Robust Environment Perception for the Audi Autonomous Driving Cup

Conference Paper · November 2016

DOI: 10.1109/ITSC.2016.7795744

CITATIONS
4

READS
1,079

8 authors, including:

Florian Kuhnt
FZI Forschungszentrum Informatik
20 PUBLICATIONS **143** CITATIONS
SEE PROFILE

Micha Pfeiffer
German Cancer Research Center
6 PUBLICATIONS **8** CITATIONS
SEE PROFILE

David Zimmerer
German Cancer Research Center
8 PUBLICATIONS **37** CITATIONS
SEE PROFILE

J. Marius Zöllner
Karlsruhe Institute of Technology
230 PUBLICATIONS **1,797** CITATIONS
SEE PROFILE

2016 IEEE International Conference on Intelligent Transportation Systems, 1-4 November 2016.

Robust Environment Perception for the Audi Autonomous Driving Cup

Florian Kuhnt, Micha Pfeiffer, Peter Zimmer, David Zimmerer,
Jan-Markus Gomer, Vitali Kaiser, Ralf Kohlhaas and J. Marius Zöllner[1]

***Abstract*— One of the biggest challenges towards fully automated driving is achieving robustness. Autonomous vehicles will have to fully recognize their environment even in harsh weather conditions. Additionally, they have to be able to detect sensor and algorithm failures and react properly to keep the vehicle in a safe state.**

These two challenges are addressed exemplarily on miniature cars. We extend the approach of Compositional Hierarchical Models [1] by temporal fusion to achieve a robust environment perception. The increased association problem is overcome by a grid-based approximation and a voting system. System performance assessment surveils the system's performance and reacts with driving function degradation or activation of specialized algorithms.

The approach was evaluated at the final of the Audi Autonomous Driving Cup 2016. A video shows the advanced driving capabilities under harsh environment conditions and the source code is available for download.

I. Introduction

Autonomous Driving is one of the central research topics today. Advanced Driving Assistance Systems [2] evolved to complex systems with the capability to drive autonomously under sufficient circumstances [3] [4] [5]. To make fully automated driving possible for end users, one of the biggest challenges is to achieve robustness [6]: A vehicle has to perform safely in every situation including harsh environment conditions, unambiguous situation perception, limited sensor performance and non-compliant behavior of traffic participants. Two of the most promising approaches to cope with the challenges are:

1) Combining redundant but complementary sensors and features to derive an environment model in a hierarchical, probabilistic way.
2) Assess the performance of single sensors and modules to be aware of the system's performance state and being able to react on sensor degradation by driving function degradation.

In this paper we show a probabilistic hierarchical approach for environment perception in combination with a system performance assessment. The probabilistic hierarchical model allows for a robust situation assessment even under harsh conditions and when information is missing. In the rare scenarios where situation assessment fails, this can be recognized by the system performance assessment and recovering strategies can be applied.

[1]The authors are with FZI Research Center for Information Technology, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany {kuhnt, pfeiffer, zimmer, zimmerer, gomer, kaiser, kohlhaas, zoellner}@fzi.de

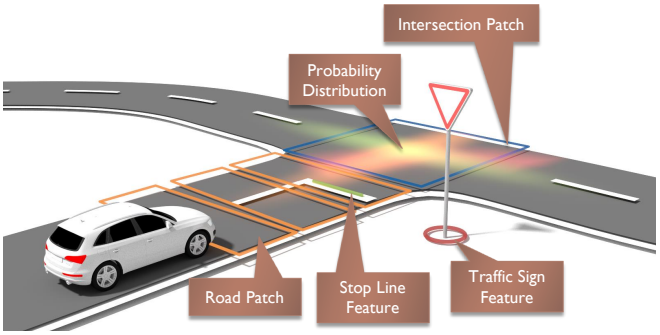

Fig. 1: For robust road layout perception a probabilistic hierarchical model is used: Detectable features like stop lines (green) or traffic signs (red) and road patches (orange) are combined using spatial constraints to infer the most probable locations of higher level objects like intersections (blue).

The concept was implemented within the scope of this year's Audi Autonomous Driving Cup[2] (AADC) and thus tested in the final event of the contest. The competition includes basic autonomous driving tasks like lane keeping, turning at intersections and parking maneuvers. In the final round of the event additional harsh conditions including unexpected environments, sensor irritations and non-compliant driving behavior of other traffic participants are added. Hence, the applicability of our approach could be demonstrated within the contest. The presentation of the concept was awarded with the price for best technical presentation and the overall performance led to the second place in the whole competition.

II. Requirements: The Audi Autonomous Driving Cup

The Audi Autonomous Driving Cup (AADC) is an international student competition, in which teams of five students each have six months to program a self driving car. All teams are given two computerized model cars (scale 1:8 of the original car) with fixed hardware setup and have to control the sensors and actors to solve driving tasks.

A. Tasks

While staying in the right lane is a basic prerequisite there are several tasks which have to be completed during the competition:

- Emergency stop: Stop in front of an unexpected occurring obstacle.

