Drivable Space Estimation: Highway Border Detection

FINAL RESEARCH REPORT

Prof. Vijayakumar Bhagavatula
& Zhiding Yu

Contract No. DTRT-13-GUTC-26

DISCLAIMER

Drivable Space Estimation: Highway Border Detection

Vijayakumar Bhagavatula & Zhiding Yu

Introduction and Motivation

For autonomous driving (AD), it is important to estimate the drivable space, defined as the spatial region within which a vehicle can be safely maneuvered without hitting obstacles or having the danger of losing control. Knowing the drivable space is particularly important for traffic safety, for it plays a central role in subsequent decision making (e.g., motion planning) in autonomous vehicles. The purpose of this research is to propose vision based models and approaches centered on the theme of drivable space estimation.

Drivable space can be influenced by a variety of factors. A general principle is that the ground surface should be relatively flat, and solid enough to support the vehicle. The requirement on flatness indicates that any abrupt high-rising objects are obstacles to the vehicle. Such objects may include both moving ones such as pedestrians, bicycles and vehicles, and static ones like curbs, guard rails, barriers, pylons, trees and buildings. These objects form physical hard limits of the drivable space, while lane markers can be viewed as soft drivable space limits. This work targets scenarios with (at least) basic road infrastructures. In other words, our work mainly aims to handle traffic scenes with paved roads and ground surfaces.

Significant efforts have been devoted to pedestrian and vehicle detection in the past decade. We note, however, that detection of static objects such as different types of physical road borders has not been as well studied. Detecting road borders is an important module for autonomous systems since it provides road structure cues and localization information. As a result, this work proposes a standardized road border detection framework.

Project Description

An important problem for autonomous driving is the detection of road borders in highway scenarios. We observe some issues associated with the literature on road border detection: 1. Unlike many other detection problems, few standardized detection frameworks and benchmark datasets have been proposed to facilitate comparative studies. 2. Despite the recent advances in other detection problems with better models and feature representations, not many of them have been applied to address road border detection. With the development of intelligent transportation systems and the increasing need of scene understanding, it is beneficial to treat it as a stand-alone vision problem similar to pedestrian and vehicle detection .

Technical Approach

We propose a standardized scanning window detection framework similar to pedestrian detection. This allows us to systematically study different feature representations with detection performance quantitatively evaluated under the same standard. Inspired by the recent success of channel features and feature learning in detection problems, we propose a layered feature learning approach to extract discriminative window features that gives excellent detection performance.

Scanning window framework

We start from the most basic setting with the right view data collected by our system, approaching it as the scanning window detection of three types of road borders (concrete barrier, guard rail, and pavement boundary) and the closest lane marker. Upon designing the detection framework, special aspects of this problem are noticed: 1. Unlike pedestrians and vehicles, borders can spread over a very long distance and extend way beyond the width of a frame. As a result, traditional way of bounding a finite sized object with the tightest rectangle box may not be directly applicable. 2. Even if we define the visible portion within a frame as “one complete object", the width of this object may vary so much that it poses considerable challenges to detection (by requiring large numbers of window size configurations). 3. The localization accuracy with object-wise single bounding box can be significantly influenced by border curvatures and tilts. The above issues motivate us to use densely overlapping windows with fixed aspect ratios to represent borders, where each window only represents a local portion of the border. At testing, we explicitly allow multiple nearby windows firing together, in contrast to traditional object detection where non-maximum suppression (NMS) is performed to force single fired windows locally. One could see that such a framework renders much more flexibility in representing the dynamically changing borders.

To handle borders at multiple scales, we construct an image pyramid and scan each level with fixed-size scanning windows. Alternatively, one may also choose to scan the original images with varying window sizes. However, introducing the image pyramid and fixed-size scanning windows helps to unify features within each window to the same scale and gives better performance. For both training and testing, the number of pyramid levels in an octave is empirically set to 4. Based on the distribution of object sizes in the dataset, we set the number of pyramid levels to 11 where the bottom level corresponds to the original image. For every pyramid level, we set a fixed vertical step size for scanning windows, while normalizing the horizontal step sizes with the corresponding pyramid scale so that horizontal step sizes are unified when windows at different levels are projected back to the original image. In other words, at the original scale, larger projected windows will show proportionally larger vertical steps, while all windows show equal horizontal steps. We generate two sets of scanning windows for border and lane marker. Different types of borders are covered by the same set of scanning.

