Speed estimation using SVM classification and edge detection

Mathias Dierickx, Tim Ranson, Maarten Tindemans, and Bjorn Vandenbussche

Ghent University, Faculty of Engineering and Architecture
Valentin Vaerwyckweg 1, 9000, Ghent

Abstract. This is the abstract: TODO

Keywords: SVM classification, Local Binary Patterns, line detection

1 Introduction

TODO

2 Background

2.1 Texture analysis using Local Binary Patterns

Local Binary Patterns (LBP) [5] is a non-parametric descriptor whose aim is to efficiently summarize the local structures of images. As a non-parametric method, LBP summarizes local structures of images efficiently by comparing each pixel with its neighboring pixels. The most important properties of LBP are its tolerance regarding monotonic illumination changes and its computational simplicity. LBP was originally proposed for texture analysis [6], and has proved a simple yet powerful approach to describe local structures.

The original LBP operator labels the pixels of an image with decimal numbers, called Local Binary Patterns or LBP codes, which encode the local structure around each pixel (see figure 1). Each pixel is then compared with its eight neighbors in a 3x3 neighborhood by subtracting the center pixel value. The resulting strictly negative values are encoded with 0 and the others with 1. A binary number is obtained by concatenating all these binary codes in a clockwise direction starting from the top-left one and its corresponding decimal value is used for labeling. The derived binary numbers are referred to as Local Binary Patterns or LBP codes.

One limitation of the basic LBP operator is that its small 3x3 neighborhood cannot capture dominant features with large scale structures. To deal with the texture at different scales, the operator was later generalized to use neighborhoods of different sizes [7]. The histogram of LBP labels can then be exploited as a texture descriptor.
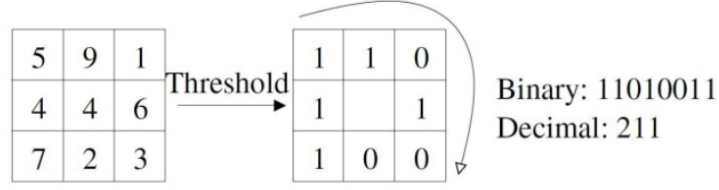
Fig. 1: An example of the basic LBP operator

2.2 SVM classification

Support Vector Machines (SVM's) [3] are a learning method used for binary classification. The basic idea is to find a hyperplane which separates the d-dimensional data perfectly into its two classes.

TODO: Tim

2.3 Precision, recall, F-scores

TODO: Bjorn

2.4 Line detection

TODO: Mathias

3 Method

In order to determine the maximum speed of a vehicle two image processing techniques are used: SVM classification and edge detection. We first try to rely on edge detection to find the contours of the lanes where driving is permitted (i.e. where no obstacles are found). We estimate the reliability of the edge detection and incorporate road detection using the SVM classifier when needed.

For the implementation of our method, we use OpenCV 3.1 [4] and C++.

3.1 Parameters for SVM classification

The goal of this classifier is to determine what pixels are part of the road and which are not. We will describe multiple tunable parameters in order to get optimal classification for road detection. These parameters determine both the length and values of the feature vectors, which are then fed to the SVM classifier. We compare classification results varying these parameters for the feature vector in section 4.1.

Block size Each frame is split into multiple blocks of predetermined size. We chose to vary the block sizes between 8x8px, 16x16px and 32x32px. On the one hand, larger block sizes would too easily contain multiple structures of different classes, resulting in poorer classification. For example a block on the edge of the road containing part of the street and grass. On the other hand, smaller block sizes would omit too many details to be descriptive enough to use for classification.

The block size affects the level of detail taken into account while calculating the feature vector, as the values for the feature vector operate within one block size.

TODO: insert photos block sizes

LBP With LBP we try to characterize the structure of roads within a block. The histogram of LBP values within that block can be added to the feature vector in order to improve classification. These histograms are calculated per channel (e.g. 3 channels for RGB) and are added consecutively to the feature vector.

Color Because working with LBP values only doesn't take color intensity into account [8], histogram values of color intensity per block and channel are added to the feature vector.

3.2   Kernels for SVM classification

Judging from the OpenCV documentation[1] there are two SVM kernels fit for a two-class classification problem.

