Machine Vision and Applications (2011) 22:359–375
DOI 10.1007/s00138-009-0231-x

ORIGINAL PAPER

In-vehicle camera traffic sign detection and recognition

Andrzej Ruta · Fatih Porikli · Shintaro Watanabe · Yongmin Li

Received: 9 March 2009 / Revised: 16 July 2009 / Accepted: 19 October 2009 / Published online: 3 December 2009

Abstract In this paper, we discuss theoretical foundations and a practical realization of a real-time traffic sign detection, tracking and recognition system operating on board of a vehicle. In the proposed framework, a generic detector refinement procedure based on mean shift clustering is introduced. This technique is shown to improve the detection accuracy and reduce the number of false positives for a broad class of object detectors for which a soft response's confidence can be sensibly estimated. The track of an already established candidate is maintained over time using an instance-specific tracking function that encodes the relationship between a unique feature representation of the target object and the affine distortions it is subject to. We show that this function can be learned on-the-fly via regression from random transformations applied to the image of the object in known pose. Secondly, we demonstrate its capability of reconstructing the full-face view of a sign from substantial view angles. In the recognition stage, a concept of class similarity measure learned from image pairs is discussed and its realization using *SimBoost*, a novel version of AdaBoost algorithm, is analyzed. Suitability of the proposed method for solving multi-class traffic sign classification problems is shown experimentally for different feature representations of an image. Overall performance of our system is evaluated based on a prototype C++ implementation. Illustrative output generated by this demo application is provided as a supplementary material attached to this paper.

Keywords Traffic sign recognition · Confidence-weighted mean shift · Regression tracking · SimBoost

1 Introduction

Road signs are an inherent part of the traffic environment. They are designed to regulate flow of the vehicles, give specific information to the traffic participants, or warn against unexpected road circumstances. Perception and fast interpretation of road signs is crucial for the driver's safety. Public services responsible for the traffic infrastructure maintenance mount the signs on poles by the roadside, over the highway lanes, and in other places in a way ensuring that they are easy to spot without distracting the driver's attention from manoeuvring the vehicle. Also, the sign pictograms are designed and standardized in accordance with the rule of maximizing simplicity and distinctiveness. However, under certain conditions such as high visual clutter, adverse illumination, or rainfall, perception of traffic signs can be significantly hampered. Purely physiological factors such as excitement, irritation, or fatigue are known to further reduce the visual concentration of a human and can hence put the driver's life at risk, while driving at high speed in particular.

Electronic supplementary material The online version of this article (doi:10.1007/s00138-009-0231-x) contains supplementary material, which is available to authorized users.

A. Ruta (✉) · Y. Li
School of Information Systems, Computing and Mathematics,
Brunel University, Uxbridge, Middlesex UB8 3PH, UK
e-mail: aruta@agh.edu.pl; Andrzej.Ruta@brunel.ac.uk

Y. Li
e-mail: Yongmin.Li@brunel.ac.uk

F. Porikli
Mitsubishi Electric Research Laboratories, 201 Broadway,
Cambridge, MA 02139, USA
e-mail: fatih@merl.com

S. Watanabe
Advanced Technology R&D Center,
Mitsubishi Electric Corporation, Amagasaki, Japan

For the above reasons, automation of the road sign detection and recognition process was found a natural direction to follow as soon as video processing became attainable on a computing machine. Today, it is considered a critical task in the contemporary visual driver assistance systems. However, reliability of such systems still does not meet our expectations and a large space for improvement is left.

1.1 Related work

Different approaches were adopted in the past for detecting road signs. In the older studies, e.g. [1,2], as well as in many recent ones, e.g. [3,4], it was common to employ a heuristic encompassing prior knowledge about the traffic signs in order to (1) define how to pre-segment the scene to find the interest regions, and (2) define the acceptable geometrical relationships between the sign parts with respect to shape and color. The major deficiency of these methods was a lack of solid theoretical foundations and high parametrization. In other studies, e.g. [5], neural networks were used to model the above-mentioned shape and appearance properties of traffic signs. A more convincing, parameter-free method for detecting road signs was proposed by Bahlmann et al. [6]. They utilized the AdaBoost algorithm [7] and the rejection cascade framework [8] to learn the most discriminative, color-parametrized Haar wavelet filters for road sign representation. Their system demonstrated a good detection rate and was reported to yield very few false alarms at an average processing speed of 10 fps. In several studies, e.g. [1,9,5], the problem of tracking of the observed road signs over time was addressed. However, the proposed frameworks, with the exception of the two-camera system in [9], never went beyond a relatively simple scheme based on a predefined motion model and some sort of geometrical Kalman filtering.

