Real-Time Detection of Road Markings for Driving Assistance Applications

Ioana Maria Chira, Ancuta Chibulcutean
Students, Faculty of Automation and Computer Science
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
{ioana.mchira, ancuta.chibulcutean}@yahoo.com

Radu Gabriel Danescu
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
Radu.danescu@cs.utcluj.ro

Abstract— **For a Driving Assistance System, which is designed to improve safety on the roads, knowledge about the type of lane border markings and other painted road objects is required. Furthermore, information about the position of the painted objects can be used by other systems to create a correct perception of the road. This paper describes a LabVIEW based system for detection, measurement and classification of painted objects. The system is able to calibrate monocular video sources, define the features of the painted objects that we look for, and detect them in real time using geometric pattern matching and edge detection. The solution can be easily deployed on a dedicated real-time computing architecture, to be used as a standalone driving assistance sensor.**

Keywords-lane detection; road markings; pattern matching; driving assistance; LabVIEW

I. INTRODUCTION

The correct perception of the road ahead can improve the way the car is manipulated. Recognizing from a certain distance the painted road objects, knowing the number of lanes and their type, can help the driver assistance system take the correct decision in changing or remaining on the same lane. For these reasons this work is focused on extracting information about the road ahead, treating each painted sign as an individual object, instead of detecting lanes as a whole. The location and the type of the painted objects found can be used by another application that will analyze them and give a correct position of the car and a good perspective of the road.

Lane detection and tracking is the focus of many researchers, ever since the camera and computer technology allowed the problem to be tackled. In lane detection, several key problems have to be solved: choosing an appropriate lane model, developing an image processing based technique for delimiting features extraction, lane model matching to the extracted features, and probability based tracking. There are multiple solutions for each of the sub-problems, and a comprehensive survey of these techniques can be found in [1]. The model-based tracking approach to lane detection is robust against noise, and allows the use of less than perfect features, extracted with simple and fast algorithms. However, the quality of the extraction of lane delimiting features remained an active concern. In [2] a comparative analysis of the most popular marking extraction techniques is presented, and the best results are found to be provided by the algorithms that compare the brightness of the feature pixel with the brightness of its neighbors at specific distances, which account for the size of the marking to be found and for the perspective effect.

Unfortunately, not all traffic scenarios can be handled in the model-detect-track approach. These situations call for a more refined approach to lane delimiting features extraction. For these reasons, some researchers have dedicated their work to the problem of identifying the lane delimiters, and the painted road objects, as standalone elements. Identifying these objects individually makes the system independent on the lane shape, and may provide valuable information to a driving assistance system.

The method presented in [3] uses histogram-based image segmentation and then applies decision trees on basic geometrical features to decide whether a painted object is a lane marking or not, and also to decide whether a set of objects are part of the same lane border. In [4] the painted road objects are extracted as polygons, and their basic properties such as length, parallelism and direction are used to decide whether they are possible lane delimiters or pedestrian crossings. In [5] we find a technique for extracting the rectangular painted objects by the use of an analytical model of the expected pixel intensity based on rectangle parameters, and the search for the parameters that minimize the error between the expectation and the image data. The aim of this technique was to identify lane delimiters in intersection scenarios.

A system that is able to identify and classify a wider range of painted road objects, not only lane marking delimiters, is presented in [6]. The segmentation is based on intensity levels and connected components analysis, and the classification uses a radial basis function classifier.

Instead of using the painted road markings for lane detection, some authors use the already detected lane as a guide in the search for specific objects such as arrows. The work presented in [7] uses the hypothesis that the arrows are in the middle of the lane, and therefore restrict their possible locations. The detection and classification of arrows is done by pattern matching on inverse perspective mapped images.

The purpose of this paper is to describe a system that uses the powerful capabilities of LabVIEW for real-time automated

recognition of the most relevant road markings. The LabVIEW functions of geometric pattern matching, edge detection and calibration information are combined with custom C++ modules, and a robust, real-timer interactive system is produced.