TODO: Tim

3.3   Road marks

The training dataset was labeled considering blocks containing road marks once as part of the road class and once as part of the non-road class. Initially road marks were labeled as non-road, in order to get a good classification of road (i.e. different types of asphalt). As these road marks aren't really considered an obstacle for a car driver, ideally they should be classified as road. This makes up for a more difficult classification problem. To quantify this effect we apply classification both with and without road marks.

3.4   Removing details

Detecting edges is very straightforward using an edge detection algorithm like Canny Edge Detection [1]. However, there are a lot of edges on the road itself, thinking of road markings and cobbles. The edges on the road, considered as

---

[1] http://docs.opencv.org/3.1.0/

noise, and the edges of the road are not unambiguously separable. The width of the patterns that cause the edges on the road is rather small. Using morphological image processing operations, it is possible to remove these small patterns. In figure 2b, a smoothed cobbled road is shown.

(a) Original frame   (b) Eroded cobbled road

Fig. 2: Perform eroding to remove the edges on the road.

Eroding is a morphology operator to make the objects on the foreground, which are the brightest, smaller [2]. Figure 4b shows an example of a thinned white line. When the kernel is large enough, white lines can be filtered out completely. The same technique is used to smooth the surface of cobbled roads. A cobblestone consist of a bright center, surrounded by darker joints. By eroding the cobbled road, the erode function will minimize the center of the cobble, which will cause enlarged joints. With a large kernel, the joints will spread until they overlap. The best results were achieved with an ellipsoidal kernel of 100 x 30 px.
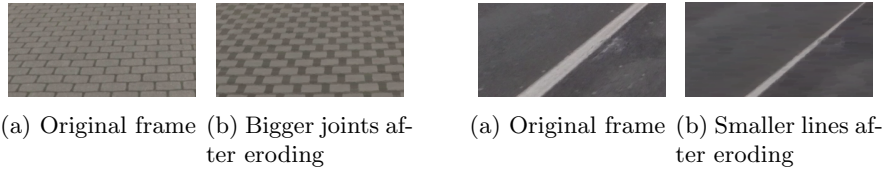

(a) Original frame (b) Bigger joints af-
                    ter eroding

Fig. 3: The effect of eroding cobbled roads.

(a) Original frame (b) Smaller lines af-
                    ter eroding

Fig. 4: The effect of eroding road markings.

Unfortunately, the erode function will cause the darker objects to expand. The original size of the objects has been modified, as seen in 5b. This will falsify the eventual edge detection. In this example, the observed car is reported to close. In order to restore the original measures, dilation is executed. This is the opposite morphology operator of eroding. The combination of eroding and dilation is called opening. When dilation is executed with the same kernel size of the erode function, the original size of the dark objects will decrease to their original size, as seen in 5c. Note that, after eroding, some details of the shape of the objects are lost. In order to provide a safety margin, the dilate function is executed with a slightly smaller kernel.

(a) Original frame
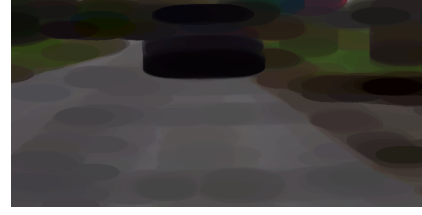
(b) Dark objects are bigger after eroding

(c) Restore original sizes with dilation

Fig. 5: Perform eroding and dilation to remove details.

3.5 Canny Edge Detection

To detect the edges of the road, the Canny Edge Detection algorithm is used. Figure 6 shows an example.

Fig. 6: Canny edge detection

This algorithm requires a minimum and a maximum threshold value, which were determined experimentally. The best results were achieved with a maximum threshold that is twice the minimum threshold. The minimum threshold is determined by the required minimum sensitivity. When the algorithm is configured too sensitive, some imperfections on the road will be detected as edges. On the other hand, a certain level of sensitivity is necessary to detect less clear road edges. For example, a transition from road to dirt has a very small distinction of color and brightness, as seen in 7b. The minimum threshold should be high enough to detect this transition as edge.
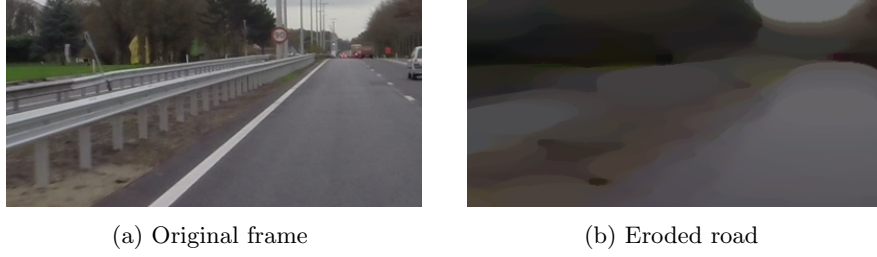
(a) Original frame   (b) Eroded road

Fig. 7: Unclear transition from road to non-road.