[2]AADC: www.audi-autonomous-driving-cup.com

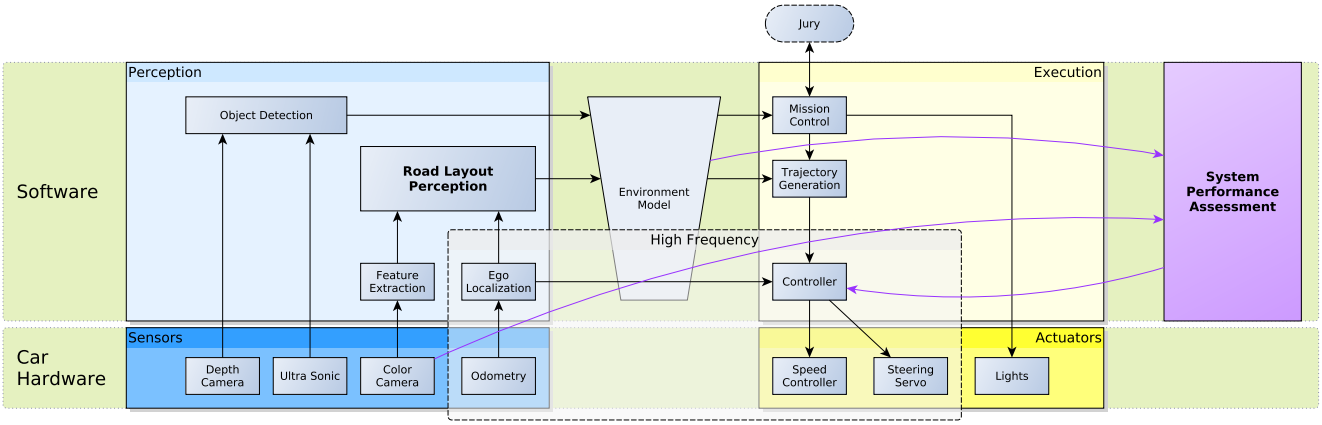

Fig. 2: The system architecture follows the *perception-cognition-action* approach: Perception provides an environment model that can be used by execution modules. For robustness, a *system performance assessment* surveils the components to react in case of failures.

- Adaptive cruise control: Adjust the vehicle speed to maintain a safe distance to a vehicle ahead.
- Overtaking: Avoid and overtake obstacles on the road.
- Intersection: Stop at an intersection, observe other road users and cross according to the road traffic regulations.
- Parking: Park either parallel or perpendicular to the road into the next free parking space.

A list of commands to carry out is generated by the jury and passed to the car via a wireless connection.

B. Cars

The cars contain different sensors which capture the environment (Fig. 2): Ten ultrasonic sensors around the car, a triple axis accelerometer with gyro and quadrature wheel encoders for odometry and a color and depth camera as the main sensor for situation assessment. The camera can output a 640x480 pixel image with a fame-rate of 30fps and a field of view of about $50°$. To process the sensor values, a miniPC with a 1.9 GHz Intel Atom E3845 and 8GB RAM is mounted on the car. The calculated target speed and steering angle are forwarded and applied to a steering servo and a brushless speed controller.

C. Environment

The road is composed of 1x1 m dark gray tiles with white markings. There are four basic classes of tiles: straight road patches, curved road patches, intersecting roads, and parking spaces. The contour lines have a width of 30 mm, the center lines have a width of 20 mm and the stop lines have a width of 50 mm.

In addition there are traffic signs, such as stop signs, give way signs, parking area signs, roundabout signs and pedestrian crossing signs, which have to be respected. Each traffic sign has a machine-readable code attached to it which simplifies detection. Further elements like obstacles, plants, tunnels and light sources to achieve different light conditions were added to the parcour.

III. System Overview

Our system architecture for the autonomous miniature vehicles follows the typical *perception-cognition-action* approach of recent autonomous vehicles [6] including *ego vehicle localization*, *road layout perception* and *object detection* to create a coherent *environment model*, which serves as a basis for a central *mission control* based on a state machine and a *trajectory generation* module creating drivable trajectories that are executed by a *high frequency controller* (Fig. 2). The key features that make the system robust are

- the probabilistic *road layout perception* that incorporates several environment features to estimate the most probable road layout (Sec. IV) and
- the attached *system performance assessment* module that observes the systems performance independently from the basic algorithms to detect failures and respond with proper function degredation or activation of suitable specialized algorithms (Sec. VI).

A brief overview about the other components is given in section V.

IV. Road Layout Perception

To solve the tasks posed during the cup, a robust road detection algorithm is needed. This is not simple to accomplish: The camera field of view is very limited – in many cases, only small portions of the road can be seen. The camera's resolution is also often too low to reliably distinguish all features in an image. To overcome these challenges, a probabilistic approach is used which can cope with the many uncertainties.

Compositional Hierarchical Models (CHM) [1] have been introduced during recent years to cope with the challenges of multi-lane road perception in highway situations but also in urban environments, especially when roads split or merge. In the case of urban intersections, established basic lane tracking algorithms [7] are not sufficient. A unified framework is needed that incorporates low-level features with high-level model knowledge in a probabilistic way. With the aid of part-sharing [8] and depth-first message passing [9], CHMs

are capable to perceive real highway and urban scenarios in sufficient runtime [1].