Different types of borders are covered by the same set of scanning windows, whose vertical sizes are fixed to 48 pixels. Meanwhile, lane markers are covered by smaller windows with vertical sizes set to 32 pixels. For both sets of windows, the aspect ratios are set to 2:3. Each window includes two context margins at the top and bottom part of the window. The margin sizes are set to 25% of the total window height.

Obtaining ground truth window labels
Once scanning windows are generated, we assign labels to each window by measuring how much the window is aligned with object annotations. Every border window is associated with several border labels, each of which corresponds to one specific border type indicating whether this window is a positive sample. For lane marker windows, each is only one associated with one lane marker label.

We also observe that a significant portion of the dataset contains consecutive frames of concrete barriers that are too small to be covered even by the smallest scanning windows. Traditionally, many object detection benchmarks choose to directly ignore small objects by not including them in the evaluation. We could have followed such tradition but we chose not to ignore these for the following reasons: 1. While these concrete barriers are small, they still show relatively strong border patterns that are identifiable. 2. If they are ignored, a considerable portion of the dataset images are treated as ones without any border, even though the borders are obviously there. In other words, any system that does not account for small barriers will not be able to handle highway scenarios with far away barriers. To address the above issue, we introduced an additional label called “small concrete barrier”. During training and testing, it is treated as a stand-alone object class besides regular concrete barriers and other border types.

Fig. 1 shows some examples of the labeled positive windows and their reference points returned by the above proposed alignment criteria.

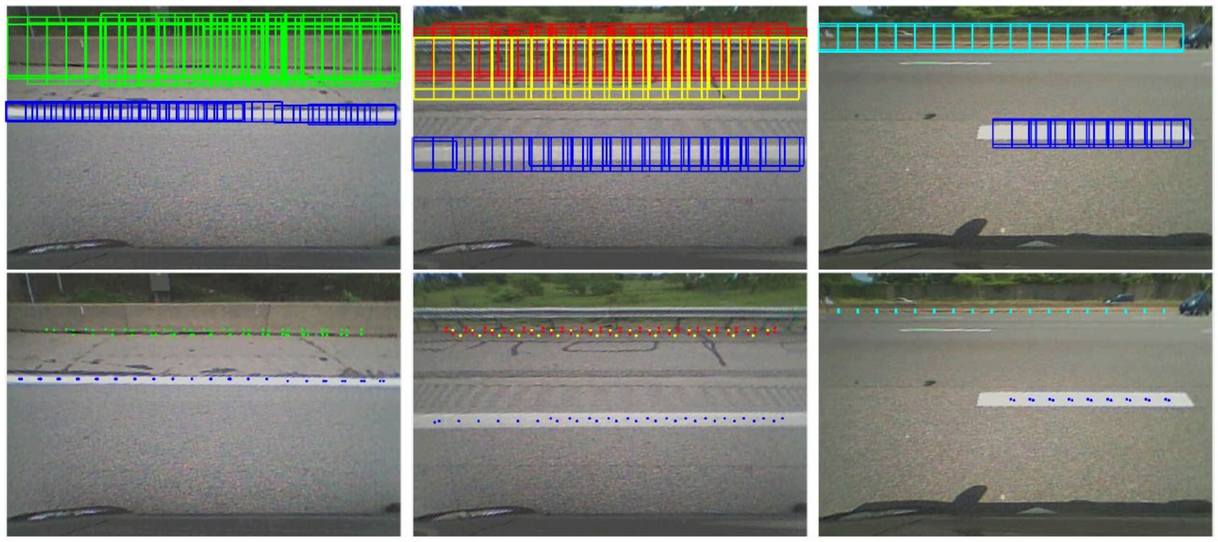


Fig. 1. Examples of positive scanning windows (first row) and their reference points (second row). Each reference point corresponds to one positive window indicating the localization information of this window. For concrete barrier (green), small concrete barrier (cyan) and guard rail (red), their reference points are the window bottom. For pavement boundary (yellow) and lane marker (blue), their reference points indicate the window centers.

