© [2007] IEEE. Reprinted, with permission, from Sehestedt, Stephan., Kodagoda, Sarath., Alempijevic, Alen & Dissanayake, Gamini. 2007, 'Efficient Lane Detection and Tracking in Urban Environments', 3rd European Conference on Mobile Robots (ECMR 2007), pp. 78-83. This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the University of Technology, Sydney's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Efficient Lane Detection and Tracking in Urban Environments

Stephan Sehestedt* Sarath Kodagoda* Alen Alempijevic* Gamini Dissanayake*
*The Arc Centre of Excellence for Autonomous Systems, The University of Technology, Sydney, Australia

***Abstract*— Lane detection in urban environments is a challenging task. That is mainly due to the non existence of unique models, poor quality of lane markings due to wear, occlusions due to the presence of traffic and complex road geometry. In this work we present a novel lane detection and tracking algorithm for urban road scenarios based on weak models, which is implemented by a particle filter. The algorithm is implemented and experiments were carried out on Sydney urban roads. The results show the robustness of the algorithm to the problems inherent in urban road environments.**

***Index Terms*— Lane Tracking, Particle Filters, Lane Detection, Urban Environments**

I. INTRODUCTION

Today we observe an increasing demand for traffic safety systems to minimize the risk of accidents. There are a large number of vision based systems for lateral and longitudinal vehicle control, collision avoidance and lane departure warning, which have been developed during the last decade around the world (some examples are [1], [2], [3], [4] and [5]). Recently announced DARPA Urban Grand Challenge is yet another proof of enthusiasm in autonomous urban driving.

The development of advanced driver assistance systems and ultimately autonomous driving requires the ability to analyse the road scene. One prerequisite for this is the detection of lanes and subsequent tracking of lanes. Lane detection is the problem of analyzing a single image and determine the lane markings. Lane tracking is process of using temporal knowledge in lane detection. Traacking helps in reducing the computational burden whilst improving robustness.

Large numbers of research work are reported in the literature, which effectively solved the lane detection/tracking problem especially for highway like situations. However, they have shortcomings to be used in urban road scenarios. Several numbers of different constraints are commonly used to be able to detect and track lane markings, such as lanes being straight [7] or only slightly curved [8]. Such an assumption holds for freeways but is certain to fail in urban areas. Furthermore, geometric models were applied [8], [9], which describe the shape of a lane. These models rely on the visibility of markings. However, in urban areas we often have to deal with occlusions and bad or missing markings over extended periods of time, which can be catastrophic for such assumptions.

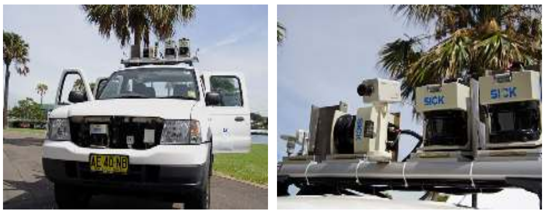

(a) (b)

Fig. 1. a) The testbed vehicle and b) the sensors mounted on the roof.

Therefore as outlined, strong assumptions about the lane geometry must be expected to be violated and weaker models are preferable. We have previously proposed a lane marking detection algorithm [6] for urban environments giving due regards to the inherent problems. We here extend it to incorporate temporal information by using a Particle filter based approach. The particles move from the bottom to the top of the 2D image plane of a inverse perspective mapped image detecting lanes. Each sample represents the possible position of a piece of lane marking and it's probability as a weight. The temporal information is incorporated by employing a second motion model and observation prediction.

The experiments were carried out to test the effectiveness of the algorithm in real situations. The CRUISE (Cas Research Ute for Intelligence, Safety and Exploration) is used for data collection in Sydney urban roads. A vertically mounted SICK laser and a CCD camera are used as perceptual sensors.

The remainder of this publication is organised as follows. In section II we briefly outline the functionality of the particle filter. In section III we present our solution to lane detection and in section IV we extend this to lane tracking. Section V contains some experimental results. Finally, we discuss future work in section VI and the conclusions in section VII.

