Robust Calculation of Ego-Vehicle Corridors for Vision-Based DAS

Ruyi Jiang[1], Reinhard Klette[2], Shigang Wang[1], and Tobi Vaudrey[2]

[1] The Department of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China
[2] The Department of Computer Science, The University of Auckland Auckland, New Zealand

Abstract. An important component of driver assistance systems (DAS) is lane detection, and has been studied since the 1990*s*. However, improving and generalizing lane detection solutions remains to be a challenging task until recently. A (physical) lane is defined by road boundaries or various kinds of lane marks, and this is only partially applicable for modeling the space an ego-vehicle is able to driving in. This paper proposes a concept of a (virtual) corridor for modeling this space. A corridor depends on information available about the motion of the ego-vehicle, as well as about the (physical) lane. This paper suggests robust corridor detection using hypothesis testing based on maximum a posterior (MAP) estimation. Then, boundary selection and road patch extension are applied as post-processing. Furthermore, a simple but efficient corridor tracking method is also discussed. This paper also informs the readers about experiments using images of some challenging road situations illustrating the usefulness of the proposed corridor detection and tracking scheme.

Keywords: driver assistance, lane detection, corridor detection, maximum a posterior estimation, hypotheses testing

1 Introduction

Lane detection plays a significant role in driver assistance systems (DAS), as it may help to estimate the geometry of the road ahead, as well as the lateral position of the ego-vehicle on the road. Lane detection is used in intelligent cruise control, for lane departure warning, road modeling, and so on. Lane detection has been widely studied for driving on a freeway [4, 12] or an urban roads [16], for single [4, 17] or multiple [1, 13] lanes, with [2] or without [18] marks, based on region (texture [21] or color [5]) or edge [14] features. Various models have been applied to describe the borders of a lane, such as *piecewise linear segments* [14], *clothoids* [4, 12], *parabola* [7], *hyperbola* [19, 11], *splines* [17, 18], or *snakes* [18, 20]. Several lane detectors have been implemented and named in literature, such as GOLD [2], SCARF [5], RALPH [15], MANIAC [6], or LANA [10, 11].

Typically, lane detection or lane tracking is used for localizing lane boundaries in given road images, and based on physical road features. This is sometimes insufficient to model the road in front of the ego-vehicle for driving assistance, such as in the case of a lane changing. An understanding of the road area that is driven through is of great importance for driver assistance. This paper introduces a new concept named "corridor" to model the space the ego-vehicle is expected to drive through. A *corridor* is defined as a road patch in front of the ego-vehicle that will be driven through shortly, with a width which is a bit wider than the known width of the ego-vehicle (see Figure 1). Compared with a lane, a corridor is only partially defined by physical road boundaries or lane marks, and also by the state of the ego-vehicle, such as driving direction and lateral position with respect to road boundaries.

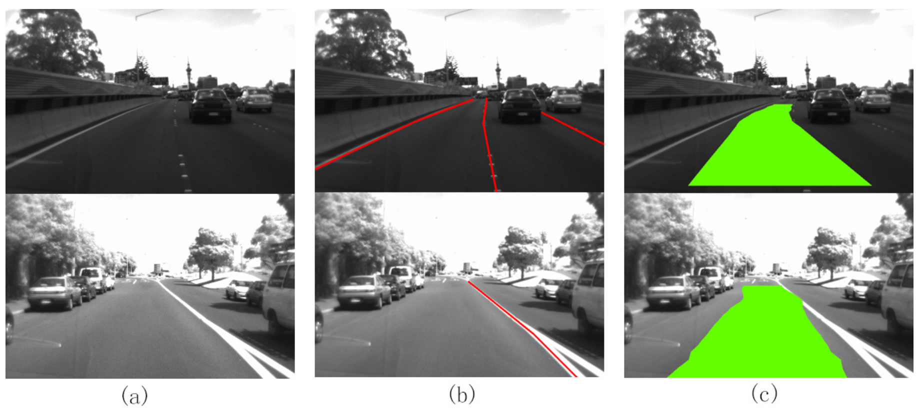

Fig. 1. Comparison of lanes and corridors. (a) Original images (top and bottom). (b) Lanes (red lines). (c) Corridors as defined in this paper (green section).

This newly proposed concept allows one to specify an innovative corridor detection method which can deal with road situations: such as variable road width, non-parallel lane boundaries, non-existing or invisible lane marks, lane changings, and some difficulties caused by illumination. The detection problem of a corridor is identical to the detection of corridor boundaries. Instead of modeling both left and right boundaries (so far, in lane detection they are commonly assumed to be parallel), that boundary, either left or right, which is detected best (with respect to some optimality characteristics) is chosen in the new method as a guide. This paper also suggests a post-processing method for road patch extension, which identifies a corridor of "sufficient" space in front of the vehicle. Finally, a simple and efficient method is used to track a corridor through a sequence.