3.6   Combination of classification and edge detection

TODO: Maarten

4   Results

4.1   SVM classification

We iterated through several parameter values (as discussed in 3.1) to find optimal classification for road detection. For each combination of parameters we train and test on different combinations of datasets. For each combination we calculate the F-scores. These scores are used to compare the classification results (higher F-scores indicate better classification).

Of the four datasets, different combinations of training sets were selected (each time shown as x-axis on the graphs). The trained classifier was then tested on the remaining datasets.

Kernels As previously mentioned in section 3.2, there are two SVM kernels we can use for this classification problem. On average, we tend to get a nearly 11% increase in F-scores using RBF kernel instead of linear kernel. It is safe to say the RBF kernel outperforms the linear kernel for this classification problem in all of our test scenarios.

Block size In general, block sizes 16x16px and 32x32px tend to show the better results. As an example we present the F-scores for LBP and color with road marks in figure 8. Other combinations follow the same trend. For the goal of this application, a smaller block size will be more practical (i.e. better accuracy). This is why prefer the block size of 16x16px.

LBP The use of LBP is crucial to our implementation in order to get a viable classifier for road detection. If we compare results (block size 16x16px) of feature vectors on LBP only to the combination of LBP and color, we get different results depending on whether or not we include road marks.
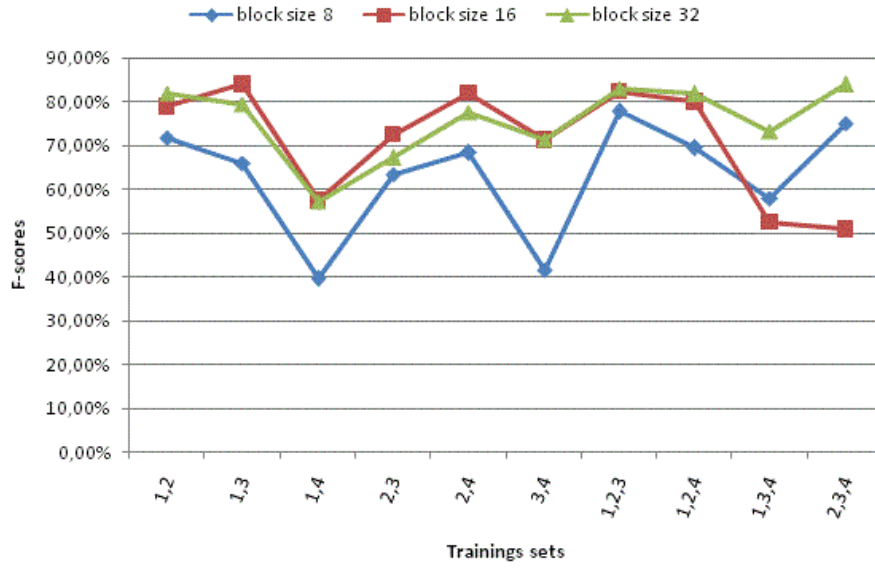
Fig. 8: F-scores for LBP and color with road marks using RBF kernel

If we do not include road marks in the training data, we get slightly better results when combining LBP and color for the feature vectors (as shown in figure 9). If on the other hand we do include these road marks, using only LBP shows equal or better results than the combination of LBP and color (as shown in figure 10).

A possible explanation for this phenomenon is that when only asphalt is considered to be road, color is a much more consistent factor to weigh in to the classification. However when including road marks as part of the road, color intensity is no longer an added value and even tends to mislead the classifier.