II. PARTICLE FILTERS

Monte Carlo Methods or Particle Filters allow to approximate arbitrary multimodal probability distributions recursively. To estimate the state of a system a set of samples X at time t is used. This set $X_t = \langle x_t^i \mid i = 1...N \rangle$ and it's associated weights ω_t^i represent the belief at time t. The weights are

---

This work is supported by the ARC Centre of Excellence program, funded by the Australian Research Council (ARC) and the New South Wales State Government.

The authors gratefully acknowledge the partial funding of this work by the German Academic Exchange Service (DAAD) with a PhD student scholarship.

Stephan Sehestedt is with the ARC Centre of Excellence for Autonomous Systems, Sydney, Australia (e-mail: s.sehestedt@cas.edu.au)

Sarath Kodagoda is with the ARC Centre of Excellence for Autonomous Systems, Sydney, Australia (e-mail:s.kodagoda@ cas.edu.au)

Alen Alempijevic is with the ARC Centre of Excellence for Autonomous Systems, Sydney, Australia (e-mail:a.alempijevic@cas.edu.au)

Gamini Dissanayake is with the ARC Centre of Excellence for Autonomous Systems, Sydney, Australia (e-mail:g.dissanayake@cas.edu.au)

TABLE I
THE THREE STEPS OF THE PARTICLE FILTER ALGORITHM

1) Prediction: Draw $x_t^i \sim p(x_t^i \mid x_{t-1}^i, u_{t-1})$.
2) Update: Compute the importance factors $\omega_t^i = \eta p(y_t \mid x_t^i)$, with η being a normalization factor to ensure that the weights sum up to 1. Here, y_t is a sensor reading at time t.
3) Resample.

computed according to a sensor model, which contains the information of how likely it is that a sample x represents the true state. The computation of the posterior is then done in three steps: 1. Prediction. 2. Update 3. Resampling.

The additional resampling step ensures that the resources, i.e. the particles, are concentrated in areas of high probability. Thus, the samples are used in the areas of interest. However, due to this step the particle filter also tends to converge to one state, which means that in the basic implementation this filter would not be suitable to track multiple hypotheses over extended periods of time. Clearly in our application we need to able to track multiple lane markings, also without prior knowledge of how many.

For a more wholesome summary of Monte Carlo methods refer to [10], [11].

A. Clustered Particle Filters

In order to be able to track multiple lane markings we apply a clustered particle filter, similar to what is presented in [12]. There the sample set is divided into clusters that each represent one hypothesis. A cluster is divided into two clusters when a subset of samples has a certain distance to the others and if this subset has a high average weight. If the clusters get near to each other, they get fused to one single set.

This approach uses a modified proposal distribution, where we take account of the existence of multiple hypotheses. According to [10], the weights are calculated as

$$\omega_t^i = \eta \frac{p(y_t \mid x_t^i) p(x_t^i \mid x_{t-1}^i)}{q(x_t^i \mid x_{t-1}^i, y_t)} \tag{1}$$

η is a normalizing constant, which ensures that the weights sum up to 1. In that way, ω_t^i is only dependent on x_t^i, y_t, and y_{t-1}. To take account of the existence of clusters, the normalizing constant η is now dependent on the individual clusters and becomes η_i and consequently the weights are now computed as

$$\omega_t^i = \eta(x_t^i) p(y_t \mid x_t^i) \tag{2}$$

For C_t^j being the sum of weights of the jth cluster

$$\eta(x_t^i) = 1/C_t^j \tag{3}$$

This method is known in statistics literature as two stage sampling. The application to mobile robot localization is described in [12].

III. Lane Marking Detection

In this section our approach to lane marking detection is presented. Furthermore, we outline all the background necessary to implement the proposed method.

A. Principle

Detection of lane markings in image data is not a trivial task. That is mainly due to the non existence of unique models, poor quality of lane markings due to wear, occlusions due to the presence of traffic and complex road geometry.

This leads us to the conclusion that any method using a too strong model of the road (lane), will fail eventually. Thus, weak models are an advantageous choice. As a result of this, particle filters are a good choice for the task of lane marking tracking due to their ability to handle poor process models.