This paper is structured as follows: Section 2 introduces our new concept of a corridor, and compares differences with the concept of a lane. Section 3 explains a robust corridor detection method using hypotheses testing based on MAP estimation.

Section 4 explains the tracking method. Experimental results are presented in Section 5. Finally, our conclusions are stated in Section 6.

2 Definition of a corridor

A (physical) lane is regarded as an area on the road surface that is completely confined by road boundaries or lane marks. For example, [3] defines a lane by criteria or hypotheses all depending on those features of the real road. This common approach for identifying a lane critically depends on the detection of physical features, and it is, however, insufficient in situations where feature detection or interpretation is difficult (or simply impossible). We aim at understanding the road area also in situations where physical features are insufficient to identify a lane.

Lane detection is not always identifying the correct road area if this process does not yet incorporate "explanations" for detected lane boundaries. For example, when the ego-vehicle is changing lanes [see Fig. 1, top of (b)], there are two lanes in front of the vehicle, and the ego-vehicle is driving partially on each of them. Another example is when there is none, or only one lane boundary, and it is difficult to tell where the lane is in such a case, just based on visible road features [see Fig. 1, bottom of (b)].

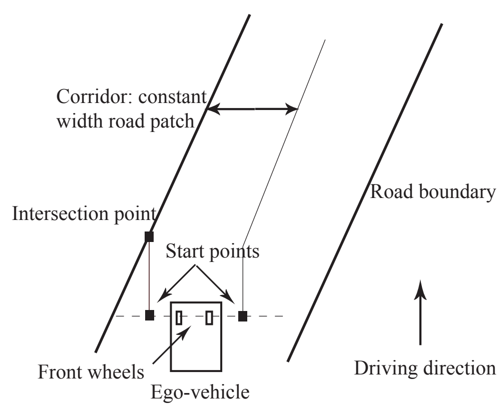

Fig. 2. Illustration for the corridor definition. The corridor should be of constant width, as far as lane boundaries and expected driving directions allow to identify such a road patch in a birds-eye view of the road.

A corridor (see Fig. 2) is a road patch in front of the ego-vehicle that will be driven through shortly. Instead of totally determining by physical lane boundaries or lane marks, the driving direction and the lateral position of the ego-vehicle are also taken into account. In order to identify sufficient driving space for the ego-vehicle, the width of a corridor is chosen to be a bit larger than that of the ego-vehicle. The corridor starts at the current position of the ego-vehicle (defined by lateral position on the road, driving direction, and width of ego-vehicle). When this initial road patch of constant width hits a road boundary or lane marks at intersection point (see Fig. 2, intersection point), then it will smoothly bend accordingly, to follow those physical features defined by the minimum deviation from the original direction. In this way, the corridor is partly decided by physical road features, and partially by the state of the ego-vehicle. Two examples of detected corridors are shown on the right in Fig. 1.

For the detection of a corridor we may combine methods as already available for detecting a lane and for analyzing the ego-vehicle motion. For initialization, we need two start points (see Fig. 2 for the definition of both start points, left and right of the ego-vehicle) and a search direction (see Fig. 2 for the driving direction which initializes the search direction).

Section 3 and Section 4 specify a possible method for corridor detection and tracking.

3 Robust corridor detection

For the detection of a corridor we have to identify a road patch in front of the ego-vehicle by its geometric boundaries. Figure 3 illustrates the overall flow of the proposed algorithm with an example.

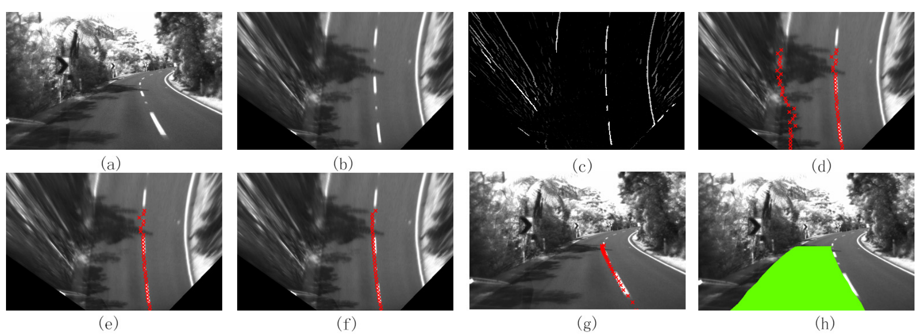

Fig. 3. Illustration of the proposed corridor detection algorithm. (a) Input image. (b) Birds-eye-view image. (c) Edge map of the birds-eye-view image. (d) Detected left and right boundaries (red lines). (e) Boundary selection. (f) Smoothing of corridor boundary using a sliding mean. (g) Projection of the boundary into the input image. (h) Identified road patch (i.e., the corridor in green).