For sign classification, a baseline approach involves cross-correlation template matching. It was used, e.g., in [1,9]. This technique is known to be useful only on condition that the object in the tested image can be well aligned with the templates. In practice, this is often difficult to achieve in the automatic sign detection systems, especially when the target is seen against cluttered background or is affected by geometrical distortion. Other feature-based methods involve neural networks [2,10–14] or kernel density estimation [15] and were shown to offer relatively good classification accuracy. Gao et al. [16] employed the biologically-inspired vision models to represent both color and shape features of the traffic signs. They achieved a promising recognition rate for static images of signs affected by substantial noise and geometric transformations. An interesting concept of a trainable, class-specific similarity measure was introduced recently by Paclík et al. [17]. This measure is based on the discriminative local image regions where the target class differs possibly the most from all other considered classes. A classifier utilizing this measure was shown successful in solving relatively simple road sign classification problems. A similar approach was further presented by Ruta et al. [4] who adapted this method to infer the discriminative sign representations using a single template image per class.

In this paper, we present a unified framework for detection, tracking and recognition of traffic signs, which alleviates the shortcomings of many previous approaches. At the detection stage, we focus on the problem of high sensitivity of the existing object detection techniques. A generic refinement procedure based on a modified mean shift clustering is proposed and evaluated with two different sign detectors. The best-performing refined detector is selected for the prototype system implementation. For tracking of the existing road sign candidates, we employ a trainable regression function that correlates the target appearance with the parameters of its affine deformations. As a result, geometrical sign distortions can be compensated on-line, making our detector pose-invariant and hence more accurate. Ability of the proposed tracker to reconstruct the full-face view of a sign seen from various view angles is shown experimentally using synthetic image sequences. Finally, we build a traffic sign classifier based on the concept of a trainable class similarity. A novel AdaBoost-like algorithm, called *SimBoost*, is utilized to learn a robust sign similarity measure from image pairs labeled either “same” or “different”. This measure is further directly used within the nearest neighbor framework to distinguish between multiple road sign classes. The discriminative power of the classifiers trained using SimBoost is demonstrated for different feature representations of the image. Apart from testing the proposed detection, tracking and recognition approaches as stand-alone algorithms, we also build a demo implementation of a real-time system integrating all three components. This system is evaluated using real-life video captured from a moving vehicle in urban traffic scenes.

The rest of this paper is divided into five parts. In Sect. 2, our road sign detection method is discussed. In Sect. 3, we develop a pose-invariant sign tracker. Section 4 explains how the concept of a trainable similarity is used to construct a robust traffic sign classifier. In Sect. 5, an extensive experimental evaluation of our algorithms is presented. Finally, in Sect. 6, we conclude our work.

2 Sign detection

Traffic sign detection is a difficult problem as it involves discriminating a large gamut of diverse objects from a generally unknown background. Taking this diversity into account, we focus in this work on a subset of circular signs that are well-constrained in terms of the size, shape, and contained ideogram. In Sect. 2.1, a fast, application-specific quad-tree focus operator is introduced. We use it to quickly discard the irrelevant fragments of the scene and locate the sparse regions that might contain traffic signs. In Sect. 2.2, we briefly discuss two practically useful sign detection methods and their common limitations. In Sect. 2.3, a detection refinement scheme is proposed in order to improve the selectivity and the accuracy of these detectors, which are on their own over-sensitive.

