Fast Detection of Multiple Objects in Traffic Scenes With a Common Detection Framework

Qichang Hu, Sakrapee Paisitkriangkrai, Chunhua Shen, Anton van den Hengel, and Fatih Porikli

***Abstract*—Traffic scene perception (TSP) aims to extract accurate real-time on-road environment information, which involves three phases: detection of objects of interest, recognition of detected objects, and tracking of objects in motion. Since recognition and tracking often rely on the results from detection, the ability to detect objects of interest effectively plays a crucial role in TSP. In this paper, we focus on three important classes of objects: traffic signs, cars, and cyclists. We propose to detect *all the three* important objects in a single learning-based detection framework. The proposed framework consists of a dense feature extractor and detectors of three important classes. Once the dense features have been extracted, these features are shared with all detectors. The advantage of using one common framework is that the detection speed is much faster, since all dense features need only to be evaluated once in the testing phase. In contrast, most previous works have designed specific detectors using different features for each of these three classes. To enhance the feature robustness to noises and image deformations, we introduce spatially pooled features as a part of aggregated channel features. In order to further improve the generalization performance, we propose an object subcategorization method as a means of capturing the intraclass variation of objects. We experimentally demonstrate the effectiveness and efficiency of the proposed framework in three detection applications: traffic sign detection, car detection, and cyclist detection. The proposed framework achieves the competitive performance with state-of-the-art approaches on several benchmark data sets.**

***Index Terms*—Traffic scene perception, traffic sign detection, car detection, cyclist detection, object subcategorization.**

I Introduction

VISION-BASED traffic scene perception (TSP) is one of many fast-emerging areas in the intelligent transportation system. This field of research has been actively studied over the past decade [57]. TSP involves three phases: detection, recognition and tracking of various objects of interest. Since recognition and tracking often rely on the results from detection, the ability to detect objects of interest effectively plays a crucial role in TSP. In this paper, we focus on three important classes of objects: traffic signs, cars, and cyclists. Fig. 1 shows a typical on-road traffic scene with the detected objects of interest and illustrates some positive examples from the three mentioned classes.

Fig. 1. Top image: A typical on-road traffic scene with the detected objects of interest. Bottom images: Each block represents one class of objects of interest. From left to right, the first block contains traffic sign examples, the second contains car examples, and the third contains cyclist examples.

The aim of traffic sign detection is to alert the driver of the changed traffic conditions. The task is to accurately localize and recognize road signs in various traffic environments. Prior approaches [8], [9], [32] use color and shape information. However, these approaches are not adaptive under severe weather and lighting conditions. Additionally, appearance of traffic signs can physically change over time, due to the weather and damage caused by accidents. Instead of using color and shape features, most recent approaches [42], [63] employ texture or gradient features, such as local binary patterns (LBP) [2] and histogram of oriented gradients (HOG) [7]. These features are partially invariant to image distortion and illumination change, but they are still unable to handle severe deformations.

Car detection is a more challenging problem compared to traffic sign detection due to its large intra-class variation caused by different viewpoints and occlusion patterns. Although sliding-window based methods have shown promising results in face and human detection [7], [62], they often fail to detect cars due to a large variation of viewpoints. Recently the deformable parts model (DPM) [16], which has gained a lot of attention in generic object detection, has been adapted successfully for car detection [20], [26], [49]. In addition to the DPM, visual subcategorization based approaches [10], [31], [45] have been applied to improve the generalization performance of detection model.

Manuscript received March 10, 2015; revised June 27, 2015 and September 16, 2015; accepted October 9, 2015. Date of publication December 3, 2015; date of current version March 25, 2016. The Associate Editor for this paper was J. M. Alvarez.

Q. Hu is with the University of Adelaide, Adelaide, S.A. 5005, Australia, and also with NICTA, Alexandria, N.S.W. 1435, Australia (e-mail: qichang.hu@adelaide.edu.au).

S. Paisitkriangkrai is with the University of Adelaide, Adelaide, S.A. 5005, Australia.

C. Shen and A. van den Hengel are with the University of Adelaide, Adelaide, S.A. 5005, Australia, and also with the Australian Centre for Robotic Vision, Brisbane, Qld. 4001, Australia.

F. Porikli is with NICTA, Alexandria, N.S.W. 1435, Australia, and also with the Australian Centre for Robotic Vision, Brisbane, Qld. 4001, Australia.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2015.2496795

1524-9050 © 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Cyclist detection is a new attractive application in the domain of TSP. At present, only few methods are designed purposely for cyclist detection. Many existing pedestrian detection approaches [7], [11], [20] can be adapted for cyclist detection because appearances of pedestrians are very similar to appearances of cyclists along the road. Compared to pedestrian detection, the new problem is more difficult because the various appearances and viewpoints increase the diversity of cyclists. Therefore, existing pedestrian detectors hardly achieve the acceptable performance for cyclist detection.

Most previous methods have designed specific detectors using different features for each of these three classes. The approach we claim here differs from these existing approaches in that we propose a single learning based detection framework to detect all the three important classes of objects. The proposed framework consists of a dense feature extractor and detectors of these three classes. Once the dense features have been extracted, these features are shared with all detectors. The advantage of using one common framework is that the detection speed is much faster, since all dense features need only to be evaluated once in the testing phase. The proposed framework introduces spatially pooled features [48] as a part of aggregated channel features [13] to enhance the feature robustness to noises and image deformations. In order to further improve the generalization performance, we propose an object subcategorization method as a means of capturing the intra-class variation of objects.

A. Related Works

1) Generic Object Detection: Object detection is a challenging but important application in the computer vision community. It has achieved successful outcomes in many practical applications such as face detection and pedestrian detection [2], [7], [62], [66]. Complete survey of object detection can be found in [7], [16], [22], [62], [67]. This section briefly reviews several generic object detection methods.

One classical object detector is the detection framework of Viola and Jones which uses a sliding-window search with a cascade classifier to achieve accurate location and efficient classification [62]. The other commonly used framework is using a linear support vector machine (SVM) classifier with histogram of oriented gradients (HOG) features, which has been applied successfully in pedestrian detection [7]. These frameworks achieve excellent detection results on rigid object classes. However, for object classes with a large intra-class variation, their detection performance falls down dramatically [48].

In order to deal with appearance variations in object detection, a deformable parts model (DPM) based method has been proposed in [16]. This method relies on a variant of HOG features and window template matching, but explicitly models deformations using a latent SVM classifier. It has been applied successfully in many object detection applications [20], [59], [69]. In addition to the DPM, visual subcategorization [10] is another common approach to improve the generalization performance of detection model. It divides the entire object class into multiple subclasses such that objects with similar visual appearance are grouped together. A sub-detector is trained for each subclass and detection results from all sub-detectors are merged to generate the final results. Recently, a new detection framework which uses aggregated channel features (ACF) and an AdaBoost classifier has been proposed in [11]. This framework uses exhaustive sliding-window search to detect objects at multi-scales. It has been adapted successfully for many practical applications [42], [45], [48].

2) Traffic Sign Detection: Many traffic sign detectors have been proposed over the last decade with newly created challenging benchmarks. Interested reader should see [43] which provides a detailed analysis on the recent progress in the field of traffic sign detection. Most existing traffic sign detectors are appearance-based detectors. These detectors generally fall into one of four categories, namely, color-based approaches, shape-based approaches, texture-based approaches, and hybrid approaches.

Color-based approaches [8], [9], [32] usually employ a two-stage strategy. First, segmentation is done by a thresholding operation in one specific color space. Subsequently, shape detection is implemented and is applied only to the segmented regions. Since RGB color space is very sensitive to illumination change, some approaches [15], [32], [39] convert the RGB space to the HSI space which is partially invariant to light change. Other approaches [9], [30] implement segmentation in the normalized RGB space which is shown to outperform the HSI space [23]. Both the HSI and the normalized RGB space can alleviate the negative effect of illumination change, but still fail on some severe situations.

Shape-based approaches [27], [38], [58] detect edges or corners from raw images using canny edge detector or its variants. Then, edges and corners will be connected to regular polygons or circles by using Hough-like voting scheme. These detectors are invariant to illumination change, but the memory and computational requirement is quite high for large images. In [8], a genetic algorithm is adopted to detect circles and is invariant to projective deformation, but the expensive computational requirement limits its application.

Texture-based approaches firstly extract hand-crafted features computed from texture of images, and then use these extracted features to train a classifier. Popular hand-crafted features include HOG, LBP, ACF, etc [2], [7], [11]. Some approaches [35], [51], [63] use the HOG features with a SVM, others [42] use the ACF features with an AdaBoost classifier. Besides the above approaches, a convolutional neural network (CNN) is adopted for traffic sign detection and achieves excellent results in [56].

Hybrid approaches [18], [53] are a combination of the aforementioned approaches. Usually, the initial step is the segmentation to narrow the search space, which is same as the color-based approaches. Instead of only using edges features or texture-based features, these methods use them together to improve the detection performance.