II. OBJECTIVES

On European roads there are multiple types of markings like arrows, lane markings and delimiters, from which we took the most common ones and created template images for the pattern matching functions. It is important not only to detect and classify the painted road objects, but also to provide information about their position, so we created a separate application that sets calibration information for the images.

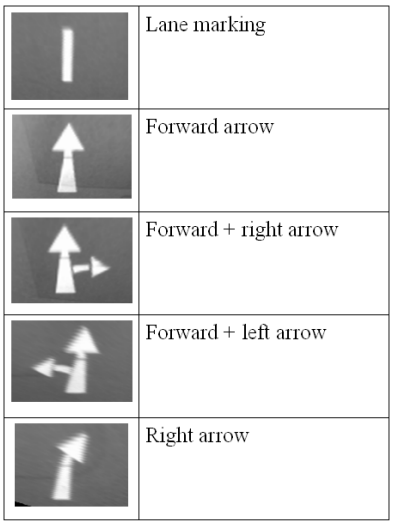

Figure 1. Painted road objects to be detected

Our aim is to develop a system that will reliably detect, measure and classify the types of objects shown in figure 1 in the shortest time possible. For a better view of the road and more accurate results the image of the road is transformed from perspective to inverse perspective. We used two different types of geometric pattern matching to detect the arrows and the lane markings as their geometry is different. For the arrows we used feature based geometric pattern matching and for the lane markings we used edge based geometric pattern matching. To detect the continuous lane marking we used simple edge detection. The result is validated using information about the position of the objects in real world coordinates.

The main steps of the application are: calibration of the vision system, computation of the necessary parameters for inverse perspective transformation, selection of the features for the template images and selection of the parameters for the matching process, object detection and classification.

Figure 2. Scene with painted road objects.

III. LABVIEW

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) is a graphical programming language that uses icons instead of lines of text to create applications. In contrast to text-based programming languages, where instructions determine the order of program execution, LabVIEW uses dataflow programming, where the flow of data through the nodes on the block diagram determines the execution order of the VIs and functions.

LabVIEW programs are called virtual instruments, or VIs, because their appearance and operation imitate physical instruments. LabVIEW contains a comprehensive set of tools for acquiring, analyzing, displaying, and storing data, as well as tools to help the user troubleshoot the code.

In LabVIEW, we develop a user interface, or front panel, with controls and indicators. Controls are knobs, push buttons, dials, and other input mechanisms. Indicators are graphs, LEDs, and other output displays. After we build the user interface, we add code using VIs and structures to control the front panel objects. The block diagram contains this code.

IV. CALIBRATION

The output of the application will provide information about real world location of the detected objects. In order to have information about the position in real word coordinates of the painted road markings, the image needs to contain calibration information. Also, the inverse perspective mapping function requires a look-up table with information about the corresponding pixel values of the real world coordinates to be projected in the destination image.

To set the calibration information for an image we used LabVIEW IMAQ Learn Calibration Template VI. This function described in [10] calibrates an image by defining a Reference Coordinate System for the real world coordinates, a number of 10 reference points - pixel coordinates and their real world correspondents - and the type of distortion to be corrected (perspective, nonlinear, simple calibration, corrected).

For the computation of the look-up table we had to define the limits of the 3D space – on X and Z axis– where the search will be made. Because we refer to the road surface, we considered Y=0, X as the sideways coordinate and Z is the distance. After defining *XMIN, XMAX, ZMIN, ZMAX* the user can choose between a LabVIEW mapping functions or a custom built function based on a predetermined projection matrix, to compute the look-up table. Each pixel coordinate i,j in the destination image is related to a point in the 3D world, by equations 1 and 2.

$$X = XMIN + \frac{XMAX - XMIN}{Width} * j \tag{1}$$

$$Z = ZMIN + \frac{ZMAX - ZMIN}{Height} * (Height - i - 1) \tag{2}$$

$$\begin{bmatrix} uw \\ vw \\ w \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (3)$$

$$u = \frac{uw}{w} \quad v = \frac{vw}{w} \quad (4)$$

The values u and v are the coordinates from the perspective image and **P** is the projection matrix, computed from the camera's intrinsic and extrinsic parameters.