The algorithm starts with mapping the input image into a birds-eye-view perspective view. A low-level image processing method, as introduced in [2], is then adopted to detect edges in the birds-eye-view image. Next, MAP-based hypotheses testing is conducted to detect points on left and right corridor boundaries separately, applying constraints based on information about the edges in the birds-eye-view image as well as about the car's state. After that, a comparison between these two boundaries will select the one with the better characteristics. Points on the selected corridor boundary are then smoothed by a sliding mean, and then back-projected from the birds-eye-view image into the input image, as the input image is more suitable for presentation to the driver. Finally, using a road patch extension based on the identified boundary, a corridor is produced which is in front of the ego-vehicle, with controllable patch width defined by the width of the ego-vehicle.

3.1 Birds-eye-view mapping

As in [8, 9], a four-points correspondence is used for the mapping from the input image into the birds-eye-view image. We use the locally planar ground plane assumption for this mapping. An important reason for using a birds-eye-view mapping is that the driving direction is (basically) vertical in the birds-eye-view image.

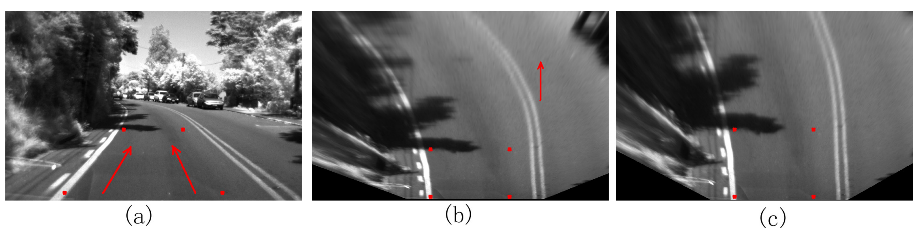

Fig. 4. Birds-eye-view mapping. (a) Input image. (b) and (c) are birds-eye-view images based on different distance definitions. Four-points correspondence (points shown in red) is established in a calibration stage; the driving direction (see the arrows) is always vertical in the generated birds-eye-view image.

The mapping is achieved by selecting four points when calibrating the ego-vehicle's camera(s), and by using the locally planar ground plane assumption. The four points from the input image are in driving direction such that they would form corners of a rectangle in the birds-eye-view image (see Fig. 4) and make sure that the driving direction is vertical. In this way, the birds-eye-view image provides a clear indication of the driving direction of the ego-vehicle. Another benefit of the birds-eye-view image is that a used distance scale can be adjusted by selecting different sets of four correspondence points (i.e., by scaling the "length" of the rectangle). This proved to be useful for detecting discontinuous lane markers as well as for further forward looking situations. Also, lane marks in the birds-eye-view image have a constant width, which may be used for for edge detection in low-level image processing [2].

3.2 Corridor detection using MAP-based hypotheses testing

The procedure for detecting a corridor is composed of three stages: initialization, prediction, and hypotheses testing. After initialization (at the selected start points), we will not continue with searching scanline by scanline in the original image (as in [22]) or with an inverse-perspective-mapped image (as in [16]); we search for potential corridor boundary points in the initialized driving direction, using a fixed distance interval in the birds-eye-view image. Doing so proved to be convenient, also due to the fact that the lateral position and the distance of points from the ego-vehicle in the birds-eye-view image are already known due to calibration.

Individual steps of the procedure predict three points, using the previously detected points. Then, a *search region* $\mathbb{S}$ is used with fixed width, centered at a predicted point; hypotheses testing of pixels in the search region $\mathbb{S}$ uses MAP estimation. In this way, each predicted point leads to a detected point at the corridor boundary, with a maximum *a posterior* probability in driving direction. The distance threshold in front of the ego-vehicle (for corridor definition) can easily be controlled, and is regarded as the *forward looking distance limit*.

Initialization. The selection of the first point on a boundary is a difficult task when initializing a lane detection process. As in [16], a particle filter is applied for the search of this first point. Note that one of the main difference between corridor and lane detection is that a start point *is* predefined in corridor detection.

In the initialization stage, lateral positions of potential boundary points are assigned to the defined two start points (see Fig. 2), which are determined based on calibration results. The distance between these two start points is assumed to be larger than that between both front wheels. This initializes the search

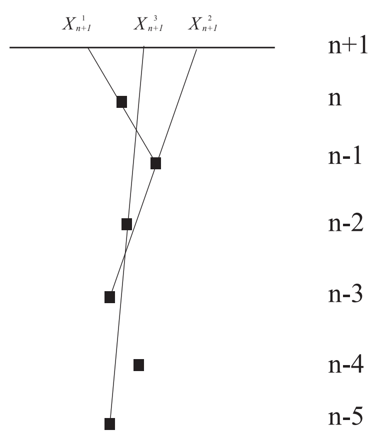

Fig. 5. At the prediction stage, three points X^1, X^2, and X^3 are generated using the previously detected black points.

for the corridor, but not for the lane. Furthermore, based on calibration, a few more initial points (in driving direction from the start points) are predefined (at constant distance increments $\triangle$) to ensure that the following prediction may work.

