lane detection image is flashed. The system can also conveniently be used in an offline mode to replay recorded scenes, which is important for error analysis. Note that this display is designed for R&D purposes and should be simplified for a real driver assistance system.

C. Dense Vision

The visual cues used for the IMO detection are computed during the "dense vision" stage (see Fig. 3). Dense vision algorithms process the visual signal in its entirety, as opposed to sparse vision algorithms that only operate on interest points such as edges or corners. The cues used here are dense disparity (the horizontal difference in image location for corresponding pixels in the rectified left and right image) and optical flow (the 2-D image motion of each pixel). The algorithms used rely on the phase of quadrature pair Gabor filter responses [34] [see Fig. 5(b)], which were extracted at multiple orientations and scales, to establish correspondences. The GPU implementation [35] of a phase-based algorithm [36] is used to compute optical flow [see Fig. 5(c)]. This algorithm integrates the temporal phase gradient across orientation and gradually refines its estimates by traversing a Gabor pyramid from coarser to finer levels. The optical flow is computed for the left video stream only. The dense disparity algorithm [see Fig. 5(d)] is very similar to the optical flow algorithm but operates on phase differences between the left and right image, as opposed to temporal phase gradients. These aspects are described in more detail in [37].

D. IMO Detection and Warning

Recent (offline) approaches for the detection of IMOs have achieved very good results using model-based techniques [38], [39]. However, limitations include the difficult detection of distant IMOs since they occupy only small patches in the image and are thus difficult to match and, further, objects for which no model is provided cannot be detected. Here, we use a model-free mechanism that is more general in the sense that it will respond to any sufficiently large (11×11 pixels) moving object. The IMO detection component combines dense vision cues (optical flow and stereo; see Section III-C) in real time to compute egomotion (the rotation and translation of the camera) and independent motion (the parts of the image that move with respect to the static environment) to detect an IMO in front.

The whole process is complex and cannot be described in detail in this paper. See [40] for further information. Here, we will give only a short overview summarized in Fig. 5. A nonlinear instantaneoustime model [41] is used to extract egomotion from the optical flow. To obtain optimal estimates, an iterative minimization procedure that relies on M-estimation is used [42] for outlier compensation. A total of 32 different initializations are explored to deal with local minima. The data-intensive parts of the algorithm entirely run on the GPU. Independent motion is detected by evaluating the depth/flow constraint [43] at each pixel. Deviations from this constraint point to inconsistencies between the optical flow, disparity, and egomotion and result from noise or independent motion. The

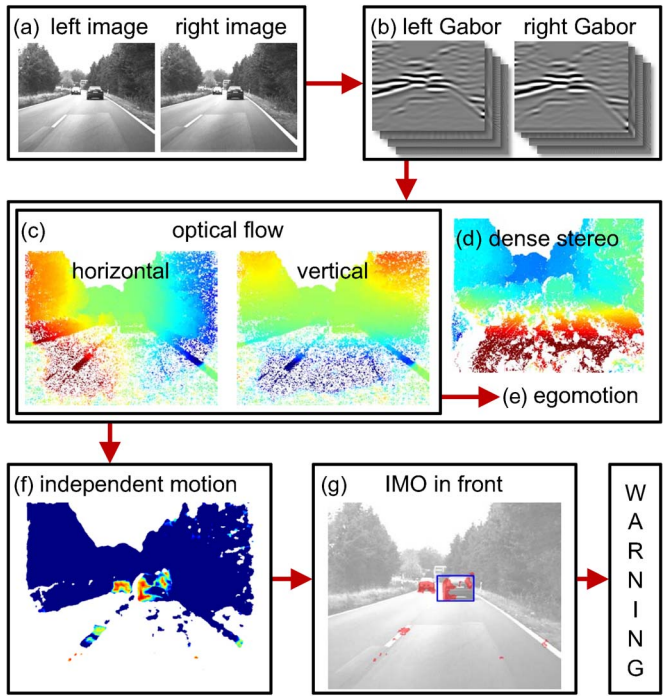


Fig. 5. Real-time IMO detection. Multiorientation multiscale Gabor filter responses ($\mathbf{B}$) are extracted from the stereo image pair ($\mathbf{A}$) and used to compute dense optical flow ($\mathbf{C}$) and stereo disparity ($\mathbf{D}$). The horizontal and vertical optical flow is color coded from -15 (dark red) to $+15$ pixels (dark blue), and the stereo disparity is color coded from -50 (dark red) to $+20$ (dark blue) pixels. Combined with egomotion ($\mathbf{E}$, which is extracted from the optical flow, not shown), these cues allow the extraction of independent motion ($\mathbf{F}$, likelihood increases from blue to red). This independent motion signal is gathered in a fixed region of the image ($\mathbf{G}$), and when it exceeds a threshold, a warning is issued.

deviations are assigned to independent motion if they comply with a 3-D translation model in a local region surrounding the pixel [see Fig. 5(f)]. To detect a moving vehicle in front, the (pixelwise) independent motion signal is accumulated inside a fixed region in the image [see the blue rectangle in Fig. 5(g)]. A warning is issued when more than 30% of the pixels within this box are considered independently moving.

E. Lane Detection and Filtering

A large number of lane detection algorithms have been reported, which can roughly be divided into feature- and model-based methods (see, e.g., [2]). The former detect street lanes bottom-up, i.e., based on certain low-level features such as intensity gradients, edges, and color, e.g., [44], whereas the latter aim at identifying image parts corresponding to a predefined lane or street model, e.g., [45] and [46]. Both approaches have known advantages and disadvantages: feature-based methods can detect arbitrary shapes but might add erroneously detected image parts into the returned street detection. The more restricted model-based methods are more robust to disturbances such as occlusions, noise, and shadows. However, they are restricted to predefined lane shapes, making them less flexible. To detect lanes in incoming image frames, we employ a simple and fast (real time, i.e., 20 Hz) feature-based algorithm, which works similar to contour tracers used in computer vision: First, edges and pixel orientations are computed by using standard

computer vision techniques (Canny and Sobel operator [47], [48]). Second, line segments are constructed by joining nearby edge pixels with similar orientations. Then, nearby line segments are further joined, resulting in longer lines. Thus, a list of lines that might contain small interruptions is obtained. The parametrization of a lane can be shown in Fig. 7(a). The left and right lanes are expected to be the longest of these lines, starting in a particular area at the bottom of the image. During initialization, this area is manually determined and further tracked using a Kalman filter [49]. This is very simple and requires almost no *a priori* knowledge or initial image preprocessing, i.e., it works on raw intensity images containing street lanes from a single camera. Implementation details are given in [50]. To correct against false positives, we apply an additional filter that uses 3-D information to verify if the detected lane is on the ground plane. This is achieved by using the computed dense stereo map (see Section III-C), which attaches 3-D information to each point of the extracted lane in the form of disparity values. There is a linear relationship between disparity values and horizontal image coordinates, and the filter checks whether the extracted lane fulfills this criterion. Since the disparity contains noise, we use RANdom SAmple Consensus (RANSAC) [51] to fit a line to the disparity data of the extracted lane. If an estimate with slope in a tolerable range can be fitted, we believe that the lane is on the ground plane. Otherwise, it is rejected.