Incorporating geometric structural information

The above framework generates scanning windows of different scales everywhere in the image. One thing we may consider is to incorporate the geometric constraint of window sizes versus vertical positions in the image. The basic idea is that far away borders (borders located higher in the image) are generally smaller than the closer ones. Therefore, many redundant windows can be filtered out based on the relation between window size and vertical position. Incorporating such geometric constraint brings multiple advantages: 1. The number of scanning windows is significantly reduced, reducing the computation load of the detection algorithm. 2. Better detection performance can be achieved since many potential false positives are avoided. 3. Incorporating the constraint helps to unify positive pavement boundary windows to the same scale.

Layered Discriminative Feature Learning

We learn the function between window height and vertical position via polynomials regression. For concrete barrier, guard rail and lane marker, we sample the heights and vertical positions from training set object annotations, and fit them with 3rd-order polynomials. The functions learned by regressions are shown in Fig. 2.

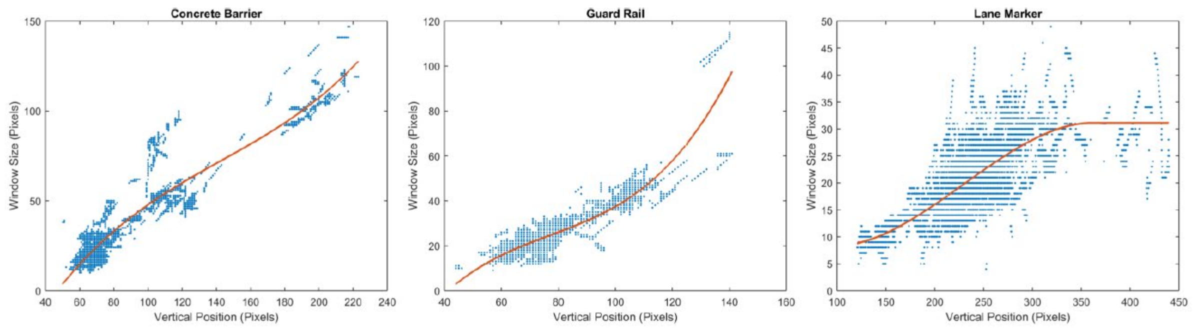


Fig. 2 The functions learned by polynomial regressions to fit window heights versus vertical positions. The origin of the vertical position is at the top of the image.

For pavement boundary, regression can not be directly applied due to the lack of height definition. We therefore reuse the learned parameters from concrete barrier to predict the corresponding window sizes. Since small concrete barriers make a subset of all concrete barriers, their geometric constraint is part of the concrete barrier constraint.

For every object, a threshold is then set to determine the acceptable range of window size, given any vertical position and the window size predicted by regression. This process is applied to the complete set of generated scanning windows to filter out redundant ones and return a type-wise valid set. For small concrete barrier, the complete set only consists of the smallest windows, as described previously. Since border windows need to cover multiple border types, the final window set is selected as the union of valid sets over all border types.

At the heart of detection is feature representation, a step which largely determines the algorithm performance. Significant work has been devoted to this area. A well-known type of hand-crafted feature is the histogram of oriented gradients (HOG) which is widely used for its good performance. The success of HOG mainly lies in its careful design towards capturing invariant representations. While visual appearance may vary due to certain reasons (such as shadows and changing luminance), gradient directions often remain invariant under these circumstances.

In addition, processing steps such as quantization and normalization can further enhance the representation robustness significantly. Despite its good performance, the representation power of HOG is limited by its handcrafted nature since the feature extraction processes in hand-crafted methods are often designed empirically. More recently, feature learning became a widely chosen scheme to obtain good features. Unlike hand-crafted features, feature learning methods explicitly optimize towards certain objective. The expectation is that if the objective is designed reasonably, the learning process can automatically mine robust and discriminative information important for good classification.

A family of highly representative feature learning methods are channel feature methods. These methods have achieved state-of-the-art detection performance on many detection tasks. In fact, the word “channel feature” itself refers to stacked channels of image features, often obtained via hand-crafted process. Typical ones include filter bank responses, edge detection results, and thresholded image channels. While channel features can be hand-crafted, they are often further processed by feature selection approaches such as boosted decision trees to select the final features. When designing channel features, a general principle is that the information contained in them should be rich or even over-complete. Such redundancy is necessary to maximize the representation ability of the learned features.

