Real-time Detection and Classification of Arrow Markings using Curve-based Prototype Fitting

Georg Maier and Sebastian Pangerl and Andreas Schindler

Abstract— **Driver assistance systems using situation analysis require a reliable environment perception in order to improve the traffic safety.**

Many previous works dealt with lane recognition techniques. In this paper we focus on the problem of detecting and classifying supplementary lane markings like painted arrows. We present a general geometric approach using curve-based prototype fitting. Although our method can process data from various sensor sources, we focus on the usage of a monocular gray value camera. Regions of interest are determined based on some preliminary knowledge in order to reduce the search space. Finally, prototypes encoded as arc splines are used for the comparison with the extracted contours of object candidates, which enables both detection and classification.

We show that the concept of arc spline models yield a data efficient and real-time capable application.

I. INTRODUCTION

High accident rates in urban environments are caused by lane change maneuvers, especially in the surrounding of intersections (cf. [1]). A myriad of studies were dedicated to lane recognition, lane departure warning, lane change assistance systems and other lane based analysis in order to support the driver in dangerous situations. The information located on the road surface can contribute to these applications substantially. However, only few systems were presented for the extraction of supplementary road markings. Arrow markings, that are painted on the road, regulate the driving directions at the upcoming intersection or junction. They advise road users for their choice of the right lane at an early stage. Additionally, they permit road users to drive on multiple lanes side by side in a more secure way since they define the possible driving directions. Therefore, advanced driver assistance system can improve their performance and reliability by taking into account supplementary information painted on the road surface.

In this paper, we focus on the detection and the classification of painted arrows on the lane. We present a general geometric approach that uses prototype fitting for the recognition of predefined arrow models in real traffic scenes. Our approach can be easily extended to the recognition of other painted road markings like characters (STOP) and numbers (speed limits).

A. Related Work

Some research approaches have been presented over the last decades and they vary significantly in their methods for the detection and classification of arrow markings. Most of the studies have in common a two step approach, consisting of the extraction of candidates and a classification step. In [2] arrow candidates are created by analyzing the projection histograms on inverse perspective images. Afterwards the classification is based on multi-class support vector machines. Another approach is presented in [3], where a gray level segmentation is used for the recognition. The classification is performed by decision tree evaluation and size constraints. Reference images of arrows serve as templates for a template matching approach in [4]. Instead of pixel-based templates the system in [5] refers to a geometric pattern matching for the classification of painted objects. The authors of [6] evaluate strategies for the extraction of different road markings and present an extension of the median local threshold algorithm by stereo-vision.

However, none of the papers listed above has dealt with a curved based geometric approach for detection and classification.

B. Outline

The outline of this paper is as follows. In Section II we sketch the individual components and steps of our method. The principles of prototype fitting are described in Section III. Our classification approach is elucidated in Section IV, while some results are shown in Section V. Finally, we summarize our work and present some future ideas (Section VI).

II. METHOD OVERVIEW

Our approach follows a two step strategy, divided into a preprocessing and a recognition phase. An overview is pictured in Fig. 1. The preprocessing component extracts candidates for arrow markings from a given gray value image. In the second part, these candidates are analyzed using prototype fitting techniques and a classification based on fitting measures. Since the main contribution of this paper is on the recognition, we present the preprocessing and the candidate creation step in a short form. These blocks are replaceable with other strategies. Instead of the monocular gray value camera other sensors can be used like stereo systems or laser scanners.

A. Lane recognition and region of interest (ROI)

Since image processing tasks are often expensive in terms of processing power, we are keen to narrow down the search space for the extraction of arrow marking candidates. According to the German road construction regulations (cf. [7]),

All authors are with FORWISS, Institute for Software Systems in Technical Applications of Computer Science, University of Passau, 94032 Passau, Germany {gmaier, pangerl, schindler}@forwiss.uni-passau.de