The idea of the proposed method is to use an inverse perspective mapped image (IPM image) to run a particle set from the bottom to the top and observe the presence of lane markings in each line. Furthermore, we make sure that the filter is able to track multiple lines and to store each estimated line as a trail. In this way we produce a correct data association, e.g. we associate every detected piece of lane marking to one trail, which then represents the marking of one lane.

Various issues need to be solved to apply this method. A clustering routine needs to be implemented which allows to detect a number of lane markings at any time. We need to be able to split clusters into subclusters to correctly detect markings in special situations like the appearance of an additional lane, where a line splits up into two lines. Above this, we want to be able to handle high degrees of curvature also in the presence of broken markings with large gaps.

The basic principle is illustrated in Fig. 3. 3(a) shows the original image from which the IPM image (3(b)) is computed. In Fig. 3(b) we do not have any prior knowledge about the lane markings and therefore the filter is initialized with a uniform distribution. As the particles move up in the image (according a process model), the filter eventually converges to the position of the marking. In Fig. 3(c) and 3(d) the filter tracks the lane marking correctly.

In the following subsections we will present the information on the process model, the observation model and clustering. Above this, we present the usage of an uncertainty measure for the estimate of a marking. To extend this method to obtain information about a lane, and not only the marking, one may add the measure of lane width to the state space vector, if available (e.g. DARPA Urban Grand challenge).

B. Inverse Perspective Mapping

Lane detection is generally based on the localization of a specific pattern (the lane markings) in the acquired image and can be performed with the analysis of a single still image. The method for low level image processing employed in our work is based on the Generic Obstacle and Lane Detection (GOLD) implementation. Details of this method are summarized in [4], [6] and [14]. In Fig. 2(a) and Fig. 2(b) we show the result of the inverse perspective mapping.

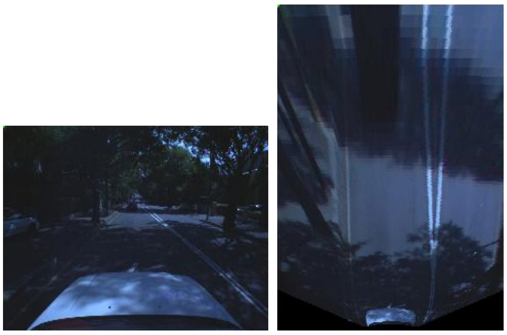
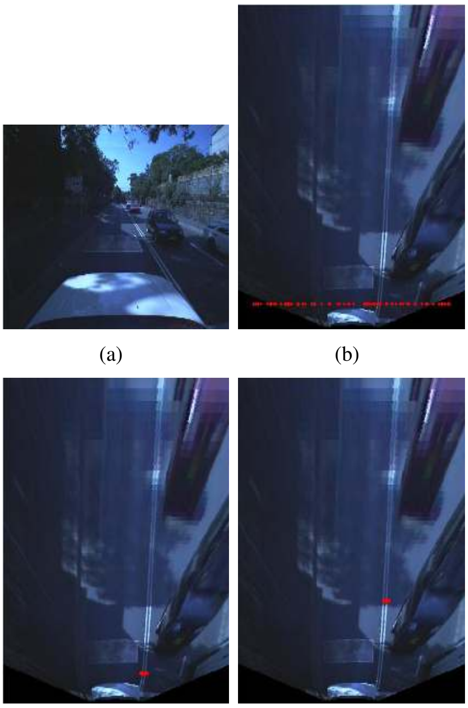

(a) (b)

Fig. 2. a) The original image and b) The inverse persepctive mapped imaged.


(a) (b)

(c) (d)

Fig. 3. a) The original image. b) Initialization phase, the particles (in red) are uniformily distributed. c) and d) The filter converges and then follows the marking

In this processing step we also use the data of a vertically mounted laser to correct for uneven road surface and the changing pitch of the vehicle. For more details see [6].