Another set of related works is deep learning. Deep learning based object detection/recognition has recently become popular due to the strong representation ability of deep neural networks. Over the past several years, they have significantly boosted the performance in many tasks such as object recognition, object detection, face verification and hand-written digit recognition. The success of deep neural networks lies in their layered learning architecture. Such design allows a hierarchical process where features are learned from local to global, and renders strong visual representation ability. However, because of the deep architecture and large numbers of parameters, the computation costs of deep learning methods are often very high.

We propose a learning framework which gives robust feature representation for scanning windows and renders good detection performance. Our proposed method bears resemblance to many channel feature methods, but incorporates a layered learning architecture similar to neural networks. To generate channel features, the method uses the oriented gradient descriptor similar to HOG, but extends its computation to multiple filter response channels in order to mine richer channel information.

Channel feature extraction

Given an image or feature channel, the descriptor at every pixel is obtained by computing the gradient and encoding it with a vector. As shown in Fig. 3, the gradient energy is encoded by assigning its value to a descriptor entry, while the quantized gradient angle is encoded by the position of the assigned entry. The descriptor bin (dimensions) number determines both the quantization resolution as well as the rotation invariance level. We set the bin number to 6 to achieve trade-off between these two aspects.

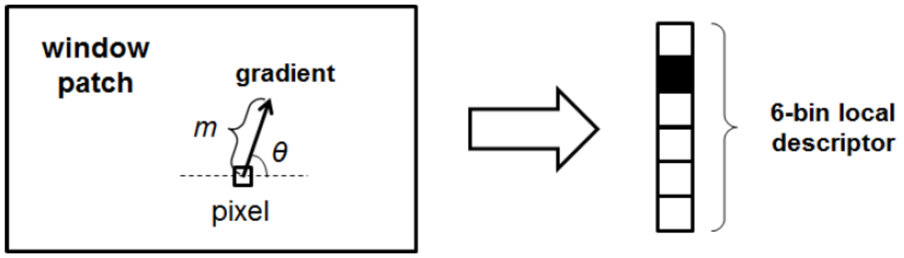


Fig. 3 Illustration of how pixel-wise gradient information is encoded by a descriptor.

In conventional HOG, local descriptors are generally computed on the grayscale channel. To incorporate richer information and maximize the representation ability via feature learning, we propose to compute the descriptors on multiple response channels obtained by a filter bank.

Fast local feature computing with random rectangle pooling

In conventional HOG, the feature is extracted by pooling and normalizing the oriented gradient descriptor within hand-designed rectangular cells. Pooling and normalization are the keys to good performance since they return more robust estimate of local statistics by averaging and normalizing out the noise. The size and location of HOG cells also play important roles in determining the performance. For example, features pooled with rectangles covering major object boundaries are generally more informative. For HOG, manually designing the pooling cells does not guarantee the optimal feature performance and limits the representation ability.

Feature learning provides a good alternative to address the problem. What we need to do is to provide a rich set of local features and let the learning algorithm itself throw away redundant information and extract the useful ones. This can be done by pooling local features from random rectangles with dynamic positions and sizes. The positions and sizes are varied such that the rectangles can cover a large number of different areas. In this work, we considered a set of 64 varying rectangles. These rectangles can be regarded as generalized HOG cells where local oriented gradient descriptors are pooled and normalized. In particular, within each rectangle, we sum the descriptors from the same filter response channel at every pixel covered by this rectangle. The resulting summed descriptor is then normalized with its L1 norm. Concatenating the pooled and normalized descriptors from every input channel, each rectangle returns a 54-d vector which is called a "local feature". Therefore, a total of 64 local features are generated during random rectangle pooling.

A computational advantage of rectangle pooling is the convenience and fast speed to obtain these local features through integral images. Upon obtaining the integral images of every output channel, the summation within each rectangle can be computed rapidly by adding and subtracting the channel feature vectors at the four corners of the rectangle. The strategy is particularly efficient when there are lots of rectangles overlapping with each other. To select good features, a number of methods have adopted strategies where channel features and bagging are used to generate large numbers of decision trees with shallow depths. The decision trees serve as weak classifiers and are ensembled by boosting to form a strong classifier.