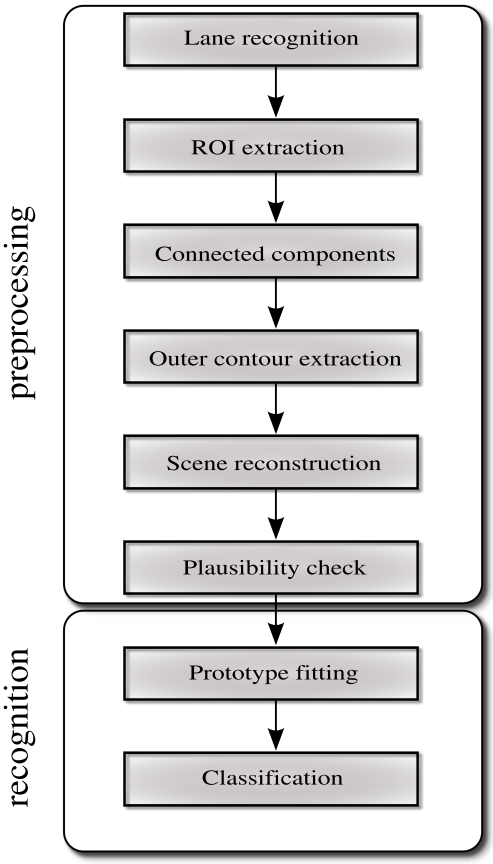


Fig. 1: Overview of the processing chain

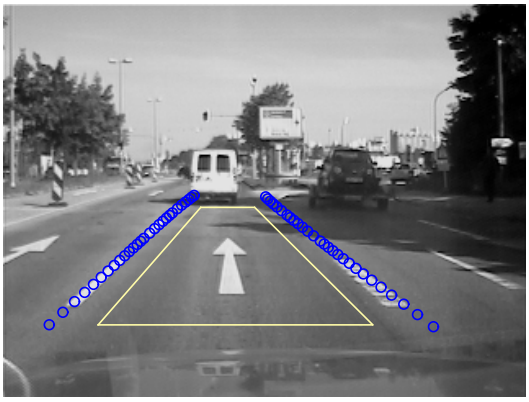


Fig. 2: The drivable area in front of the vehicle as a result of lane recognition system.

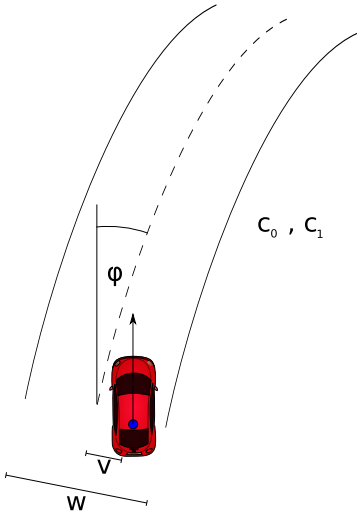


Fig. 3: Road model for video-based lane recognition including the parameter lane width (w), relative yaw angle (φ), vertical offset (v) from the middle of the lane and two curvature parameter (c_0, c_1).

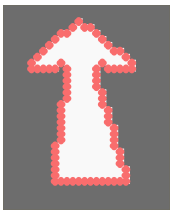


Fig. 4: The largest connected component is extracted from the mask image and its outline is reprojected on the street plane.

arrow markings are located at the center of the lane. In order to define a suitable ROI for them, we first apply a lane recognition system which works on both marked lanes ([8]) and on lanes without line markings ([9]) even under difficult scenarios. Fig. 3 shows some details on the road model parameter used for the lane recognition. We combine the estimated drivable area in front of the ego vehicle with preliminary knowledge on the road plane to finally select a region of interest in which arrow markings may appear. In out implementation we have chosen a region of interest from 4 to 20m in front of the car. For the classification an arrow candidate must be located entirely inside this region. An example of the resulting ROI together with the recognized lane is illustrated in Fig. 2.

B. Connected Components

As the arrow markings are painted with high contrast to the road surface, we are looking for connected image parts that significantly differ from the asphalt. Connected components are extracted within the ROI using some segmentation threshold which can be gained by N-level-fitting techniques (cf. [10]). For the further processing only the outline of the connected components is used as it describes the geometric shape of the candidate. The result of this step is a contour pixel list bordering the extracted component (Fig. 4).

C. Scene Reconstruction and Plausibility Check

While the arrow markings are painted on the road surface, their observed candidates are given at first in the image coordinate system (*ICS*). In order to enable a device independent recognition approach, we reconstruct the local environment by reprojecting the extracted contour pixel list on the road surface. This is done by intersecting rays, defined by the

optical center of the camera and some image coordinate, with the assumed driving plane. Thereby, a point list in street coordinate system (*SCS*) is created, that represents the reprojected contour list of the arrow candidate. For that purpose high precise camera calibration is needed ([11]) enriched by a video-based ego motion estimation in order to compensate short term movements of the vehicle like pitching and rolling.