One standard benchmark for traffic sign detection is the German traffic sign detection benchmark (GTSDB) [28] which collects three important categories of road signs (prohibitory, danger, and mandatory) from various traffic scenes. All traffic signs have been fully annotated with the rectangular regions of interest (ROIs). Researchers can conveniently compare their work based on this benchmark.

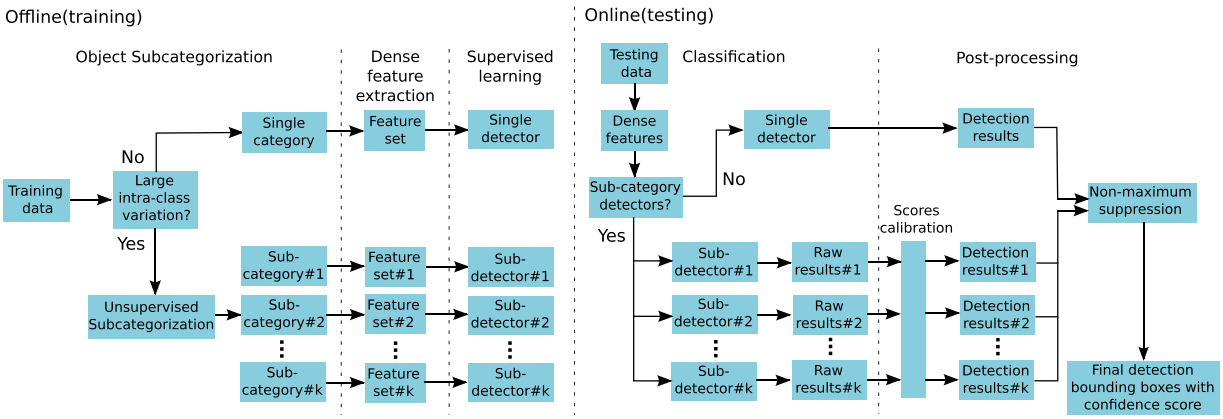


Fig. 2. Overview of the proposed detection framework. Left diagram is the training section and right diagram is the testing section.

3) Car Detection: Many existing car detectors are vision-based detectors. Interested reader should see [57] which discusses different approaches for vehicle detection using mono, stereo, and other vision-sensors. We focus on vision-based car detectors using monocular information in this paper. These detectors can be divided into three categories: DPM-based approaches, subcategorization-based approaches and motion-based approaches.

DPM-based approaches are built on the deformable parts model (DPM) [16] which has been successfully applied in car detection [59]. In [20], a variant of DPM discretizes the number of car orientations and each component of the mixture model corresponds to one orientation. The authors of [26] train a variant of DPM to detect cars under severe occlusions and clutters. In [49], occlusion patterns are used as training data to train a DPM which can reason the relationships between cars and obstacles for detection.

Visual subcategorization which learns subcategories within an object class is a common approach to improve the model generalization in car detection [10]. It usually consists of two phases: feature extraction and clustering. Samples with similar visual features are grouped together by applying clustering algorithm on extracted feature space. Subcategorization-based methods are commonly used with DPM to detect cars from multiple viewpoints. In [31], subcategories of cars corresponding to car orientation are learned by using locally linear embedding method with HOG features. In [45], cars with similar viewpoints, occlusions, and truncation scenarios are grouped into the same subcategory using a semi-supervised clustering method with ACF features.

Motion-based approaches often use appearance cues in monocular vision since monocular images do not provide any 3D and depth information. In [4], adaptive background model is used to detect cars based on motion that differentiated them from the background. The authors of [65] propose an adaptive background model to model the area where overtaking cars tend to appear in the camera's field of view. Optical flow [40], which is a popular tool in machine vision, has been used for monocular car detection. In [33], a combination of optical flow and symmetry tracking is used for car detection. Optical flow is also used in conjunction with appearance-based techniques in [6].

The KITTI vision benchmark (KITTI) [19] is a novel challenging benchmark for the tasks of monocular, stereo, optical flow, visual odometry, and 3D object detection. The KITTI dataset provides a wide range of images from various traffic scenes with fully annotated objects. Objects in the KITTI dataset includes pedestrians, cyclists, and vehicles.

4) Cyclist Detection: Many existing cyclist detectors use pedestrian detection techniques since appearances of pedestrians are very similar to appearances of cyclists along the road. These detectors are mainly derived from the fixed camera-based approaches.

Fixed camera-based approaches are designed for traffic monitoring using fixed cameras [54], [55], [64]. In [54], corner feature extraction, motion matching, and object classification are combined to detect pedestrians and cyclists simultaneously. In [64], a stereo vision based approach is proposed for pedestrian and cyclist detection. It uses the shape features and matching criterion of partial Hausdorff distance to detect targets. The authors of [55] propose a cyclist detector to detect two wheels of bicycles on road, but this approach is limited to detect crossing cyclists.

II. Our Approach

Despite several important techniques have been proposed on object detection, the conventional sliding-window based method of Viola and Jones [62] is still the most successful and practical object detector. The VJ framework consists of two main components: a dense feature extractor and a cascade classifier. In this paper, we build a common object detection framework for traffic scene perception based on the VJ framework, but our framework can employ a number of different classifiers to detect target objects of different classes. Apart from basic components of the VJ framework, we propose an object subcategorization method to improve the generalization performance and employ spatially pooled features [48] to enhance the robustness and effectiveness.

Fig. 2 shows an overview of our framework. In the training phase, we firstly check the intra-class variation of the input object class with respect to object properties, e.g. size, orientation, aspect ratio, and occlusion. If the variation is considerable large, we apply the object subcategorization method to categorize

the object class into multiple subcategories and train one sub-detector for each subcategory. Otherwise, we train a single detector for the entire object class. In the testing phase, raw detection results from all sub-detectors need to be calibrated before merging them together. Non-maximum suppression is used to eliminate redundant bounding boxes. If the framework employs detectors of different classes, detection results need to be carefully merged together.

A. Object Subcategorization

For object classes with a large intra-class variation like cars, the appearances and shapes of cars change significantly as viewpoints change. In order to deal with these variations that cannot be tackled by the conventional VJ framework, we present an object subcategorization method which aims to cluster the object class into visually homogeneous subcategories. The proposed subcategorization method applies an unsupervised clustering method to one specific feature space of the training samples to generate multiple subcategories. This method simplifies the original learning problem by dividing it into multiple sub-problems and improves model generalization performance.

1) Visual Features: A variety of hand-designed features can be used to perform the clustering algorithm, such as HOG and ACF [7], [11]. HOG is successful at capturing the shapes of objects while does not consider color information. ACF combines both color information and gradient information, which is shown to outperform HOG [13]. In our experiments, a total of 10 feature channels are used for clustering: LUV color channels (3 channels), histogram of oriented gradients at 6 bins (6 channels), and normalized gradient magnitude (1 channel). To extract features from the training samples, all samples are resized to the median object size.

2) Geometrical Features: Besides the visual features, geometrical information of objects can be extracted from traffic scenes using a variety of sensors and methods. In the KITTI dataset, objects in images from a velodyne laser scanner were annotated with 3D bounding boxes and 3D orientations. Ohn-Bar *et al.* [46] propose an analysis of different types of geometrical features, which shows that the geometrical features outperform the visual features for clustering, even for the CNN features. We use the following set of geometrical features to represent the object instances in our experiments.

3D Orientation: The appearances and shapes of objects change significantly as viewpoints change. We include the 3D orientation (relative orientation between the object and the camera) in clustering, aiming at grouping objects with similar visual appearance together.

Aspect-Ratio: The aspect-ratio (width/height) of objects is strongly correlated with the geometry of objects being detected. We use this feature because learning models at different aspect-ratios significantly improve the generalization performance.

Truncation Level: The truncation level refers to the percentage of the object outside of the image boundaries. This feature strongly affects appearances of objects.

Occlusion Index: Instead of using subtle occlusion patterns defined in [46], we use an occlusion index to indicate whether an object is not occluded, partially occluded, largely occluded or an unknown situation. We simplify the occlusion patterns because some occlusion features cannot be defined for each occluded object, such as occlusion level, relative orientation and relative 3D point between occluded objects and occluders. The above features are only available when the object is occluded by other labeled occluders. However, many occluders are unlabelled in the KITTI dataset.

3) Clustering: A clustering method is used to generate a predefined number of clusters on a specific feature space. Traditional clustering schemes, such as k-means or single linkage, suffer from the cluster degeneration which means that a few clusters claim most data samples [29]. The cluster degeneration problem can be alleviated by using spectral clustering. Spectral clustering followed by k-means often outperforms the traditional schemes. We implement the normalized spectral clustering using the algorithm proposed in [44]. The quality of clustering results is very sensitive to the predefined number of clusters. Unfortunately, how to determine the appropriate number of centroids is still an open question. We experimentally determine the number of clusters for each application.

B. Feature Extraction

The proposed framework introduces spatially pooled features [48] as a part of the aggregated channel features [13] and employs them as dense features in the training phase. All feature channels are aggregated in 4×4 blocks in order to produce fast pixel lookup features.

