Online Self-supervised Learning for Road Detection

Marco L.L. Rompen, Willem P. Sanberg, Gijs Dubbelman, Peter H.N. de With
Eindhoven University of Technology
SPS/VCA, Department of Electrical Engineering
Eindhoven, Netherlands
m.l.l.rompen@student.tue.nl, {w.p.sanberg, g.dubbelman, p.h.n.de.with}@tue.nl

Abstract

We present a computer vision system for intelligent vehicles that distinguishes obstacles from roads by exploring online and self-supervised learning. It uses geometric information, derived from stereo-based obstacle detection, to obtain weak training labels for an SVM classifier. Subsequently, the SVM improves the road detection result by classifying image regions on basis of appearance information. In this work, we experimentally evaluate different image features to model road and obstacle appearances. It is shown that using both geometric information and Hue-Saturation appearance information improves the road detection task.

1 INTRODUCTION

Vehicles are relying increasingly on computer vision to ensure a safer way of driving. Examples are Advanced Driver Assistance Systems (ADAS), such as lane departure warning and pedestrian detection. These and related technologies will be improved and extended in the future, leading to autonomously driving vehicles. Such intelligent vehicles have the potential to significantly reduce traffic congestions, accidents, and pollution levels.

In this work, we focus on computer vision methods necessary to distinguish obstacles from drivable surfaces. Our approach is designed to work with asphalt roads, paved roads, country roads, or any other drivable terrain. This is enabled by online and self-supervised learning of the visual appearance of drivable surfaces versus obstacles. In contrast to [2], it does not depend on human-annotated train images, and is fully adaptive to different road surface types, environments, weather and lighting conditions. An obstacle detection system based on a stereo camera [1] provides a basis for reliably separating obstacles from roads by exploring a weak labeling, which is then improved by a Support Vector Machine (SVM) classification using color features. A similar approach was used in [3], where the focus was purely on outdoor terrain in which the differences between drivable surfaces and obstacles are typically more pronounced than in urban (gray world) environments, which is our main focus. In [4] also a similar approach was used, building on the assumptions that the road in front of the vehicle is always drivable, has uniform appearance and differs significantly from obstacles. By combining [3] and [4] and improving it with [1] a system is created applicable to the urban environment.

The complete system is described in Section II and III. The results of our experiments are provided in Section IV and finally Section V brings forward our conclusions and recommendations.

2 SYSTEM OVERVIEW

Our vision-based drivable surface (i.e. road) detection system with its key processing blocks is depicted in Fig. 1. Its general concepts are discussed here. Detailed descriptions of its key processing blocks are provided in Section III.

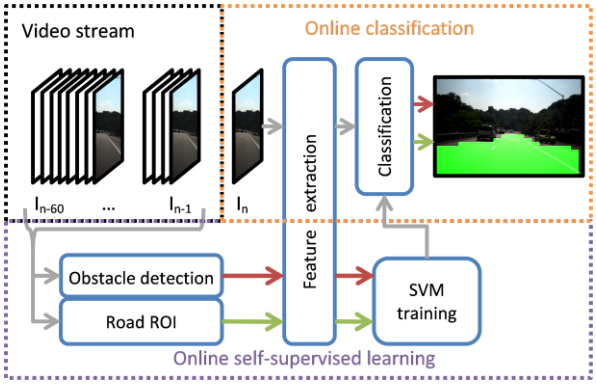


Fig. 1. Overview of the integrated road and ostacle detection system.

The input to the system is a continuous binocular video stream. It keeps a buffer of the past 60 frames $I_{n-60},\ldots,I_{n-1}$ to online learn an appearance model that is used to classify pixels in the current frame I_n as being part of a drivable surface or part of an obstacle.

The classifier used by the system is a two-class SVM. It requires a set of training examples to learn a decision surface separating the training samples of the two classes {road, obstacles}. This decision surface is then used to classify new samples. In contrast to the typical usage of SVM, where training samples are obtained by manually annotating images, in our system the training samples are automatically obtained from the stereo-based *obstacle detection* processing block.