The geometric shape and size of arrow markings are standardized on German roads [7]. Hence some plausibility checks in form of an outlier elimination can be applied in order to eliminate some false candidates before the recognition algorithm is executed.

III. Prototype Fitting

In order to compare a candidate, i.e. an extracted contour, with the particular prototypes for classification, we use a generalization of the *Iterated Closest Point (ICP) algorithm* (see [12]), the so-called *prototype fitting*, which was introduced in [13]. Whereas current ICP algorithms are based on set representations, our approach encodes the prototypes as a curve which allows a fast computation of point to curve distances and which is composed of a preferably low number of segments. As a result, we yield a faster, more accurate and less memory-intensive method for matching a prototype to a certain point set.

A. Fundamentals in Prototype Fitting

When we consider an object like an arrow marking - after having them re-projected - its position in the plane is not the same as the location of the corresponding prototype. For instance, the observed object might be rotated, translated and even somewhat scaled. In general, we want to transform our prototype in a way such it fits the extracted contour as well as possible. I.e., we search for a transformation $\Phi : \mathbb{R}^2 \to \mathbb{R}^2$ minimizing the sum of squares $\sum_{i=1}^{n} \operatorname{dist}(\Phi(P), y_i)^2$ with available contour points $y_1, \ldots, y_n$ of the observed object and a prototype $P \subset \mathbb{R}^2$, where dist denotes the euclidean distance. Naturally, the existence of such optimal motions can only be assured if we make some restrictions and assumptions on the feasible transformations. Assuming that our camera calibration makes us re-projecting the points accurately, we focus on motions in plane that consist of translations, isotropic scalings and rotations, which are mappings of the form $T_{c,s,t} : \mathbb{R}^2 \to \mathbb{R}^2,\ x \mapsto A_{c,s} \cdot x + t$ with scaled rotation

$$A_{c,s} := \begin{pmatrix} c & -s \\ s & c \end{pmatrix} \text{ for all } c, s, \in \mathbb{R}.$$

and translation $t \in \mathbb{R}^2$. Hence we search for optimal parameters $c, s \in \mathbb{R}$ and $t \in \mathbb{R}^2$.

In order to adapt the extrinsic camera calibration for this detection and classification task, we also could include more sophisticated transformations, like non-isotropic scalings or projections (cf. [13]). However, even for a larger class of admissible transformations, the pose estimation regarding rotation, scaling and translation is important to have a first match. Hence focusing on this class is sufficient within this scope. Then, the problem formulated above can be solved very fast by an iterative approach.

To begin with, let us assume that some initial transformation parameter c_0, s_0, t_0 are known such that the transformed prototype approximately fits to the contour points. In order to obtain such an initialization we refer to Section III-C.

In any case, after having found an initial transformation, we can compute the best approximating points x_i of y_i with respect to the set $P^{(1)} := T_{c_0,s_0,t_0}(P)$ and calculate the parameters $s_1, c_1 \in \mathbb{R}$ and $t_1 \in \mathbb{R}^2$ minimizing

$$\sum_{i=1}^{n} \left\| T_{c,s,t}(x_i) - y_i \right\|^2,$$

which can be calculated in a closed form (cf. [14]). Again, we can compute the best approximating points $x_i^{(2)}$ of y_i with respect to $T_{c_1,s_1,t_1}(P^{(1)}) =: P^{(2)}$ and solve the least squares problem as above. We carry on with this alternating procedure while the difference between the predecessor error and the current error is greater than a given threshold or a given maximum iteration number has been reached. As a matter of course, the parameters and thresholds depend considerably on the real application itself, which we will discuss later on.

B. Prototype Encoding

In order to achieve an efficient computation of this iterative method it is decisive that the calculation of the best approximating points with respect to the prototype is very fast. As already discovered in [12] and [13] the necessity of a fast determination of best approximating points doesn't depend on a special choice of the optimization method. Indeed, the efficiency of the calculation of closest points depends on the encoding of the prototype P. Therefore, we suggest a description of P as a union of curves having a low number of segments and providing a fast calculation of best approximating points. Furthermore, high flexibility for modeling the desired geometric pattern is needed. Since almost all ICP methods are based on point encodings of the prototype, sophisticated point selection approaches are needed to achieve efficiency improvements. Obviously, a curve representation, as described above, has considerable advantages over these techniques regarding accuracy, time and storage space.

