A Fast and Robust Approach to Lane Marking Detection and Lane Tracking

Christian Lipski Björn Scholz Kai Berger Christian Linz Timo Stich
Marcus Magnor
Computer Graphics Lab, TU Braunschweig
Mühlenpfordtstrasse 23
38106 Braunschweig
Germany
{lipski,scholz,berger,linz,stich,magnor}@cg.tu-bs.de

Abstract

We present a lane detection algorithm that robustly detects and tracks various lane markings in real-time. The first part is a feature detection algorithm that transforms several input images into a top view perspective and analyzes local histograms. For this part we make use of state-of-the-art graphics hardware. The second part fits a very simple and flexible lane model to these lane marking features. The algorithm was thoroughly tested on an autonomous vehicle that was one of the finalists in the 2007 DARPA Urban Challenge. In combination with other sensors, i.e. a lidar, radar and vision based obstacle detection and surface classification, the autonomous vehicle is able to drive in an urban scenario at up to 15 mp/h.

1 Introduction

Detecting lane markings on roads in an urban environment is a difficult but very important task. Towards this goal, we present a lane detection system that is capable of analyzing several high-resolution images simultaneously in real-time. Our lane fitting algorithm uses a very versatile lane model and is robust against outliers and artifacts. It also takes into account lane markings of adjoining lanes. It copes with different road setups, lane markings and lighting situations. The lane detection process is divided into four subsequent parts, Fig. 1. First, the raw images are downloaded from the cameras via the IEEE1394b interface. Second, they are uploaded to graphics hardware, the color information is retrieved from the raw Bayer pattern, and the images are transformed into a single top view perspective, Fig. 2. Third, lane marking features are detected in the image, Fig. 4. In the last step, a lane model is fitted to match the found features.

2 Related Work

During the last ten years, many techniques for processing and analyzing images for lane detection purposes have been proposed. Although the idea of transforming a single camera image into a top view perspective using dedicated hardware was introduced in [1, 2], most approaches work on the original images. Many consider edges in grayscale images as sufficient features for lane markings [6, 8, 10]. Other methods for feature detection include morphological filters [1], steerable filters [7] and histogram analysis [4]. A challenging task is to create a lane model based on the features found in the image. Simple approaches try to fit a single line to the data [5], more complex approaches model the street as B-Snakes[10] or concentric circular arcs [6]. Surprisingly, only very few attempts have been made to work on color images [3] which is indispensable when colored lane marks should be detected. In order to cope with limitations of monocular systems, the use of additional geometric information has been proposed [5]. Graphics hardware can be used to accelerate image processing and detection [9].

3 Preprocessing

The preprocessing stage of the lane detection system provides an HSV top view image of the region ahead of the vehicle, Fig. 2.

3.1 Data Acquisition

Three cameras with field of view of 58° cover the area in front of the car. A 22° tele lens camera provides a high-resolution view of the street ahead of the car. The four 1376x600 8-bit raw Bayer images are synchronously acquired via the IEEE1394b interface at 14 frames per second. The images are uploaded to the graphics card and converted

978-1-4244-2297-5/08/$25.00 ©2008 IEEE

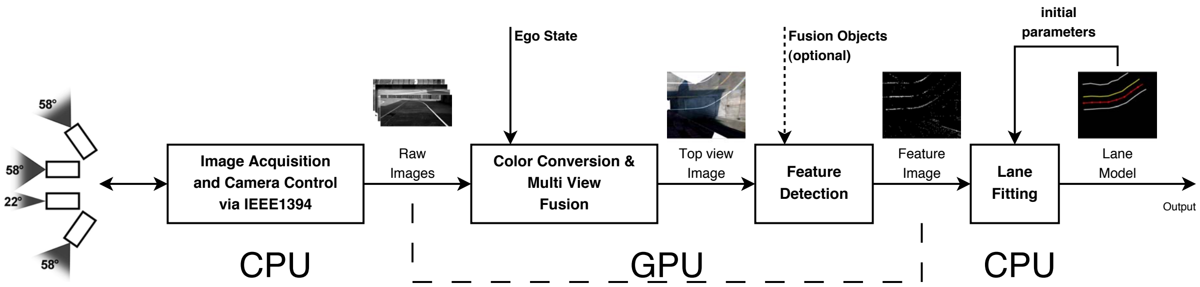


Figure 1. The four stages of the lane detection algorithm.

to the RGB color space using bilinear interpolation. As the lane fitting algorithm works in a global coordinate system, the position and rotation of the vehicle, also referred to as 'Ego State', must be available. A transformation function $f_{ego} : p_{car} \mapsto p_{world}$ can be defined if the Ego State is known, where p_{car} is a point in the car's reference system, and p_{world} is a point in a global Cartesian reference system. An Inertial Measurement Unit corrected by a GPS signal was used to generate the EgoState.