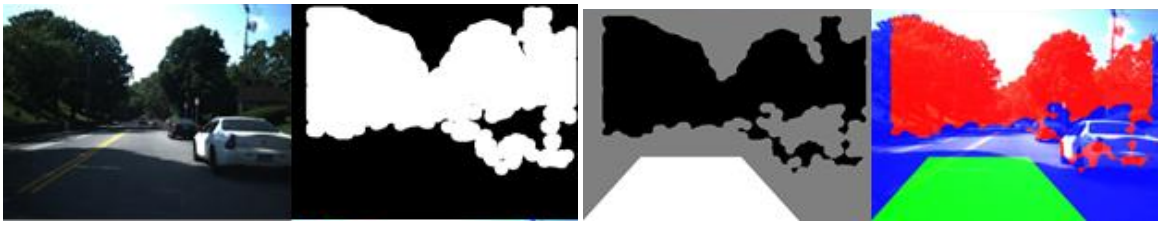
This stereo subsystem estimates disparity maps from which the depth of pixels can be obtained. From this geometric information, the local slope of surfaces is calculated. If the slope exceeds a certain threshold, pixels are labeled as obstacles [1]. As stereo-based depth estimation suffers from many artifacts, the obstacle labeling is not perfect. Therefore, we use these (weak) labels, automatically obtained from stereo geometric information, to train an SVM on appearance-based information. The appearance model for pixels is a histogram representing the color or gradient distribution in a local rectangular area around each pixel. In this work different color spaces and histogram creation strategies are experimentally evaluated.

3 ROAD DETECTION WITH SELF-SUPERVISED LEARNING

The online learning and classification pipeline starts with generating training masks, using the *obstacle detection* and *road Region Of Interest* (ROI) processing blocks, as depicted in Fig. 1. The next step is *feature extractio*n followed by online *SVM training* and finally *classification*. These four key steps in the system are described in detail below.

3.1 Training mask

Training masks assign labels {*road, obstacle, unknown*} to pixels to facilitate creating training examples for the online self-supervised SVM classifier. The training mask creation exists of two processing blocks, *obstacle detection* and *road ROI*, which are combined into one mask for every train frame. The obstacle map provided by the *obstacle detection* processing block is eroded to reduce the number of pixels that are wrongly labeled as obstacles. The *road ROI* is based on the assumption that the surface directly in front of the car is always drivable, unless the obstacle detector determines otherwise. All parts not specifically designated as obstacle or road are given the *unknown* label. In Fig. 2 an example frame with its obstacle map and its training mask is depicted.



(a) (b) (c) (d)

Fig. 2. Example of training mask generation. (a) The rectified left image of the stero input frame. (b) The obstacle map (white is obstacle and black is non-obstacle) generated by the *obstacle detection* processing block. (c) The training mask containing the weak labels. In this mask black is a weak obstacle label, white is a weak road label, and gray is unknown. (d) Training mask as an overlay on the rectified left image.

3.2 Feature extraction

The *feature extraction* processing block has two functions. Its first function is to provide unlabeled training samples for the *SVM training* processing block. The labels for the training samples are obtained from the training masks. Next to that, it provides features for the road *classification* processing block.

The *feature extraction* processing block divides frames in (non-overlapping) square blocks of 17-by-17 pixels. For each of these blocks it computes a histogram, i.e. the image feature, as a representation of their appearance. In this work, we consider different histogram content and creation methods.

In order to fill the histograms we need to assign an Index (I) and Magnitude (M) to each pixel on basis of its (multidimensional) value. The index (I) is used to determine the corresponding bin (B) of the pixel in the histogram, which can be a non-linear mapping. The Magnitude (M) of a pixel is its contribution to its corresponding bin in the histogram. For color-based histograms, the value of a pixel is used as the Index (I) and the Magnitude (M) is defined as 1. For histograms of gradients (HOG), the orientation of the gradient is used as the Index (I) and the Magnitude (M) of a pixel is the magnitude of its gradient.

A histogram can be created by mapping each feature space dimension to a histogram dimension, thereby forming a multidimensional histogram. An alternative is to concatenate all feature space dimensions into one histogram dimension, thereby forming a 1-D histogram. The potential benefit of a multidimensional histogram is that it can capture correlations between feature space dimensions. In a 1-D histogram all individual feature spaces are modeled as being independent of each other. With this we gain computational

efficiency, but we potentially lose valuable statistical information. In this work we experiment with both approaches.

Another aspect of histogram creation that we evaluate is the mapping from pixel indexes (I) to histogram bins (B). For this we compare a simple linear mapping with an adaptive data dependent method. The adaptive method can be seen as an offline dimension reduction strategy [6]. Its aim is to only have meaningful histogram bins. This is performed by taking random pixels out of the dataset and determining the bin boundaries such that every bin has an equal (non-zero) number of pixels in it.

