A marking-based, flexible approach to intersection detection

Christian Duchow

Institut für Mess- und Regelungstechnik

University of Karlsruhe

Karlsruhe, Germany

Email: duchow@mrt.uka.de

Abstract— This paper describes a flexible, marking-based approach to intersection detection. Edge contours are extracted from an image acquired with an onboard camera. The contours are mapped to the flat earth model so as to remove the effect of the perspective mapping. Using properties like spatial context, orientation and length, the individual markings are grouped to contiguous straight lines and finally to lanes. The result is a description of an intersection's geometry without any assumptions about the number, width and orientation of lanes.

I. Introduction

Vehicle environment sensing has been a focus of research for many years (see, for example, [1]). In the past few years, attention has moved from lane detection on highways to more complex scenarios that may include intersections (see [2], [3]). The capability of sensing intersection geometries offers great possibilities for application in driver assistance systems. It would be feasible to guide the driver through complex traffic scenarios, thereby relieving the driver. The driver would be able to concentrate more on maintaining an overview of the complete situation, therefore, the hazardous potential of such situations would be reduced. To give an example of a possible intersection situation, Fig. 1 displays a typical intersection problem. The image in Fig. 1 consists of 640×480 pixels and was acquired with a camera of focal length $f = 820$ pixels. The camera has a horizontal field of view of $45°$. The image shows road lanes parallel to the viewing direction of the camera, and there is also a road going to the right. In addition, there are lanes for crossing pedestrians present in the scene: One located on the road to the right, another one crosses the lane on which the own vehicle is travelling. A complete interpretation of that image would sense all of that.

The complexity of these situations prohibits the restriction to specific generic models such as clothoid geometry, because such simplifications limit the applicability to only certain types of intersections. Instead, it is necessary to deploy a general approach which can handle different geometrical types of intersections. In [4], an algorithm was described which extracts straight edge segments from an intensity image acquired with an onboard camera. The straight edge segments are paired in order to model the left and right long edges of a rectangular lane marking. After that, the edge pairs are grouped to yield lines describing the intersection geometry. This paper adopts the approach of [4] in the sense that lane markings are extracted and aggregated without imposing a strict model on road geometry. The description and aggregation of single white lane markings to line segments is conducted in the ground plane. The method of total least squares is used in that process. Thus, the effect of the perspective mapping which could spoil the correct interpretation of the image is neutralized. Furthermore, the resultant line segments are then processed to yield lanes. This approach is capable of detecting the own lane, lanes which are parallel to the own lane as well as lanes that are oriented arbitrarily to the own lane. Thus, it is possible to detect lanes that run perpendicular to the own lane, and this capability is crucial in the context of detecting intersection geometry. Also, the detected lanes are classified into lanes for pedestrians, lanes for cars, and lanes which could not be classified.

The overall approach is schematically depicted in Fig. 2. The scheme illustrates the strategy of solving the problem of sensing intersection geometry by breaking it down into smaller tasks. Detailed descriptions of each processing step are given in the following sections.

The rest of this paper is organized as follows: The detection of the individual lane markings is described in Section II. Section III describes the aggregation of the markings to line segments. Section IV finally describes the perception of lanes based on the line segments. Section V presents some results on real test data. The paper concludes with a short summary of the results and an outlook on future work.

II. Extraction of lane markings

The rectangular form of an individual lane marking is fully described by five parameters in the ground plane (position in y, position in z, length, width, orientation). To further illustrate the description, consider Fig. 3 showing the circled lane marking from Fig. 1.

The determination of the lane marking parameters is started by detecting edge contours in the image. These edge contours may be of arbitrary shape. The detection of the edge contours is done in the following way: First, the image is convolved with a small gaussian filter of size 3×3 in order to avoid detecting false edges. Next, the image derivatives are computed using the Sobel operator. Then, edges are detected using the Canny edge detector (see [6]) with a very low value for the low threshold and a moderate value for the high threshold. The

Fig. 1. A complex intersection scene consisting of parallel and perpendicular car lanes and pedestrian crossings. The marked elements correspond to Fig. 3 and Fig. 7

result of choosing the thresholds like that is that a contour is required to have at least partially a strong edge but that the edge detector tries to follow the edge as far as possible. Thus, objects which consist of weak edges only are not found, but lane markings that may happen to have partially weak edges are still found. The result of the edge detection is then processed with a contour retrieving algorithm. The result of this processing for Fig. 3 is shown in Fig. 4.