Prediction of corridor boundary points. In our definition of a corridor we assumed a smooth boundary. We will ensure this by using the sliding mean (see below). Based on the smooth boundary, the following procedure can be used to predict potential boundary points by using previously detected ones. For robustness, three points are predicted in each step using different previously detected ones. $X_n(u_n, v_n)$ denotes a detected lane boundary point at the nth interval in driving direction, where values v_n increase with assumed step size $\triangle$, and only u_n needs to be determined, for every X_n. In order to obtain the three boundary points X_{n+1}, we use three predictions $X^i_{n+1} = (u^i_{n+1}, v_{n+1})$, for $i = 1, 2, 3$, as follows (see Fig. 5):

$$\begin{aligned} u^1_{n+1} &= 2u_n - u_{n-1} \\ u^2_{n+1} &= 2u_{n-1} - u_{n-3} \\ u^3_{n+1} &= 2u_{n-2} - u_{n-5} \end{aligned}$$

Global distance scaling (based on the used rectangle) in the birds-eye-view can be achieved by modifying the parameter *triangle*. The prediction method has the same effect, and can be used to detect discontinuous road features. Experiments showed that this prediction method may also generate irrelevant outliers, and thus we constrain predicted points to some range:

$$u^i_{n+1} = \begin{cases} 2u_{n-i+1} - u_{n-2i+1} & \text{if } (u_{n-i+1} - u_{n-2i+1}) < T \\ u_{n-i+1} + T & \text{else} \end{cases}$$

Each of these three predictions will have its own search region, and then undergo hypotheses testing independently to obtain the corresponding MAP points. These three points are then compared to produce just one estimated point at distance v_{n+1}, namely the point with the largest MAP.

Hypotheses testing based on MAP. A 1D search region $\mathbb{S}$ (with fixed width in the row of the edge map of the birds-eye-view image) is attached to every predicted point X^i_{n+1}. Let $x(u, v)$ denote pixels in the search region of the edge map of the birds-eye-view image. A likelihood function $p(z|x)$, with z for observed features, denotes the probability of observing a lane boundary edge at pixel location $x = (u, v)$. Then, the MAP estimation can be written as follows:

$$x^* = arg \max_{x \in \mathbb{S}} \; p(x|z)$$

Using Bayes' theorem, we obtain that

$$x^* = arg \max_{x \in \mathbb{S}} \; p(z|x)p(x)$$

Here, $p(x)$ is a prior probability density function. Assuming smoothness between neighboring boundary points, $p(x)$ is defined as (a_1, b_1 are constants):

$$p(x) = \frac{1}{a_1} \exp(-b_1(u - u^i_{n+1})^2)$$

The determination of the likelihood function $p(z|x)$ uses edge information. [22] states that "edge-based methods often fail to locate the lane boundaries in images with strong distracting edges". However, edges are still useful as a source of information to discriminate a lane from its surroundings; it is reasonable to assume that the stronger an edge, the more likely it is that it is part of a lane boundary; see [16]. Let S be the edge strength at $x(u, v)$,and let $x_{max}(u_{max}, v)$ be the pixel with the greatest edge strength S_{max} in $\mathbb{S}$. Then, we use the following (a_2, b_2, b_3 are constants):

$$p_(z|x) = \frac{1}{a_2} \exp(-\frac{1}{b_2} \cdot (S - S_{max})^2 - \frac{1}{b_3} \cdot (u - u_{max})^2)$$

Experiments show that detected lane boundaries are (typically) distracted by other strong edges such as at bright areas or shadows on the road, other objects on the road or on the roadside – if only edge information is used. The smoothness assumption (for corridor boundaries) and the proposed prediction method can relieve the distractions caused by edges on the road.

3.3 Post processing for robust detection

Based on the detected corridor boundaries in Section 3.2, post processing will produce a reasonable and smooth corridor. Optimal boundary selection will select a better boundary, points of which will then be smoothed by a sliding mean. After back-projecting to the input image, a road patch defined by these boundaries will be extended as being the corridor.

Optimal corridor boundary selection. Till now, both a left and a right corridor boundary have been detected. Considering different deterioration features in the left and right part of the road as well as if those boundaries are parallel or not, it can be expected that both boundaries may not define a patch of constant width on the road.