3.2 Multi-View Fusion

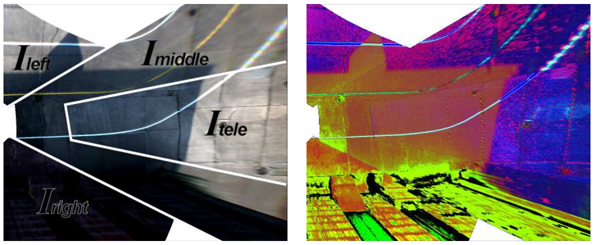
Because local changes of the light intensity are an indicator for white lines, and local changes in the saturation indicate colored lane markings, the RGB images are converted to the HSV color space. This color space encodes saturation and color in separate channels. Knowing the intrinsic and extrinsic parameters of the camera, and including the orientation of the vehicle (pitch and roll), a lookup function that converts top view coordinates to image coordinates can be used to create a single HSV top view image. The lookup operation is applied to each source image. In regions where the projected images overlap, the precedence $I_{tele} > I_{middle} > I_{left} > I_{right}$ is maintained, Fig. 2. The region of interest covers the area of up to 30m in front of the vehicle and 12m to the left and right at a scale of 35 pixel per meter.

4 Features

Lane markings can be described as a thin pattern of local differences of the road surface that cover long distances. Therefore, the basic idea of the feature detection is to identify these local differences in regions of 8x8-pixels that resemble road patches of approximately 25cmx25cm. Analyzing the HSV top view image, the feature detection's output is a downsampled feature image that encodes the quality, the direction and the color, i.e., white or yellow, of the lane features, Fig. 4. As lane markings exist in various colors, qualities as well as widths, and appear differently under changing lighting conditions, only few stringent assumptions apply. When analyzing the top view image for features, we check three criteria that have to be present:

1. The local contrast v_{diff} must exceed a certain threshold. The local contrast is the difference between the local minimal and maximal value $v_{diff} = v_{max} - v_{min}$.
2. Analyzing a local adaptive histogram, the distance b_{diff} between the two largest bins b_{high} and b_{low} must have a minimum value. This is because it can be assumed that b_{low} contains pixels depicting the street and b_{high} identifies the lane marking.
3. The pixels in b_{high} must have an evident shape and orientation. For several discrete orientations, the ratio of the variances of the pixels' x- and y-coordinates is checked.

A detailed description is given in algorithm 1. As this algorithm is prone to discretization errors, supersampling improves the quality of the feature detection.



(a) RGB top view

(b) HSV top view

Figure 2. The four different images (a, RGB color space used for visualization) are merged to a single HSV top view image (b).

5 Lane Tracking

The main goal of the lane fitting algorithm is to find a parameter set for a lane model that explains the features found in the current top view image and the previous frames. In order to create a global model of the lane, all feature points are mapped to world space coordinates and inserted to a list l_p. This is done using the function $f_{ego} : p_{car} \mapsto p_{world}$ defined by the current Ego State. Old data, i.e. feature points

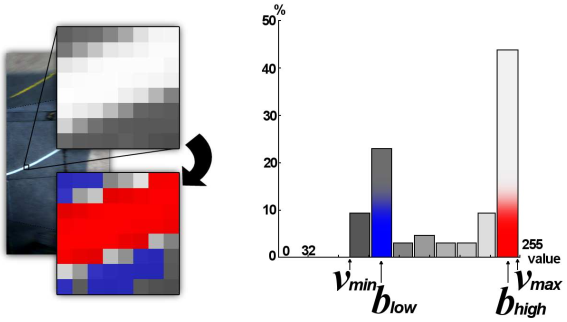
```
Data: An 8x8 region of a HSV top view image, thresholds
      $t_{con}, t_{hist}, t_{dir}$ and $t_{col}$
Result: A feature quality $q$, direction $a \in \{0, 22.5, ..., 157.5\}$ and
        color $c \in \{white, yellow, undecided\}$
for the saturation and lightness channel do
    $v_{diff} = v_{max} - v_{min}$; $v_{max}$ and $v_{max}$ are the maximal
    and minimal values of the current channel
    if $v_{diff} < t_{con}$ then
        break;
    compute adaptive histogram;
    determine two largest bins $b_{high}$ and $b_{low}$, Fig. 3b ;
    $b_{diff} = b_{high} - b_{low}$;
    if $b_{diff} < t_{hist}$ then
        break;
    set of pixels $p_{high}$ = pixels in $b_{high}$;
    determine center of mass $R$ of $p_{high}$;
    initialize $r_{max}$ and $a_{max}$ to 0;
    for $i = 0; i <= 157.5; i = i + 22.5$ do
        rotate $p_{high}$ around $R$ by $i$ degrees. determine ratio of
        variances $r = \frac{Var(X)}{Var(Y)}$;
        if $r > r_{max}$ then
            $r_{max} = r; a_{max} = i$;

    if $r_{max} < t_{dir}$ then
        break;
    label this region as a feature;
    if current channel is lightness then
        $q_{white} = b_{diff}; a_{white} = a_{max}$
    else
        $q_{yellow} = b_{diff}; a_{yellow} = a_{max}$

if $q_{white} > t_{col}$ & $q_{white} > q_{yellow}$ then
    $c = white; a = a_{white}$
if $q_{yellow} > t_{col}$ & $q_{yellow} > q_{white}$ then
    $c = yellow; a = a_{yellow}$
$q = max(q_{white}, q_{yellow})$;
```