As indicated by the entry "generate lane expectation" in Fig. 3, a final feedback mechanism aids the stability of the lane detection by generating lane expectations based on the human behavior. The velocity and steering angle of the car is used to derive its rigid body motion, which is then used to predict the position of a detected lane one frame ahead.

The expected lane is then used for filtering the lane detection in the following frame by comparing both and rejecting the detection if it differs too much from the prediction.

F. PAR

The PAR serves as the system's memory. It stores its (driving) experience and retrieves it at a later stage. The idea is based on the assumption that a human executes a stereotypical driving behavior according to the street trajectory that he or she sees. A straight street ahead will be followed by straight steering for a while and probably some acceleration. A sharp turn will cause the driver to decelerate and to steer accordingly. To let the system learn this, we store a description of the street ahead, together with *sequences* of human driving data that he or she issued after having observed that street. This is the training phase. Next follows the retrieval phase, during which the system can use incoming street trajectory descriptions to query the PAR (similar to a pattern-matching process) and obtain adequate driving sequences. The fact that we use sequences, instead of single step actions, allows us to compute an expected human driving behavior that reaches, to some extent, into the future, i.e., we predict future sequences of human driving actions. To demonstrate the system's use for driver assistance, we issue warnings if the actual human driving data differs too much from the computed expected driving. In the following, we explain what data we use and then formalize the individual steps.

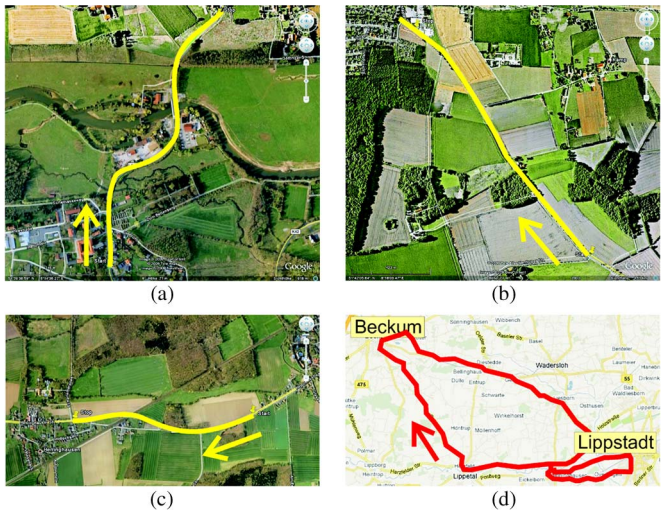


Fig. 6. Tracks on which training data were recorded. (a) s03 (1 km). (b) s05 (2 km). (c) s06 (1 km). (d) Long tour between Lippstadt and Beckum (69 km).

Data, Default Driving, and Preprocessing: We use a data set from a single driver recorded and provided by Hella KGaA Hueck & Co[2]. It consists of three repeatedly driven tours to which we refer as s03, s05, and s06 [c.f. Fig. 6(a)–(c)] and a track that we denote "long tour" [see Fig. 6(d)], which was driven twice.

Steering and acceleration are the data predicted from the car CANbus (see Section III-A). Due to trends and a too high variance found in general in the velocity data, we used the acceleration signal, which we found to be more predictable.

The goal is to learn the default human behavior concerning steering and acceleration for lane following in context-free situations. By this, we mean driving without additional traffic and without special situations, such as intersections. This requires filtering the original driving data to clean it from context-afflicted situations, which can be achieved by using the IMO detector to identify and exclude situations with IMOs in front, as well as by using inconsistencies in the lane detection to identify and exclude situations such as intersections. Hence, PAR entries for lane following are based on context-free driving examples.

In addition to the acquisition of context-free situations, we also remove action sequences that are obvious outliers as described in the following. In Fig. 7(b) and (c), 15 steering and acceleration signals from the same track and driver are shown, along with their means plotted in black. We observe a much higher variance in the acceleration data than for steering, which is quantified by the mean signal-to-noise ratio (SNR, we compute $E[|\mu|/\sigma]$, with μ being the mean and σ the standard deviation), where a high SNR indicates good predictability of a signal and a low SNR indicates a bad one. For the shown data, we obtain an SNR value for steering of 7.43 and acceleration of 0.62. By removing outliers (which are detected by an automatic procedure during which the individual signals are compared with the mean), the latter can be increased to 1.3. In Fig. 7(d), black signals are those acceleration signals found to

[2]Parts of this set are available at the webpage [52].

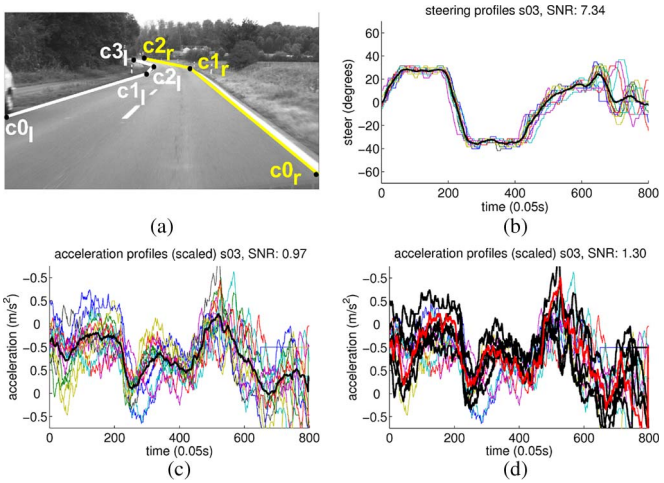


Fig. 7. (a) Extracted street boundaries described by polylines and the corner points. The vectors of the corner points $(c_{0_{l/r}}, c_{1_{l/r}}, \ldots)$ for the left and right lanes constitute the visual state description. (b) Steering and (c) acceleration signals from 15 runs from the same track and driver. The mean is plotted in black. "SNR" refers to signal-to-noise ratio. (d) The sequence closest to the mean is shown in red. Black signals are considered to be sufficiently similar.

be sufficiently similar, and the others (differently colored plots) are those that were sorted out. The sequence closest to the mean is plotted in red. We use these data for training the PAR, as explained in Section III-F2.