In this situation, this paper suggests that a better corridor boundary is selected for robustness and practicality reasons, according to the following criteria. For a straight road, the criterion may be something like "less lateral variation" to pay more attention to the straight boundary. Actually, as we have made no assumption of a straight road, two used criteria are actually useful for selecting the corridor boundary: first, the preference of "stronger edge strength" (As edge information is used for corridor detection, a boundary with points showing stronger edge strength will be selected.); second, the "minimization of variation in boundary direction" (Due to requested smoothness of corridor boundaries, a boundary with less change in direction will be selected with higher probability.)

Sliding mean. As no specific road geometry is assumed, no curve model is used to fit the detected boundary points. A simple sliding mean is applied for ensuring smoothness of corridor boundaries. We obtain smoothed points X'_n from the point sequence $\{X_n\}$ by applying the following:

$$X'_n = X'_{n-1} + (X_n - X_{n-s})/s$$

The constant s determines the step size.

Road patch extension. Following the corridor definition as provided in Section 2, once a "dominant" corridor boundary (left or right) is selected and mapped back to the input image, a road patch will be calculated, having one of its sides identical to the selected boundary, and by calculating the other side with the pre-defined width. The width is adaptable as it should provide enough space for the car to drive through. Compared with a constant road width assumption, this method will provide a similar result if driving on a lane of constant width, and a fixed width identical to the lane's width. However, at other occasions, when the constant road width assumption is not applicable, the provided method still detects a reasonable corridor using the road patch extension.

4 Corridor tracking

This section presents a corridor tracking approach which is not utilizing a time-consuming particle filter, as in [8, 16, 18, 19], nor a model-based Kalman filter, as in [1]; both techniques are commonly used for lane tracking. After a corridor is detected (as discussed in Section 3), a practical way to represent a corridor is by using points on the central line of the corridor (in the birds-eye-view image) and its width. Such a point sequence $\{C_n, n = 0, 1, \ldots, N\}$ (N is defined by the lookahead distance) of the center line can be calculated from some of the corridor's boundary points (in the birds-eye-view image), also using the constant corridor width. Tracking of a corridor is composed of two modules: continuous corridor tracking, and discontinuous corridor tracking.

4.1 Continuous corridor tracking

Note that a corridor estimates a road patch that will be driven through shortly by the ego-vehicle. This means that a corridor detected at time t will have been partly driven through at time $t+1$. The ratio of the already driven part depends on the cycle time between two frames as well as the ego-vehicle's speed. If the ego-vehicle does not change much the driving direction (i.e., the yaw angle), and is also not in the process of a lane changing, then there will be a continuous corridor update between subsequent frames. Note that the point sequence $\{C_n\}$ is in the birds-eye-view image. Corridor tracking is then easy, defined by tracking of the ego-vehicle's motion state for an adjustment of sequence $\{C_n\}$, and composed of two steps: adjustment caused by the driven distance and the variation in driving direction; possibly also by detecting new points. For points $\{C_n\}$ in frame $t+1$ and $\{C'_n\}$ in frame t, because of the driven distance, it follows that

$$C_i = C'_{i+m}, \qquad i = 0, 1, \dots, N-m$$

Here, m is determined by the driven distance, and usually it is small. Furthermore, points $\{C_n, n = 0, 1, \dots, N-m\}$ will all have an added shift in lateral position (according to n) caused by the variation of driving direction between these two frames. For the detection of center line points $\{C_n, n = N-m+1, \dots, N\}$, the same method is used as introduced in Section 3.2. The only difference is that left and right boundary points are calculated starting at points on the center line. By combining points from the last frame and points detected in the current frame, a corridor can be efficiently updated using the given sequence of frames.

4.2 Discontinuous corridor tracking

However, a corridor will not always change continuously between subsequent frames, which is obvious from its definition. If the change in driving direction is above some threshold, then the corridor may differ greatly compared to the continuous corridor of the last frame. Another situation is when an ego-vehicle is in the process of lane changing. The example in Fig. 6 gives an illustration of this discontinuous case. In order to deal with such situations, a simple re-initialization by corridor detection is applied. The change of driving direction can be calculated from ego-vehicle's motion model. Any occurrence of a lane changing, or of any other boundary variation, can be identified by tracking the intersection points (see Fig. 2). A pass through this intersection point means a change of a corridor. Then, corridor detection will be applied as re-initialization, to restart a new process of tracking. Actually, as corridor detection is really time-efficient (see time measurements in Section 5), re-initialization will not harm a continuous tracking of the corridor. Furthermore, driving direction and intersection point allow an easy way to re-initiate using backtracking. – Figure 7 summarizes the proposed scheme of corridor tracking.