The basis of the CHM is a pairwise Markov Random Field that describes the dependencies between variables as potentials. Using the hierarchical representation a complex scene is decomposed into parts and subparts (Fig. 1): The global problem is split into problems of lower complexity. A priori information can easily be incorporated into spatial constraints between subparts. Additionally local differences, for example illumination changes, have a limited effect on the overall perception. Since Markov Random Fields are probabilistic graphical models, the inference algorithm is often described using a factor graph representation [10].

In the following we present the adaptions of the concept of CHMs to the limited miniature world of the AADC:

- using a *reduced hierarchical model*,
- describing spatial constraints and dependencies between variables as *probability distributions*,
- adding *temporal fusion* via temporal spatial constraints,
- replacing the depth-first message passing inference strategy by our own approach,
- and utilizing a *grid-based state space*.

A. Variable Definition

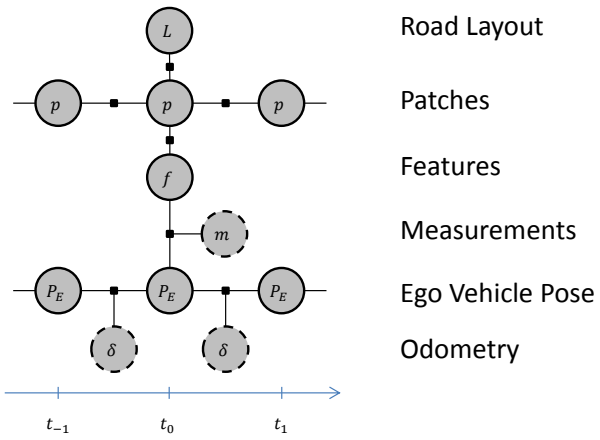

Fig. 3: Variables used in the temporal extension of the CHM.

While the original CHM made heavy use of the hierarchic component to cope with the variety of real road layouts, we reduce the hierarchic component to fit the requirements of the AADC. Instead of features, patches, lane segments and multi-lane road layouts, we use the following variables (Fig. 3):

1) Feature Variables: All feature variables f match the sensory evidences m observable from the environment. The feature variables consist of a position $(x, y) \in \mathbb{R}^2$ and orientation $\phi \in [0, \pi)$ in a world coordinate system and a feature type $\tau_f \in T_f := \{\text{center\_line}, \text{side\_line}, \text{stop\_line}, \text{parking\_line}, \text{traffic\_sign}\}$ to differ between different features:

$$f = (x, y, \phi, \tau_f) \tag{1}$$

2) Patch Variables: Patch variables p are defined in the same coordinate system as the feature variables. Each patch variable fully describes a hypothesis for a single patch. Besides position $(x, y) \in \mathbb{R}^2$, $\phi \in [0, \pi)$ and patch type $\tau_p \in T_p := \{\text{road}, \text{intersection}, \text{parking}\}$ they include a width w and length l:

$$p = (x, y, \phi, \tau_p, w, l) \tag{2}$$

For width and length, prior knowledge can be incorporated according to the three different types of patch variables:

- A *road patch* represents the full width of the uniform road with an adjustable fixed length. This is sufficient since during the AADC the number of lanes, as well as the width of the road, are fixed. This road patch is not only used for straight roads – multiple road patches are also used to approximate curves.
- An *intersection patch* matches the width and length of the intersection tiles.
- A *parking lot patch* matches the width and length of parking lots which are known to be parallel or perpendicular to the road.

B. Spatial Constraints

Both the feature variables and the patch variables contain uncertainties: The features are derived from measurements and contain measurement errors (in the worst case, wrong features can be generated). The patch hypotheses are derived from a mix of a-priori knowledge, the current car pose – which is approximated – and the uncertain features.

This is why edges in the factor graph define weak spatial constraints between the corresponding entities, and probability distributions are used to model these constraints. In general, a spatial constraint between two variables $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ can be denoted by a potential

$$\psi_{i,j}(\boldsymbol{v}_i, \boldsymbol{v}_j) = P_{[\theta]}(\boldsymbol{v}_j, S(\boldsymbol{v}_i)), \tag{3}$$

where P is a general probability distribution of the two variables $\boldsymbol{v}_j$ and a transformation S from $\boldsymbol{v}_i$ with additional parameters θ. Depending on the feature-type, the probability distributions can be gaussian, multi-modal or even completely non-gaussian.