The LabVIEW program uses IMAQ Convert Real World to Pixel function, which transforms real-world coordinates X and Z, computed from equations (1) and (2) - to pixel coordinates, according to the calibration information acquired from the IMAQ Learn Calibration Template function.

V. Setup Features For Geometric Matching Template

In order to identify the lane markings and the arrows we used geometric pattern matching functions from LabVIEW Vision, which require template images as input. Geometric matching helps us to quickly locate objects with good geometric information in an inspection image. For this purpose we defined a template image for each road painted object. Because the geometry of the lane marking and the arrow is different, we used two types of geometric pattern template: feature based and edge based [8]. The template image is saved as a *.png* image along with the selected parameters and the geometric features extracted.

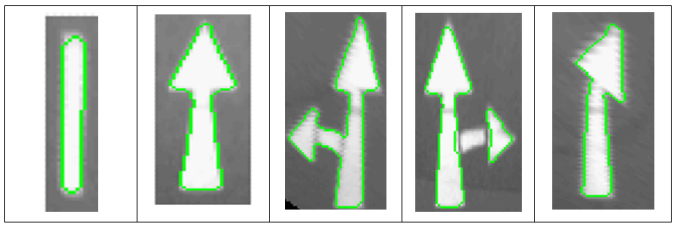

Figure 3. Geometric Features extracted from template images

First thing we need to do after defining the patterns is to set the parameters that are used during the matching process. For this we used existing functions from LabVIEW Vision: IMAQ Setup Match Geometric Pattern for the arrows, and IMAQ Setup Match Geometric Pattern 2, IMAQ Advanced Setup Match Geometric Pattern 2 for the lane markings [9]. These functions allow us to define how curves are extracted from the inspection image during the match phase and the conditions under which we want to find the template matches. IMAQ Advanced Setup Match Geometric Pattern 2 function provides ways to optimize and fine-tune parameters used during the matching phase and how curves are extracted from the image.

VI. Road Markings Detection

Figure 4 illustrates the complete processing of a frame from acquisition to painted road objects detection and computation of real world coordinates. The circles represent steps of the processing and the rectangles represent information stored during processing.

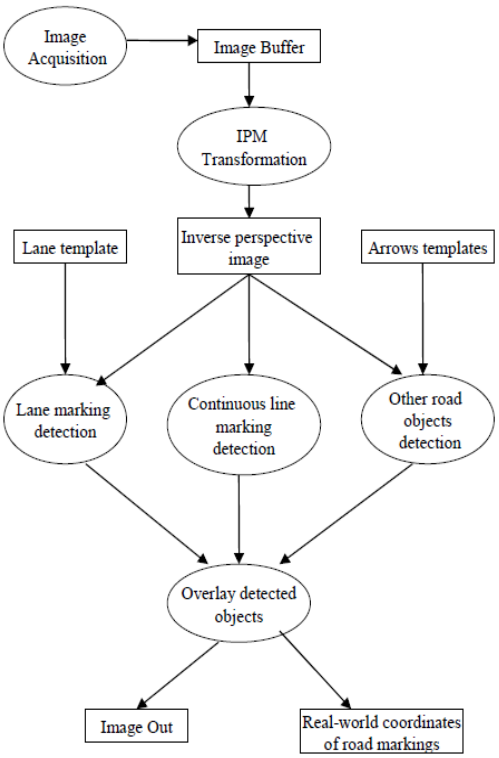

Figure 4. Flow of events for road markings detection

The LabVIEW top level diagram for the road objects recognition system, implementing the flow described in figure 4, is shown in figure 5.