Driving data assigned to similar situations are averaged during training. Similarity is defined by the resemblance between the left and right street lanes of the observed image, and one already contained in the PAR, as well as a short sequence of previous steering data. A formal definition is given in Section III-F3. Hence, for tracks that are being driven multiple times, the system defines the default human driving behavior over the mean of the action sequences obtained from driving on the same track, as shown by the black plot in Fig. 7(b). Note, however, that learning starts with the first entry in the PAR, and additional information just improves the performance of the perception–action mapping. Single-example performance is reliable for steering but remains restricted for acceleration control (see Section V). Note that, as the curve shapes are fairly similar, after some learning, the system is able to generalize into unseen curves. This will be shown later (see Fig. 11).

Since the signals predicted by the PAR should correspond to the default behavior of the human, i.e., the mean action signal, we evaluate the performance of the PAR by comparing its output against the human action sequence closest to the mean signal, i.e., the red plot in Fig. 7(d) (which we do not include in the training data).

PAR Training: The information stored as experience in the PAR is a state vector $\mathbf{s}$ containing a description of extracted left and right street lanes $(\mathbf{v}_{\text{left}}, \mathbf{v}_{\text{right}})$ from a current image frame I_t. Here, t denotes the time of the image observation.[3] Because we found that only a description of the street ahead is not sufficient to distinguish different driving situations from each other, e.g., a straight street can lead to acceleration or no acceleration, depending on the current velocity, we also store a short sequence of previous steering $(\mathbf{s}_{\text{past}})$. We use a fixed number of $m = 50$ discrete steering actions that preceded I_t. Thus, the state vector is as given in (1). To each such state vector, we assign sequences of future human driving actions executed after the observation of I_t, which we refer to as $\mathbf{s}_{\text{fut}}$ and $\mathbf{a}_{\text{fut}}$. The length of the sequences stored is supposed to resemble the number of actions necessary to cover the part of the street observable in I_t. Since we do not know exactly how many actions this corresponds to, we use a fixed value $n = 100$, which is 5 s of driving corresponding to 97 m at a speed of 70 km/h, which we consider reasonable for country road driving. Thus, a PAR entry $\mathbf{e}$ is as given in the following:

$$\mathbf{s} = \{\mathbf{v}_{\text{left}}, \mathbf{v}_{\text{right}}, \mathbf{s}_{\text{past}}\}, \qquad \text{state vector} \tag{1}$$

$$\mathbf{e} = \{\mathbf{s}, \mathbf{s}_{\text{fut}}, \mathbf{a}_{\text{fut}}\} \qquad \text{PAR entry.} \tag{2}$$

The action sequences $\mathbf{s}_{\text{past/fut}}$ and $\mathbf{a}_{\text{fut}}$

$$\mathbf{s}_{\text{past}} = [s_{t1}, s_{t2}, \ldots, s_{\text{tm}}] \tag{3}$$

$$\mathbf{s}_{\text{fut}} = [s_t, s_{t+1}, \ldots, s_{t+n}] \tag{4}$$

$$\mathbf{a}_{\text{fut}} = [a_t, a_{t+1}, \ldots, a_{t+n}] \tag{5}$$

with s and a denoting single steering and acceleration signal values (actions).

The descriptions of the left and right street lanes $(\mathbf{v}_{\text{left/right}})$ are linear approximations (polylines) of the extracted lanes in image coordinates, which were obtained by applying the Douglas–Peucker method [53]. We store this as vectors containing the corner points of the polylines, as visualized in Fig. 7(a). Thus

$$\mathbf{v}_{\text{right}} = [c_{0_r}, c_{1_r}, \ldots, c_{l_r}] \tag{6}$$

$$\mathbf{v}_{\text{left}} = [c_{0_l}, c_{1_l}, \ldots, c_{l_l}] \tag{7}$$

with l_l and l_r denoting the lengths of $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$.

During the training phase, a new entry $\mathbf{e}$ [see (2)] is added to the PAR if it represents new knowledge, i.e., if no entries are already available containing similar information. An entry is added if 1) there is no other entry already in the PAR, with $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$ having the same lengths as those contained in the probed entry (in other words, if a completely new street trajectory is observed), or 2) if there are such entries but none of them has a state vector similar to that probed for adding. Two state vectors are similar if the differences between their components are each below a fixed threshold, that is, if the following are fulfilled:

$$\epsilon\_\mathrm{v} \leq thresh\_\mathrm{v} \tag{8}$$

$$\epsilon\_\mathrm{s}_{\text{past}} \leq thresh\_\mathrm{s}_{\text{past}} \tag{9}$$

where $\epsilon\_\mathrm{s}_{\text{past}}$ is the summed squared difference between the entries of $\mathbf{s}_{\text{past}}$ of two state vectors, and $\epsilon\_\mathrm{v} = \epsilon\_\mathrm{v}_l + \epsilon\_\mathrm{v}_r$ and $\epsilon\_\mathrm{v}_{l/r}$ are the sums of the normalized, weighted, and summed

[3]Time is measured in discrete time steps according to the camera's image capturing frequency.

Euclidean differences between $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$ of two state vectors, see

$$\epsilon\_\text{v}_{l/r} = \frac{1}{l_{l/r}} \sum_{i=0}^{l_{l/r}} \omega[i] \sqrt{\left(\mathbf{v}[i]_{\text{left/right}} \mathbf{v}^*[i]_{\text{left/right}}\right)^2} \quad (10)$$

where the star in $\mathbf{v}^*[i]_{\text{left/right}}$ indicates that it is an entry of another state vector, and $\boldsymbol{\omega}$ is a weight vector with $\boldsymbol{\omega}[i] \geq \boldsymbol{\omega}[i+1]$. The weighting punishes differences between two lanes close to the image bottom more than differences appearing closer to the horizon.

The PAR is complete if a predefined number of entries is reached, or the two cases in which entries are added to the PAR do not occur no longer. Note that training is done offline (hence, not during driving). This would also be desired in any commercial system as the training procedure is demanding (due to the removal of context-dependent scenes) and should thus take place when the car is not operating.