The definition of P, S and the parameters θ largely depends on the actual spatial dependencies of the entities in the real world. The following spatial constraints are applied between the previously defined features and patches:

1) Feature-Patch constraints: All feature-patch constraints describe the spatial dependencies between observable features and patches. They are represented by

$$\psi_{i,j}(f_i, p_j) = P_{[\theta]}(p_j, S(f_i)). \tag{4}$$

Lateral features like center lines and side lines are connected to road patches and have a tight lateral coupling while their longitudinal dependency is with high variance (Fig. 4). Similarly, stop lines are connected to intersection patches and parking lines to parking space patches with high longitudinal precision. The estimation of the orientation of traffic signs

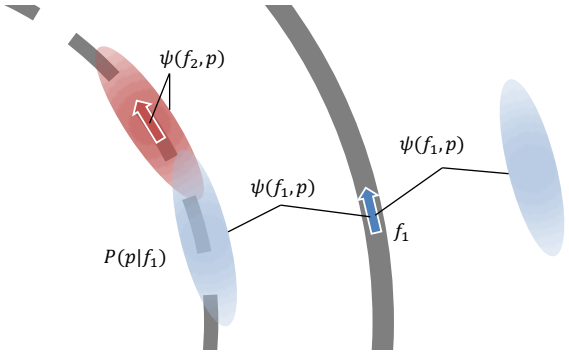

Fig. 4: Spatial constraints ψ describe the relationship between features f and patches p. Exemplarily a side line feature f_1 (blue arrow) and a center line feature f_2 (red arrow) are shown. While the spatial constraint for the center line feature can be modeled with a single gaussian distribution, the one for the side line feature has a bimodal gaussian distribution.

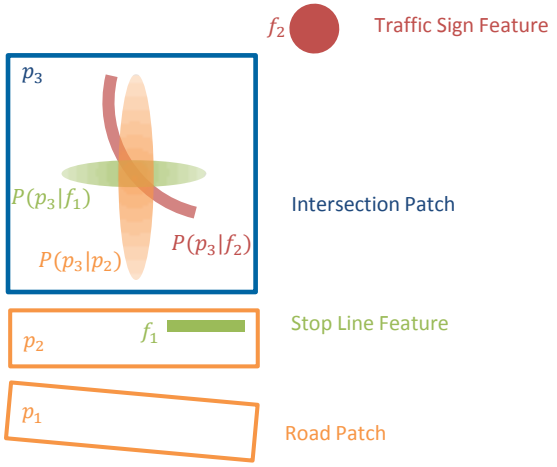

Fig. 5: Spatial constraints for intersection patches on the example of Fig. 1. The combination of previously found road patches p_1, p_2, stop line features f_1 and traffic sign features f_2 give a precise estimation of the intersection patch $P(p_3|f_1, f_2, p_2)$.

can be very imprecise due to the marker detection algorithm. This issue was overcome by assigning a spatial constraint in the form of a quarter circle to them (i.e. large angle variance), as can be seen in Fig. 5.

2) Patch-Patch constraints: All patch-patch constraints are represented by

$$\psi_{i,j}(p_i, p_j) = P_{[\theta]}(p_j, S(p_i)). \tag{5}$$

Road-Road-Constraints enforce the consistency of successive road patches. The road patches at the arms of an intersection are coupled to the intersection so that they help localizing the intersection (Fig. 5). Similarly, parking spaces are connected to road patches.

C. Temporal Constraints

Temporal constraints are used to incorporate information from previous time-frames. All spatial constraints from patch hypotheses found in the past are re-used in the current frame. Additionally, a temporal constraint is added for each patch

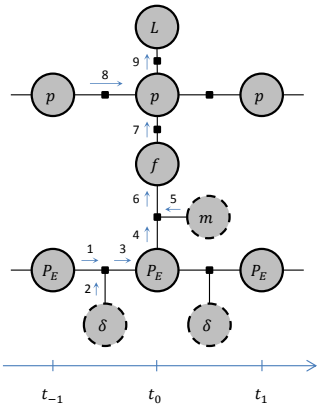

Fig. 6: Numbered arrows show message passing schedule. Step 1 - 6 are part of the feature extraction (Sec. V-A). Step 7 and 8 are combined with the intra-frame patch spatial constraints using a grid-based voting space.

hypothesis (see step 8 in Fig. 6) which describes the position of the patch itself. In this case, the transformation S of the probability distribution is the identity transformation:

$$\psi_{T\,i,j}(p_i, p_j) = \mathcal{N}(p_j; p_i, \Sigma_{i,j}). \tag{6}$$

This allows tracking of a single patch in multiple frames, which drastically improves the stability of its localization.

D. Inference and the Association Problem

The inference strategy largely depends on the variables of interest and on the state space representation. In the case of road layout perception, the variables of interest are the patch variables (Sec. IV-A.2) that will be estimated using sensor evidences represented by the feature variables (Sec. IV-A.1). Spatial constraints (Sec.IV-B) bring these variables into context. A corresponding message passing schedule is shown in Fig. 6.

In our algorithm, the first six steps of the message passing schedule are part of the image processing procedure, which, using the current estimated car pose and the current camera image, returns a vector of features f (Sec. V-A). The next task is then to find a probability distribution $P(p_j|f)$ of all patch hypotheses p_j, which we choose to describe as a factorization of the spatial constraints:

$$P(p_j|f) = \prod_i \psi_{i,j}(f_i, p_j) \cdot \prod_i \psi_{i,j}(p_i, p_j) \cdot \prod_i \psi_{T\,i,j}(p_i, p_j) \tag{7}$$

When inferring patches from features and past patches, these have to be associated correctly. This association problem has a high complexity which can be addressed by depth-first message passing for single-frame estimations [9]. Since we have introduced additional temporal dependencies, we have this association problem on the level of all the spatial constraints $\psi_T(p, p)$, $\psi(p, p)$ and $\psi(f, p)$ (Fig. 7). Therefore we introduce a grid-based approximation of the whole probability distribution over all patch hypotheses.