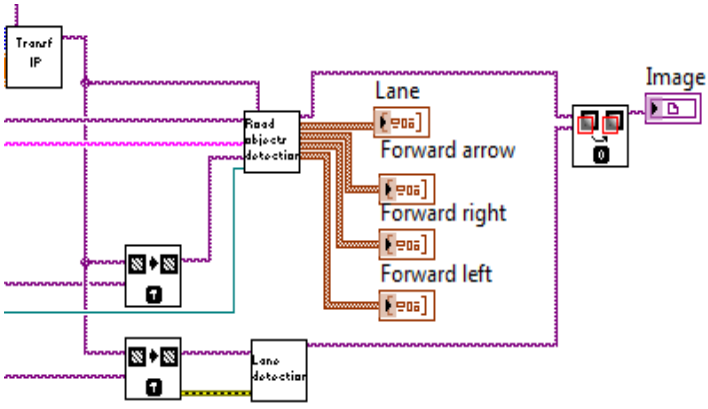

Figure 5. LabVIEW program for road markings detection

A. Inverse Perspective Mapping

Inverse perspective mapping is a method that takes the source image, acquired by a video camera, and transforms it into a top view of the same scene. For this algorithm to be applied the scene has to contain the road plane, and the

relationship between the camera and this plane has to be known. The look-up table previously computed (using equations 1-4) is used to accomplish this.

Having the values of the perspective coordinates *u* and *v* computed at calibration step, the value of the intensity of the source image will be inserted in the destination image at the position described by the coordinates *i* and *j* coordinates. This step will be executed for all the pixels in the destination image.

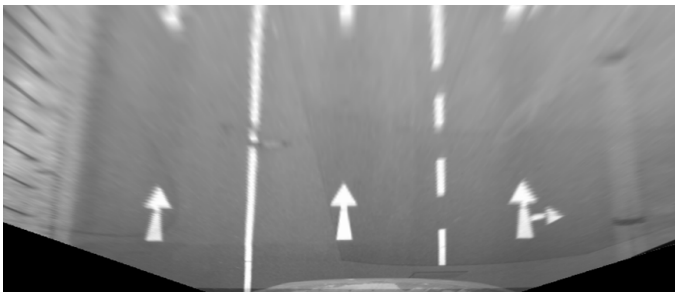

Figure 6. Inverse perspective image

B. *Geometric Pattern Matching*

We use geometric pattern matching to quickly locate road painted objects within a gray scale image because they have good geometric information. Another reason to choose this method was that the template matches are found regardless of lighting variation, blur, noise, occlusion, and geometric transformations such as shifting, rotation, or scaling of the template. Geometric matching also provides information about the number of matches found, their position in pixels and real-world coordinates (if the image contains calibration information), the percentage change in size (scale) of each match and the amount by which each object in the match is occluded.

Because the geometry of the lane markings and arrows is different we used two types of geometric matching: edge based for the lanes and feature based for the arrows. Both techniques are based on curves extracted from the image. The difference is in the way the curve information is used to perform the matching. The edge-based geometric matching method computes the gradient value of the edge at each point along the curves found in the image and uses the gradient value and the position of the point from the center of the template to perform the matching. The feature based geometric matching method extracts geometric features from the curves and uses these geometric features to perform the matching.

Curves are extracted from the image using the algorithm described in [8] at Curve Extraction. Advanced parameters can be set using IMAQ Advanced Setup Match Geometric Pattern 2 LabVIEW function. Setting these parameters improves the processing time by 2-3 ms per frame.

1) *Detection of Lane Markings*

The lane marking is the most commonly encountered road object. In order to detect the lane markings, we use an edge-based geometric pattern matching technique, as this method is guaranteed to find the object in the inspection image as long as a significant portion of the shape remains similar to the shape of the template object. This technique utilizes the generalized Hough transform method for matching and has two stages: a learning stage and a matching stage. The learning stage is described in the Setup Features for Geometric Template chapter. The matching stage consists of three steps. The first step is edge point extraction and the other two steps are generalized Hough matching and match refinement [8].

Despite the fact that an edge based geometric template uses more memory than a feature-based geometric template, there is a significant improvement in classification performance.

After we classify an object as a lane marking we validate it and then overlay its bounding box on the resulting image. Real-world coordinates of the detected lane are computed using equations (1) and (2).