Finally, the histograms are normalized. For 1-D histograms that are made up of multiple feature space dimensions, the separate histograms of each feature dimension are first normalized separately, after which the complete histogram is normalized. This ensures that training of the SVM classifier starts with a situation in which each independent feature space is equally weighted. An overview of all histogram based feature spaces that we use for our experiments are provided in Table I. They are a combination of gradient histograms and histograms over the color spaces: RGB, HSV, YIQ (NTSC) and subspaces of these.

TABLE I LIST OF TESTED FEATURE COMBINATIONS

Name	Number of feature space dimensions	Bins per feature space dimension	Histogram dimensionality	Total number of bins
HS100	2	10	2	100
HS100	2	50	1	100
HS144	2	12	2	144
HS128	2	64	1	128
HSV96	3	32	1	96
HS-HOG96	3	32	1	96
HSV216	3	6	3	216
YIQ216	3	6	3	216
RGB216	3	6	3	216
IQ144	2	12	2	144
RGB96	3	32	1	96

3.3 Training

The goal of the *training* processing block is to train a SVM, online and self-supervised, that can be used to classify pixels as being part of a road or an obstacle. A training sample consists of a feature and a label. Its feature is a histogram computed over a 17-by-17 pixel block and its label is the value {*road, obstacle, unknown*} of the center pixel of the block in the automatically obtained training mask. Blocks for which their label is unknown are ignored during training.

The goal of SVM is to fit a hyper plane in such a way that two classes {*road, obstacle*} are maximally separated by the hyper plane [7]. In this work we experiment with a linear SVM and a Radial Basis Function (RBF-) SVM. The linear-SVM fits a hyper plane directly in the feature space. Its benefit is that it is more computational efficient than RBF-SVM but it can only model linear decision boundaries. Such linear decision boundaries are not appropriate for every classification task; some tasks require non-linear decision boundaries. An RBF-SVM implicitly maps the feature space to a higher dimensional feature space such that a non-linear decision boundary in the original feature space can be represented by a linear decision boundary in the higher dimensional feature space. This is known as the *kernel trick*. Although kernel methods like RBF are relatively efficient, they still take significantly more

computations than a linear-SVM. For more information on SVM and the kernel trick we recommend [7].

3.4 Classification

On basis of an SVM trained on past image data, i.e. $I_{n-60},\ldots,I_{n-1}$, the task is to classify pixels in the current frame I_n as being part of a road or an obstacle. This process starts with feature extraction on I_n providing a feature for each non-overlapping 17-by-17 pixel block in I_n. These features are then classified by the SVM resulting in a classification of each pixel block in I_n. In this work, we assume that the road is always below the horizon; therefore we only classify pixel blocks that are below the horizon. Every pixel block above the horizon is determined to be part of an object.

After classification, we apply post processing to incorporate some level of regularization. This starts with creation of a binary mask from the classification result in which 1 represents a road and 0 an obstacle pixel block. Regularization is then performed by applying median filters and enforces local smoothing of the classification result at the pixel block level.

4 EVALUATION

The primary goal of our experiments is to evaluate different image features (color-based and gradient-based histograms) for the road detection task. We also experiment with using a linear-SVM and a RBF-SVM. Our system is currently implemented in MATLAB utilizing 4 cores and a GPU trough MATLAB's parallel computing toolbox.

(a) (b) (c)

Fig. 3. The automaticaly obtained train mask sets (a) in which red is a weak obstacle label, green a weak road label and blue is unknown. The classification result (b) and the ground truth (c).

Our experimental evaluation is based on 20 datasets with 123 manually annotated ground truth frames in total. The datasets are recorded at a resolution of 640-by-480 pixels at a frame rate of 30Hz. For a single experiment, we take a ground truth image G_n and train the SVM on 60 preceding frames $I_{n-60},\ldots,I_{n-1}$ and classify the current frame I_n. This classification is then compared pixel-wise against G_n, resulting in an error rate, i.e. the percentage of wrongly classified pixels in each frame. This per-frame error rate is averaged over all frames in a particular dataset, resulting in a per-dataset average error rate. An example of this classification process is shown in Fig. 3.

We use five different performance criteria to evaluate our system:

1. *Average error rate:* The average over the per-dataset average error rates. This assures that each dataset is equally weighted in the metric regardless of its number of ground truth

frames. Frames for which the SVM did not converge, and which were not classified, are given an error of 100%.

2. *Average maximum error rate:* The average over the per-dataset maximum error rates. Frames for which the SVM did not converge are ignored.
3. *Average maximum false negative rate:* Similar as average maximum error rate but now only considering false negatives (road classified as obstacle).
4. *Average maximum false positive rate:* Similar as average maximum error rate but now only considering false positives (obstacle classified as road).
5. *Total number of unclassified frames:* Frames for which the SVM did not converge. This is an important measure for the robustness of the system.