One possible choice of such a curve, fulfilling these criteria, is an *arc spline* which is a composition of circular arcs and line segments $\gamma := \gamma_1 \ldots \gamma_N$. Since arc splines are determined by only a few parameters and they are invariant with respect to rotations, scalings and translations, they can be applied well to geometric pose estimation and detection problems as shown in [14]. In contrast to parametric curves, which need iterative strategies, the best approximating point x_i of y_i to the nearest circular segment γ_j can be calculated in a closed form. Therefore, the computational costs are very cheap which is a crucial attitude for making the prototype fitting fast, as already seen. Thus, a representation of the prototype in form of an arc spline composed of as few as

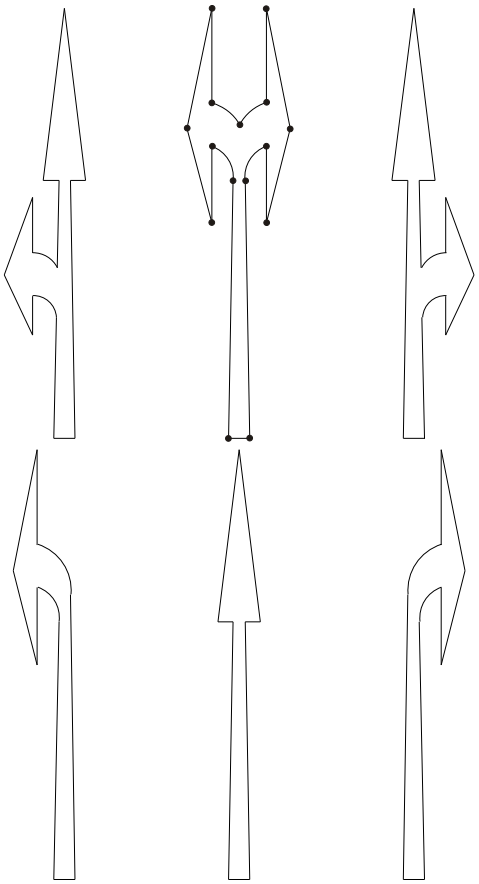


Fig. 5: Arc spline models of the arrow markings (ahead-left, left-right, ahead-right, left, ahead, right). In the middle image of the first row the breakpoints of the arc spline are accentuated.

possible segments is a suitable description leading to an efficient approach.

The geometry of painted arrows on German roads is explicitly characterized in [7]. According to this specification we generated arc spline models of the several prototypes (cf. Fig. 5).

In [14] a method is proposed for the computation of an arc spline approximating a sequence of points guaranteeing a minimum number of segments as well as a freely definable approximation accuracy given by a so-called tolerance channel along the points. Due to the relationship of this problem to the *minimum link problem* the resulting arc spline is called *(continuous) minimum arc path*, abbreviated by *CMAP*. Though this algorithm guarantees the minimally possible number of segments with respect to any accuracy, it doesn't satisfy real time requirements. However, as the prototype can be generated off-line, the computational time doesn't play an important role.

C. Initial Solution

As in every nonlinear optimization problem, a good start value is crucial for the performance of the solving method. Hence we deal in this section with the problem of estimating an initial transformation, which provides a suitable first match of the prototype and the observed object.

In case of detecting and classifying supplementary lane markings we can surely adapt an initial guess using preliminary knowledge deduced from the vehicle state. Besides, we suggest the following strategy in case of a translated, rotated and possibly scaled object: An initial translation t_0 is due to the difference of the barycenter of the extracted points and the barycenter b of the prototype, i.e.:

$$t_0 := \frac{1}{n}\sum_{i=1}^{n} y_i - b.$$

Note that the barycenter of a prototype described by an arc spline can be computed in a closed form (cf. [14]).