Next, the contour points are mapped onto the ground plane. This back projection restores the more or less round blobs to a rectangular shape, as seen in Fig. 5. A close look at Fig. 5 reveals that the lane marking as perceived by this processing is not parallel to the viewing direction of the camera. In reality, the lane marking should be parallel to the viewing direction. This error in orientation is induced by the quantization error of the image acquisition. Since the whole lane marking is represented by only a few pixels in the image, the direction information is quantized very roughly accordingly. Thus, one would actually expect a rather large quantization error with respect to the direction of the lane marking. However, this is not an algorithmic flaw, but rather a nuisance which comes along with the image acquisition.

A total least squares line is fitted to the back-projected rectangle points (see, for example, [5]). Using total least squares rather than least squares is crucial since the rectangles are arbitrarily oriented and ordinary least squares is not capable of handling vertically oriented lines. The resulting line g is represented in Hessian normal form:

$$g : \begin{bmatrix} \cos \alpha \\ \sin \alpha \end{bmatrix} \cdot \begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} \cdot \begin{bmatrix} x \\ z \end{bmatrix} = D \qquad (1)$$

where α is the angle with the x-axis and D is the distance to the origin. In order to resolve ambiguities in D and α, D is constrained to positive values only as is feasible for a measure of distance, and α ranges from $-\pi$ to π. See Fig. 6. The total least squares estimation estimates the parameters a, b and D such that

$$\sum_{i=1}^{n} e_i^2 \rightarrow \min \qquad (2)$$

where e_i is the distance from the i-th point to the estimated line. To obtain meaningful results, the constraints $a^2 + b^2 = 1$ and $D > 1$ were imposed.

The line is clipped to a line segment corresponding to the rectangle. The center of this line segment is taken as the center of the lane marking. This approximation yields the position, length and orientation of the rectangle. Now, the contour points are projected on the normal to the total least squares line. The

grey value
image

↓

lane marking
segment detection

knowledge: lane → test
marking is white

additional information: → test
stereo information

knowledge about
dimensions of → test
possible lane markings

Line segment
aggregation

lane patch
detection

classified
road lanes
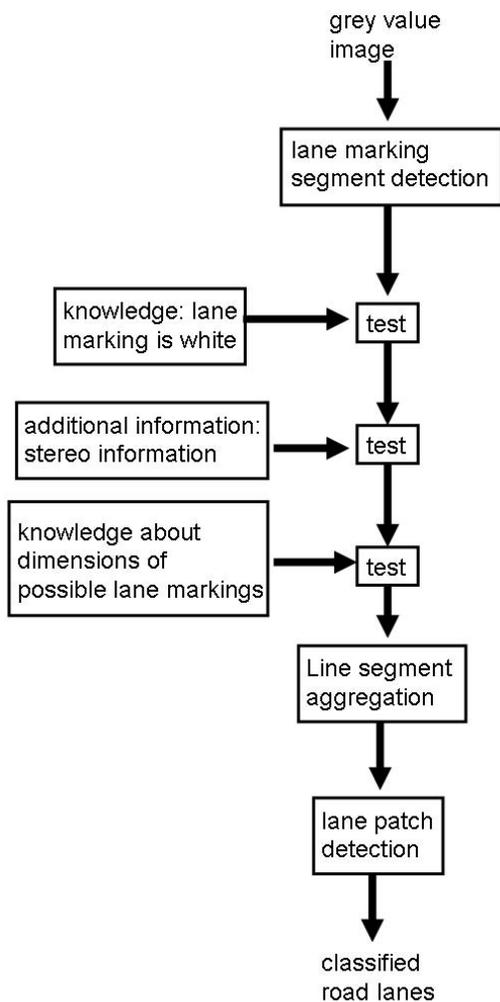
Fig. 2.   Schematic outline of the algorithm
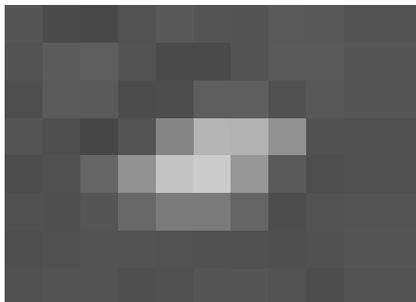
Fig. 3.   An image of a lane marking at a distance of 33m
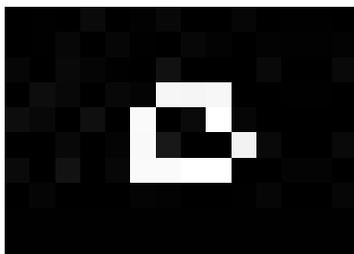
Fig. 4.   The lane marking after edge and contour detection