2.1 Quad-tree focus of attention

In order to detect the new road sign candidates emerging in the scene, it is first necessary to reduce the search area. Dense scanning of the entire image wastes processor time and is hence unlikely to work in real time, even using a detector based on the well-known Haar wavelets [8], probably computationally the cheapest available image descriptors. One generic method for quick elimination of the irrelevant regions of an image is a rejection classifier cascade introduced by Viola and Jones [8]. Not denying the potential of this technique, we should yet note that it still involves a sequential, pixel-by-pixel processing of the input image and requires complex, time-consuming training. Below, we briefly outline a much simpler generic search reduction technique that is tuned to our specific application, and which can be used solely or in chain with Viola and Jones, as well as other methods.

The proposed quad-tree attention operator associates a scalar feature value $\mathrm{v}(x, y)$ with each pixel of the image I: $\mathbf{V}(I) = \{\mathrm{v}(x, y) : x = 1, \dots, W, y = 1, \dots, H\}$, where $W \times H$ is the image size. A region $R(x_1, y_1, x_2, y_2)$ is considered relevant if the sum of the contained pixels' feature values is greater that a predefined threshold $t_{\min}$. If an integral feature image is available:

$$\Sigma(I) = \{\upsilon(x, y) : \upsilon(x, y) = \sum_{i \le x, j \le y} \mathrm{v}(i, j),\quad x = 1, \dots, W, y = 1, \dots, H\}, \tag{1}$$

then this sum can be computed using only 4 array referencing operations and 3 additions/subtractions:

$$\mathrm{v}(R(x_1, y_1, x_2, y_2)) = \upsilon(x_2, y_2) - \upsilon(x_1, y_2) - \upsilon(x_2, y_1) + \upsilon(x_1, y_1). \tag{2}$$

If the threshold $t_{\min}$ is set to an appropriately low value that can be used to reliably discriminate between the relevant and irrelevant fragments of the scene at the smallest considered scale, then RoIs can be rapidly identified using the following recursive algorithm:

Algorithm 1 Quad-tree RoI extraction.

```
input: image $I_{W \times H}$, minimum “amount” of feature contained in a RoI,
    $t_{\min}$, minimum region size, $s_{\min}$
output: set of RoIs, $S$
build a feature map $\mathbf{V}(I)$
build an integral feature map $\Sigma(I)$
initialize an empty set of relevant smallest-scale regions $\mathbf{C} = \emptyset$
call $ProcessRegion(R(1, 1, W, H), t_{\min}, s_{\min}, \mathbf{C})$
cluster regions in $\mathbf{C}$
populate $S$ with bounding rectangles of found clusters
```

Algorithm 2 Procedure *ProcessRegion*.

```
input: region $R_{w \times h}$, minimum “amount” of feature contained in a RoI,
    $t_{\min}$, minimum region size, $s_{\min}$, a set of relevant smallest-scale
    regions, $\mathbf{C}$
compute the amount of feature in $R$
if $\min\{w, h\} \geq s_{\min}$ then
    if $\mathrm{v}(R) > t_{\min}$ then
        set $w = w/2, h = h/2$
        for each quarter $Q_j$ of $R$ do
            call $ProcessRegion(Q_j, t_{\min}, s_{\min}, \mathbf{C})$
        end for
    end if
else
    add $R$ to $\mathbf{C}$
end if
```

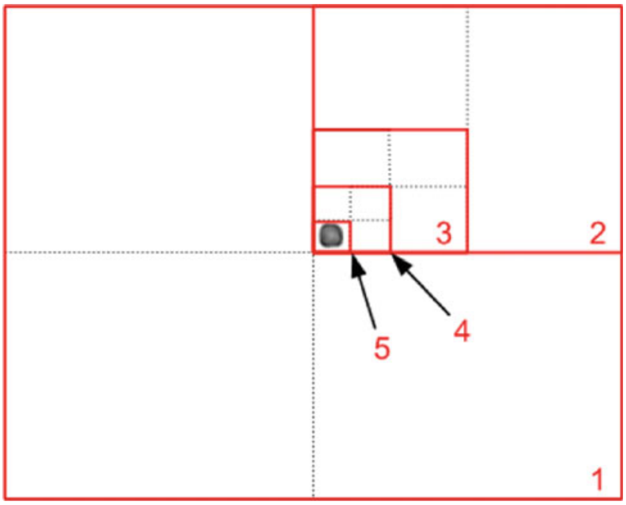


Fig. 1 Quad-tree interest region finding algorithm. The consecutive numbers correspond to the older of quarters being processed