PAR Retrieval: To retrieve information from the PAR, it is queried with a current state vector, which is compared with all PAR entries whose $\mathbf{v}_{\text{left}}$ and $\mathbf{v}_{\text{right}}$ have the same lengths as those in the queried state vector. The action sequences attached to the most similar PAR entry are returned. Similarity is defined by computing the differences between the single entries of two state vector entries, i.e., ϵ_v, ϵ_s_{past}, as previously defined, and the most similar entry is that with the lowest overall differences.

Thus, the return parameters of a query are either 1) the differences ϵ_v and ϵ_s_{past} between the most similar PAR entry and the query, and the action sequences $\mathbf{s}_{\text{fut}}$ and $\mathbf{a}_{\text{fut}}$ assigned to the most similar entry or 2) an indication that no match could be retrieved. The latter occurs when either there was no entry that the query could be compared with or the best found match was unacceptable, i.e., the assigned differences exceeded predefined thresholds.

Prediction of Action Sequences: Because the PAR is queried every time step, sequences of driving behavior more or less appropriate to the queried situation are obtained. The degree to which the returned actions correspond to the queried situation is indicated by the returned differences ϵ_v, ϵ_s_{past}. Since it is unlikely that identical state vectors are obtained multiple times, even on the same track, a mechanism for generalization is required. That is, the system must compute adequate driving actions based on more or less appropriate PAR returns. We postpone this step until retrieval time as typical for lazy-learning algorithms [33], [54]–[56], which are often used in imitation learning (compare [16]). The final expected human driving sequences are generated by keeping the latest k query results for steering and acceleration, and simply averaging over values belonging to the same time step (see the gray box in Fig. 8). Assuming that these values are contained in a buffer $\mathbf{g}_{\text{buf}}$ (see Fig. 8), a single predicted action is computed, as given in

$$a_t = \frac{1}{|\mathbf{g}_{\text{buf}}|} \sum_{i=0}^{|\mathbf{g}_{\text{buf}}|1} \mathbf{g}_{\text{buf}}[i]. \quad (11)$$

Thus, every action command in the resulting prediction is a linear interpolation between the examples learned before. This is similar to the k-nearest neighbor algorithm, which uses k closest training examples to compute a value for the variable of interest. Thus, the learning begins to work with one single entry in the PAR and improves on repeatedly seen tracks just as a human driver would. For a final smoothing, we apply a moving average filter (window size $= 10$) on the resulting signal.

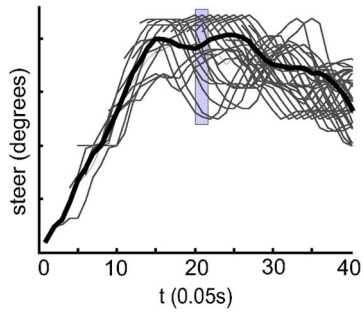


Fig. 8. Gray lines are the PAR returns for steer, and the computed expected human driving sequence is drawn in black. The gray rectangle indicates a vector $\mathbf{g}_{\text{buf}}$ containing all action signals of a certain time step used for averaging (see text).

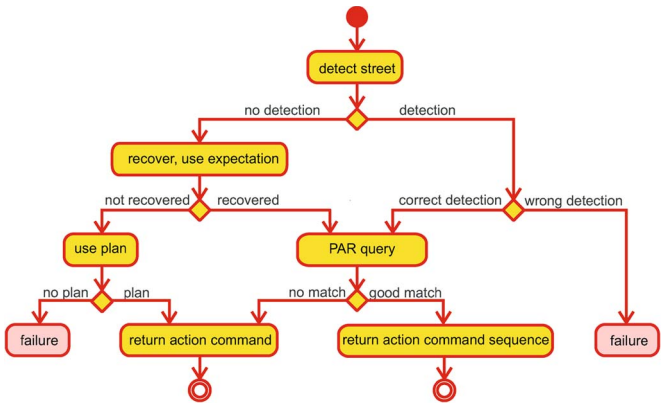


Fig. 9. Algorithmic flow of the driving system Unified Modeling Language (UML) activity diagram.

We implemented a simple warning mechanism to demonstrate its use for driver assistance. A warning is generated if the actual human steering deviates too much from the expected driving given by the computed prediction and specified by a threshold determined empirically based on the offline analysis of the test drive sequences.

Fig. 4 shows an example of the actual driving, the prediction, and the thresholds.

G. Algorithmic Flow

The algorithmic flow of the system concerning action prediction is summarized in Fig. 9.

No output can be generated if lanes could not repeatedly be detected or if they were repeatedly misdetected, i.e., other items erroneously identified as a lane. Some critical cases can be compensated by other mechanisms, e.g., the generated lane expectation can be used in case of an undetected lane. If this prediction is considered unreliable, the previous action plan can be used for as long as it contains entries. In this case, the control is open loop, and only single-action commands can be issued. In the optimal case, the system returns a sequence of action commands, as described in Section III-F4.

TABLE I
COMPARISON BETWEEN THE PRESENCE OF AN ACTUAL LANE MARKER IN THE SCENE AND THE OUTPUT OF THE DEVELOPED LANE-DETECTION ALGORITHM

Lane Marker	Present		Not Present:
Total 11,400	11,055 (100%)		355
Detected	10,933 (98.9%)	only left: 619 (5.6%) only right: 939 (8.5%)	0
Not Detected	122 (1.1%)		355
False Positive	0		0

IV. RESULTS

The performance of the IMO detection is assessed in two ways, which together demonstrate the range of situations that the algorithm can handle. On one hand, simulated scenes are used with precisely known parameters to allow a rigid quantitative evaluation. On the other hand, several real-world scenes are processed to assess the qualitative performance. We evaluate the detection of two moving objects and the magnitude of their speed in a demanding simulated scene and find that both objects are detected with a density of more than 50%, compared with all pixels that belong to the objects. The magnitude of the speed is calculated with an accuracy of better than 98%. The qualitative performance was evaluated by applying the algorithm to a diverse set of real-world driving situations involving both country roads and complex urban environments. A variety of moving objects (cars, bikes, and pedestrians) were successfully detected, whereas the car itself underwent a wide range of egomotion speeds and steering actions. Further information can be found in [40].