1) Aggregated Channel Features (ACF): Given an input image I, a channel C of I is a feature map, where the output pixels are computed from corresponding pixels of the input image. Aggregated channel features are extracted from multiple image channels using pixel lookup method. Many image channels are available for extracting features. For example, a trivial channel of a grayscale image is the image itself. For a color image, each color channel can be used as a channel. Other channels can be computed using various transformations of I. In order to accelerate the speed of feature extraction, all transformations are required to be translational invariant. It means that the transformation needs only to be evaluated once on the entire image rather than separately for each overlapping detection window.

ACF uses the same channel features as **ChnFtrs** [13]: LUV color channels (3 channels), histogram of oriented gradients (6 channels), and normalized gradient magnitude (1 channel). ACF combines the richness and diversity of statistics from these channels, which is shown to outperform HOG [11], [13]. Prior to computing these 10 channels, we smooth the input image I to suppress fine scale structures as well as noises.

LUV Color Channels: LUV color space contains 3 channels, L channel describes the lightness of the object, U channel and V channel represent the chromaticity of the object. Compared to RGB space, LUV space is able to partially invariant to illumination change. So the proposed detector can work under different light conditions. Images can be converted to LUV space by using a specific transformation.

Gradient Magnitude Channel: A normalized gradient magnitude is used to measure the edge strength. Gradient magnitude

$M(x, y)$ at location (x, y) is computed by $\sqrt{I_x^2 + I_y^2}$, where I_x and I_y are first intensity derivatives along the x-axis and y-axis, respectively. Since the gradient magnitude is computed on 3 LUV channels independently, only the maximum response is used as the gradient magnitude channel.

Gradient Histogram Channels: A histogram of oriented gradients is a weighted histogram where bin index is determined by gradient orientation and weighted by gradient magnitude [13]. The histogram of oriented gradients at location (x, y) is computed by $M(x, y) \cdot \mathbb{1}[\Theta(x, y) = \theta]$, where $\mathbb{1}$ is the indicator function, $M(x, y)$ and $\Theta(x, y)$ are the gradient magnitude and discrete gradient orientation, respectively. ACF quantizes the orientation space to 6 orientations and computes one gradient histogram channel for each orientation.

2) Spatially Pooled Features: Spatial pooling is used to combine multiple visual descriptors obtained at nearby locations into a lower dimensional descriptor over the pooling region. We follow the work of [48] which is shown that pooling can enhance the robustness of two hand-crafted low-level features, covariance features [60] and LBP [2].

Covariance Matrix: A covariance matrix is a positive semi-definite matrix which provides a measure of the relationship between multiple sets of variates. The diagonal elements of a covariance matrix represent the variance of each feature and non-diagonal elements represent the correlation between different features. In order to compute the covariance matrix, we use the following variates proposed in [48]:

$$[x, y, |I_x|, |I_y|, |I_{xx}|, |I_{yy}|, M, O_1, O_2]$$

where x and y indicate the pixel location. I_x and I_y are first intensity derivatives along the horizontal-axis and vertical-axis respectively. Similarly, I_{xx} and I_{yy} are second intensity derivatives, respectively. M is the gradient magnitude $\sqrt{I_x^2 + I_y^2}$. O_1 is the edge orientation $\arctan(|I_x|/|I_y|)$ and O_2 is an additional edge orientation in which,

$$O_2 = \begin{cases} \text{atan2}(I_y, I_x) & \text{if atan2}(I_y, I_x) > 0 \\ \text{atan2}(I_y, I_x) + \pi & \text{otherwise} \end{cases}$$

where the atan2 function is defined in terms of the arctan in the following:

$$\text{atan2}(y, x) = \begin{cases} \arctan\frac{y}{x} & x > 0 \\ \arctan\frac{y}{x} + \pi & y \geq 0, \quad x < 0 \\ \arctan\frac{y}{x} - \pi & y < 0, \quad x < 0 \\ +\frac{\pi}{2} & y > 0, \quad x = 0 \\ -\frac{\pi}{2} & y < 0, \quad x = 0 \\ \text{undefined} & y = 0, \quad x = 0 \end{cases}$$

The covariance descriptor of a region is a 9×9 covariance matrix which can be computed efficiently because the computational cost is independent of the size of the region. We also exclude the variance of pixel locations (x and y coordinates) and the correlation coefficient between pixel locations (x and y coordinates), since these features do not capture discriminative information. Due to the symmetry, each covariance descriptor finally contains 42 different values.

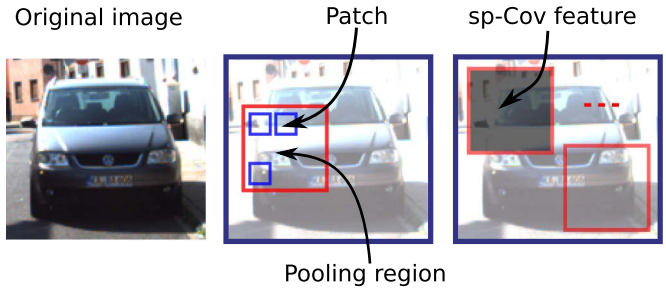


Fig. 3. Architecture of the spatially pooled covariance features.

Spatially Pooled Covariance: The spatial invariance and robustness of the covariance descriptors can be improved by applying pooling method. There are two common pooling methods in this context: average pooling and max pooling. Max pooling is used in our framework as it has been shown to outperform average pooling in image classification [5]. Max pooling uses the maximum value of a pooling region to represent the pooled features in the region. It aims to retain the most salient information and discard irrelevant details and noises over the pooling region. The image window is divided into multiple dense patches (refer to Fig. 3). Covariance features are computed over pixels within each patch. Then, we perform max pooling over a fixed-size pooling region and use the pooled features to represent the covariance features in the pooling region. In fact, multiple covariance matrices within each pooling region are summarized into a single matrix which has better invariance to image deformation and translation. The pooled features extracted from each pooling region is called the spatially pooled covariance (sp-Cov) features in [48].

Implementation: To expand the richness of our feature representation, we extract sp-Cov features using multi-scale patches with the following sizes: 4×4, 8×8 and 16×16 pixels. Each scale will generate an independent set of visual descriptors. In our experiments, the patch step-size is set to be 1 pixel, the pooling region is set to be 4×4 pixels, and the pooling spacing stride is set to be 4 pixels.

Local Binary Pattern (LBP): LBP is a texture descriptor which uses a histogram to represent the binary code of each image patch [2]. The original LBP is generated by thresholding the 3×3-neighbourhood of each pixel with the value of centre pixel. All binary results are concatenated to form an 8-bit length binary sequence with 2^8 different labels. The histogram of these 256 different labels can represent a texture descriptor. By following the work of [48], we convert the input image from the RGB space to LUV space, and extract the uniform LBP [66] from the luminance (L) channel. The uniform LBP, which is an extension of the original LBP, can better filter out noises.

Spatially Pooled LBP: Similar to the sp-Cov features, the image window is divided into multiple dense patches and LBP histogram is computed over pixels within each patch. In order to enhance the invariance to image deformation and translation, we perform max pooling over a fixed-size pooling region and use the pooled features to represent the LBP histogram in the pooling region. The pooled features extracted from each pooling region is called the spatially pooled LBP (sp-LBP) features in [48].

Implementation: To extract LBP, we apply the LBP operator on the 33-neighbourhood at each pixel. The LBP histogram is extracted from a 4 × 4 pixels patch. We extract the 58-dimension LBP histogram using a C-MEX implementation of [61]. In our experiments, the patch step-size, the pooling region, and the pooling spacing stride are set to 1 pixel, 8 × 8 pixels, and 4 pixels, respectively. Instead of extracting LBP histograms from multi-scale patches, the sp-LBP and LBP are combined as channel features.

C. Supervised Learning

Once dense features have been extracted, we are in a position to train a classifier. Instead of training a standard AdaBoost classifier, we use a shrinkage version of AdaBoost as the strong classifier and use decision trees as weak learners. To train the classifier, the procedure known as bootstrapping is applied, which collects hard negative samples and re-trains the classifier. If the object subcategorization is applied to the object class, we train one classifier for each subcategory. The pseudo code of the learning algorithm is presented in Algorithm 1.

Algorithm 1 Shrinkage version of AdaBoost

Input: The training set $S = \{(\vec{x}_1, y_1), \cdots, (\vec{x}_i, y_i), \cdots, (\vec{x}_N, y_N)\}$, $\vec{x}_i \subseteq \mathbb{R}^n$, $y_i \in \{-1, +1\}$, $i = 1, 2, \cdots, N$.