These methods have shown the effectiveness of such strategy by achieving state-of-the-art performance on pedestrian detection. However, the flat learning architecture and shallow tree depth lacks a local to global representation hierarchy, and tend to ignore joint representation ability between multiple channel feature dimensions. In contrast, deep neural networks do not have these drawbacks but they generally require much more computation. We argue that going with such deep architectures on border detection problems may not be necessary. Instead, what we propose is a two-layer learning framework. By proposing this layered framework, we hope to retain a local to global hierarchy with relatively small computation requirement, and overcome some of the disadvantages of at learning architectures. Fig. 4 shows the proposed learning framework.

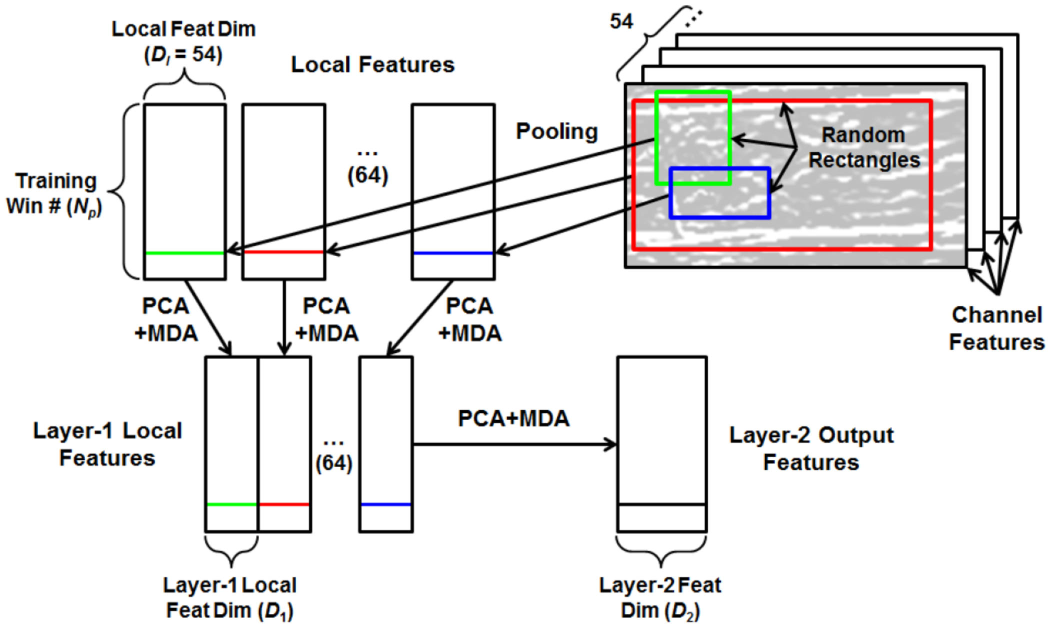


Fig. 4 The proposed learning framework

Given the labeled instances of training window channel features, we take the local features as input and perform layer-1 supervised dimensionality reductions. In particular, we denote the set of rectangles covering the same portion of areas in different windows as “co-located rectangles”, stacking the local features from all co-located rectangles and forming them as a local feature training set. Suppose there are Np labeled training windows, the local feature training set returned by every set of co-located rectangles is an Np _ 54 dimensional matrix. On every such training set, the parameters of layer-1 dimensionality reduction are learned, with learned output features denoted as “layer-1 local features”. There are two major purposes for layer-1 learning: 1. Local features often contain redundant information. By performing supervised dimensionality reduction, we seek to discard the redundant information, and return a more compact and discriminative representation. 2. The learning process takes the whole local feature vectors as inputs, therefore the inter-dimension joint representation is better considered in our framework.

The layer-1 local features are then concatenated as the input for layer-2 supervised dimensionality reduction. The layer-2 learning serves as a bridge connecting local representation and global representation, returning the learned output features as the final scanning window features.