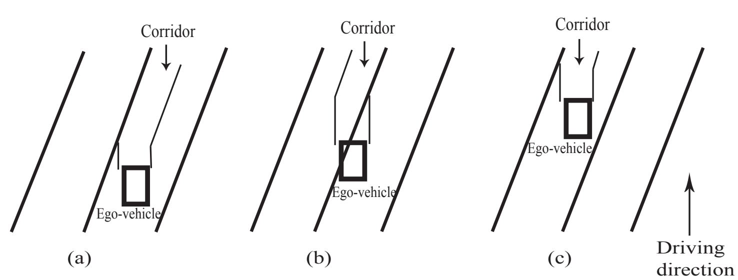

Fig. 6. Illustration of corridor discontinuousness between frames. (a) Before lane changing. (b) During lane changing. (c) After lane changing.

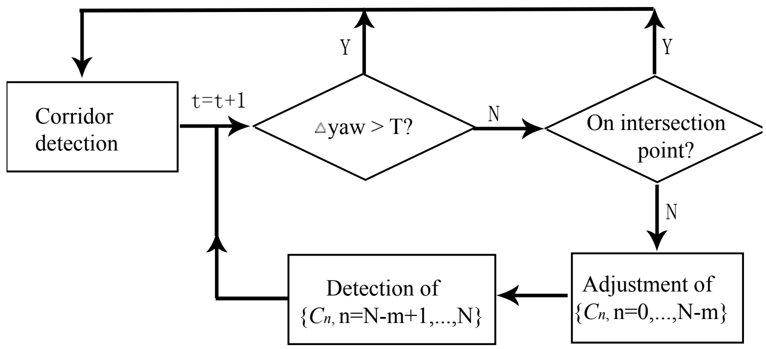

Fig. 7. The proposed scheme for corridor tracking. "Yaw" means yaw angle, and "On intersection point" means that the ego-vehicle drives on the intersection point of the current (i.e., previous frame) corridor.

5 Experimental results

Experiments were conducted on images and sequences recorded with our test vehicle "HAKA1" (see Fig. 8). Though a pair of stereo cameras was installed in HAKA1 for those test drives, only the right input image sequence was used for corridor detection.

Fig. 8. (a) The test vehicle 'High Awareness Kinematic Automobile 1' (HAKA1). (b) A stereo camera pair on the bar behind the windscreen.

Corridor detection results are illustrated in Fig. 9 for a few selected (e.g., challenging) road situations. For better understanding of the shown situations, intermediate results of raw boundary detections are also shown.

Images in the first row show a simple situation with relatively perfect lane marks, but with shadows on the road. The detected corridor is similar as the lane.

Images in the third row illustrate a lane change. This provides a good demonstration of a corridor.

The other images illustrate some particular difficulties, such as when there is only one road boundary visible, a "noisy" road surface, discontinuous lane