3.3 SVM configuration

In the first experiment we compare a linear-SVM with a RBF-SVM on six randomly selected datasets. The two important parameters of the SVM are the box constraint (C) and the kernel width (σ) of the RBF. For these parameters, sweeps were performed and we only report the performance that was obtained with the optimal parameter values (C=90 and σ =30). The results of the RBF-SVM over several feature spaces are shown in Table II and that of the linear-SVM in Table III. It can clearly been seen that the RBF-SVM significantly outperforms the linear-SVM on multiple performance criteria.

TABLE II RESULTS RBF-SVM (SORTED ON AVERAGE ERROR RATE)

RBF-SVM	Average error rate (%)	Average maximum error rate	Average maximum false negative rate (%)	Average maximum false positive rate (%)	Total number of unclassified frames
HS144	5.62	10.93	10.28	3.40	0
HS128	5.72	10.92	10.77	1.97	0
HSV96	6.05	13.96	13.95	1.96	0
HS-HOG96	7.71	12.82	11.69	2.50	1
HSV216	8.18	10.46	10.01	3.56	1
YIQ216	9.06	23.40	23.25	4.34	0
RGB216	10.18	24.11	22.23	6.77	1
IQ144	10.34	23.78	23.15	2.52	0
RGB96	21.40	19.68	18.36	3.77	5

TABLE III RESULTS LINEAR-SVM (SORTED ON AVERAGE ERROR RATE)

Linear-SVM	Average error rate (%)	Average maximum error rate	Average maximum false negative rate (%)	Average maximum false positive rate (%)	Total number of unclassified frames
HS-HOG96	11.90	11.60	11.60	1.78	3
HSV96	15.86	10.84	10.07	1.89	5
HSV216	17.97	19.30	19.06	2.24	5
HS144	19.51	9.51	9.29	2.91	7
HS128	19.90	13.46	13.42	1.87	7
RGB216	20.12	10.85	7.95	5.34	5
IQ144	39.32	19.10	19.03	1.51	15
RGB96	43.96	18.67	16.50	3.88	16
YIQ216	45.83	13.94	13.94	1.24	18

An interesting observation that can be made from Table II and Table III is that the best performing linear-SVM method has twice the average error rate of the best performing RBF-SVM method but the feature spaces are different. For the linear method, feature spaces that have more dimensions (e.g. both color and gradient) perform better. The RBF-SVM method performs better on feature spaces with fewer dimensions. A closer inspection reveals that the performance of the linear-SVM approach is harmed by the high number of frames for which it did not converge. For our system, robustness is as important as accuracy, so the total

number of unclassified frames is an important performance criterion and makes the linear-SVM approach unattractive

Looking more in depth in the results of linear-SVM method, we see that the HOG based (HS-HOG96) and intensity based features (HSV96 and HSV216) outperform the features that do not use gradients or intensity. This shows that intensity and gradients are better linearly separable than e.g. Hue and Saturation for our application domain. However, this benefit is lost when using non-linear decision boundaries of RBF-SVM.

3.4 Feature sets

In our second experiment we focus on different feature spaces. We only discuss the results obtained with the RBF-SVM that are provided in Table II. The results show that feature sets with at least Hue and Saturation give the best results. Even so, only using Hue and Saturation provides optimal results. It is interesting to note that the HS128 feature, that uses a 1-D histogram with 128 bins in total, provides practically the same performance as the HS144 feature, which uses a 2-D histogram with 144 bins. This indicates that modeling dependencies between the Hue and Saturation subspaces adds little to no useful statistical information for the classifier. The slightly better average error rate of HS144 compared to HS128 is probably solely due to using more bins. Qualitative results obtained with using a RBF-SVM and the HS144 feature space are provided in Fig. 4.

It can also be observed that gradient information used in the HS-HOG96 feature adds no benefits when compared to the HSV96 feature for the RBF-SVM. Apparently, only using Hue and Saturation and allowing for a non-linear separation boundary is better than using gradient/intensity information either with a linear or a non-linear separation boundary. This can be explained from the fact that gradient information is more sensitive to perspective distortion than color information.

3.5 Histogram creation

In this experiment we evaluate the potential benefit of linear compared to adaptive histogram creation strategies over all 123 ground truth frames. We use the Hue Saturation feature space and evaluate 1-D and 2-D histograms with a RBF-SVM. Here we make sure that each tested approach has an equal number of 100 histogram bins. The results are provided in Table IV.