The result of this method is exploited as a sensor model in this work, because it produces high quality observations even in areas of shadow and changing light conditions. Another reason to use the 2D image plane of an IPM image is the easier implementation of a process model as described below. For further details refer to the section about observation models below.

To decrease the computational effort, the IPM image has a decreased resolution. In our current implementation we are using a 200x400 pixels resolution, which means that we are dropping some information.

C. The Process Model

In the process model we define how the particles move in the image. For every incoming observation, the filter starts from the bottom of the image. In every time step the particles are then moved to the next line of the image according to a defined Markov model, which incorporates general knowledge about road design.

In this model we define that a straight lane is more likely than a lane of any degree of curvature. Furthermore, a low degree of curvature is more likely than a high degree of curvature. This property is derived from the observation that most road segments are straight or only slightly curved and larger degrees of curvature are usually only present for relatively short times. Finally, we also take into account that there is a certain maximum degree of curvature, which will by definition not be exceeded.

A simplified Markov model for this is illustrated in Fig. 4 as an example of how it works in principle. This simple version lacks the distinguishment between different degrees of curvature and should be regarded for illustrational purposes only. Qualitatively, we define that if the particle was moving straight it is then more likely to move straight again than moving to the left or the right. Moreover, if a sample moved left or right, then it is equally likely to move into this direction again or to just move straight.

D. Observation Models

It is reasonable to apply a number of observation models to gain additional robustness for the estimate. Currently we are using an edge image, which also encodes the strength of the edges. The model assumes that the stronger an edge is the more likely it is to be part of a lane marking. This assumption is reasonable, because markings are generally features on the road which are designed to be outstanding. Thus, these features should always be very distinguishable from the surroundings. So the model can be defined as

$$\omega_{edge}(t) = p(y_t \mid x_t) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{-\frac{1}{2}\frac{y_t-\mu}{\sigma}} \tag{4}$$

where y_t denotes the observation, in this case the edge strength at a particles position x_t at time t. The edge strength is defined to be between 0.0 and 1.0. μ is the expected edge strength and is set to 1.0.

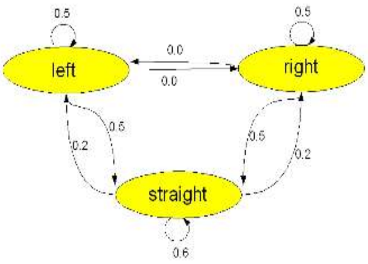


Fig. 4. A simplified Markov model for lane marking detection.

Additionally, we use prior knowledge about the colour of the lane markings, i.e. we know whether these are white or yellow and can therefore use the distance to this colour as a quality measure, where for this the original image can be used. Again, this is a reasonable assumption because the lane markings colour is different to that of the background (usually high contrast).

The definition of this model, called ω_{colour} looks similiar to 4. Only $y_t - \mu$ denotes the difference to the target colour. The weight of the sample is then calculated as $\omega_{edge} * \omega_{colour}$.

E. Uncertainty

There are certain reasons why we want to keep track of the uncertainty in our estimates. Whenever we do not have enough data for a good estimate we want to be able to not regard it in further considerations. That means, if data is bad or incomplete, which gives rise to high uncertainty, then it is necessary to have a measure for the uncertainty.

This measure is derived from the variance in the sample set. Obviously, when there is no or bad data, the samples will be distributed over a larger area. This indicates that the produced estimate is less accurate.

Adding the uncertainty measure we can now see the full functionality of the filter, which is illustrated in Fig. 5. Here the particles are shown as red dots, the estimates are shown in green and the uncertainty in yellow.

In Fig. 5(b) the filter is initialized with a uniform distribution. Fig. 5(c) shows when the filter converges the uncertainty decreases. As long as there is a meaningful observation, the uncertainty remains low. However, in a gap the uncertainty grows as in Fig. 5(d). After reconverging at the end of the gap the uncertainty is low again. Hence, this measure allows us to extract dashed and non dashed lines and also enables the filter to produce meaningful outputs with bad data.