Then, it is reasonable to compute an initial rotation R_0 by the angle between the principal axes of the prototype and the contour points. The calculation of the principal direction of a set of points meets calculating the best approximating line and in case of an arc spline there even exists a closed form solution. As generally known, the computation of a best approximating line is a eigenvalue problem of a symmetric positive definite 2×2 matrix, which can be solved very fast. In fact, an ambiguity of 180 degrees is possible, as we obtain lines but not directions. However, knowing the principal direction of our vehicle and having a rough estimation of the arrow's direction, which has to be detected, the wrong direction can be identified easily.

In addition, the root mean square deviation δ_C of the point set $C := \{y_1, \ldots, y_n\}$ and the corresponding barycenter μ_C is given by $\delta_C := \sqrt{\sum_{i=1}^{n} \operatorname{dist}(y_i, \mu_C)^2}$ Again, the likewise value δ_P of the prototype P can be computed in a closed form in case of an arc spline. Thus, a suitable choice for an initial scaling factor λ_0 is the ratio $\lambda_0 := \delta_C/\delta_P$. This way we can generate a satisfying starting solution such that the whole iterative strategy works very efficiently. In practically all out experiments, we could observe the convergence to the global minimum of the error function after only a few steps.

IV. Classification

After the preprocessing step (cf. Fig. 1) a point set corresponding to a potential candidate is available. Next, the prototype fitting algorithm is performed for the comparison of the contour with each of the prototypes P_j. For each j let P_j' be the result when applying the optimal transformation to P_j. By evaluating the fitting errors

$$E_j := \frac{1}{n}\sum_{i=1}^{n} \operatorname{dist}\left(y_i, P_j'\right)$$

and taking into account the Hausdorff distance $h(C, P_j')$

$$M_j := \max\left(\max_{x \in P_j'} \operatorname{dist}(x, C), \max_{i=1,\ldots,n} \operatorname{dist}\left(y_i, P_j'\right)\right),$$

we perform our classification. Then, we assign the current candidate to the j_0-th prototype if M_{j_0} is smaller than a predefined threshold ε, and E_{j_0} is the minimum fitting error regarding all prototypes P_j satisfying $M_j < \varepsilon$. If none of

the prototypes satisfies the condition $M_j < \varepsilon$, the current contour is supposed not to correspond to any arrow marking ($j_0 := 0$). The whole classification strategy is summarized in Algorithm 1.

Algorithm 1 Classification

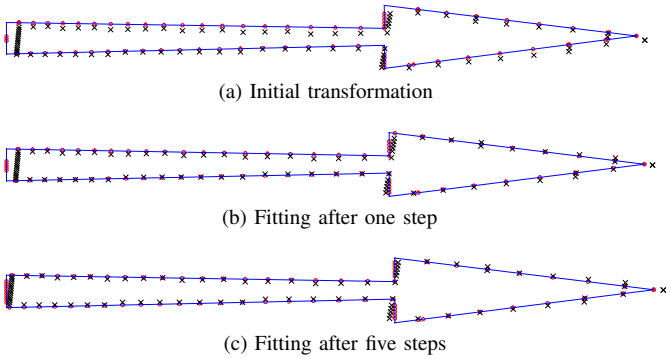
Input: Prototypes $P_1, \ldots, P_N$, threshold $\varepsilon > 0$, contour points $C := \{y_1, \ldots, y_n\}$,
Output: Classified prototype number $j_0 \in \{0, \ldots, N\}$

```
L = ∞
j0 = 0
for j = 1 to N do
    apply an initial transformation for Pj and C
    run prototype fitting regarding Pj and C
    P' ← Φ(Pj) //  Φ is the optimal transformation
    E ← Σ dist(yj, P')^2 //  least squares fitting error
    M ← h(C, P') //  Hausdorff distance
    if M < ε and E < L then
        j0 ← j
    end if
end for
return j0
```

V. Results

In this section we present some evaluations of our method. Exemplary traffic scenes are depicted in Fig. 2 and 9 including the calculated ROI which contains an arrow marking. Fig. 10 shows the corresponding mask image and the best fitting prototype with respect to the reprojected contour points. The reconstructed contour point list of the extracted connected component is passed to the prototype fitting algorithm after some plausibility checks. Fig. 6 shows some iterations of the prototype fitting, starting with the initial transformation according to Section III-C. The fitting error as a function of the number of iterations is shown in Fig. 7 for all six different arrow prototypes. Obviously, the fitting error for the arrow pointing ahead is much smaller than the fitting error of the other prototypes. After five iterations the fitting error for the best fitting prototype changes in the forth positions after decimal point. Fig. 8 shows the corresponding Hausdorff distances of the prototypes during fitting process. As the prototype fitting minimizes the least squares error but not the Hausdorff distance, the latter can even increase during the iterations. Nevertheless, the arrow pointing ahead is also the best fitting arrow according to the Hausdorff metric, which turns the overall classification to the expected result.