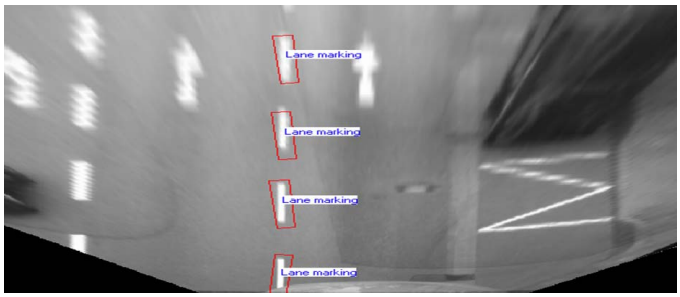

Figure 7. Detected lane markings

2) *Detection of Arrows*

For detecting more complex road painted objects we use a feature-based geometric matching technique. In the learning stage we used IMAQ Learn Multiple Geometric Patterns VI, which combines the descriptions of the patterns we want to search for during the matching phase into a multiple geometric template and uses the multiple geometric template to search for these template images in the target image.

After the road objects are detected, they need to be validated in order to increase the reliability of the application. The validation is made by comparing the coordinates of the objects position. If two distinct template matches overlay, the most specific one is retained – a forward right arrow will often be classified as a forward arrow, whereas a forward arrow will never be classified as a forward right, forward left, right or left arrow. After validation, real-world coordinates are computed from pixel coordinates using equations (1) and (2).

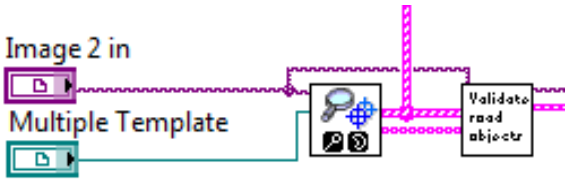

Figure 8. Multiple Geometric Pattern Matching followed by object validation.

C. *Detection of Continuous Lane Markings*

The continuous lane delimiters are not discrete objects, and therefore they cannot be identified by the pattern matching techniques. To detect the continuous lane markings we use

LabVIEW's *Find Edge* function – which finds straight edges (lines) within a region in an image. This function allows us to define a region of interest, parameters for the edge detection algorithm and the information that is overlaid on the result image like search direction, edge polarity, kernel size, minimum edge strength, interpolation type. Also we can set line fit options, which specify the options used to detect the straight edges, like the number of lines to find and the method to use (we choose Hough Edge Rake [11]).

Hough-Based methods use a Hough transform, which is a standard technique used in image analysis to find curves that can be parameterized such as straight lines. It will identify a continuous line even if the image has noise or parts of the road object are erased. This is an important advantage as objects are often occluded by the vehicles on the road or are worn out. The processing time can be improved by setting the step size (the gap in pixels between the search lines) as high as possible and also by specifying the minimum number of points as a percentage of the number of search lines that need to be included in the detected straight edge.

In order to identify the straight edges, the function looks for dark-light-dark transition patterns in a gray scale image. In the image below, the green rectangle shows the region of interest defined, the blue lines represent the search lines and the red lines represent the found edges.

The results given by the Edge Detection function are in pixel coordinates and they are transformed in real word coordinates using the calibration information. In the next step the continuous line is validated if the distance between two detected lines is in the range of 12 to 20 cm.

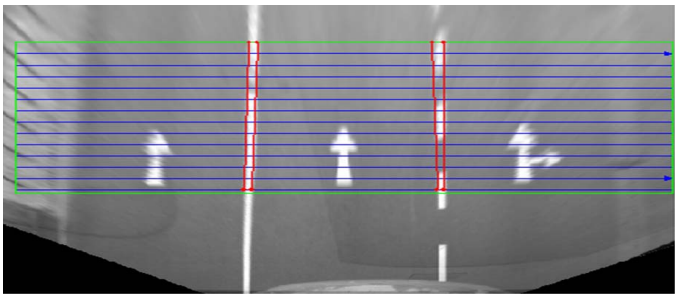

Figure 9. Edge detection results.