The developed lane detection algorithm was evaluated on a randomly chosen sample of each tour of the available data set, which comprised a variety of country roads with both clear and damaged lane markers (a total of 11.400 frames with humanly detectable lane markers present in 11.055 frames). In 98.9% of the 11.055 frames, a valid lane description was extracted. In 5.6% of these cases, only the left marker and, in 8.5%, only the right marker were detected. Both markers were found in 84.8% of these cases. This information is summarized in Table I. Examples of detected lanes are shown in Figs. 4, 7(a), and 12.

To evaluate how well the action predictions match the human default behavior, we use the provided data set.

After filtering it, as explained in Section III-F1, approximately 80 min of training data were obtained, resulting in a PAR with 90 470 entries, which is adequate for evaluating the system performance. First, we test the performance on a known track, i.e., one that the system had seen before. For that, we train the PAR with all runs but that closest to the mean, which we consider to resemble the human default behavior, as explained in Section III-F1, and we use for testing. The smoothed steering and acceleration signals from the algorithm are plotted against the signal generated by the human for s03, s05, and s06 in Fig. 10. As an additional measure of similarity, we compute the correlation coefficient of the two signals (human and prediction). For steering and acceleration prediction, we obtain 0.99 and 0.81 for s03, 0.93 and 0.73 for s05, and 0.97 and 0.67 for s06. Thus, all predictions are very good. However, the

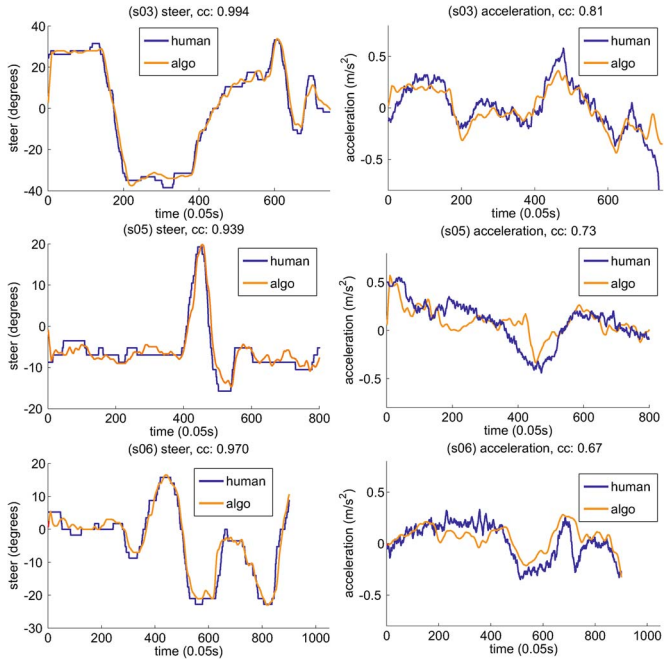


Fig. 10. Results for (left column) steering and (right column) acceleration prediction for known tracks. "cc" denotes the correlation coefficient value between the human-generated signal and the synthesized signal.

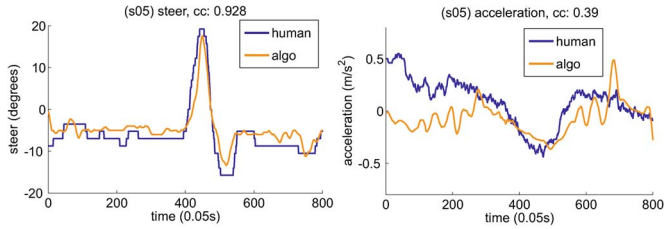
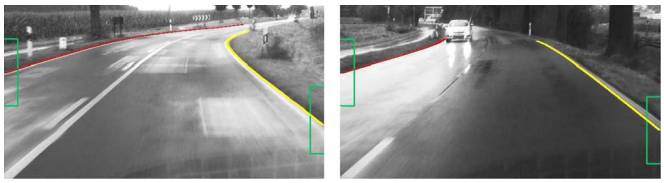


Fig. 11. Results for steering (left) and acceleration (right) prediction on an unknown track.

steering signal is better approximated than acceleration, which is as expected from the computed SNR in Section III-F1.

For testing the performance on an unknown track, we train the PAR with all available data, except that from the track we test for. The result is shown in Fig. 11. We chose s05, which contains a very sharp turn (see Fig. 12), leading to two typical phenomena, which are discussed here. The resulting steer and acceleration predictions in comparison to the human signal are shown in Fig. 11. The steering prediction is very close to the human signal, but the acceleration is less reliable. In particular, it can be seen from the plotted acceleration prediction in Fig. 11 that the sharp curve [which is entered around time step 200, c.f., Fig. 12(a)] is accompanied by a deceleration in the human signal (between time steps 220 and 380) and that this is nicely reflected by the algorithm. However, before and after the deceleration part, the predicted signal considerably differs from the human data.

This phenomenon results from the fact that street parts with a lower curvature allow a great variance in the driver's choice of speed, depending on hard-to-access variables including "mood" and "intention," where curves, particularly sharp curves, significantly restrict the driver's behavior and thus make it more predictable. We therefore conclude that acceleration prediction



(a) (b)

Fig. 12. (a) Entering a sharp turn in s05 at $t = 200$. (b) Too-short detected lanes (with the left lane shown in black and the right lane in white.

with the presented method is only useful for curve negotiation. The second observation is that there are unwanted peaks in the predicted acceleration curve. This happens because the system sometimes hooks on to different but similar-looking road segments in the PAR, as shown in Fig. 12(b). Although the lanes are correctly extracted, the lookahead is not sufficient to properly distinguish this situation from almost straight streets. We found that our driver sometimes reacted to upcoming curves up to 8 s before entering them, requiring the system to have a similar lookahead to correctly predict the driver. Humans can see this far and make out even very faint features of the road layout, which leads to such early reactions; computer vision systems, however, cannot. The detected lanes at such distances and the detected lane segments will, for smooth and less descriptive situations, remain ambiguous, leading to false matches, which causes these peaks.

According to the algorithmic flow shown in Fig. 9, critical cases are a frequently undetected street, as well as the case in which the PAR did not contain a good-enough match. In these cases, it is possible to work off a previously generated action plan, but only as long such a plan is available. For the presented tests, the street detection rates were high: 100% for s03, 96% for s05, and 98% for s06. However, the case that no PAR match was retrieved occurred relatively frequently: for s03, in 32%; for s05, in 12%; and for s06, in 39% of all cases. For testing the case of driving on an unknown street on s05, no PAR match was retrieved in 39%. Despite these high rates, it can be seen from Figs. 10 and 11 that, for the entire duration action, signals were reliably produced. It is an important aspect of the system that its ability to predict sequences adds considerable robustness to its performance.