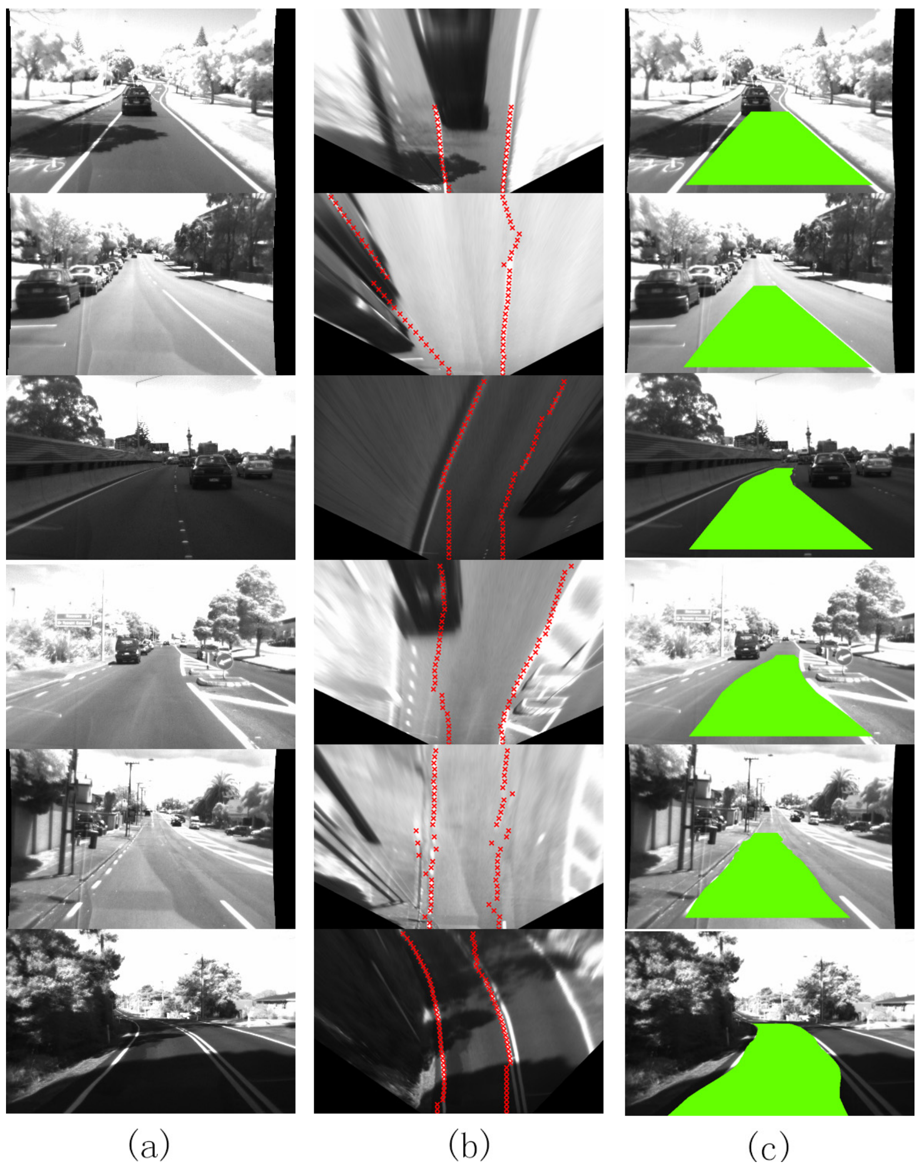

Fig. 9. Experimental results. (a) Original image. (b) Both raw corridor boundaries (in red). (c) Final detected corridor (in green).

marks, or a road without central lane marks. The raw detected left boundary in the second row is far from perfect as being affected by lane-mark-like reflections on the ego-vehicle's windscreen. However, the corridor was constructed based on the right boundary.

Corridor detection only takes less than 0.1 seconds for a 752×480 image, using an off-the-shelf computer without runtime optimization. As no time-consuming computation is needed for detection, it is very "reasonable" in its computational efficiency.

Results for corridor tracking are presented in Fig. 10, for a few frames of some sequences. For discontinuous tracking, corridor detection is commonly used for re-initialization. Continuous corridor tracking is much faster than the discontinuous steps.

Obtained experimental results (see Fig.5) show that corridor detection and tracking provides a good indication of the road patch that the ego-vehicle is expected to drive through shortly, even under difficult road situations.

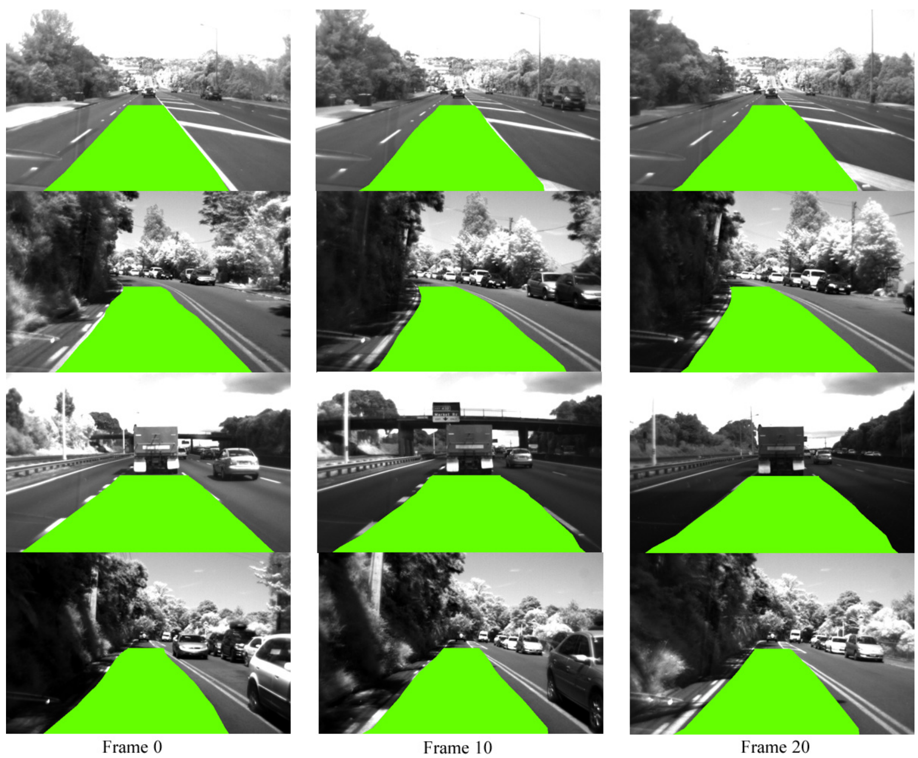

Fig. 10. Experimental results.