In figure 9 we see the results of edge detection. Sometimes the interrupted lane border causes similar edge response, when the lane delimiting objects are close together. However, it is no danger to the final results, as the system will also detect each object individually, and the results will be thus corrected.

VII. RESULTS

For testing our approach, different image sequences were evaluated which contained country roads in different visibility conditions. Because the image is transformed from perspective to inverse perspective the painted road objects are detected at the maximum distance of 23 m ($ZMAX = 25$ m). The maximum distance is dependent on the maximum values chosen for inverse perspective matching algorithm.

It is difficult to devise an objective metric for the classification performance. The number of correctly detected and classified objects is highly dependent on the quality of the road markings. If the road mark is worn out it won't be detected at all or it will be erroneously classified as road marking. The same happens when the object is entering or exiting the image frame, as it is not completely visible. Furthermore, the reliability of the results depends also on the template images used for pattern matching and on the features selected for them. If we compare the total number of occurrences of an object with the number of correctly classified instances, the results are convincing, the correct rate of classification being about 85%. The percentage increases to 95% if we consider a maximum distance of 13 m, where the road markings are most visible.

The complete acquisition and processing of a single frame takes less than 35 ms, thus allowing the processing of about 28 frames per second. Further increase in processing speed can be obtained if the system is deployed on a real-time device, such as the National Instruments PXI, which can act as a standalone driving assistance module, capable of interconnecting to the vehicle CAN bus and to other data sources.

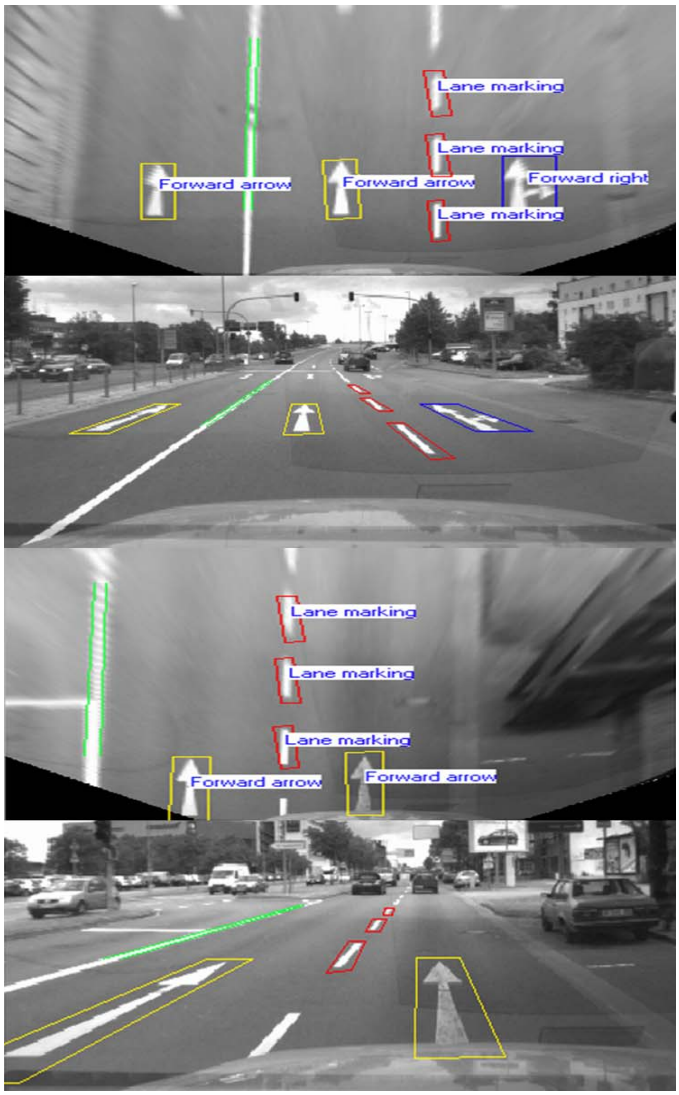

Figure 10. Results – Correctly classified objects.