This grid-based representation is used to overcome the association problem on a per timestep basis. After estimating

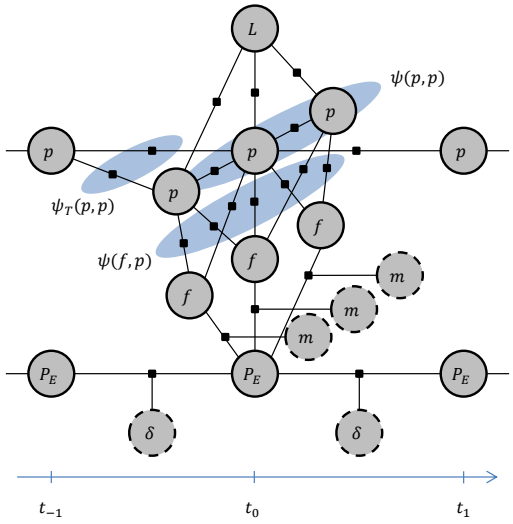

Fig. 7: For every time step t several measurements m are observed, relative to the ego vehicle pose P. For every measurement, a feature f can be estimated, which itself can support several patch hypotheses p. The complexity problem in associating features with past and current patch variables $(\psi_T(p,p), \psi(p,p), \psi(f,p))$ is overcome by using a grid-based space approximation for all patch variables p.

the patch space, single parametric patch observations are extracted and merged into the environment model holding the road layout consisting of temporally smoothed patches. The smoothing also reduces discretization errors and thus allows a quite rough grid representation.

E. Grid-based Patch Voting Space

The patch space for finding the most probable patch hypotheses in Eq. 7 is approximated by a 4-dimensional grid. Since width and height of patches are given by prior knowledge, the unknowns to estimate for a patch hypothesis are the position and angle as well as the type of the patch:

$$p \in X \times Y \times \Phi \times \mathrm{T}_p \tag{8}$$

Since new features are only found in the area seen by the camera, the ranges of X and Y are chosen in such a way that they represent the space in front of the car. Depending on the position and angle of the camera, a point $P_c = (x_c, y_c)$ in front of the car is chosen as the center of the discretized search space. X and Y can then be written as:

$$X = \{x_i \,|\, x_i = x_c + i\Delta s\}$$
$$Y = \{y_i \,|\, y_i = y_c + i\Delta s\}$$

and

$$i = \{-N/2, -N/2+1, \ldots, N/2-1\}$$

where the side-length of each cell Δs and the grid size N are chosen in such a way that X and Y describe a large area in front of the car. Since all patches have a rotation-symmetry of π, only the angle range from 0 to π needs to be used and the angle can be discretized by:

$$\Phi = \{0, \frac{\pi}{M}, \frac{2\pi}{M} \ldots, \pi - \frac{\pi}{M}\}. \tag{9}$$

Using this discretized patch space, the patch distribution $P(p_j|f)$ can be approximated by applying the spatial constraints on the observed features and combining their results in the grid.

To implement this concept on a low-performance PC we introduce a voting system where each spatial constraint is approximated by a vote in the discretized grid.

To do so, we take the logarithm of both sides of Eq. 7 and use the logarithmic rule $\log(ab) = \log(a) + \log(b)$:

$$\log(P(p_j|f)) = \sum_i \log(\psi_{i,j}(f_i, p_j)) \tag{10}$$
$$+ \sum_i \log(\psi_{i,j}(p_i, p_j)) \tag{11}$$
$$+ \sum_i \log(\psi_{T\,i,j}(p_i, p_j))) \tag{12}$$

In this new function, the probability distributions are combined via additions instead of multiplications. To further decrease computation time, the probability distributions are approximated by box-functions. The probability distributions of all spatial constraints for all features and detected patches are added together in the patch- or voting-space. The result is blurred using a two dimensional blur filter to mimic the behavior of smooth probability functions. The local maxima in the search space now represent new patch hypotheses. An acceptance-threshold is applied to ensure a patch is only found if enough features and patches voted for it.

Using this method, the weights and sizes of the votes can be parametrized to model for example the confidence of different perception and prediction algorithms. Features which occur less often can cast votes with a higher weight and features which are found to be less stable can have wider votes with a lower weight. When the system performance assessment detects failures, the parameters can be changed to reach higher precision (Sec. VI). Additionally, a-priori information at system start can easily be incorporated by distributing inital patch votes, information like *starting from a parking lot* or *starting on the right lane*.

V. System Integration

To perform the tasks demanded by the competition the concept has to be included into an overall architecture. In the following different important parts of the architecture including feature extraction, environment representation, mission control and trajectory generation are explained briefly.

A. Feature extraction

The road perception system needs features which are generated by processing the camera images. Features each hold a position and angle, as well as the feature-type.