Even though a performance measurement is dependent on many conditions, especially on hardware requirements, we give a rough estimation in order to have a better basic orientation of the complexity. Since this approach was used as a proof of concept for curve based prototype fitting, no effort was made for optimizing any software issues. Further performance improvements will be done within the project Ko-PER. Anyway we can give some figures which are restricted only to the recognition phase (Fig. 1). A scenario with 82 contour points consumes for one fitting iteration between 0.05 and 0.09 ms. The measurement was taken on a 2.67 GHz processor.



(a) Initial transformation

(b) Fitting after one step

(c) Fitting after five steps

Fig. 6: Iterations of prototype fitting. The higher the number of iterations the smaller are the transformation changes. Prototype fitting converges to a stable state.

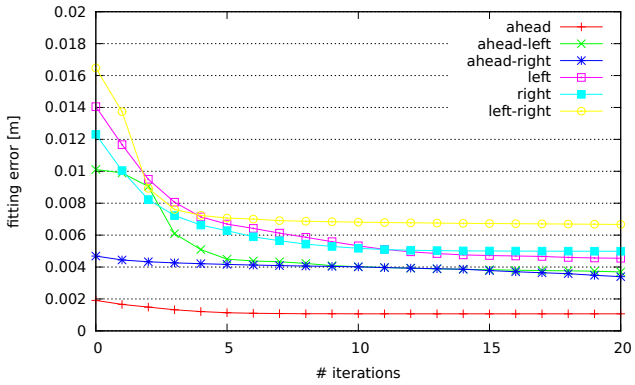


Fig. 7: Fitting error as a function of the number of iterations for all arrow prototypes.

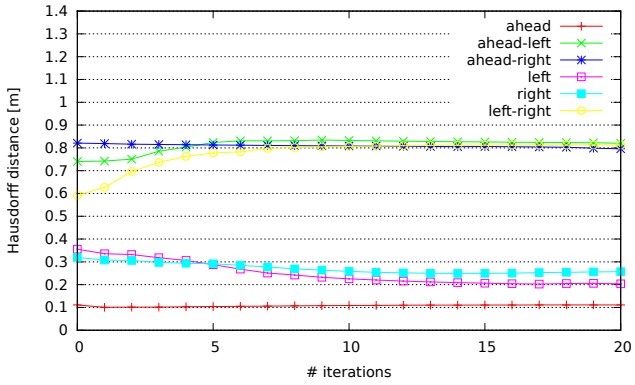


Fig. 8: Hausdorff distance as a function of the number of iterations for all arrow prototypes.

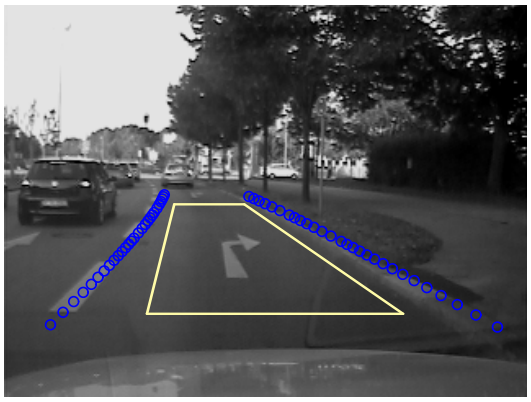


Fig. 9: A scenario with a right arrow.

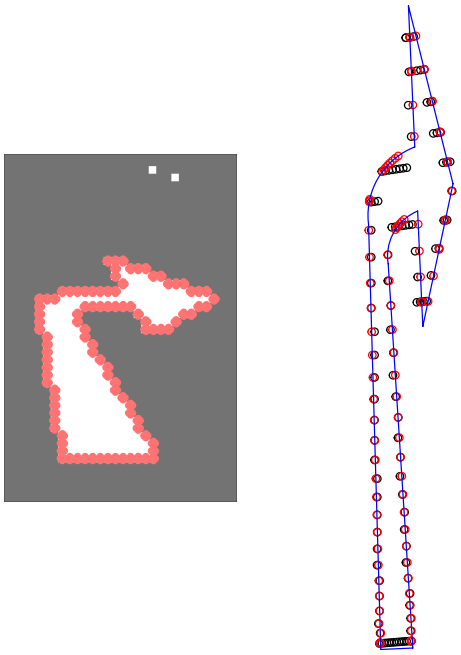


Fig. 10: The left figure shows the extracted pixel list of the connected component. The black points on the right figure indicate the pixel list reprojected to the street plane. The red ones are the perpendicular points on the prototype.