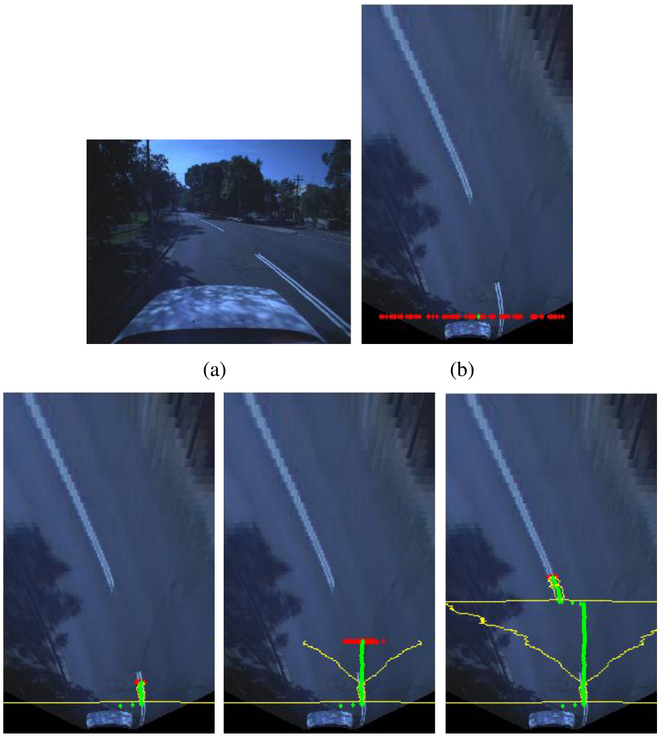
IV. LANE MARKING TRACKING

In this section we present one possible approach to extend the functionality of the lane marking detection to tracking. As tracking we denote the use of previously obtained information in subsequent time steps. Note that in this section one time step means one image. The tracking algorithm is summarized in Fig. 6, which is described in the following. The yellow box contains the processing of a single image as described in the previous section.

A. Motion Model

In order to implement tracking we need a second motion model which we apply to carry information on to the next obtained image. For this we use the ego motion of the vehicle, which allows us to predict where on the bottom of the image the particles should be initialized. Furthermore, we also use the uncertainty measure from the previous time step.

Hence, we use two motion models. One to move the particles within one image to detect lanes and one to move particles to the following image. This second motion model is shown as *Motion Model 2* in Fig. 6.



(a) (b)

(c) (d) (e)

Fig. 5. a) The original image. b) Initialization phase, the particles (in red) are uniformily distributed and uncertainty is high. c) The filter converged, so the uncertainty is low d) Due to a gap in the marking the uncertainty grows, and decreases again as soon as the filter starts converging again.

B. Observation Model

Above the particle information, we can also carry on the detected markings and project these into the newly obtained camera image according to the vehicles ego motion. This step we call *Observation Prediction*, as shown in Fig. 6.

We identified two ways to implement this step. The first possibility is to remember the edge strength and colour of the estimates, information which can be obtained from the best particle or an average including samples around the best particle during the detection phase. Alternatively, we may only consider the uncertainty of an estimate, which however would imply a loss of information.

In the first case we can use the same observation models as shown in section III-D. The only difference is that we use a higher variance due to uncertainty in the odometric data. Thus, when observing the newly found markings and the projected ones, the new estimate will average between these.

C. Initialization

In the first time step, the filter is initialized as described in lane detection. For this difficult task we need many particles to pick up all markings. For the following time steps, we already know where to look for markings and thus we place the particles only in the places of interest. Hence, the number of particles needed is lower, which results in lower computational effort.

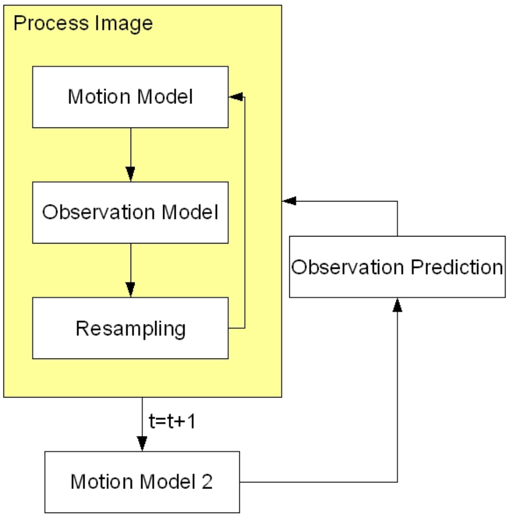


Fig. 6. The tracking algorithm.