Algorithm 1: Feature Detection Algorithm

gathered during previous frames, may be kept if the features of a single image are too sparse.

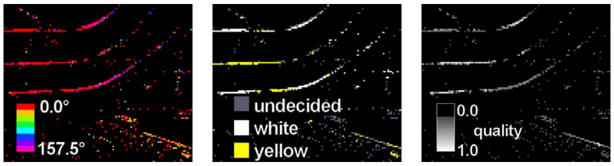
5.1 Lane Model

The lane model consists of connected lane segments. Each segment s_i is described by a length l_i (given parameter), a width w_i and an angle $d_i = \alpha_i - \alpha_{i-1}$ describing the difference of orientation between this segment and the previous one, Fig. 6. The first segment is initially placed on the current coordinates of the vehicle and facing the driving direction, assuming that the vehicle is actually located on the street. Knowing the position c_0 of the initial segment as well as the lengths l_i and the angular changes d_i of all segments, the position p_i and global orientation α_i of each segment can be computed. Each segment contains information whether the vehicle's lane is confined by lane markings and whether additional lanes to the left and right exist. Straight streets, sharp curves and a mixture of both can all be described by the model.



(a) 8x8 regions are analyzed (b) 8 bin histogram of the 8x8 region

Figure 3. 8x8-pixel regions of the top view image (a, up) are tested for possible features. The distance between the two largest bins b_{low} (b, blue) and b_{high} (b, red) of the histogram determines the quality of the feature. The pixels gathered in b_{high} must be arranged in a directed shape (a, red area).



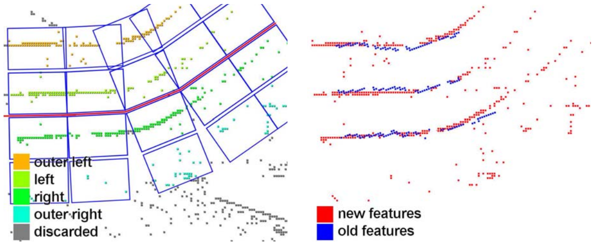
(a) direction information (red channel) (b) color information (green channel) (c) quality information (blue channel)

Figure 4. The direction (a), color (b) and quality (c) of the features are encoded in a RGB image downloaded from the graphics card. For visualization purposes, the channels encoding the direction (a) and color (b) are colorized

5.2 Lane Fitting

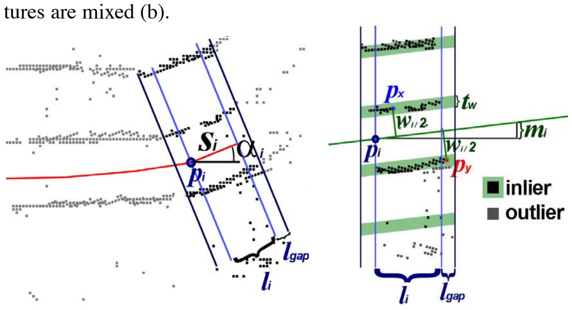
For each frame, the existing lane model or an initial guess is used to define four regions of interest, Fig. 5a. These are the regions expected to contain the own lane's markings and the lane markings of the adjoining lanes. If a feature is inside such a region, it is labeled as $outer\ left$, $left$, $right$ or $outer\ right$. Otherwise, it is discarded. Afterwards, features from previous frames are mixed with the new data, Fig. 5b.

The first currently visible segment s_f of the lane model is determined, older segments are no longer considered. If the list of lane segments is empty, it is initialized with $s_0 \leftarrow s_f$. Starting from s_f, each segment s_i is estimated (or reestimated if it has already been estimated previously). Therefore, an initial guess of the orientation α_i of s_i is made, Fig. 6. All local features relevant for estimating s_i are rotated by α_i around the starting point p_i of s_i. A RANSAC algorithm is used to estimate the parameter d_i and w_i: Iteratively, two feature points p_x and p_y are chosen. Assuming that they lie on the lane markings they were labeled for, the gradient $g_i = m_i/l_i$ as well as the width w_i are derived from their coordinates. All features that are also sufficiently described



(a) labeling and discarding (b) mixing

Figure 5. Regions of interest (a, blue boxes) determine to which lane marking features are assigned. Afterwards, old and new features are mixed (b).