viewing
direction
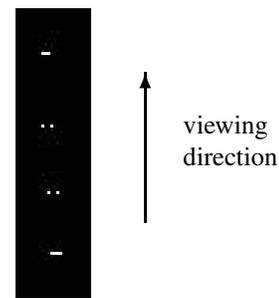
Fig. 5.   The contour points after mapping onto the ground plane
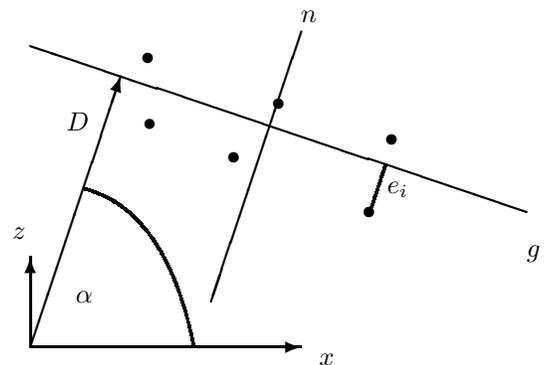
Fig. 6.   Contour points, TLS line g and line n

normal to the line g is given by line n:

$$n : \left[\begin{array}{c} -b \\ a \end{array} \right] \cdot \left[\begin{array}{c} x \\ z \end{array} \right] = D_n \qquad (3)$$

The range of the projected points on the line n is taken to be the width of the rectangle. Of course, estimating the width in this way causes a bias when the contour points are corrupted with noise, however, the width of the rectangles is not relied on heavily in the future processing.

As the contours do not necessarily have to originate from lane markings, several tests are performed in order to validate or reject a detected lane marking. As the edge detector also finds dark objects which cannot correspond to white lane markings, it is evaluated whether the object in question is brighter than its surroundings or darker. In addition, a minimal length and a maximum width is required. Objects that are much wider than any possible type of lane marking are not considered. Apart from this test, the width is not considered further. Finally, in dense traffic or narrow streets, there might be a substantial amount of edges that stem from objects outside the ground plane. Hence, this algorithm was combined with a stereo algorithm in order to detect these edges. Not surprisingly, edges that came from objects sufficiently above the ground plane were detected quite well and eliminated. This way, the results were improved in such situations. However, as this is only a feature for practical reasons and not specific to the main problem of interest, this is not described in detail. Also, the algorithm does not generally use the stereo feature. The results presented in this paper were all produced without relying on the stereo information, thereby accelerating the computation.

It was examined whether requiring the objects to have a rectangular shape would help to identify true and false rectangles. While true and false rectangles in the neighborhood of the vehicle were classified correctly with this approach, it was found that this was not the case for objects further away from the car. At some distance away from the car, this test is not a good measure of distinction between lane markings and other objects, because even lane markings at that distance may produce incomplete contours so that the shape of the contours is indeed not rectangular. The process of edge detection and contour following produces these incomplete results. Therefore, the idea of testing the rectangles for a complete rectangular shape was not pursued any further.