Initialize:The weighted distribution D of training set in 1st round, $D_1 = (w_{1,1}, \cdots, w_{1,i}, \cdots, w_{i,N}), w_{1,i} = 1/N$, $i = 1, 2, \cdots, N$.

for $t = 1 \cdots T$ **do**

· Train the weak learner $h_t(\cdot)$ using the weighted distribution D_t,

$$h_t(\cdot) : \mathbb{R}^n \to \{-1, +1\}$$

· Compute the error rate e_t of $h_t(\cdot)$ in traning set S.

$$e_t = \sum_{i=1}^{N} w_{t,i} \cdot \mathbb{1}(h_t(\vec{x}_i) \neq y_i)$$

· Compute the coefficient w_t of $h_t(\cdot)$ and update it by multiplying shrinkage parameter ν.

$$w_t = \frac{1}{2} \log \frac{1 - e_t}{e_t}$$

$$a_t = \nu \cdot w_t$$

· Update the weighted distribution of the training set

$$D_{t+1} = (w_{t+1,1}, \cdots, w_{t+1,i}, \cdots, w_{t+1,N})$$

$$w_{t+1,i} = \frac{w_{t,i}}{Z_t} \exp\left(-a_t y_i h_t(\vec{x}_i)\right), i = 1, 2, \cdots, N$$

where Z_t is a normalization factor,

$$Z_t = \sum_{i=1}^{N} w_{t,i} \exp\left(-a_t y_i h_t(\vec{x}_i)\right)$$

end

Output: Final classifier

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} a_t h_t(\vec{x})\right).$$

Shrinkage: The accuracy of AdaBoost can be further improved by applying a weighting coefficient known as shrinkage [25]. The shrinkage version of AdaBoost can be viewed as a form of regularization for boosting. At each iteration, the coefficient of weak learner is updated by

$$H_t(\vec{\boldsymbol{x}}) = H_{t-1}(\vec{\boldsymbol{x}}) + \nu \cdot w_t h_t(\vec{\boldsymbol{x}}). \tag{1}$$

Here $h_t(\cdot)$ is a weak learner of AdaBoost at the t-th round and w_t is the coefficient of the weak learner. $\nu \in (0, 1]$ is a learning rate which controls the trade-off between overall accuracy and training time. The smaller the value of ν, the higher the overall accuracy as long as the number of weak learners is sufficiently large. Compared to the standard AdaBoost, shrinkage often produces better generalization performance [17].

Bootstrapping: To improve the performance of the learned classifier, we perform three bootstrapping iterations in addition to the original training phase. The initial training phase randomly sample negative samples from training images with positive regions cropped out, and further bootstrapping iterations add more hard negatives to the training set. The learning process consists of 4 training iterations with increasing number of weak learners and the final model consists of 2048 weak learners.

D. Post-Processing

Raw detection results are generated by applying trained detectors to test images, but these results often contain some noises and redundant information. To improve detection performance, some techniques are used to post-process raw detection results.

1) Calibration of Confidence Scores: If we have multiple sub-detectors and apply them to test data, detection results of each sub-detector are required to merge together to generate the integrated results. However, the classifier of each sub-detector is learned with different training data, confidence scores of raw detection results output by individual classifiers need to be calibrated appropriately to suppress noises before merging them together. We address this problem by transforming the output of each classifier by a sigmoid regression to generate comparable score distributions [36], [52]. For sample i in subcategory k, its confidence score is the output of the ensemble classifier which is defined as

$$s_i^k = \sum_{t=1}^{T} a_t h_t\left(\vec{x}_i^k\right) \tag{2}$$

its calibrated score is defined as

$$g_i^k = \frac{1}{1 + \exp\left(A_k \cdot s_i^k + B_k\right)} \tag{3}$$

where A_k, B_k are the learned parameters for the k-th subcategory of the following regularized maximum likelihood problem:

$$\arg\min_{A_k, B_k} - \sum_{i=1}^{N_k} \left[t_i \log g_i^k + (1 - t_i) \log\left(1 - g_i^k\right)\right] \tag{4}$$

$$t_i = \begin{cases} \frac{N_+ + 1}{N_+ + 2} & \text{if } y_i = +1 \\ \frac{1}{N_- + 2} & \text{if } y_i = -1 \end{cases}, \quad i = 1, \ldots, N_k. \tag{5}$$

The g_i^k in equation (4) can be cancelled by reformulation:

$$\arg\min_{A_k, B_k} \sum_{i=1}^{N_k} \left[(t_i - 1)\left(A_k \cdot s_i^k + B_k\right) + \log\left(1 + \exp\left(A_k \cdot s_i^k + B_k\right)\right)\right]. \tag{6}$$

N_k is the total number of training examples for the k-th subcategory-specific classifier, N_+ is the number of positive

examples, and N_- is the number of negative examples in the k-th subcategory.

2) Non-Maximum Suppression (NMS): NMS aims to suppress redundant bounding boxes among the raw detection results. When multiple bounding boxes overlap, NMS will eliminate the lower-scored detections and retain the highest-scored detection. Pascal overlap score [14] is used to determine the overlap ratio a_0 between two bounding boxes. The overlap ratio a_0 is defined as

$$a_0 = \frac{\text{area}(B_1 \cap B_2)}{\text{area}(B_1 \cup B_2)} \tag{7}$$

where B_1 and B_2 are two different bounding boxes. If the overlap ratio a_0 exceeds a predefined threshold, bounding box with the lower confidence score is discarded.

3) Fusion of Detection Results: The proposed framework can detect multiple objects simultaneously using detectors or sub-detectors of different classes. We need to consider how to merge detection results from different detectors. Since an object may be detected redundantly using multiple sub-detectors or a single detector at multiple scales, NMS is usually used to eliminate these redundant detections in the merging process. However, NMS is not suitable for merging detections from different classes. Assume that a car is occluded by a cyclist. If their overlap ratio exceeds the threshold, NMS will simply delete the lower-scored detection, and retain the higher-scored detection. It means that one true positive will be removed in this case.

To remedy the above problem, we propose a fusion method to merge all detection results in two steps. Instead of applying NMS to detection results from all detectors, we apply NMS to detections of each single class (traffic sign, car, cyclist) separately to filter out redundant bounding boxes generated by either a single detector or multiple sub-detectors of the class. Next, we directly combine filtered bounding boxes from different classes without using NMS to generate the final detection results. This fusion method can eliminate the overlapped false positives of each single class while it keeps the true positives from different classes as much as possible.

III. Experiments

A. Traffic Sign Detection on GTSDB Dataset

In this section, we conduct an experiment on traffic sign detection and evaluate our detector on the German Traffic Sign Detection Benchmark (GTSDB) [28].

1) Dataset: The GTSDB dataset contains 600 images for training and 300 images for testing. Images are captured from various scenes (highway, urban, rural) and various time slots (morning, afternoon, dusk, etc). The dataset contains more than 1000 traffic signs from different categories. Three main categories of traffic signs (prohibitory, danger, mandatory) are selected as the target classes in the IJCNN 2013 [28] competition and in our experiments. The resolutions of traffic signs vary from 16×16 pixels to 128×128 pixels.

2) Evaluation Criteria: Pascal overlap score [14] is used to find the best match between each predicted bounding box and each ground truth. The minimum overlap ratio a_0 is set to be 60% on the GTSDB. Only the bounding box with the highest confidence score is counted as true positive if multiple bounding boxes satisfy the overlap criterion, the others are ignored. To compare the performance of different detectors, we follow the evaluation metric of the GTSDB which uses the area under the precision-recall curve (AUC) as a final score.

TABLE I
Performance (AUC) Difference Between Training on Original Training Set and Jettered Training Set

	Prohibitory	Danger	Mandatory	Avg.
Original dataset	98.76%	93.65%	86.86%	93.09%
Jettered dataset	100.00%	98.00%	97.57%	98.52%

TABLE II
Performance (AUC) of Detectors With Different Shrinkage Values. $*$ The Model Consists of 4096 Weak Learners While Others Consist of 2048 Weak Learners

Shrinkage	Prohibitory	Danger	Mandatory	Avg.
$\nu = 0.5$	98.13%	95.28%	90.32%	94.58%
$\nu = 0.2$	99.38%	96.80%	92.79%	96.32%
$\nu = 0.1$	**100.00%**	**98.00%**	**97.57%**	**98.52%**
$\nu = 0.05$	99.99%	97.81%	95.16%	97.63%
$\nu = 0.05^*$	99.99%	98.00%	96.76%	98.25%

3) Parameter Selection: To alleviate the effect of the illumination change, we apply the automatic color equalization algorithm (ACE) [21] to globally normalize all images. The resolution of the traffic sign model is set to 20×20 pixels and the dimension of model padding is set to 30×30 pixels. This border provides an additional amount of context that helps improve the detection performance [7], [12]. Additionally, we increase the number of positive samples by adding jittered versions of the original samples, which significantly improves the detection performance. For prohibitory and danger signs, flipped versions are added to the training set. For mandatory signs, samples are randomly perturbed in translation ($[-2, 2]$ pixels), in scale ($[0.8, 1]$ ratio), in rotation ($[-5, 5]$ degrees), and flipping. We demonstrate the performance gain on the test set in Table I. Negative samples are collected from the GTSDB training images with the corresponding traffic sign regions cropped out.