VI. Conclusion And Future Work

We have presented a method for the detection and classification of arrow markings painted on the road surface. After a preprocessing step, which results in a point list with coordinates in the SCS, our approach uses prototype fitting for the recognition of predefined shapes. Due to the advantageous mathematical properties of arc splines, we can ensure an efficient comparison of arrow candidates with the prototypes when encoding them as arc splines. First evaluations show that our approach achieves the recognition task in real traffic situations in a promising way regarding to computational time, data volume and classification rate. A more detailed evaluation of our method with a wide range of different scenarios will be carried out within the scope of the project Ko-PER.

Future extensions of our method will include the improvement of outlier detection in the image processing part as well as the development of compensation approaches to more difficult scenarios like shadowy lanes. Furthermore, tracking of arrow hypothesis will be implemented in order to improve the classification by taking into account temporal integration.

VII. Acknowledgments

This work results from the joint project Ko-PER, which is part of the project initiative Ko-FAS, and has been funded by the German Bundesministerium für Wirtschaft und Technologie (Federal Department of Commerce and Technology) under grant number 19 S 9022E.

We would like to thank BMW Group Research and Technology for the cooperation in this project and for providing sensor data of various test scenarios.

References

[1] S. Bundesamt, "*Verkehrsunfälle - Fachserie 8 Reihe 7*," September 2010.

[2] N. Wang, W. Liu, C. Zhang, H. Yuan, and J. Liu, "The detection and recognition of arrow markings recognition based on monocular vision," in *Proceedings of the 21st annual international conference on Chinese control and decision conference*, ser. CCDC'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 4416–4422. [Online]. Available: http://portal.acm.org/citation.cfm?id=1714810.1714972

[3] R. Danescu and S. Nedevschi, "Detection and classification of painted road objects for intersection assistance applications," in *13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Sept 2010, pp. 433–438.

[4] S. Vacek, C. Schimmel, and R. Dillmann, *Road-marking analysis for autonomous vehicle guidance*. ECMR, 2007, no. 1, pp. 1–6.

[5] I. M. Chira, A. Chibulcutean, and R. G. Danescu, "Real-time detection of road markings for driving assistance applications," in *International Conference on Computer Engineering and Systems (ICCES)*, Dec 2010, pp. 158–163.

[6] Y. Sebsadji, J.-P. Tarel, P. Foucher, and P. Charbonnier, "Robust road marking extraction in urban environments using stereo images," in *Proceedings of IEEE Intelligent Vehicle Symposium (IV'2010)*, San Diego, California, USA, 2010, pp. 394–400, http://perso.lcpc.fr/tarel.jean-philippe/publis/iv10b.html.

[7] Bundesministerium für Verkehr, "Richtlinien für die Markierung von Straßen," 1993.

[8] T. Tatschke and J. Urlhardt, "Model-based road lane extraction from monocular image sequences," in *PReVENT Fusion Forum eJournal Vol. 2.*, 2008.

[9] A. Schindler and V. Lauren, "Video-based recognition of unmarked lanes via texture interpretation and n-level-set-fitting," in *12th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2009, pp. 1–6.

[10] T. Hanning and G. M. Pisinger, "A pixel-based segmentation algorithm of color images by n-level-fitting," in *CGIM 2002, 5th IASTED International Conference*, 2002.

[11] T. Hanning, *High Precision Camera Calibration*, 2009.

[12] P. Besl and H. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[13] K. Donner, "Image interpretation based on local transform characterization," *Pattern Recognition and Image Analysis*, vol. 7, no. 4, 1997.

[14] G. Maier, *Smooth Minimum Arc Paths. Contour Approximation with Smooth Arc Splines*. Aachen: Shaker, 2010.