Of course, eliminating the effect of the perspective mapping is not a new idea. In [7], an interesting algorithm is described based on the inverse perspective mapping. However, in this work, not the image is mapped, but only the edge points. Hence, the mapping is less computationally expensive as this is a lot less points than the whole image. Secondly, the edge detection is performed in the image rather than in the inversely mapped image where an edge detection might be even more difficult because of interpolation issues.

III. Aggregation of straight line segments

Now that the lane markings have been found, they need to be put into context. Since the vehicle's own lane as well as any possibly crossing lanes need to be detected, it is not possible to start the search at the bottom of the image, as is done in research focused on the detection of highway environments (see [1]). Instead, every rectangle initiates a candidate line and checks whether there are other adjacent rectangles when following the rectangle's orientation. The rectangles found are required to have similar length. Of course, since each rectangle starts a candidate line, this results in several candidate lines describing geometrically equivalent lines. However, this is no problem as multiply found lines are simply eliminated.

A. Issues

Let us consider the image of a lane marking given in Fig. 7. This lane marking is taken from the area in Fig. 1 which is marked by a box. The lane marking is in reality a rectangle going from right to left, as can be guessed from the image. The result of the edge detection is also given. Obviously, the edge detection process estimates the length and the width of the rectangle as too large. During the inverse perspective mapping, the bias in the width is strongly increased, while the bias in length is not increased as much. Therefore, the short side of the real rectangle becomes the long side of the rectangle in the ground plane. This can be seen in Fig. 8. This causes the total least squares line to be oriented exactly in the direction perpendicular to the direction it should be! Admittedly, this is not the behavior that one wishes to observe. For short distances, the detection is as expected, but at moderate distances, this problem does exist. At far distances, these short markings which run perpendicular to the driving direction do not yield edges in the first place.

(a) Image (b) After edge and contour detection
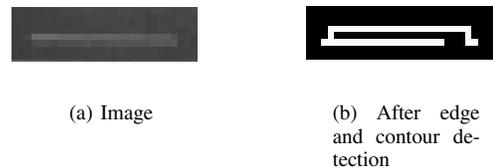
Fig. 7. Image and edge contour points of a lane marking
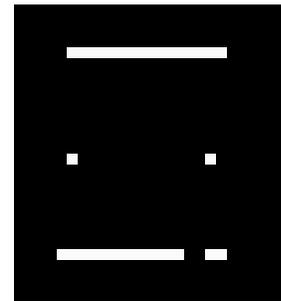
Fig. 8. The contour points after mapping to the ground plane

Now that we have stated the problem, let us investigate its consequences and possible solutions. The first consequence is that the width of the rectangle is substantially biased. The second consequence is that the length of the rectangle is not estimated well either, but the bias is not as bad since the estimated rectangle is rather a square than a rectangle, thus, the estimated length corresponds approximately to the true length. Since the width is not used for the aggregation of lines, the bias in width does not do much damage. The bias in length does not do much damage as well as this bias is not as bad. The consequence which is the most unpleasant is that the estimated direction of the lane marking is perpendicular to the direction it should be. This cannot be neglected since the orientation of a lane marking determines the position of possible neighboring lane markings. Thus, when a candidate line is initialised with a rectangle, it is tested whether the rectangle is almost a square. If that is the case, then in fact two candidate lines are initialised: one in the direction of the rectangle, the other in the direction perpendicular to the direction of the rectangle. This solution solves the problem at hand: The line going into the wrong direction does not find any other rectangles and is eliminated; the line which follows the true direction does find other rectangles and is submitted to further processing.

Markings, which do not fit the model of a rectangle are not detected using this method, of course. That is, markings of the form ⊓ or T are not detected.

IV. Combining lines to lanes

In order to flexibly detect lanes for cars, lanes for bicycles and other types of lanes, lanes must be allowed to have a rather large range of width. That is, lanes are allowed to be as narrow as 1.8m and as wide as 4.4m. These requirements are accounted for in the following.