6 Conclusions

A new concept of a corridor was introduced in this paper, and a possible corridor detection and tracking method is proposed. Compared with a lane, a corridor also pays attention to the driver's intention, which is indicated by the car's lateral position on the road and the driving direction. Still, a corridor is partly constrained by the physical lane marks or boundaries, and it will follow those if suitable.

A main difference between lane and corridor detection methods is that corridor detection starts at fixed points (two start points) and searches in driving direction. Road patch extensions combined with better boundary selections may be applied when only one corridor boundary can be detected, and there is no assumption about a constant road width.

Acknowledgments: This work is supported by the National Natural Science Foundation of China under Grant 50875169.

References

1. R. Aufrére, R. Chapuis, and F. Chausse: A model-driven approach for real-time road recognition, *Machine Vision and Application*, **13**:95–107, 2001.
2. M. Bertozzi, A. Broggi: GOLD:A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Trans. Image Processing*, **7**:62–81, 1998.
3. M. Bellino, Y.L. de Meneses, P. Ryser, and J. Jacot: Lane detection algorithm for an onboard camera. In Proc. *SPIE Photonics in the Automobile*, volume 5663, pages 102–111, 2005
4. E. D. Dickmanns and B. D. Mysliwetz: Recursive 3-D road and relative ego-state recognition. *IEEE Trans. Pattern Recognition Machine Intelligence*, **14**:199–213, 1992.
5. J. D. Grisman and C. E. Thorpe: SCARF: A color vision system that tracks roads and intersections. *IEEE Trans. Robotics and Automation*, **9**:49–58, 1993.
6. T. M. Jochem, D. A. Pomerleau, and C. E. Thorpe: MANIAC: A next generation neurally based autonomous road follower. In Proc. *Int. Conf. Intelligent Autonomous Systems*, 1993.
7. C. R. Jung and C. R. Kelber: A lane departure warning system based on a linear-parabolic lane model. In Proc. *IEEE Symp. Intelligent Vehicle*, pages 891–895, 2004.
8. Z. Kim: Realtime lane tracking of curved local road. In Proc. *IEEE Intelligent Transportation Systems*, pages 1149-1155, 2006
9. Z. Kim: Robust lane detection and tracking in challenging scenarios. *IEEE Trans. Intelligent Transportation System*, **9**:16-26, 2008
10. C. Kreucher and S. Lakshmanan: LANA: A lane extraction algorithm that uses frequency domain features. *IEEE Trans. Robotics and Automation*, **15**:343–350, 1999.
11. C. Kreucher, S. Lakshmanan: A frequency domain approach to lane detection in roadway images. In Proc. *Int. Conf. Image Processing*, pages 31–35, 1999.
12. D. Khosla: Accurate estimation of forward path geometry using two-clothoid road model. In Proc. *IEEE Symp. Intelligent Vehicle*, volume 1, pages 154–159, 2002.

13. C. Lipski, B. Scholz, K. Berger, C. Linz, and T. Stich: A fast and robust approach to lane marking detection and lane tracking. *IEEE Southwest Symp. Image Analysis and Interpretation*, pages 57–60, 2008.
14. M. A. Nasirudin and M. R. Arshad: A feature-based lane detection system using hough transform method. *Int. Symp. Intelligent Transportation Systems*, pages 166–169, 2007.
15. D. Pomerleau: RALPH: Rapidly adapting lateral position handler. In Proc. *IEEE Symp. Intelligent Vehicle*, pages 506–511, 1995.
16. S. Sehestedt, S. Kodagoda, A. Alempijevic, and G. Dissanayake: Efficient lane detection and tracking in urban environments. In Proc. *European Conf. Mobile Robots*, pages 126–131, 2007.
17. Y. Wang, D. Shen, and E. K. Teoh: Lane detection using Catmull-Rom spline. In Proc. *IEEE Int. Conf. Intelligent Vehicles*, pages 51–57, 1998.
18. Y. Wang, E. K. Teoh, and D. Shen: Lane detection and tracking using B-Snake. *Image and Vision Computing*, **22** : 269–280, 2004.
19. Y. Wang, L. Bai, M. Fairhurst: Robust road modeling and tracking using condensation. *IEEE Trans. Intelligent Transportation Systems*, **9**:570–579, 2008.
20. Y. Yagi, M. Brady, Y. Kawasaki, and M. Yachida: Active contour road model for smart vehicle. In Proc. *Int. Conf. Pattern Recognition*, volume 3, pages 3819–3822, 2000.
21. J. Zhang and H. Naqel: Texture-based segmentation of road images. In Proc. *IEEE Int. Conf. Intelligent Vehicles*, pages 260–265, 1994.
22. Y. Zhou, R. Xu, X. Hu, and Q. Ye: A robust lane detection and tracking method based on computer vision. *Measurement Science Technology*, **17**:736–745, 2006.