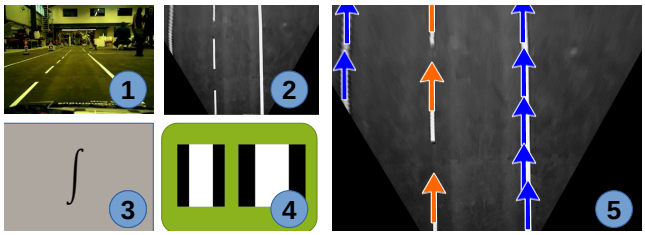

Fig. 8: The line feature extraction strategy. 1: The input image. 2: Inverse Perspective Mapping. 3: Integral images are calculated from the IPM. 4: Haar-Features as applied to the image, for a thin center line and a thicker side line, 5: Examples of the found side line (blue) and center line (red) features displayed on top of the IPM.

1) Line features: Line features are essential for road, intersection and parking lot detection. Therefore, first an Inverse Perspective Mapping is performed on the gray scale camera image. The result is a top-view image of the area in front of the car. On this image Haar Line filters [11] are used to generate features. The lines which these features represent can have an arbitrary angle in the image. Hence, in another pass, Haar Line features rotated by $\frac{\pi}{4}$ are also calculated [12]. The responses of both filtering passes are used to approximate Haar Line features of arbitrary angles.

Since the width of the center, side and stop lines differ on the Audi track, different Haar Line features, which represent the different line widths, are used. The responses of the features are normalized, such that the impact of the different surface areas of the features are taken into account. For each pixel, the line feature with the highest normalized response wins and is considered in the voting process. This makes the filter robust against a change in lighting. Confidence values for each feature can be calculated by dividing the filter response by the (theoretical) maximum response. These confidence are used as weights for the features during the voting process. For speed reasons, the filters are not applied to the top-view images directly, but instead integral images are used. To further decrease computation time, not all pixels are analyzed. The described process is also shown in Fig. 8.

2) Traffic signs: At the AADC, each traffic sign has a machine-readable marker attached to it. The ArUco library [13] is used to determine the type of the traffic sign, its location and orientation. While the position can be recognized very precisely, the uncertainty in orientation is considered in the observation potentials of the CHM.

B. Environment Model

Patches, obstacles, traffic signs and semantic information are stored in an environment model. Those are continuously updated to be as accurate as possible depending on new detections, the proximity to the car and the time of the last measurements.

C. Trajectory Generation

The trajectory is generated using the patches in the found road hypothesis: Each patch stores predefined path segments per lane, entry point and desired exit point. Combining these path segments, a trajectory can be created according to the current maneuver plan.

The trajectory is resampled, interpolated by a B-Spline interpolation and simplified with the Ramer Douglas Peucker algorithm to get a smooth curve which is then passed to the lateral and longitudinal controllers.

D. Mission Control

The high level decision making process is performed by a hierarchical and event-based state machine. State transitions can be either temporally triggered by timers or spatially triggered by entering or leaving defined semantic regions.

The primary states are:

- Driving
- Traversing an intersection
- Parking
- Overtaking an obstacle
- Emergency

The state machine also parses the list of commands which the car receives from the jury. The knowledge about the next maneuver is used to configure the road perception algorithm: For example, when the next maneuver is the crossing of an intersection, the search for parking lots is disabled to decrease computation time.

VI. System Performance Assessment

The Performance Assessment module surveils the basic system, detects performance losses and initiates proper responses. Three exemplary rules were implemented to master the AADC:

A. Sensor Performance Assessment

Due to extreme lighting conditions it is possible that the frame rate of the front camera sensor drops. In this case the whole perception lags information and the environment cannot be estimated in time. The sensor performance assessment detects the frame-rate drops and degrades the driving function by reducing the driving speed.

B. Feature Extraction Performance Assessment

The number of different features extracted in each image is monitored. In case the number of features drops (for example due to different lighting conditions and lower contrast in the image), the thresholds used for feature extraction are adjusted dynamically. This ensures that enough features are available at all times for the voting system to work with.

C. Perception Performance Assessment

If no sensor degradation was detected but the road layout perception cannot find a proper road, the perception performance assessment detects this and reacts with a special emergency behavior: Driving speed is reduced to gain more time and feature extraction precision is maximized to get the highest possible quality. If the road perception still fails to find the road, new votes are introduced which are used to guess the road position depending on previous patches. These votes are very high compared to patch and feature

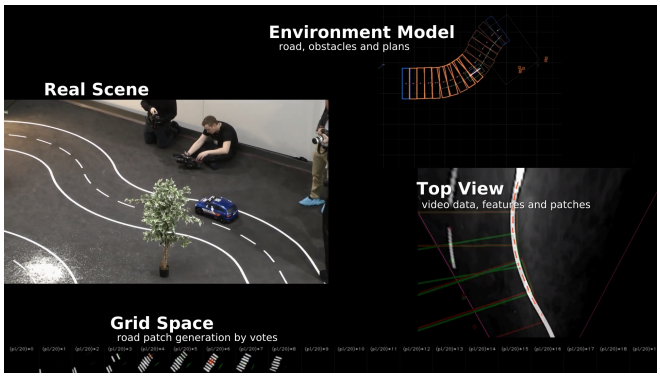

Fig. 9: Exemplary frame of the attached video. The video data recorded by the vehicle and used for perception is displayed in the lower right. Inverse perspective transform is already applied. Furthermore lane features (red lines) and patches (red and green boxes) are displayed. In the upper right the local map of the current state is shown. Road patches (red boxes) as well as the ego vehicles position (white arrow and green box) and the planned trajectory (white line strip) are displayed. In the bottom voting spaces for different orientations are shown.

votes and make sure that only very few features are needed to reach the patch acceptance threshold. If this still does not help in finding a new road, the car carefully starts following a short predefined search trajectory and starts the searching process again at a new position, while iteratively increasing these new emergency votes.