4) Experimental Design: We investigate the experimental design of the proposed detector on traffic sign detection. Since traffic signs are divided into three subcategories, we train one sub-detector for each subcategory. We train all detectors on the GTSDB training set and evaluate them on the GTSDB test set. All experiments are carried out using combined features $(\text{ACF} + \text{sp}-\text{Cov} + \text{sp}-\text{LBP})$ as dense features, AdaBoost with shrinkage value of 0.1 as the strong classifier, and depth3-decision trees as weak learners (if not specified otherwise).

Shrinkage: We evaluate the performance of AdaBoost with 4 different shrinkage values from $\{0.05, 0.1, 0.2, 0.5\}$. We decrease the reject threshold of soft cascade by a factor of ν as coefficients of weak learners have been diminished by a factor of ν. The area under precision-recall curve of different detectors are shown in Table II. We observe that applying a small shrinkage value often improves the detection performance and the best performance is achieved by setting $\nu = 0.1$. However, without increasing the number of weak learners, setting the shrinkage value to be too small $(\nu = 0.05)$ can degrade the

TABLE III
PERFORMANCE (AUC) OF DETECTORS WITH DIFFERENT DEPTHS OF DECISION TREES

Depth	Prohibitory	Danger	Mandatory	Avg.
depth-1	99.98%	97.41%	75.47%	90.95%
depth-2	99.99%	97.98%	95.49%	97.82%
depth-3	**100.00%**	**98.00%**	97.57%	**98.52%**
depth-4	99.99%	96.77%	**98.10%**	98.29%

TABLE IV
PERFORMANCE (AUC) OF DETECTORS WITH VARIOUS FEATURE COMBINATIONS

Feature combination	Prohibitory	Danger	Mandatory	Avg.
ACF (LUV+O+M)	98.72%	94.58%	92.65%	95.32%
sp-LBP+ACF	99.99%	95.07%	96.12%	97.06%
sp-Cov+LUV	99.30%	96.67%	95.56%	97.18%
sp-Cov+ACF	98.73%	95.23%	95.61%	96.52%
sp-Cov+sp-LBP+ACF	**100.00%**	**98.00%**	**97.57%**	**98.52%**

TABLE V
DETECTION PERFORMANCE (AUC) OF VARIOUS DETECTORS ON GTSDB TEST SET WITH 60% OVERLAP RATIO

Method	Prohibitory	Danger	Mandatory	Avg.
Ours	**100.00%**	98.00%	**97.57%**	**98.52%**
Wang *et al.* [63]	**100.00%**	**99.91%**	**100.00%**	**99.97%**
Mathias *et al.* [42]	**100.00%**	**100.00%**	**96.98%**	**98.99%**
BolognaCVLab [28]	99.98%	**98.72%**	95.76%	98.15%
Liang *et al.* [35]	100.00%	98.85%	92.00%	96.95%
Timofte *et al.* [58]	61.12%	79.43%	72.60%	71.05%
Viola-Jones [62]	90.81%	46.26%	44.87%	60.65%

performance as the boosting cannot converge within a limited number of boosting iterations.

Depth of Decision Trees: We trained 4 different traffic sign detectors with decision trees of depth 1 to depth 4. Table III shows the detection performance of different detectors. We observe that increase the depth of decision trees provides a performance gain, especially for the mandatory category. However, the depth-3 decision trees achieve better generalization performance and are faster to train than depth-4 decision trees.

Combination of Features: To compare the discriminative power of different feature representations, we evaluate the performance of various feature combinations. The results are shown in Table IV. We observe that a combination of the sp-Cov features and LUV outperforms the ACF features and combining more features can further improve the detection performance. The best result is achieved using a combination of all features $(\text{sp}-\text{Cov}+\text{sp}-\text{LBP}+\text{ACF})$.

5) Comparison With State-of-the-Art Detectors: Detection performance of various detectors on the GTSDB test set are shown in Table V. The proposed detector achieves the comparable results with state-of-the-art detectors despite its simplicity. These detectors [42], [63] that offer better performance employ multi-scale models in detection. The authors of [63] train multiple subcategory-specific classifiers for each type of mandatory signs to achieve the best performance.

B. Car Detection on UIUC Dataset

Next, we conduct an experiment on car detection and compare detection performance of different detectors on the UIUC dataset [1]. The UIUC dataset captures images of side views of cars with a resolution 40×100 pixels. The training set contains

TABLE VI
DETECTION PERFORMANCE OF VARIOUS DETECTORS ON UIUC MULTI-SCALE TEST SET

Method	F-Measure	Det. rate	No. false pos.
Ours	98.6%	99.28%	3
Pruning [47]	98.6%	97.8%	1
AdaBoost [62]	98.6%	98.6%	2
AdaBoost+LDA [68]	98.6%	97.8%	1
CS-AdaBoost [41]	95.3%	95.5%	9

TABLE VII
COMPARISON OF CAR DATASETS. THE FIRST FOUR COLUMNS INDICATE THE AMOUNT OF TRAINING/TESTING DATA IN EACH DATASET. NOTE THAT KITTI DATASET IS TWO ORDERS OF MAGNITUDE LARGER THAN OTHER EXISTING DATASETS. THE NEXT FIVE COLUMNS PROVIDE ADDITIONAL PROPERTIES OF EACH DATASET

	Training		Testing		Properties				
	# cars	# images	# cars	# images	color	Annotations	multi-views	occ. labels	trunc. labels
UIUC Car	550	1050	139	108					
MIT Car	516	516	-	-	✓				
Street Parking	-	881	-	-	✓	✓	✓	✓	
Pascal VOC	1250	713	1201	721	✓	✓	✓	✓	✓
KITTI Car	27k	7481	-	7518	✓	✓	✓	✓	✓

550 positive samples and 500 negative samples. The test set is divided into two subsets: 170 single-scale test images, containing 200 cars at roughly the same scale as in the training set, and 108 multi-scale test images, containing 139 cars at various scales.

We follow the evaluation protocol provided along with the UIUC dataset. A bounding box is counted as true positive if it lies within 25% of the ground truth dimension in each direction. Only the bounding box with the highest confidence score is counted as true positive if multiple bounding boxes satisfy the criterion, the others are counted as false positives. In the dataset, three criteria are used to evaluate the performance: F_1-score, detection rate, and the number of false positives. F_1-score is the weighted harmonic mean of precision and recall.

The dimension of UIUC car model is set to 40×100 pixels without marginal padding as the car images are clipped to the same size. We expand the positive samples by flipping car images along the vertical axis. Since viewpoints of cars in the UIUC dataset are limited to side-views, we train a single detector without applying subcategorization method. Table VI shows the results of different detectors on the multi-scale test images. We observe that our detector achieves the best detection rate with slight more false positives on this dataset.

C. Car Detection on KITTI Dataset

To further demonstrate the effectiveness and robustness of the proposed detector on car detection, we evaluate our detector on a more challenging object detection benchmark, KITTI dataset [19].

1) Dataset: The KITTI dataset is a recently proposed challenging dataset which consists of 7481 training images and 7518 test images, comprising more than 80 thousands of annotated objects in traffic scenes. Table VII provides a summary of existing car datasets. We observe that the KITTI dataset

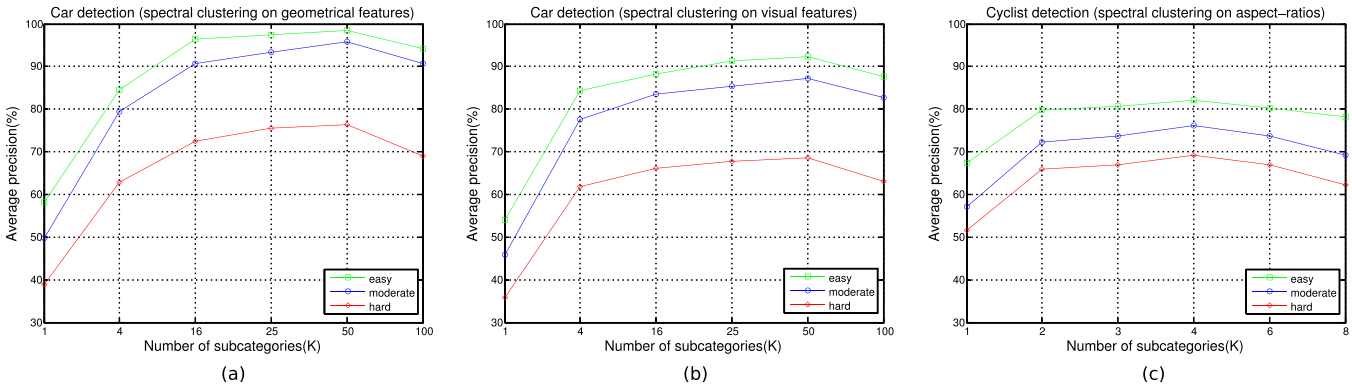


Fig. 4. Detection performance (AP) of various detectors with different number of subcategories on the KITTI validation set. (a) Car detector (spectral clustering + geometrical features). (b) Car detector (spectral clustering + visual features). (c) Cyclist detector (spectral clustering + aspect-ratios).

provides a large number of cars with different sizes, viewpoints, occlusion patterns, and truncation scenarios. Due to the diversity of these objects, the dataset has three subsets (Easy, Moderate, Hard) with respect to the difficulty of object size, occlusion and truncation. Since the detection performance are ranked based on the moderately difficult results, we use the moderate subset as the training data in our experiments. The moderate subset contains 15710 cars, with the heights of the cars vary from 25 pixels to 270 pixels and the aspect ratios vary between 0.9 and 4.0. Since annotations of test data are not provided by the KITTI benchmark, we split the KITTI training images into training set (first 4000 images) and validation set (remaining 3481 images).