Overall we can conclude that using only LBP with road marks shows the optimal results for our classification problem.
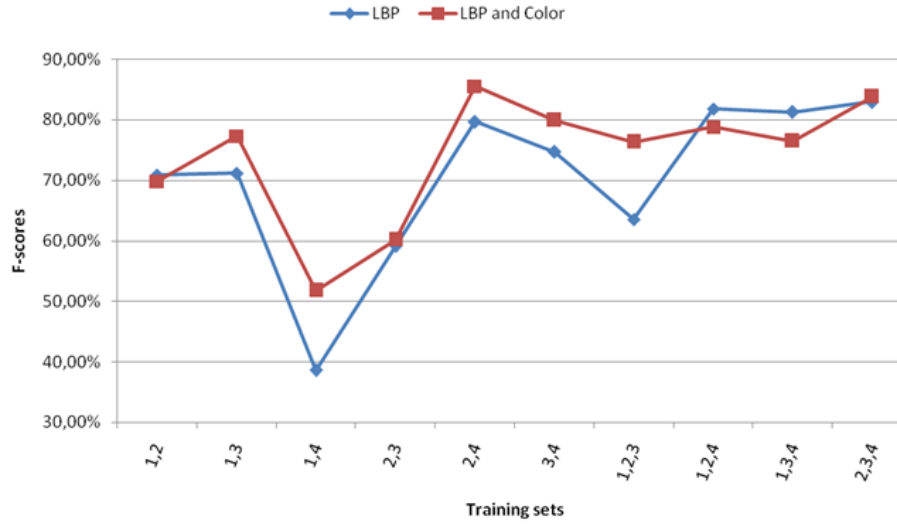
Fig. 9: Comparision of F-scores for 16x16px block size without road marks using RBF kernel

In retrospect, the optimal parameters for our classification problem are:

- RBF kernel
- Block size 16x16px
- Using only LBP histogram values for the feature vectors
- Including road marks as part of the road
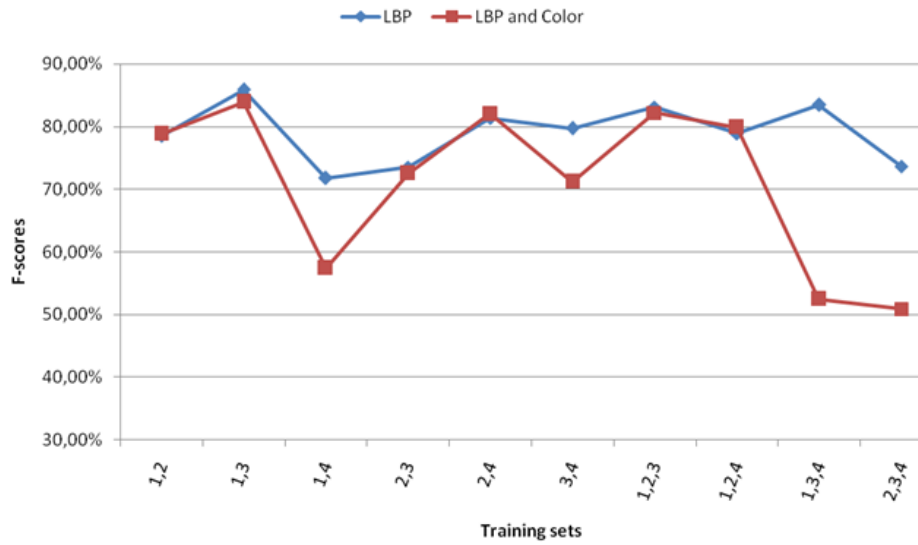
4.2   Edge detection

5   Conclusion

Fig. 10: Comparision of F-scores for 16x16px block size with road marks using RBF kernel

References

1. Canny edge detection OpenCV 3.1.0 documentation.
2. Eroding and dilating OpenCV 3.1.0 documentation.
3. Dustin Boswell. Introduction to Support Vector Machines. 2002.
4. G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
5. Di Huang, Caifeng Shan, Mohsen Ardabilian, Yunhong Wang, and Liming Chen. Local Binary Patterns and Its Application to Facial Image Analysis: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 41(6):765–781, nov 2011.
6. Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture analysis algorithms in textile inspection applications. *Pattern recognition*, 29(1):51–59, 1996.
7. Timo Ojala, Matti Pietikainen, and Topi Mäenpää. Multiresolution Gray Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Trans. Pattern Analysis and Machine Intelligence,*, 24(7):971–987, 2002.
8. M. Pietikainen, T. Maenpaa, and Jaakko Viertola. Color texture classification with color histograms and local binary patterns. *Workshop on Texture Analysis in Machine Vision*, pages 109–112, 2002.