(a) estimating segment s_i (b) RANSAC fitting

Figure 6. p_i, α_i, l_i and l_{gap} identify the features relevant for s_i. After rotating around α_i, a RANSAC fitting eliminates outliers among the features.

by g_i and w_i are counted as inliers. This process is repeated n times and the parameter set with most inliers is used to define s_i. A quality function q takes into account the ratio of inliers and outliers, the amount of inliers, the quality of the features and states the quality of the segment. The quality is computed for every region of interest ($outer\ left$, $left$, $right$ and $outer\ right$). If the maximum of these qualities exceeds a threshold t_q, the segment is considered as valid and the next segment s_{i+1} is estimated. After all segments are estimated, Fig. 7, a proposal about the lane markings' colors can be made by looking at the inliers' average color.

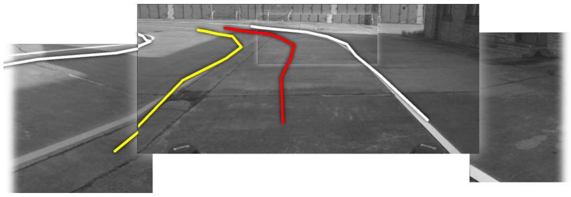


Figure 7. The lane model reprojected onto the original images.

6 Results and Evaluation

The algorithm has been thoroughly tested on several sites in Northern Germany and Texas. Its output enabled an autonomous vehicle to stay in lane while driving at 15 mph. A frame rate of 10 fps could be maintained using a 2 GHz Pentium with a GeForce 7600 GTS graphics card. The testing sessions included different weather and lighting conditions. The amount of false positives could be reduced significantly by utilizing the vehicle's other sensors. The objects detected by lidar and radar sensors were used to mask out regions in the feature image where other cars, walls, cones and poles caused irritating artifacts in the top view image. The vehicle was one of eleven finalists of the 2007 DARPA Urban Challenge.

7 Conclusion

We presented a multi-camera lane detection algorithm that makes use of a conventional PC and a graphics card. Our approach is capable of detecting lane markings at 10 fps. The feature detection and lane fitting approach are able to cope with different lighting situations, weather conditions, road layouts and lane markings. Extensive tests have shown that the quality of the algorithm can be improved by considering other sensor inputs. Laser scanners may generate height maps in top view space which can be used to correct image fusion in uneven terrain. Another promising approach is a post-processing pass of the feature image. First experiments with a line enhancement using probabilistic Hough lines yielded good results and stabilized the algorithm.

References

[1] M. Bertozzi and A. Broggi. GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, pages 62–81, 1997.

[2] A. Broggi and S. Berte. Vision-based road detection in automotive systems: A real-time expectation-driven approach. *Journal of Artificial Intelligence Research*, 3:325–348, 1995.

[3] F. Diebolt. *Reconnaissance des marquages routiers*. PhD thesis, L'Univ. Louis Pasteur de Strasbourg, Strasbourg, France, 1996.

[4] J. P. Gonzalez and U. Özgüner. Lane detection using histogram-based segmentation and decision trees. In *Intelligent Transportation Systems*, pages 346–351, 2000.

[5] T. Hong, T. Chang, C. Rasmussen, and M. Shneier. Road detection and tracking for autonomous mobile robots. In *Proceedings of SPIE Aerosense Conference, Vol. 4715, Orlando, Florida*, pages 35–40, 2002.

[6] K. Kluge and S. Lakshmanan. A deformable-template approach to lane detection. In *Intelligent Vehicles Symposium*, pages 54–59, 1995.

[7] J. McCall and M. Trivedi. An integrated, robust approach to lane marking detection and lane tracking. In *Intelligent Vehicles Symposium*, pages 533–537, 2004.

[8] F. Paetzold and U. Franke. Road recognition in urban environment. In *Image and Vision Computing, vol. 18*, pages 377–387, 2000.

[9] R. Strzodka, I. Ihrke, and M. Magnor. A graphics hardware implementation of the generalized hough transform for fast object recognition, scale, and 3d pose detection. *Proc. IEEE International Conference on Image Analysis and Processing (ICIAP'03),* Mantova, Italy, pages 188–193, Sept. 2003.

[10] Y. Wang, E. Teoh, and D. Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22(4):269–280, April 2004.