2) Evaluation Criteria: We follow the provided protocol for evaluation. Pascal overlap score is used to find the best match and the minimum overlap ratio a_0 is set to be 70%. Only the bounding box with the highest confidence score is kept if multiple bounding boxes satisfy the overlap criterion, the others are counted as false positives. Instead of using AUC, average precision (AP) [14] is used to evaluate the detection performance. The AP summaries the shape of the precision-recall curve, and is defined as the mean precision at a set of evenly spaced recall levels.

3) Parameter Selection: We apply the proposed subcategorization method to categorize the training data into multiple subcategories. To find the model dimensions of each subcategory, we set the base height of each model to 52 pixels. From the base height, the width of each model can be obtained by taking the median aspect ratios of cars in the corresponding subcategory. Each model includes additional 4 pixels of marginal padding on all sides. Using a model with suitable aspect ratio can significantly improves the detection performance due to better localization. We expand the positive training samples by randomly perturbing original car samples in translation ([−2, 2] pixels), and in rotation ([−2, 2] degrees). Negative samples are collected from the KITTI training images with vehicle regions cropped out.

4) Experimental Design: We investigate the experimental design of the proposed detector on car detection. We train car detectors on the training set and evaluate them on the validation set. All experiments are carried out using ACF as dense features, AdaBoost with shrinkage value of 0.1 as the strong classifier, depth-4 decision trees as weak learners, and $K = 25$ in the subcategorization method (if not specified otherwise).

TABLE VIII
PERFORMANCE (AP) OF DETECTORS WITH DIFFERENT DEPTHS OF DECISION TREES

Depth	Easy	Moderate	Hard
depth-2	96.38%	89.18%	70.87%
depth-3	97.17%	91.21%	74.44%
depth-4	**97.41%**	**93.37%**	**75.60%**
depth-5	96.67%	92.08%	72.77%

TABLE IX
PERFORMANCE (AP) OF DETECTORS WITH VARIOUS FEATURE COMBINATIONS

Feature combination	Easy	Moderate	Hard	Runtime
ACF (LUV+O+M)	97.41%	93.37%	75.60%	0.5s
sp-LBP+ACF	97.74%	94.38%	76.50%	1.5s
sp-Cov+LUV	97.76%	93.68%	75.68%	6.8s
sp-Cov+ACF	97.98%	93.48%	75.61%	6.8s
sp-Cov+sp-LBP+ACF	**98.42%**	**94.55%**	**76.66%**	7.5s

Number of Subcategories: To investigate the effect of different number of clusters in our subcategorization method, we set the number from {1, 4, 16, 25, 50, 100}. Fig. 4(a) and (b) shows the effect of increasing the number of subcategories on geometrical features and visual features, respectively. We observe that geometrical features outperform visual features in spectral clustering. We also observe that the detection performance improves as we increase the number of subcategories up to 50. However, setting the number of subcategories to be too large ($K = 100$) can hurt the performance as the average number of samples in each subcategory is not enough to train an effective model. For the rest of our experiments, we set the number of subcategories to be 25 as it gives a better trade-off between the performance and the complexity.

Depth of Decision Trees: We evaluate the performance for different decision tree depths. As can be observed in Table VIII, the depth-4 decision trees perform the best as they can provide the best generalization performance.

Combination of Features: We evaluate the performance of various feature combinations on car detection. The results are shown in Table IX. We observe that the detection performance improves as we add more features and the best performance

TABLE X
DETECTION PERFORMANCE (AP) OF VARIOUS DETECTORS ON KITTI CAR TEST SET WITH 70% OVERLAP RATIO

Method	Easy	Moderate	Hard	Runtime
Ours	**87.19%**	**77.40%**	**60.60%**	1.5s
Regionlets [67], [37]	84.75%	76.54%	59.70%	1s
SubCat [45]	81.94%	66.32%	51.10%	0.3s
AOG [34]	80.26%	67.03%	55.60%	3s
OC-DPM [49]	74.94%	65.95%	53.86%	10s
DPM-C8B1 [59]	74.33%	60.99%	47.16%	15s
MDPM-un-BB [16]	71.19%	62.16%	48.43%	60s
mBoW [3]	36.02%	23.76%	18.44%	10s

TABLE XI
PERFORMANCE (AP) OF DETECTORS WITH DIFFERENT DEPTHS OF DECISION TREES

Depth	Easy	Moderate	Hard
depth-2	80.92%	75.47%	69.46%
depth-3	89.83%	82.67%	76.65%
depth-4	**92.15%**	**86.18%**	**79.28%**
depth-5	90.98%	85.21%	78.26%

TABLE XII
PERFORMANCE (AP) OF DETECTORS WITH VARIOUS FEATURE COMBINATIONS

Feature combination	Easy	Moderate	Hard	Runtime
ACF (LUV+O+M)	92.15%	86.18%	79.28%	0.2s
sp-LBP+ACF	**92.56%**	**87.40%**	**80.01%**	0.6s
sp-Cov+LUV	85.48%	79.17%	72.20%	5.8s
sp-Cov+ACF	85.16%	80.58%	73.64%	5.8s
sp-Cov+sp-LBP+ACF	90.08%	83.80%	76.89%	6.1s

is achieved using a combination of all features ($\text{sp}-\text{Cov}+\text{sp}-\text{LBP}+\text{ACF}$). A combination of sp-LBP features and ACF features also achieves the similar performance and is five times faster than the combination of all features. We use the combination of sp-LBP features and ACF features as dense features in the testing phase since it gives a better trade-off between detection performance and runtime.

5) Comparison With State-of-the-Art Detectors: Table X shows the performance comparison of state-of-the-art detectors on the KITTI test set. Experimental results show that the proposed detector is of not only better performance than all DPM-based methods [16], [49], [59] but also less runtime. More significantly, our detector outperforms the SubCat [45] which employs a similar object subcategorization method and the Regionlets [37], [67] which employs a similar pooling strategy. We conjecture that the additional performance gain is provided by the spatially pooled features.

D. Cyclist Detection on KITTI Dataset

In this section, we conduct an experiment on cyclist detection and evaluate our detector on the KITTI dataset.

1) Dataset: The KITTI dataset contains annotated cyclist objects which are captured from various traffic scenes. Similar to cars, cyclists are divided into three subsets (Easy, Moderate, Hard) and the moderate subset is used as the training data in our experiments. The moderate subset contains 1098 cyclists, with the heights of the cyclists vary from 25 pixels to 275 pixels and the aspect ratios vary between 0.3 and 1.5.

2) Evaluation Criteria: The KITTI cyclist detection uses the same evaluation protocol with the car detection except that the minimum overlap ratio is relaxed to 50%.

3) Parameter Selection: The proposed subcategorization method is applied to cyclist detection. We define the dimensions of cyclist models using the similar method in car detection. We set the base height of each model to 56 pixels, and the width of each model is derived from the median aspect ratios of cyclists in the corresponding subcategory. Each model includes additional 4 pixels of marginal padding on all sides. We expand the positive training samples by randomly perturbing the original cyclists in translation ([−2, 2] pixels), in rotation ([−2, 2] degrees). Negative samples are collected from the KITTI training images with cyclist regions cropped out.

4) Experimental Design: We investigate the experimental design of our detector on cyclist detection. We train cyclist detectors on the training set and evaluate them on the validation set. All experiments are carried out using ACF as dense features, AdaBoost with shrinkage value of 0.1 as the strong classifier, depth-4 decision trees as weak learners, and $K = 4$ in the subcategorization method (if not specified otherwise).

Number of Subcategories: We set the number of clusters from {1, 2, 3, 4, 6, 8} in our subcategorization method. Since only the minority of cyclists are occluded and truncated, clustering on all geometrical features leads to a cluster degeneration problem. We carefully select the aspect-ratios of cyclists as the feature space to avoid the above problem. Fig. 4(c) shows the effect of increasing the number of subcategories. We observe that the detection performance improves as we increase the number of subcategories up to 4. Since the number of cyclists is much less than cars, the average number of cyclists in each subcategory becomes very small when we have a large number of subcategories, which results in an imbalanced learning problem and degrades the detection performance.

Depth of Decision Trees: We trained 4 cyclist detectors with decision trees of depth-2 to depth-5. Average precisions of different detectors are shown in Table XI. We observe that depth-4 decision trees offer the best generalization performance, as similar in the car detection.