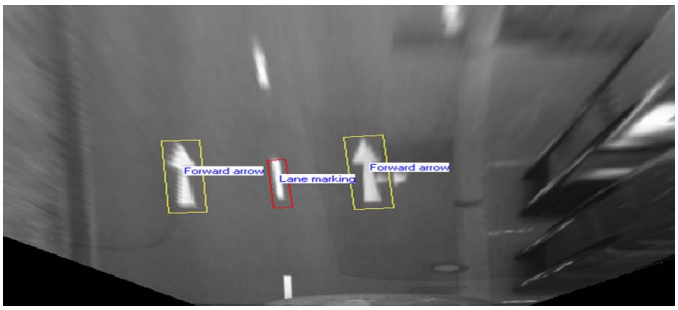

Figure 11. Results - Objects misclassified; forward right arrow misclassified as forward arrow due to distance and reduced visibility.

VIII. Conclusion and Future Work

A robust detection of road markings via on-board camera becomes more and more important for autonomous vehicle guidance. This paper presents an application developed in LabVIEW for detection and classification of five of the most common road markings. The system provides a real time objects detection in urban traffic. The advantage of the application is that it doesn't require third party pre-processing of the image. The acquired image is transformed in inverse perspective using previously computed parameters (in the set calibration information part of the application), and then the template images are used for road markings detection and classification.

The application can be optimized in multiple ways for detecting more accurate the road markings in more generic urban traffic scene. One optimization would be to use more template images for one road marking in the learning phase of geometric pattern matching. The template images can represent the object as it enters the image frame, when it is less clear. Another improvement is to combine the results from consecutive frames to preserve the class consistency of the detected road painted objects.

IX. Acknowledgment

This work was supported by CNCSIS –UEFISCSU, project number PNII – IDEI 1522/2008, and by the POSDRU program, financing contract POSDRU/89/1.5/S/62557.

References

[1] J. C. McCall and M. M. Trivedi, “Video based lane estimation and tracking for driver assistance: Survey, system, and evaluation”, IEEE Transactions on Intelligent Transportation Systems, Vol. 7, No. 1, pp. 20-37, 2005.

[2] T. Veit, J. P. Tarel, P. Nicolle and P. Charbonier, “Evaluation of Road Marking Feature Extraction”, in proc of IEEE Conference on Intelligent Transportation Systems 2008 (ITSC 08), pp. 174-181.

[3] J. P. Gonzalez and U. Ozguner, “Lane Detection Using Histogram-Based Segmentation and Decision Trees”, in proc of IEEE Intelligent Transportation Systems Conference 2000 (ITSC 2000), pp. 346-351.

[4] F. Paetzold and U. Franke, “Road Recognition in Urban Environment”, in proc of IEEE International Conference on Intelligent Vehicles, 1998, pp. 87-91.

[5] C. Duchow, “A novel, signal model based approach to lane detection for use in intersection assistance”, in proc of IEEE Intelligent Transportation Systems Conference 2006 (ITSC 2006), pp. 1162-1167.

[6] U. Franke, D. Gavrila and S. Goerzig, “Vision based Driver Assistance in Urban Traffic”, in proc of World Congress on Intelligent Transportation Systems (ITS), 2000.

[7] S. Vacek, C. Schimmel and R. Dillman, “Road-marking analysis for autonomous vehicle guidance”, in proc of European Conference on Mobile Robots, 2007.

[8] “Geometric Matching Techniques”, chapter of “NI Vision Concepts Help”, online resource at http://digital.ni.com/manuals.nsf/websearch/16712B9E3D2179698625776E0057700D

[9] “Searching and Matching VIs”, chapter of “NI Vision for LabVIEW Help”, online resource at http://digital.ni.com/manuals.nsf/websearch/161A3B0C9C130E758625776E0057C86E .

[10] “System Setup and Calibration, chapter of “NI Vision Concepts Help”, online resource at http://digital.ni.com/manuals.nsf/websearch/16712B9E3D2179698625776E0057700D.

[11] “Edge Detection”, chapter of “NI Vision Concepts Help”, online resource at http://digital.ni.com/manuals.nsf/websearch/16712B9E3D2179698625776E0057700D .