During the final review meeting of the DRIVSCO project, the system was observed by three independent international reviewers (see [57]). The driver from which the training data were obtained, i.e., who taught the system, drove the first 20 min of the long tour and back, where "back" corresponds to an unknown track situation as curves are inverted. All components were shown to reliably work together at the desired speed. The lane detection worked even under challenging weather conditions where sunshine after rain caused reflections. (A TV report of this is available online [58].).

V. Discussion and Conclusion

We have presented the DRIVSCO system, which learns basic human driving (steering and acceleration control for lane following) by observing the driver. We have evaluated its imitation performance on known and unknown tracks and shown it to reliably work for steering and, in the case of curve negotiation, for acceleration prediction as well. In addition, we have implemented a method for vision-based data-driven IMO detection. By using visual, i.e., passive, information, our system does not invade the environment and does not suffer from interference problems, which may be the case with active systems, e.g., laser range finders. By taking a data-driven approach, our system does not rely on predefined models, which makes it widely applicable.

We have demonstrated the use of the system output (steering and IMO detection) for supporting the driver. A domain in which this system may have future potential is that of intelligent driver assistance systems, which automatically adapt to individual driver behavior.

In contrast to most related work, DRIVSCO is an integrated system implemented on a real car, in real time, realized by a multithreaded parallel CPU/GPU architecture that uses data from real driving—not simulation. The learning algorithm is deliberately simple and, thus, is easy to understand and implement. In the early stages of the project, different methods based on feedforward and radial-basis-function networks have been tested, however not achieving the performance of the lazy learning approach presented here. Furthermore, lazy learning offers the advantage that all information remains human interpretable, which is convenient for error analysis, as opposed to learning algorithms, which transform information into subsymbolic knowledge, which is, for example, the case with neural networks. The system predicts *sequences* of expected human behavior, as opposed to a moment-to-moment control. This makes it 1) more stable, e.g., in case of unreliable or lacking sensory input, it can use predictions as a fall-back plan, and 2) it allows for proactive control, i.e., warnings can be issued based on the predicted behavior, instead of the current one.

The system has been specifically designed for motorways and country roads, hence driving situations with limited context. The extraction of context-free situations is demanding; therefore, the training of the system is performed offline. To apply imitation learning to more difficult driving situations (e.g., city) appears currently infeasible, as driving decisions are, in these cases, far too diverse and state-action descriptions would become too complex. Furthermore, we observed that acceleration signals are nondescriptive when driving in uncritical situations (e.g., straight road) because drivers follow their mood. This strongly contributes to the high variance observed in the acceleration data. As a consequence, longitudinal control (or warning) becomes only useful whenever a driver is forced to drive with less leeway (e.g., in front of sharp curves). This notion is important when considering the psychological acceptance of individualized driving aids. One of their central features must be to not interfere with the driver unless necessary. Hence, in uncritical situations, systems should remain silent, and acceleration should remain in the hands of the driver.

To improve the presented system, one should furthermore consider to extend the system's lookahead beyond that of machine vision. This could be achieved, for example, by integrating GPS information and digital maps.

One interesting aspect concerning industrial applications is the potential use of this system for night driving support. Under

bad illumination conditions, the human's sensing process is obviously limited; however, by using infrared light, the system's sensing is less affected, given that the lane detection process is adequately adapted to the new circumstances. Thus, the system can use its knowledge about driving acquired during the day to support the human in the more difficult situation of night driving.

REFERENCES

[1] S. Mammar, S. Glaser, and M. Netto, "Time to line crossing for lane departure avoidance: A theoretical study and an experimental setting," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 2, pp. 226–241, Jun. 2006.

[2] J. McCall and M. Trivedi, "Video-based lane estimation and tracking for driver assistance: Survey, system, and evaluation," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 20–37, Mar. 2006.

[3] A. Amditis, M. Bimpas, G. Thomaidis, M. Tsogas, M. Netto, S. Mammar, A. Beutner, N. Möhler, T. Wirthgen, S. Zipser, A. Etemad, M. Da Lio, and R. Cicilloni, "A situation-adaptive lane-keeping support system: Overview of the safelane approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 617–629, Sep. 2010.

[4] L. Li, F. Y. Wang, and Q. Zhou, "Integrated longitudinal and lateral tire/road friction modeling and monitoring for vehicle motion control," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 1–19, Mar. 2006.

[5] E. Bertolazzi, F. Biral, M. D. Lio, A. Saroldi, and F. Tango, "Supporting drivers in keeping safe speed and safe distance: The saspence subproject within the European framework programme 6 integrating project prevent," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 525–538, Sep. 2010.

[6] P. Ulleberg and T. Rundmo, "Personality, attitudes and risk perception as predictors of risky driving behaviour among young drivers," *Safety Sci.*, vol. 41, no. 5, pp. 427–443, Jun. 2003.

[7] F. I. Kandil, A. Rotter, and M. Lappe, "Driving is smoother and more stable when using the tangent point," *J. Vis.*, vol. 9, no. 1, pp. 1–11, Jan. 2009.

[8] K. A. Brookhuis, D. de Waard, and W. H. Janssen, "Behavioural impacts of advanced driver assistance systems an overview," *Eur. J. Transp. Infrastructure Res.*, vol. 1, no. 3, pp. 245–253, 2001.

[9] A. Lindgren and F. Chen, "State of the art analysis: An overview of advanced driver assistance systems (ADAS) and possible human factors issues," in *Proc. Human Factors Eco. Aspects Saf. Swedish Netw. Human Factors Conf.*, 2007, pp. 38–50.

[10] J. F. Coughlin, B. Reimer, and B. Mehler, "Driver wellness, safety & the development of an awarecar," AgeLab, Mass Inst. Technol., Cambridge, MA, 2009.

[11] S. Hoch, M. Schweigert, F. Althoff, and G. Rigoll, "The BMW surf project: A contribution to the research on cognitive vehicles," in *Proc. Intell. Veh. Symp.*, 2007, pp. 692–697.

[12] E. D. Dickmanns and V. Graefe, "Dynamic monocular machine vision," *Mach. Vis. Appl.*, vol. 1, no. 4, pp. 223–240, 1988.

[13] M. A. Turk, D. G. Morgenthaler, K. D. Gremban, and M. Marra, "VITS-a vision system for autonomous land vehicle navigation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 3, pp. 342–361, May 1988.