Combination of Features: We evaluate the performance of various feature combinations on cyclist detection. The results are shown in Table XII. We observe that the best performance is achieved using a combination of sp-LBP features and ACF features. The performance declines when we add the sp-Cov features as a part of aggregated channel features. The reason may be due to the lack of enough cyclist training samples. We use the combination of sp-LBP features and ACF features as the dense features in the testing phase.

5) Comparison With State-of-the-Art Detectors: Table XIII shows the performance comparison with state-of-the-art approaches. As shown in Table XIII, our detector outperforms all other methods on the test set. Specifically, our detector outperforms the best DPM-based method $\text{DPM}-\text{VOC}+\text{VP}$ [50] on all the three subsets by 16.29%, 14.95%, and 12.35%, respectively. Our detector also performs slightly better than the Regionlets [37], [67].

TABLE XIII
DETECTION PERFORMANCE (AP) OF VARIOUS DETECTORS ON KITTI CYCLIST TEST SET WITH 50% OVERLAP RATIO

Method	Easy	Moderate	Hard	Runtime
Ours	**58.72%**	**46.03%**	**40.58%**	0.6s
Regionlets [67], [37]	56.96%	44.65%	39.05%	1s
MV-RGBD-RF[24]	52.97%	42.61%	37.42%	4s
DPM-VOC+VP [50]	42.43%	31.08%	28.23%	8s
LSVM-MDPM-us [16]	38.84%	29.88%	27.31%	10s
DPM-C8B1 [59]	43.49%	29.04%	26.20%	15s
mBoW [3]	28.00%	21.62%	20.93%	10s

TABLE XIV
AN EVALUATION OF THE OVERALL RUNTIME OF THE PROPOSED FRAMEWORK WITH VARIOUS FEATURE COMBINATIONS

Feature combination	Feature extraction	Cars(25) detection	Cyclists(4) detection	Signs(3) detection	Total Runtime
ACF (LUV+O+M)	0.10s	0.40s	0.10s	0.05s	0.65s
sp-LBP+ACF	0.35s	1.20s	0.30s	0.10s	1.95s
sp-Cov+ACF	5.50s	1.30s	0.30s	0.10s	7.20s
sp-Cov+sp-LBP+ACF	5.75s	1.75s	0.35s	0.15s	8.00s

E. An Evaluation of the Overall Runtime

We conduct an experiment on the evaluation of the overall runtime of the proposed detection framework on the KITTI dataset. All experiments are carried out on a computer with an octa-core Intel Xeon 2.50 GHz processor. The average runtime of each component of the framework can be seen in Table XIV. For feature extraction, we observe that the ACF features can be extracted very quickly within 0.1 s. When we add the sp-LBP features, the runtime increases moderately, but these features provide an obvious performance gain in all three applications. When the sp-Cov features are employed, the runtime of feature extraction increases rapidly and dominates the total runtime of the system. For object detection, we observe that the car detector costs the most time in this framework since it has 25 sub-detectors. The traffic sign detector uses the least time since it has only 3 sub-detectors. We also observe that the runtime of detection increases as we add more complicated features in the framework. According to observe the detection results of three applications, we conjecture that using a combination of ACF features and sp-LBP features can provide a better trade-off between detection performance and system runtime.

IV. CONCLUSION

In this paper, we propose a common detection framework for detecting three important classes of objects in traffic scenes. The proposed framework introduces spatially pooled features as a part of aggregated channel features to enhance the feature robustness and employs detectors of three important classes to detect multiple objects. The detection speed of the framework is fast since dense features need only to be evaluated once rather than individually for each detector. To remedy the weakness of the VJ framework for object classes with a large intra-class variation, we propose an object subcategorization method to improve the generalization performance by capturing the variation. We demonstrated that our detector achieves the competitive results with state-of-the-art detectors in traffic sign detection, car detection, and cyclist detection. Future work could include that contextual information can be used to facilitate object detection in traffic scenes and convolutional neural network can be used to generate more discriminative feature representations.

REFERENCES

[1] S. Agarwal, A. Awan, and D. Roth, "Learning to detect objects in images via a sparse, part-based representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, pp. 1475–1490, Nov. 2004.

[2] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Proc. Eur. Conf. Comput. Vis.*, 2004 pp. 469–481.

[3] J. Behley, V. Steinhage, and A. B. Cremers, "Laser-based segment classification using a mixture of bag-of-words," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2013, pp. 4195–4200.

[4] A. Broggi, A. Cappalunga, S. Cattani, and P. Zani, "Lateral vehicles detection using monocular high resolution cameras on terramax," in *Proc. IEEE Intell. Veh. Symp.*, 2008, pp. 1143–1148.

[5] A. Coates and A. Y. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 921–928.

[6] J. Cui, F. Liu, Z. Li, and Z. Jia, "Vehicle localisation using a single camera," in *Proc. IEEE Intell. Veh. Symp.*, 2010, pp. 871–876.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2005, pp. 886–893.

[8] A. de la Escalera, J. M. Armingol, and M. Mata, "Traffic sign recognition and analysis for intelligent vehicles," *Image Vis. Comput.*, vol. 21, no. 3, pp. 247–258, 2003.

[9] A. De La Escalera, L. E. Moreno, M. A. Salichs, and J. M. Armingol, "Road traffic sign detection and classification," *IEEE Trans. Ind. Electron.*, vol. 44, no. 6, pp. 848–859, Dec. 1997.

[10] S. K. Divvala, A. A. Efros, and M. Hebert, "How important are deformable parts in the deformable parts model?" in *Proc. Eur. Conf. Comput. Vis. Workshop*, 2012, pp. 31–40.

[11] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1532–1545, Jan. 2014.

[12] P. Dollár, S. Belongie, and P. Perona, "The fastest pedestrian detector in the west," in *Proc. Bri. Conf. Mach. Vis.*, 2010, pp. 1–11.

[13] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *Proc. Bri. Conf. Mach. Vis.*, 2009, pp. 1–11.

[14] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[15] C. Fang, S. Chen, and C. Fuh, "Road-sign detection and tracking," *IEEE Trans. Veh. Technol.*, vol. 52, no. 5, pp. 1329–1341, Sep. 2003.

[16] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2009.

[17] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting (with discussion and a rejoinder by the authors)," *J. Ann. Stat.*, vol. 28, no. 2, pp. 337–407, 2000.

[18] X. W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, and N. Shevtsova, "Recognition of traffic signs based on their colour and shape features extracted using human vision models," *J. Vis. Commun. Image Rep.*, vol. 17, no. 4, pp. 675–685, 2006.

[19] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[20] A. Geiger, C. Wojek, and R. Urtasun, "Joint 3d estimation of objects and scene layout," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1467–1475.

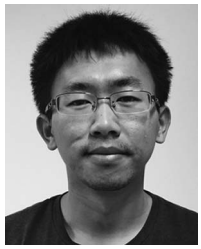
[21] P. Getreuer, "Automatic color enhancement (ace) and its fast implementation," *Image Process. Line*, vol. 2, pp. 266–277, 2012.

[22] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 580–587.

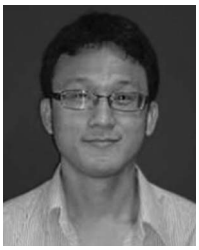
[23] H. Gómez-Moreno, S. Maldonado-Bascón, P. Gil-Jiménez, and S. Lafuente-Arroyo, "Goal evaluation of segmentation algorithms for traffic sign recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 4, pp. 917–930, Jul. 2010.

[24] A. González, G. Villalonga, J. Xu, D. Vázquez, J. Amores, and A. M. López, "Multiview random forest of local experts combining rgb