Only line segments that consist of at least three rectangles are considered when the line segments are grouped into lanes. For two line segments to be paired into a lane, the two line segments are required to be parallel, lie side by side so that one line segment could be projected onto the other along their normal direction, and they must have the required distance as given by the width described above. (See Fig. 9.) Only the area which lies exactly between the two line segments is regarded the lane area, because nothing else is known about the area where only one line was detected. Maybe the lane ends there or is occluded by another vehicle, thus, it would be too optimistic to infer that there was a lane. Every line segment searches for its left and right neighbor which meets these requirements. Since every line segment performs this search, the lanes might be detected twice, although this is not always the case. Again, after this search has been performed, multiple lanes are detected and eliminated.

Lane markings which belong to a lane for pedestrians or bycicles are shorter than lane markings that belong to a lane which is used by cars. E.g., on German roads, the lane markings for cars in an intersection may either be 1.5m long or 3m. Lane markings which refer to lanes for pedestrians are 0.5m long (see [8]). This knowledge is exploited to classify the detected lanes into lanes for cars, lanes for bycicles and pedestrians or lanes which cannot be classified. For each lane, the average rectangle length of the left and right line segment is considered; if both line segments have an average rectangle length smaller than $l_p^{max} = 0.8$m, the lane is considered a lane for pedestrians and bycicles. If both line segments have an average rectangle length of more than $l_c^{min} = 1.3$m, the lane is considered a lane for cars. If none of the above is met, the lane type cannot be determined. These thresholds have been chosen to account for errors due to quantization and edge detection. They could be chosen even more permissible, that is, l_c^{min} could be chosen smaller, but that would increase the probability that a lane for pedestrians is classified as a lane for cars.

As mentioned above, only line segments that consist of at least three rectangles are considered in the lane aggregation process. Considering line segments which consist of less than three rectangles would cause too many false line segments. Therefore, a long solid lane marking will not be detected with this approach, as the corresponding line segment is missing two rectangles. Also, a significantly curved line cannot be detected with this approach as significant curves do not fit the model of straight lines. This approach is flexible in the sense that the number and orientation of the detected lanes is not constrained.

V. RESULTS

Fig. 10 depicts an image taken from a sequence in which an intersection is crossed. All the rectangles detected in the image are visualized as white patches. Fig. 11 shows the result of the line aggregation process. The line segments themselves are shown in black and the rectangles associated to the lines are shown in white. Finally, Fig. 12 depicts the lane segments
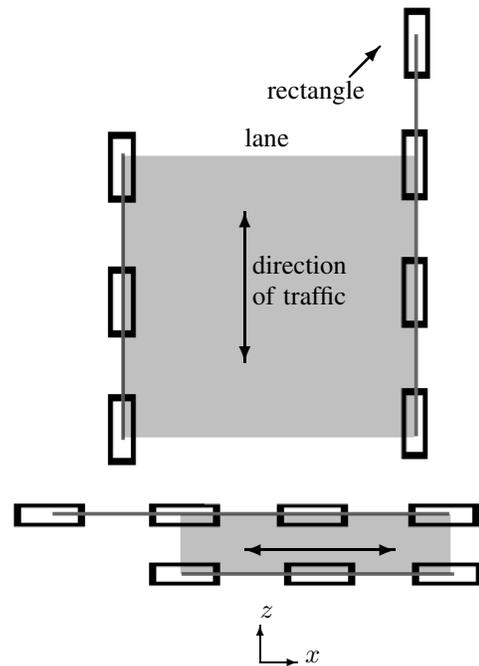

Fig. 9. Lane detection on an intersection

Fig. 10. Rectangle detection results. Detected rectangles are shown as bright patches.

derived from the line segments. The lane segments are shown as bright patches, with the assumed direction of traffic shown as black bars.