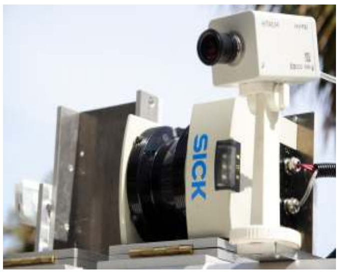


Fig. 7. The camera and vertically mounted laser.

V. EXPERIMENTAL RESULTS

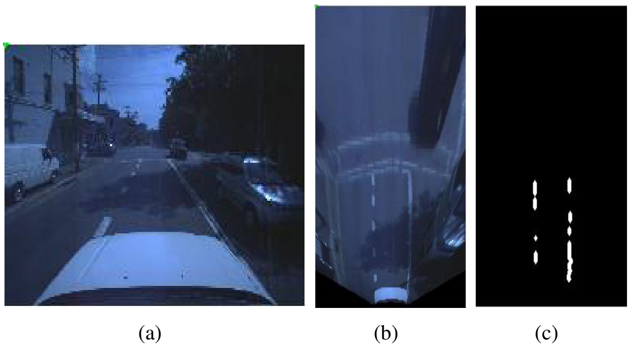
In this section we present preliminary experimental results. These were produced with early implementations in the ORCA2 Software Framework [15] on our research vehicle. The vertically mounted laser scanner and the camera can be seen in Fig. 7.

The figures in the previous chapters showed results from our Matlab implementation, whereas the figures in this section present results from our actual Orca/C++ implementation. For one time step (one image) the filter needs below 0.1s on an up to date desktop computer, which we consider real time since our camera is currently operating at 7.5Hz and a resolution of 1024 by 768 pixels. Furthermore, the current implementation of the lane tracking algorithm is in an early stage and thus not optimized.

In Fig. 8 to 14 the left image shows the original image as it is retrieved from the camera, the middle one is the IPM transformed image and the right one shows the detected and tracked lane markings. It can be seen that in normal traffic situations and in the presence of shadows our algorithm still performs well. In these cases we see increased noise levels and obstructions. It is worth noticing that in the presented data set the markings are of different quality, i.e. worn or even missing.

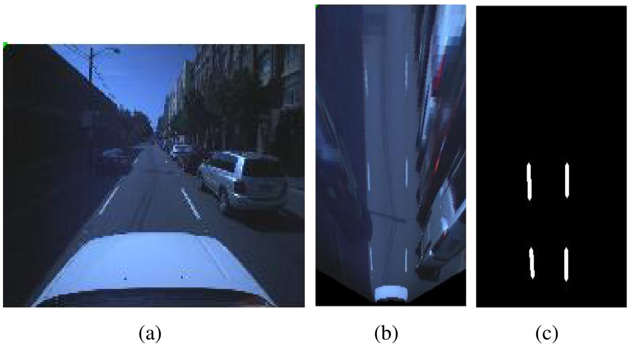
VI. FUTURE WORK

The presented approach is part of ongoing research and several things could not be discussed in this paper. Some of these are briefly discussed in this section.



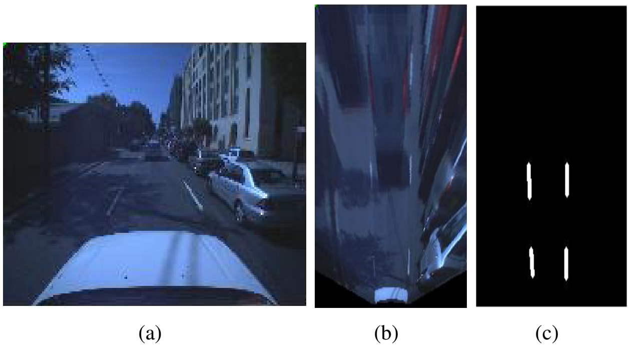
(a) (b) (c)

Fig. 8. a) The original image. b) The IPM image c) The estimated lane markings.