TABLE IV RESULTS LINEAR VS. ADAPTIVE HISTOGRAMS

Dimensions	Bin distribution	Average error rate (%)
1	Linear	4.3903
1	Adaptive	4.6354
2	Linear	4.5216
2	Adaptive	4.8883

It can be seen that using adaptive bins has no advantage over linear bins. However, in preliminary experiments where we used very few bins (10-25 bins), we observed that using adaptive bins is advantageous. However, the absolute performance when using so few bins is significantly worse than when using 100 bins. This indicates that using adaptive bins is only useful when a system has strict limitations on computation resources.

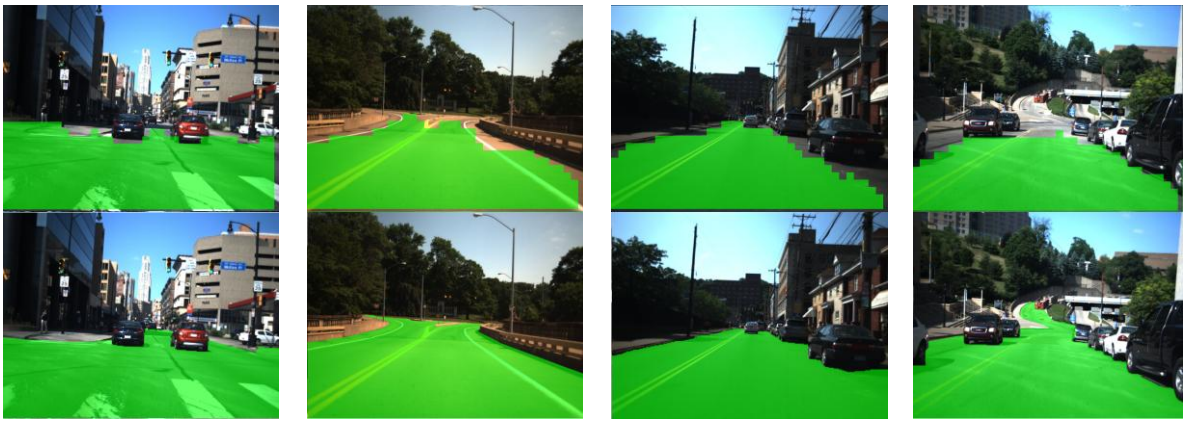


Fig. 4. Example results of our online self-supervised road detection system using a RBF-SVM with the HS144 feature space. Classification results are shown in top row and ground truth labels are shown in bottom row.

5 CONCLUSIONS

We developed an online and self-supervised computer vision system that can aid intelligent vehicles in distinguishing obstacles from surfaces on which can be driven. Critical design choices regarding the used classifier and appearance models are evaluated. It is shown that optimal accuracy of 95.6% is achieved, when using a RBF-SVM classifier with 1-D color histograms containing Hue and Saturation. In future work, we will develop a real-time version of our system and extend it with methods that regularize the classification over time.

References

[1] Dubbelman, G., van der Mark, W., van den Heuvel, J. C., and Groen, F. C., "Obstacle Detection During Day and Night Conditions using Stereo Vision.", *IEEE/RSJ International Conference* on *Intelligent Robots and Systems,* pp. 109-116, San Diego, USA, 2007.

[2] Cui, J., Guo, Y., Zhang, H., Qian, K., Bao, J., and Song, A., "Support Vector Machine Based Robotic Traversability Prediction with Vision Features", *International Journal of Computational Intelligence Systems*, *6*(4), pp. 596-608, 2013.

[3] Bajracharya, M., Tang, B., Howard, A., Turmon, M., and Matthies, L., "Learning Long-range Terrain Classification for Autonomous Navigation.", *IEEE International Conference* on *Robotics and Automation*, pp. 4018-4024, Pasadena, USA, 2008.

[4] Ulrich, I., and Nourbakhsh, I., "Appearance-based Obstacle Detection with Monocular Color Vision.", *AAAI National Conference on Artificial Intelligence*, pp. 866-871, Austin, USA, 2000.

[5] Kim, D., Oh, S. M., and Rehg, J. M., "Traversability Classification for UGV Navigation: A Comparison of Patch and Superpixel Representations.", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3166-3173, San Diego, USA, 2007.

[6] Satoh, S. I., "Generalized Histogram: Empirical Optimization of Low Dimensional Features for Image Matching.", *European Conference on Computer Vision*, pp. 210-223, 2004, Springer-Verlag Berlin Heidelberg.

[7] Bishop, C.M., "Pattern Recognition and Machine Learning.", Springer, 2007.