and lidar data for pedestrian detection," in *Proc. IEEE Intell. Veh. Symp.*, 2015, pp. 356–361.
[25] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: Data mining, inference and prediction," *J. Math. Intell.*, vol. 27, no. 2, pp. 83–85, Mar. 2005.
[26] M. Hejrati and D. Ramanan, "Analyzing 3d objects in cluttered images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 602–610.
[27] S. Houben, "A single target voting scheme for traffic sign detection," in *Proc. IEEE Intell. Veh. Symp.*, 2011, pp. 124–129.
[28] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German traffic sign detection benchmark," in *Proc. Int. Joint Conf. Neural Netw.*, 2013, pp. 1–8.
[29] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surveys*, vol. 31, no. 3, pp. 264–323, Sep. 1999.
[30] R. Janssen, W. Ritter, F. Stein, and S. Ott, "Hybrid approach for traffic sign recognition," in *Proc. IEEE Intell. Veh. Symp.*, 1993, pp. 390–395.
[31] C. Kuo and R. Nevatia, "Robust multi-view car detection using unsupervised sub-categorization," in *Proc. App. Comput. Vis. Workshop*, 2009, pp. 1–8.
[32] W. Kuo and C. Lin, "Two-stage road sign detection and recognition," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2007, pp. 1427–1430.
[33] S. Kyo, T. Koga, K. Sakurai, and S. Okazaki, "A robust vehicle detecting and tracking system for wet weather conditions using the imap-vision image processing board," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 1999, pp. 423–428.
[34] B. Li, T. Wu, and S. Zhu, "Integrating context and occlusion for car detection by hierarchical and-or model," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 652–667.
[35] M. Liang, M. Yuan, X. Hu, J. Li, and H. Liu, "Traffic sign detection by ROI extraction and histogram features-based recognition," in *Proc. Int. Joint Conf. Neural Netw.*, 2013, pp. 1–8.
[36] H.-T. Lin, C.-J. Lin, and R. C. Weng, "A note on Platt's probabilistic outputs for support vector machines," *Mach. Learn.*, vol. 68, no. 3, pp. 267–276, 2007.
[37] C. Long, X. Wang, G. Hua, M. Yang, and Y. Lin, "Accurate object detection with location relaxation and regionlets re-localization," in *Proc. Asian Conf. Comput. Vis.*, 2014, pp. 3000–3016.
[38] G. B. Loy and N. M. Barnes, "Fast shape-based road sign detection for a driver assistance system," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2004, pp. 70–75.
[39] S. Maldonado-Bascón, S. Lafuente-Arroyo, P. Gil-Jiménez, H. Gómez-Moreno, and F. López-Ferreras, "Road-sign detection and recognition based on support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 264–278, Jun. 2007.
[40] E. Martinez *et al.*, "Driving assistance system based on the detection of head-on collisions," in *Proc. IEEE Intell. Veh. Symp.*, 2008, pp. 913–918.
[41] H. Masnadi-Shirazi and N. Vasconcelos, "Cost-sensitive boosting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 294–309, Mar. 2010.
[42] M. Mathias, R. Timofte, R. Benenson, and L. J. V. Gool, "Traffic sign recognition-how far are we from the solution?" in *Proc. Int. Joint Conf. Neural Netw.*, 2013, pp. 1–8.
[43] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1484–1497, Oct. 2012.
[44] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, vol. 2, pp. 849–856.
[45] E. Ohn-Bar and M. M. Trivedi, "Fast and robust object detection using visual subcategories," in *Proc. IEEE Conf. Comput. Vis. Pattern. Recog. Workshop*, pp. 179–184, 2014.
[46] E. Ohn-Bar and M. M. Trivedi, "Learning to detect vehicles by clustering appearance patterns," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2511–2521, Mar. 2015.
[47] S. Paisitkriangkrai, C. Shen, and A. van den Hengel, "Asymmetric pruning for learning cascade detectors," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1254–1267, Feb. 2014.
[48] S. Paisitkriangkrai, C. Shen, and A. van den Hengel, "Strengthening the effectiveness of pedestrian detection with spatially pooled features," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 546–561.
[49] B. Pepik, M. Stark, P. V. Gehler, and B. Schiele, "Occlusion patterns for object class detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3286–3293.
[50] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Multi-view and 3d deformable part models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 11, pp. 2232–2245, Mar. 2015.
[51] N. Pettersson, L. Petersson, and L. Andersson, "The histogram feature-a resource-efficient weak classifier," in *Proc. IEEE Intell. Veh. Symp.*, 2008, pp. 678–683
[52] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Proc. Adv. Large Margin Classifiers*, 1999, pp. 61–74.
[53] V. A. Prisacariu, R. Timofte, K. Zimmermann, I. Reid, and L. J. V. Gool, "Integrating object detection with 3d tracking towards a better driver assistance system," in *Proc. Int. Conf. Pattern Recog.*, 2010, pp. 3344–3347.
[54] Z. Qui, D. Yao, Y. Zhang, D. Ma, and X. Liu, "The study of the detection of pedestrian and bicycle using image processing," in *Proc. IEEE Trans. Intell. Transp. Syst.*, 2003, vol. 1, pp. 340–345.
[55] S. Rogers and N. P. Papanikolopoulos, "Counting bicycles using computer vision," in *Proc. IEEE Conf. Intell. Transp. Syst.*, 2000, pp. 33–38.
[56] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *Proc. Int. Joint Conf. Neural Netw.*, 2011, pp. 2809–2813.
[57] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 4, pp. 1773–1795, Jul. 2013.
[58] R. Timofte, K. Zimmermann, and L. J. V. Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," in *Proc. App. Comput. Vis. Workshop*, 2009, pp. 1–8.
[59] J. J. Y. Torres, L. M. Bergasa, R. Arroyo, and A. Lazaro, "Supervised learning and evaluation of kitti's cars detector with DPM," in *Proc. IEEE Intell. Veh. Symp.*, 2014, pp. 768–773.
[60] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 589–600.
[61] A. Vedaldi and B. Fulkerson, "Vlfeat: An open and portable library of computer vision algorithms," in *Proc. IEEE Int. Conf. Multimedia*, 2010, pp. 1469–1472.
[62] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, May 2004.
[63] G. Wang, G. Ren, Z. Wu, Y. Zhao, and L. Jiang, "A robust, coarse-to-fine traffic sign detection method," in *Proc. Int. Joint Conf. Neural Netw.*, 2013, pp. 1–5.
[64] H. Wang, Q. Chen, and W. Cai, "Shape-based pedestrian/bicyclist detection via onboard stereo vision," in *Proc. Multiconf. Comput. Eng. Syst. Appl.*, 2006, pp. 1776–1780.
[65] J. Wang, G. Bebis, and R. Miller, "Overtaking vehicle detection using dynamic and quasi-static background modeling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. Workshop*, 2005, pp. 64–64.
[66] X. Wang, T. X. Han, and S. Yan, "An HOG-LBP human detector with partial occlusion handling," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 32–39.
[67] X. Wang, M. Yang, S. Zhu, and Y. Lin, "Regionlets for generic object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 17–24.
[68] J. Wu, S. C. Brubaker, M. D. Mullin, and J. M. Rehg, "Fast asymmetric learning for cascade face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 3, pp. 369–382, Mar. 2008.
[69] J. Yan, X. Zhang, Z. Lei, S. Liao, and S. Z. Li, "Robust multi-resolution pedestrian detection in traffic scenes," in *Proc. IEEE Conf. Comp. Vis. Pattern Recog.*, 2013, pp. 3033–3040.



Qichang Hu received the bachelor's degree in computer science from The University of Adelaide, Adelaide, S.A., Australia, in 2012. He is currently working toward the Ph.D. degree with the Australian Centre for Visual Technologies, The University of Adelaide. His research interests include deep learning, object detection, and machine learning.



Sakrapee Paisitkriangkrai received the bachelor's degree in computer engineering, the master's degree in biomedical engineering, and the Ph.D. degree from the University of New South Wales, Sydney, N.S.W., Australia, in 2003 and 2010, respectively.

He is currently a Postdoctoral Researcher with the Australian Centre for Visual Technologies, The University of Adelaide, Adelaide, S.A., Australia. His research interests include pattern recognition, image processing, and machine learning.

Chunhua Shen received the bachelor's degree from Nanjing University, Nanjing, China, the master's degree from the Australian National University, Canberra, A.C.T., Australia, and the Ph.D. degree from The University of Adelaide, Adelaide, Australia.

From 2012 to 2016, he held an Australian Research Council Future Fellowship. He is currently a Professor with the School of Computer Science, The University of Adelaide. He was with the computer vision program at National ICT Australia (NICTA), Canberra Research Laboratory, for about six years. His research interests are in the intersection of computer vision and statistical machine learning.

Anton van den Hengel received the bachelor's degree in mathematical science, the B.L. degree, the master's degree in computer science, and the Ph.D. degree in computer vision from The University of Adelaide, Adelaide, S.A., Australia, in 1991, 1993, 1994, and 2000, respectively.

He is currently a Professor with the School of Computer Science, The University of Adelaide, where he is also the Founding Director of the Australian Centre for Visual Technologies, Interdisciplinary Research Centre, with a focus on innovation in the production and analysis of visual digital media.

Fatih Porikli received the Ph.D. degree from NYU, New York, NY, USA, in 2002.

He is currently a Professor with the Research School of Engineering, Australian National University, Canberra, A.C.T., Australia. He is also acting as the Leader of the Computer Vision Group at NICTA, Sydney, N.S.W., Australia. Previously, he served as a Distinguished Research Scientist at Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. He has contributed broadly to object detection, motion estimation, tracking, image-based representations, and video analytics. He has coedited two books, namely, Video Analytics for Business Intelligence and Handbook on Background Modeling and Foreground Detection for Video Surveillance. Prof. Porikli is a Fellow of the IEEE. He served as the General and Program Chair of several IEEE conferences in the past. He is an Associate Editor of five journals, namely, the IEEE Signal Processing Magazine, SIAM Imaging Sciences, EURASIP Journal of Image and Video Processing, Springer Journal on Machine Vision Applications, and Springer Journal on Real-time Image and Video Processing. His publications won three best paper awards, and he was a recipient of the R&D 100 Award in the Scientist of the Year category in 2006.