VII. Experiments and Results

The algorithms were tested extensively before and during the final event of the AADC. Even though the lighting conditions and the road surface at the cup were very different from the testing track during the development phase, only few adjustments had to be made to the configuration parameters. A configuration which we found successful in most situations is outlined in the following.

To generate the voting space, Δs was chosen to be 3 cm and N was set to 120, resulting in a search space in front of the car of about $13\mathrm{m}^2$. This area is larger than the area which is seen by the camera, allowing us to generate patch hypotheses for which the patches' centers lie outside the camera's field of view. The angle range $\{0 \dots \pi\}$ was discretized into 20 regions (i.e. $M = 20$). To reduce the number of false-positives the number of new patch hypotheses for straight patches was limited to three, the number of intersections and parking lots to one per frame.

The source code of the implementation is available for download[1] and only depends on open source software. A video of the final run of the Team KACADU is available for download[2]. Additionally to the live video of the competition, the video features three different visualizations of internal data. Fig. 9 shows an exemplary frame of the video.

In the following some interesting parts of the video are presented and the performance of the vehicle in these situations is discussed.

[1]Source code: `url.fzi.de/aadc2016_code`
[2]Video: `url.fzi.de/aadc2016`

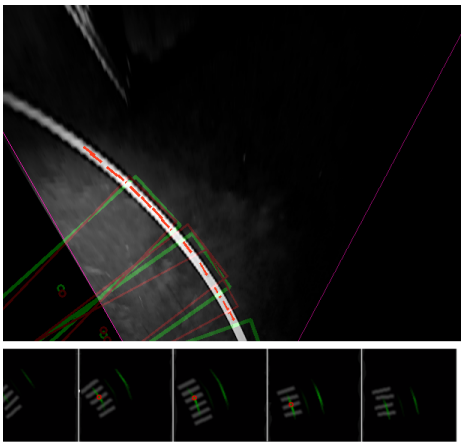

Fig. 10: Road detection on a curved road. The upper image shows the top-view generated from the input image. Only the right outer marking can be seen. These markings are detected using the Haar-Line filters which generate features (red lines on the lane marking). The lower image shows a portion of the voting space for straight patches ($\tau_p = \mathrm{road}$). Gray votes are generated by previously found patches, green votes are generated by the line features. The maxima which are found at the end of the voting process are marked with small red circles. Note that the patch votes locate new patch hypothesis mainly in the longitudinal position while feature votes locate them in the lateral position. Patches are shown as boxes in the top-view: Red patch hypotheses are the ones found in this frame, green patches are the merged, final road layout hypothesis from multiple frames. Even though only a very small portion of the road is visible, the algorithm correctly approximates the curve.

A. Basic Performance

Fig. 10 shows the result of our approach for a left curve. When on curved roads, the limited field of view leads to situations in which only the outer side line is visible in the top-view. Nevertheless, the road is detected and tracked accurately.

B. Glare Light

One of the challenges was the approach to a glare light. This generally leads to failures in perception, in our case the detection of line features was affected (Fig. 11). Due to the advanced System Performance Assessment, the vehicle was able to detect this situation, adjust to the new lighting condition and hence recover from this situation.

C. Snow

Another challenge in the final run was a snow spot representing harsh weather conditions. In the contest it was represented by a pile of rice. Because of its structure the ultra sonic sensors are confused. But also the camera introduced difficulties in this situation. Because of an error in the hardware or driver, the camera fails for several seconds. This is detected by the System Performance Assessment and the speed of the vehicle is reduced, gaining more time for the sensor to recover (Fig. 12).

VIII. Conclusions

Our approach to robust environment perception consists of two main components: The probabilistic hierarchical road perception algorithm and the system performance assessment module.

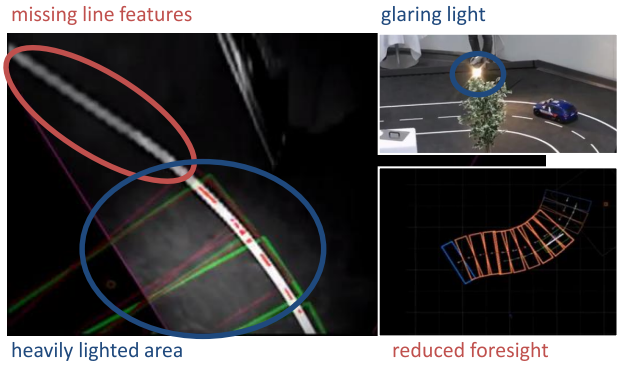

Fig. 11: Evaluation of the System Performance Assessment. The vehicle approaches a glare light (top right). Due to the light, some parts of the camera image are very bright, leading to missing lane markings and missing patches in the darker parts (left). However, the dynamic adjustment of the feature detection algorithm to the new lighting condition ensures that enough new features are found shortly after, enabling the vehicle to continue on its path. See seconds 30 to 40 in the attached video.