To learn discriminative representations in both layer-1 and layer-2, we consider a supervised dimensionality reduction method called mixture discriminant analysis (MDA) in this work. Mixture discriminant analysis is a generalized version of linear discriminant analysis (LDA) where samples from each class are further grouped into subclasses with clustering. A regular multi-class LDA is performed by treating each subclass as a single, independent class. The reasons to choose MDA over LDA are two-fold: 1. By splitting into subclasses, MDA avoids the over-simplified unimodal assumption in LDA. 2. The maximum possible dimension of the learned basis in LDA is limited by the rank of the between-class scatter matrix. MDA avoids the rank problem and increases the upper bound of the learned basis dimension.

For border scanning windows, we hope to learn a unified feature representation for all classes instead of separate ones for every class. Instead of separately extracting features for the valid window set of every border type, we now only need to run it once on the union set. Considering that the valid window sets of different border types have large portions of overlap, the unified representation will help to significantly reduce the computation for feature representation. To learn this unified representation, we take the positive windows from every border class, and randomly mine certain number of negative windows from the intersection of class-wise negative sets within each frame to form a combined training set. Therefore, the labels of the combined training set contain five classes, including four border classes and one negative class. Note that before proceeding to layer-1 feature learning, a zero mean and unit variance normalization is conducted on every local feature set, followed by principal component analysis (PCA) which retains 95% of the energy. Before layer-2 feature learning, a sigmoid function is used to normalize all the layer-1 outputs between (0; 1), followed by another PCA to reduce the dimensionality.

Testing sample feature extraction
The above learning framework targets feature extraction on training samples. When extracting the testing window features, one only needs to load the learned parameters and go through the same extraction process. These parameters include the statistics of mean and standard deviation of local features, as well as the learned MDA projection bases at layer-1 and layer-2.

Classifier Training
We use radial basis function (RBF) kernel support vector machines (SVM) to predict the test window labels. To unify both feature learning and classifier training, the training window sets in feature learning are reused to also train the SVM classifiers. Given the learned features of training window sets and their labels, binary-class SVMs are trained separately for every object class. Separate binary SVMs are used for two reasons: 1. Binary SVM allows flexible thresholds and detection performance study by varying the threshold. Therefore, certain benchmarks such as precision-recall curves can be used in evaluation. 2. For borders, there could be some small portions of overlapping positive sets between two classes. It is difficult to handle this scenario in a multi-class SVM where only one predicted class is associated with each instance. For every class, windows with \ignored" labels are not included when training the classifier.

Numerical Experiments

We applied the proposed methods to the right-view data on “standard sets”. The proposed scanning window framework is applied to both training frames and testing frames, resulting in 2076 and 3521 scanning windows respectively covering borders and lane markers within each frame. For MDAs in both layer-1 and layer-2, we use k-means method to generate a total of 30 subclasses and empirically set the output dimensions to 15.

The parameters of SVMs are automatically decided with 5-fold cross-validations. For every object class, we predict the SVM scores of testing windows and set thresholds to determine the predicted labels. By varying the thresholds and comparing the predicted labels with ground truth, we obtain the precision recall curves which measure the detector performance of each class.

Qualitative results

We visualize the default SVM label outputs. Specifically, we visualize the voting points of scanning windows detected as positive. Fig. 5 shows typical examples of the successful results with the proposed the method, where green, cyan, red, yellow and blue correspond to concrete barrier, small concrete barrier, guard rail, pavement boundary and lane marker, respectively. One could see that the proposed scanning window detection method is able to correctly localize highway borders and generate results similar to ground truth.