[14] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L. E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, P. Mahoney, "Stanley: The robot that won the darpa grand challenge," *J. Robot. Syst.*, vol. 23, no. 9, pp. 661–692, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1002/rob.v23:9

[15] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, J. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, A. Kelly, D. Kohanbash, M. Likhachev, N. Miller, K. Peterson, R. Rajkumar, P. Rybski, B. Salesky, S. Scherer, Y. Woo-Seo, R. Simmons, S. Singh, J. Snider, A. Stentz, W. Whittaker, and J. Ziglar, "Tartan racing: A multi modal approach to the DARPA urban challenge," Defense Advanced Res. Projects Agency, Arlington, VA, DARPA Tech. Rep., 2007.

[16] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[17] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems 1*. San Mateo, CA: Morgan Kaufmann, 1989.

[18] D. A. Pomerleau, "Neural network based autonomous navigation," in *Proc. NAVLAB*, 1990, pp. 558–614.

[19] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural Comput.*, vol. 3, no. 1, pp. 88–97, 1991.

[20] M. Arbib, "Neural network vision for robot driving," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA: MIT Press, 1999.

[21] M. Pasquier and R. J. Oentaryo, "Learning to drive the human way: A step towards intelligent vehicle," *Int. J. Veh. Auton. Syst.*, vol. 6, no. 1/2, pp. 24–47, Dec. 2007. [Online]. Available: http://dx.doi.org/10.1504/IJVAS.2008.016477

[22] C. Sammut, S. Hurst, D. Kedzier, and D. Michie, "Learning to fly," in *Proc. Mach. Learn.*, 1992, pp. 385–393.

[23] A. Y. Ng and S. Russel, "Algorithms for inverse reinforcement learning," in *Proc. ICML*, 2000, pp. 663–670.

[24] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, p. 1.

[25] *Official Diplecs Website*, 2010. [Online]. Available: http://www.diplecs.eu/

[26] C. H. Hwang, N. Massey, B. W. Miller, and K. Torkkola, "Hybrid intelligence for driver assistance," in *Proc. FLAIRS*, 2003, pp. 281–285.

[27] R. Freymann, "Driver assistance technology to enhance traffic safety," in *Motion and Vibration Control*. Berlin, Germany: Springer-Verlag, 2008, pp. 71–81.

[28] K. Brookhuis and D. de Waard, "Limiting speed, towards an intelligent speed adapter (ISA)," *Transp. Res. F, Traffic Psychol. Behav.*, vol. 2, no. 2, pp. 81–90, Jun. 1999.

[29] A. Tahirovic, S. Konjicija, Z. Avdagic, G. Meier, and C. Wurmthaler, "Longitudinal vehicle guidance using neural networks," in *Proc. CIRA*, 2005, pp. 685–688.

[30] D. Partouche, M. Pasquier, and A. Spalanzani, "Intelligent speed adaptation using a self-organizing neuro-fuzzy controller," in *Proc. IEEE Intell. Veh. Symp.*, 2007, pp. 846–851.

[31] H. Kwasnicka and M. Dudala, "Neuro-fuzzy driver learning from real driving observations," in *Proc. Artif. Intell. Control Manag.*, 2002, pp. 81–89.

[32] I. Markelic, T. Kulvicius, M. Tamosiunaite, and F. Wörgötter, "Anticipatory driving for a robot-car based on supervised learning," in *Proc. ABiALS*, 2008, pp. 267–282.

[33] D. W. Aha, Ed., "Editorial," in *Lazy Learning*. Norwell, MA: Kluwer, 1997, ser. Artificial Intelligence Review, pp. 7–10.

[34] J. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *J. Opt. Soc. Amer. A, Opt. Image Sci. Vis.*, vol. 2, no. 7, pp. 1160–1169, Jul. 1985.

[35] K. Pauwels and M. Van Hulle, "Realtime phase-based optical flow on the GPU," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog., Workshop Comput. Vis. GPU*, 2008, pp. 1–8.

[36] T. Gautama and M. Van Hulle, "A phase-based approach to the estimation of the optical flow field using spatial filtering," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1127–1136, Sep. 2002.

[37] S. Sabatini, G. Gastaldi, F. Solari, K. Pauwels, M. Van Hulle, J. Diaz, E. Ros, N. Pugeault, and N. Krüger, "A compact harmonic code for early vision based on anisotropic frequency channels," *Comput. Vis. Image Understanding*, vol. 114, no. 6, pp. 681–699, Jun. 2010.

[38] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool, "Coupled object detection and tracking from static cameras and moving vehicles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1683–1698, Oct. 2008.

[39] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool, "Dynamic 3d scene analysis from a moving vehicle," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Minneapolis, 2007, pp. 1–8.

[40] K. Pauwels, N. Krueger, M. Lappe, F. Woergoetter, and M. van Hulle, "A cortical architecture on parallel hardware for motion processing in real-time," *J. Vis.*, vol. 10, no. 10, pp. 1–21, Aug. 2010.

[41] T. Zhang and C. Tomasi, "On the consistency of instantaneous rigid motion estimation," *Int. J. Comput. Vis.*, vol. 46, no. 1, pp. 51–79, Jan. 2002.

[42] F. Mosteller and J. Tukey, *Data Analysis and Regression: A Second Course in Statistics*. Reading, MA: Addison-Wesley, 1977.

[43] W. Thompson and T. Pong, "Detecting moving-objects," *Int. J. Comput. Vis.*, vol. 4, no. 1, pp. 39–57, 1990.

[44] M. Bertozzi and A. Broggi, "Real-time lane and obstacle detection on the system," in *Proc. IEEE Intell. Veh.*, 1996, pp. 213–218.