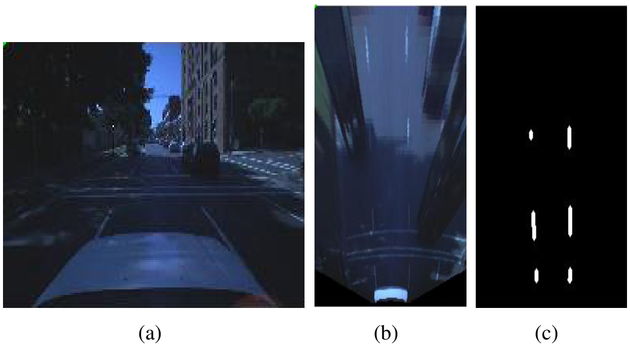
(a) (b) (c)

Fig. 9. a) The original image. b) The IPM image c) The estimated lane markings.



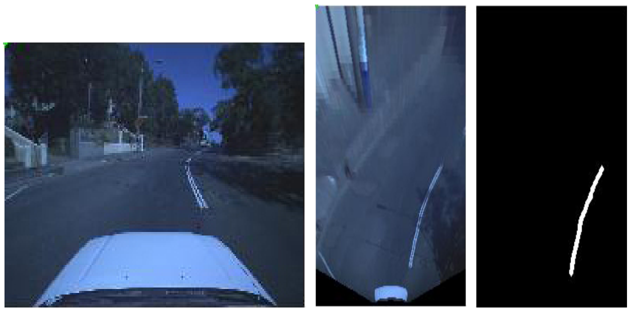
(a) (b) (c)

Fig. 10. a) The original image. b) The IPM image c) The estimated lane markings.



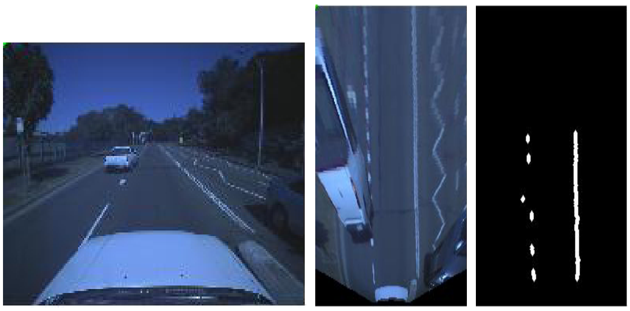
(a) (b) (c)

Fig. 11. a) The original image. b) The IPM image c) The estimated lane markings.



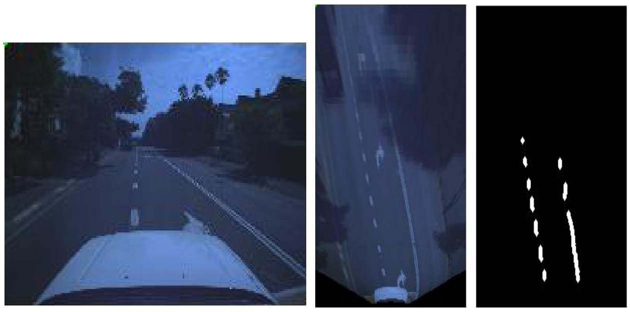
(a) (b) (c)

Fig. 12. a) The original image. b) The IPM image c) The estimated lane markings.



(a) (b) (c)

Fig. 13. a) The original image. b) The IPM image c) The estimated lane markings.



(a) (b) (c)

Fig. 14. a) The original image. b) The IPM image c) The estimated lane markings.

Currently we are working on the implementation of new sensor models. The goal is to avoid any process that would work on the whole image (as the one presented in section III-B. Instead, we only want to process parts of the image where we have samples. The main benefit is lower computational effort.

Above this we want to use the original image directly for sensor models rather than using IPM images alone, because the IPM process drops some information of the original image. In the original image colour models and template matching might be used. Template matching is especially interesting to track specific kinds of markings like double lines.