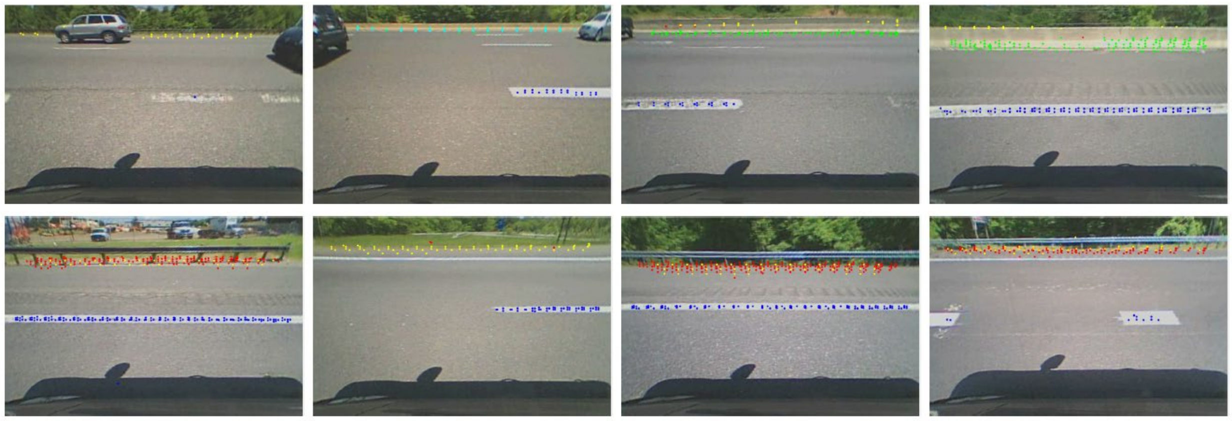


Fig. 5 Successful results returned by the proposed detection method.

A significant portion of the frames contain inner lanes with borders occluded by other vehicles. Fig. 6 shows examples of the highly occluded frames and their detection results.

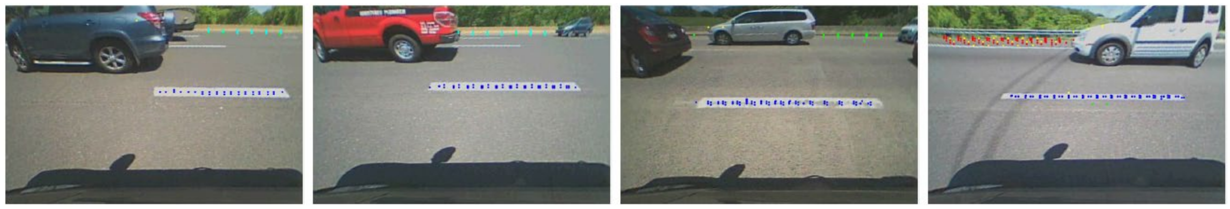


Fig. 6 Results on highly occluded frames

As the collected dataset simulates real world highway driving scenarios, a considerable portion of the frames contain a variety of challenging cases where detection becomes significantly harder and the number of incorrect results increases. Some of the examples are shown in Fig. 7. We notice that the major causes of detection failures include over-exposures, strong shadows, occlusions, samples unseen in training set, and fake patterns highly similar to real borders, etc.