Additionally, Fig. 13 shows results on another image. The vehicle's own lane is detected. The lane next to the own lane is detected, too. There is a lane for crossing pedestrians and bycicles and this lane is detected also. Furthermore, it is shown whether each lane is considered a lane for pedestrians (P) or cars (C). There is a lane for crossing pedestrians on the road to the right, but this is not detected as the individual lane markings are connected with each other by the convolution of the image with the small gaussian window of size 3×3. The road that goes to the right is not detected at all as there are no markings. This algorithm will have to be extended in

Fig. 11. Line aggregation results

Fig. 13. Lane detection results on an intersection
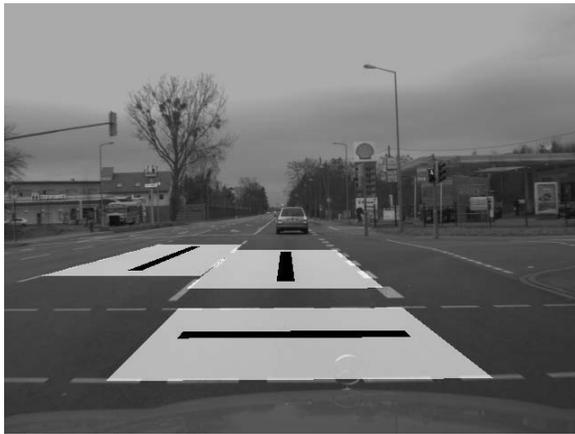
Fig. 12. Lane detection and classification results

order to be capable of detecting the road going to the right. However, one can see that a number of lanes with different orientations are detected. These results could be used to guide the vehicle in the own lane. At the same time, the knowledge about the crossing pedestrian lane could be used to watch out for crossing persons or persons that are not yet on the road but might have the intention to cross. Of course, this would require additional cameras looking to the right and the left of the car. In addition to that, the knowledge about the crossing lanes could be used for position estimation in conjunction with a digital map.

VI. CONCLUSION

The main characteristic of an intersection like presented in this paper is the periodic and structured arrangement of lane markings. Features like the number, width and orientation of lanes may vary from intersection to intersection, but the trait of periodicity and structuredness is one that is common to various different types of intersections. Thus, the proposed algorithm tries to exploit this periodicity. Requirements like the one that a line segment should feature at least three rectangles in order to be admitted to the lane producing process are chosen exactly to enhance periodic features.

There are definitely contours that do not come from lane markings and which still may pass the tests like minimal length and maximum width. These noisy contours are suppressed by the requirement of periodicity and they do not show up as false lane detections. Of course, if there were enough noisy contours positioned in a periodic scheme, these noisy contours would also yield false lane detections, but the probability for such unlucky circumstances is small.

This algorithm runs at twelve frames per second on an 1.6 GHz Intel Centrino. Real time requirements will be met simply by deploying faster hardware.

The proposed method was demonstrated on real imagery of a complex intersection and the results are promising. It was also shown that the approach enables a classification of detected road lanes. As mentioned above, in future work it is intended to estimate the markings of type ⊓ and ⊤ as well. Also, single, long marking lines will be taken into account.

REFERENCES

[1] Ernst D. Dickmanns and Birger D. Mysliwetz. Recursive 3-d road and relative ego-state recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):199 – 213, February 1992.

[2] F. Heimes, K. Fleischer, and H.-H. Nagel. Automatic generation of intersection models from digital maps for vision-based driving on innercity intersections. In *IEEE Intelligent Vehicles Symposium*, pages 498 – 503, Dearborn (MI), USA, October 2000.

[3] K. Kluge and C. Thorpe. The yarf system for vision-based road following. *Mathl. Comput. Modelling*, 22(4-7):213 – 233, 1995.

[4] W. Enkelmann, G. Struck, and J. Geisler. Roma - a system for model-based analysis of road markings. In *IEEE Intelligent Vehicles Symposium*, pages 356 – 360, Detroit (MI), USA, September 1995.

[5] Michael Krystek. Least-square straight lines in the plane. *Technisches Messen*, 71(1):19 – 23, 2004.

[6] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679 – 698, November 1986.

[7] Massimo Bertozzi and Alberto Broggi. Gold: A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, 7(1):62 – 81, January 1998.

[8] Forschungsgesellschaft für Strassen- und Verkehrswesen. *Richtlinien für die Markierung von Straßen*, 1993.

IEEE
COMPUTER
SOCIETY