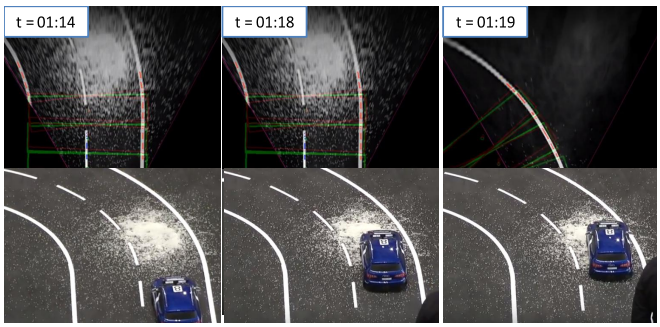

Fig. 12: Evaluation of the System Performance Assessment. The vehicle approaches a snow spot. Due to the big white area the camera sensor fails. There are no new video images for about 4 seconds (from 01:14 to 01:18 in the video). The error is detected, the vehicle slows down. Hence, it is still in a save position on the road and can proceed when the video sensor recovers (at 01:19 in the video).

The road perception is described on a factor graph model including spatial constraints between various features and temporal dependencies. The association problem of the many features and patches is solved using a grid-based approximation for the patch space.

This method proved to be very robust in the context of the Audi Autonomous Driving Cup. The probabilistic method was able to cope with difficult situations like missing road markings and different lighting conditions.

Additionally, the system performance assessment was able to detect sensor and algorithm failures and reacted with proper driving function degredations. The reduced speed helped several times to manage harsh weather conditions like sun glare or snow on the road during the final run of the cup.

The proposed approach is designed to perform in the AADC. Therefore some changes have to be considered when switching to real vehicles: We largely utilized the limited environment. Side lines and center lines are distinguished by their line width and the patch estimation relies on knowing these two types and the fixed patch width more than probably necessary. In real world the variety of environment features and road types is much higher. Also the system performance assessment will struggle with much more variance. Sensors have to be extended to provide performance measures. Degradation strategies will have to include emergency plans to come to halt from high speed.

We want to thank Audi for organizing the Audi Autonomous Driving Cup, providing a safe miniature environment to develop and test ideas that lead to robust autonomous driving.

Acknowledgements

The research leading to these results has received funding from the European Union under the H2020 EU.2.1.1.7. ECSEL Programme, as part of the RobustSENSE project, contract number 661933. Responsibility for the information and views set out in this publication lies entirely with the authors.

The authors would like to thank all partners within RobustSENSE for their cooperation and valuable contribution.

References

[1] D. Töpfer, "On Compositional Hierarchical Models for holistic Lane and Road Perception in Intelligent Vehicles," Ph.D. dissertation, 2014.

[2] K. Bengler, K. Dietmayer, B. Färber, M. Maurer, C. Stiller, and H. Winner, "Three decades of driver assistance systems: Review and future perspectives," *IEEE Intelligent Transportation Systems Magazine*, pp. 6–22, 2014.

[3] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, and C. Geyer, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[4] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knoppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb, "Making bertha drive - an autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[5] S. Klemm, M. Essinger, J. Oberländer, M. R. Zofka, F. Kuhnt, M. Weber, R. Kohlhaas, A. Kohs, A. Roennau, T. Schamm, and J. M. Zöllner, "Autonomous Multi-Story Navigation for Valet Parking," in *19th International Conference on Intelligent Transportation Systems*, 2016.

[6] Ö. S. Tas, F. Kuhnt, J. M. Zöllner, and C. Stiller, "Functional System Architectures towards Fully Automated Driving," in *IEEE Intelligent Vehicles Symposium (IV)*, 2016.

[7] R. Risack, P. Klausmann, W. Kruger, and W. Enkelmann, "Robust lane recognition embedded in a real time driver assistance system," *IEEE International Conference on Intelligent Vehicles*, pp. 35–40, 1998.

[8] L. Zhu, Y. Chen, A. Torralba, W. Freeman, and A. Yuille, "Part and appearance sharing: Recursive compositional models for multi-view multi-object detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1919–1926, 2010.

[9] D. Töpfer, J. Spehr, J. Effertz, and C. Stiller, "Efficient scene understanding for intelligent vehicles using a part-based road representation," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, no. Itsc, pp. 65–70, 2013.

[10] F. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Transactions on Information Theory*, vol. 47, pp. 498–519, 2001.

[11] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. 511–518.

[12] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 1. IEEE, 2002, pp. 900–903.

[13] R. Munoz-Salinas and S. Garrido-Jurado, "Aruco library," *URL: http://sourceforge.net/projects/aruco*.