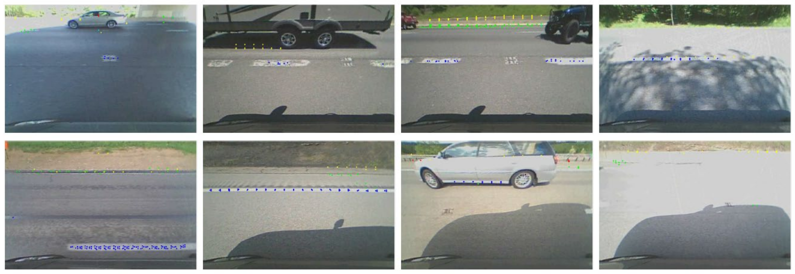
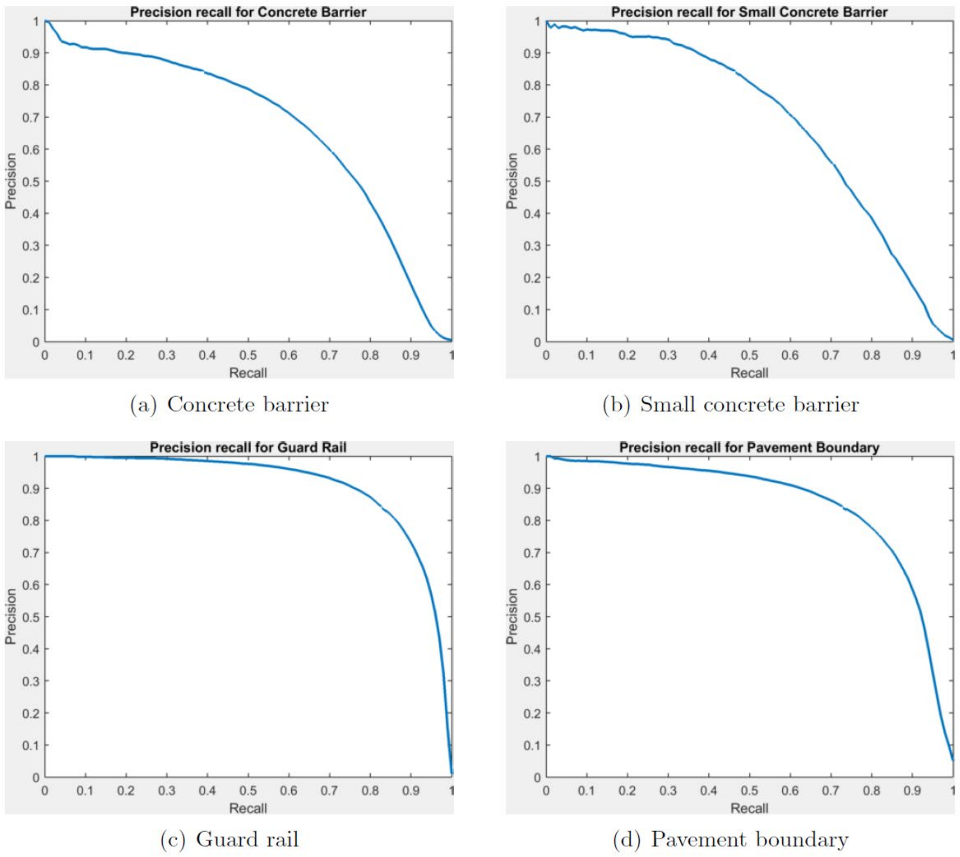


Fig. 7 Failure examples returned by the proposed detection method.

Quantitative results

Since each object detector is trained with a binary kernel SVM, we can obtain precision recall curves by varying the threshold on the SVM scores. Fig. 8 shows the precision recall curves of the detectors for each object class. We also measure the average precision (AP), which is obtained by taking the mean of precision values over the recall. In practice, we obtain this value by sampling and averaging the precisions sampled at 101 equally spaced recalls between [0; 1]. The measured AP of each object is: 1. Concrete barrier - 0.667. 2. Small Concrete Barrier - 0.683. 3. Guard Rail - 0.904. 4. Pavement Boundary - 0.846. Lane Marker - 0.902.



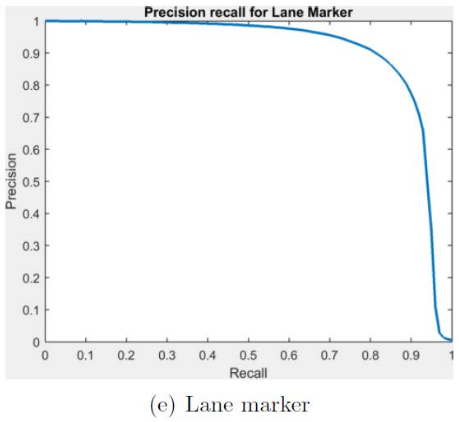


Fig. 8 The evaluated precision recall curves of each object class.

It can be observed that among all the object classes, detecting concrete barriers is the most difficult one. A major reason for this is the texture-less nature of concrete barriers. Compared to other objects, their visual appearances are less discriminative. Interestingly, the detection performance on small concrete barriers is even slightly better than regular sized ones despite the lower resolution of the input. This shows that having more global representations and incorporating more context may partly help to improve the detection performance.

Summary

We described a scanning window detection framework and its evaluation benchmark for highway border detection. This framework allow us to systematically compare the detection performance of different feature representation and classification methods. We also proposed a feature learning approach to extract learned scanning window features, and train a detector separately for each object class with a binary RBF kernel SVM. The performance of each detector is quantitatively measured by a precision-recall curve and the average precision. In the experiment, we showed that the proposed feature learning method can return features with relatively low dimensions and good representation ability, achieving good detection performance.