Finally, we think that target tracking alongside the lane marking tracking is a good way to improve results. In some cases it might happen that part of a white car appears as a lane marking, a situation which would be avoided by using target tracking. Also it would enable us to not do any detection in places where we track a target.

VII. CONCLUSIONS

In this work we presented a novel approach to lane tracking based on the use of weak models in a particle filter. As opposed to many previously presented algorithms we avoid strong assumptions about the geometry of a lane, which enables the filter to operate even in difficult situations where markings are obstructed or otherwise absent. Furthermore, even early implementations run in real-time, i.e. operate at least with the speed of our camera.

We discussed the most difficult task of lane marking detection as well as the task of lane marking tracking, where we benefit from the use of previous information. Using prior knowledge enables us to gain more robustness and at the same time to decrease the number of samples, as we already know where to place the samples at the beginning of a time step.

REFERENCES

[1] D. Pomerleau and T. Jochem, Rapidly adapting machine vision for autonomated vehicle steering, *IEEE Expert*, Volume 11(2), pp. 19 -27, 1996.

[2] D. Pomerleau, RALPH: Rapidly Adapting Lateral Position Handler, *in Proceedings of the Intelligent Vehicles Symposium 1995*, pp. 506-511, 1995.

[3] E. D. Dickmanns and B. D. Mysliwetz, Recursive 3D road and relative ego-state recognition, *IEEE Trans. of Pattern Analysis and Machine Intelligence*, Volume 14(2), pp. 199-213, 1992.

[4] M. Bertozzi and A. Broggi, GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection, IEEE Trans. on Image Processing, Volume 7(1), pp. 62-81, 1998.

[5] U.Franke, I.Kutzbach: "Fast Stereo based Object Detection for Stop and Go Traffic", *Intelligent Vehicles'96*, Tokyo, pp. 339-344, 1996.

[6] S. Sehestedt, S. Kodagoda, A. Alempijevic, G. Dissanayake, Robust Lane Detection in Urban Environments, *in Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[7] R. Wang, Y. Xu, Libin, Y. Zhao, A Vision-Based Road Edge Detection Algorithm, *in Proceedings of IEEE Intelligent Vehicle Symposium 2002*, Vol. 1, pp. 141-147, 2002.

[8] K. Kluge, Extracting Road Curvature and Orientation From Image Edge Points Without Perceptual Grouping Into Features, *in Proceedings of the Intelligent Vehicles Symposium*, pp 109-114, 1994.

[9] C. Kreucher, S. Lakshmanan, A frequency domain approach to lane detection in roadway images, *in Proceedings of the International Conference on Image Processing 1999*, Volume 2, pp. 31-35, 1999.

[10] S. Arulampalam, S. Makell, N. Gordon, T. Clapp, A Tutorial on Particle Filters for Online Non-linear/Non-Gaussian Bayesian Tracking, *in IEEE Transactions on Signal Processing*, Volume 50, pp. 174-188, 2002.

[11] A. Doucet, S.J. Godsill, C. Andrieu, On sequential simulation-based methods for Bayesian filtering, *Statistics and Computing*, Volume 10, pp. 197-208, 2000.

[12] S. Sehestedt, F. Schneider, A. Kraeussling, Monte Carlo Localization in Highly Symmetric Environments, *In Proceedings of The 3rd International Conference on Informatics in Control, Automation and Robotics*, pp. 249-254, 2006.

[13] R. Labayrade, J. Douret, J. Laneurit, R. Chapius, A Reliable and Robust Lane Detection System Based on the Parallel Use of Three Algorithms for Driving Safety Assistance, *in IEICE Transactions on Information and Systems 2006*, E89-D(7):2092-2100, 2006.

[14] Muad, A.M. Hussain, A. Samad, S.A. Mustaffa, M.M. Majlis, B.Y., Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system, *TENCON 2004*, Volume 1, pp. 207-210, 2004.

[15] The Orca2 robotics software framework:
http://orca-robotics.sourceforge.net/