[45] E. Dickmanns, *Dynamic Vision for Perception and Control of Motion*. Berlin, Germany: Springer-Verlag, 2007.
[46] M. Aly, "Real time detection of lane markers in urban streets," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 4–6, 2008, pp. 7–12.
[47] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.
[48] I. Sobel and G. Feldman, *A 3 × 3 Isotropic Gradient Operator for Image Processing*. Hoboken, NJ: Wiley, 1973.
[49] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960.
[50] I. Markeli, "Teaching a robot to drive—A skill learning inspired approach," Ph.D. dissertation, Georg-August Univ. Göttingen, Göttingen, Germany, 2010.
[51] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.
[52] *Eisat Website*, 2010. [Online]. Available: http://www.mi.auckland.ac.nz/EISATS
[53] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica, Int. J. Geogr. Inf. Geovis.*, vol. 10, no. 2, pp. 112–122, Dec. 1973.
[54] L. Bottou and V. Vapnik, "Local learning algorithms," *Neural Comput.*, vol. 4, no. 6, pp. 888–900, Nov. 1992.
[55] J. C. Santamaria, R. S. Sutton, and A. Ram, "Experiments with reinforcement learning in problems with continuous state and action spaces," *Adapt. Behav.*, vol. 6, pp. 163–217, 1997.
[56] W. D. Smart and L. P. Kaelbling, "Practical reinforcement learning in continuous spaces," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 903–910.
[57] *Future and Emerging Technologies (FET), Newsletter*, Jan. 2010. [Online]. Available: ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/fet/fet-nl-06_en.pdf
[58] *TV Report About the Drivsco System (in German)*, 2009. [Online]. Available: http://www1.ndr.de/mediathek/index.html?media=ndsmag2340

Irene Markelić received the B.Sc. and M.Sc. degrees in computer science from the University of Koblenz-Landau, Koblenz, Germany, in 2002 and 2005, respectively, and the Ph.D. degree in computer science from Georg-August-University Göttingen, Göttingen, Germany, in 2010.

She is currently a Postdoctoral Researcher with the Institute of Physics 3, Georg-August-University Göttingen. Her research interests include computer vision, developmental robotics, cognitive skill learning in humans and robots, and data mining.

Anders Kjær-Nielsen received the B.Sc. and M.Sc. degrees in computer systems engineering, in 2004 and 2007, respectively, from the University of Southern Denmark, Odense, Denmark, where he is currently working toward the Ph.D. degree with the Mærsk McKinney Møller Institute.

His research interests include real-time processing, computer vision, embedded systems, and field-programmable gate arrays.

Karl Pauwels received the M.Sc. degree in commercial engineering, the M.Sc. degree in artificial intelligence, and the Ph.D. degree in medical sciences from the Katholieke Universiteit Leuven (K.U.Leuven), Leuven, Belgium.

He is currently a Postdoctoral Researcher with the Laboratorium voor Neuro- en Psychofysiologie, Medical School, K.U.Leuven. His research interests include dense optical flow, stereo and camera motion estimation, video stabilization, and real-time computer vision.

Lars Baunegaard With Jensen received the B.Sc. and M.Sc. degrees in computer systems engineering in 2004 and 2007, respectively, from the University of Southern Denmark, Odense, Denmark, where he is currently working toward the Ph.D. degree with the Mærsk McKinney Møller Institute.

His research interests include real-time computer vision and graphics processing unit computing.

Nikolay Chumerin received the M.Sc. degree in mathematics and educational science and the Higher Educational Certificate of teacher in mathematics and computer science from Brest State University, Brest, Belarus, in 1999. He is currently working toward the Ph.D. degree with the Laboratorium voor Neuro- en Psychofysiologie Medical School, Katholieke Universiteit Leuven (K.U.Leuven), Leuven, Belgium.

His research interests include biologicallyinspired computer vision, machine learning, and brain computer interfacing.

Aušra Vidugiriene received the B.Sc. and M.Sc. degrees in computer science in 2003 and 2005, respectively, working on language technologies, from Vytautas Magnus University, Kaunas, Lithuania, where she is currently working toward the Ph.D. degree.

Her research interests include signal processing and analysis of driver's behavior.

Minija Tamosiunaite received the Engineer Diploma from Kaunas University of Technology, Kaunas, Lithuania, in 1991 and the Ph.D. degree from Vytautas Magnus University, Kaunas, in 1997.

Her research interests include machine learning and applications in robotics.

Alexander Rotter received the Diploma degree in electrical engineering and the Diploma degree in industrial engineering from Fachhochschule Sdwestfalen, Iserlohn, Germany, in 2002 and 2007, respectively.

He is currently with the Advanced Development Department, Hella KGaA Hueck & Co., Lippstadt, Germany. His research interests are vision-based driver-assistance systems, including object and lane detection, and emergency braking systems in combination with distance measurement sensors, and vision-based driver-assistance systems, object detection, and emergency braking systems, in combination with distance measurement sensors.

Marc Van Hulle received the M.Sc. degree in electrotechnical engineering (electronics) and the Ph.D. degree in applied sciences from the Katholieke Universiteit Leuven (K.U.Leuven), Leuven, Belgium, the Doctor Technics degree from Queen Margrethe II of Denmark, Odense, in 2003 from the Danmarks Tekniske Universitet (DTU), Lyngby, Denmark, and the Honorary Doctoral degree from Brest State University, Brest, Belarus, in 2009.

He is currently a Full Professor with K.U.Leuven Medical School, where he heads the Computational Neuroscience Group of the Laboratorium voor Neuro- en Psychofysiologie. In 1992, he was with the Brain and Cognitive Sciences Department, Massachusetts Institute of Technology, Boston, as a Postdoctoral Scientist. His research interests include computational neuroscience, neural networks, computer vision, data mining, and signal processing.

Norbert Krüger received the M.Sc. degree from Ruhr-Universität Bochum, Bochum, Germany, and the Ph.D. degree from the University of Bielefeld, Bielefeld, Germany.

He is currently a Professor with Mærsk McKinney Møller Institute, University of Southern Denmark, Odense, Denmark. He is a partner in several European Union and national projects PACO-PLUS, Drivsco, NISA, and Handyman. He is leading the Cognitive Vision Laboratory, which focuses on computer vision and cognitive systems, particularly the learning of object representations in the context of grasping. He has also been working on computational neuroscience and machine learning.

Florentin Wörgötter received the Ph.D. degree from the University of Essen, Essen, Germany, in 1988.

He worked experimentally on visual cortex before he turned to computational issues at the California Institute of Technology, Pasadena, from 1988 to 1990. After 1990, he became a Researcher with the University of Bochum, Bochum, Germany, where he was concerned with experimental and computational neuroscience of the visual system. Between 2000 and 2005, he was a Professor of computational neuroscience with the Department of Psychology, University of Stirling, Stirling, U.K., where his interests strongly turned toward "learning in neurons." Since July 2005, he has been leading the Department for Computational Neuroscience, Bernstein Center, Georg-August-University Göttingen, Göttingen, Germany. His research interests are information processing in closed-loop perception-action systems, including aspects of sensory processing, motor control, and learning/plasticity. These approaches are tested in walking and driving robotic implementations. His group has developed the RunBot, which is a fast and adaptive biped walking robot.