Algorithm 1 is illustrated in Fig. 1. We tailor it to our needs by associating the relevance of a given image region with the amount of contained contrast measured with respect to the appropriate color channels. The traffic signs we focus on always have a distinctive color rim. Therefore, the input image is first filtered using the appropriate set of filters intended to amplify certain colors and suppress any other. Suitable filters used in this work are:

$$\begin{aligned} f_R(\mathbf{x}) &= \max\left(0, \min\left(\frac{x_R - x_G}{s}, \frac{x_R - x_B}{s}\right)\right) \\ f_B(\mathbf{x}) &= \max\left(0, \min\left(\frac{x_B - x_R}{s}, \frac{x_B - x_G}{s}\right)\right), \end{aligned} \tag{3}$$

where x_R, x_G, x_B denote the red, green and blue components, respectively, of an input RGB pixel and $s = x_R + x_G + x_B$.

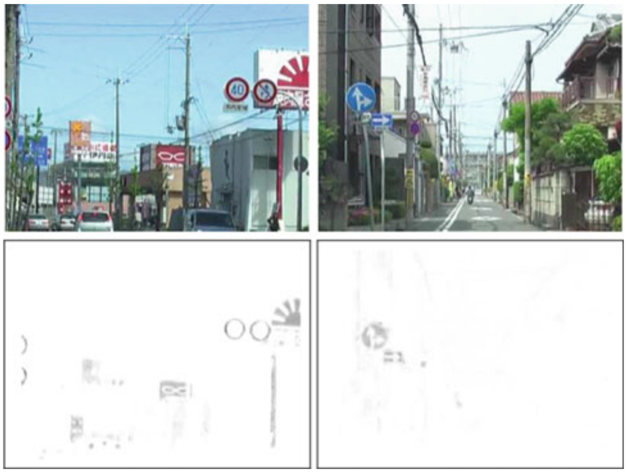


Fig. 2 The effect of applying the color filters (3) to the example RGB images (*top row*): *red color* filter (*bottom-left*), *blue color filter* (*bottom-right*)

The above filters effectively extract the red and blue fragments of the image, which is shown in Fig. 2.

The RoI selection algorithm starts with applying filters (3) to the input image. Then, two feature images $\mathbf{V}_R(I)$ and $\mathbf{V}_B(I)$ are constructed as gradient magnitude maps for each color. Similarly, two integral images $\Sigma_R(I)$ and $\Sigma_B(I)$ are build from $\mathbf{V}_R(I)$ and $\mathbf{V}_B(I)$, respectively. A region corresponding to the entire image is now checked against the total color gradient contained using a maximum of the values picked from both integral images. As it is typically far above the predefined threshold, the image is subdivided into four quarters and each quarter is recursively processed in the same way. The process is stopped either when the current input region contains less gradient than the threshold or upon reaching the minimum region size. The above-threshold lowest-level regions are clustered and the ultimate RoIs are constructed as bounding rectangles of the found clusters. This way we can very quickly discard the irrelevant fragments of the scene, e.g. sky or asphalt, which either do not contain the interest colors and/or are too low-contrasting. Note that the total amount of color-specific gradient constitutes a much stronger filter than simply the total amount of characteristic color, which fails in presence of uniform reddish or blueish regions, e.g. sky or large color billboards.

Further processing, i.e. the true object detection, is done only in the found interest regions. Below, two useful techniques for localized traffic sign detection are presented.

2.2 Sign detectors

In-vehicle road sign detector must be both sufficiently discriminative and computationally inexpensive so that it can work in real time even in the worst-case scenario, when a large part of the scene has to be scanned. We evaluate here two detection techniques which seem particularly useful for road sign detection: Haar rejection cascade and the Hough transform.

The Haar cascade of boosted classifiers for object detection has been thoroughly discussed in [8]. This technique revolves around an idea of building a multi-stage classifier in which at each new layer the layer-specific binary classifier is trained in a supervised way using all available true positive images and only these negative (background) images that were misclassified in the previous layer. This way the cascade is arranged such that in runtime the most top-level classifier can quickly reject a majority of irrelevant parts of the scene, leaving the more ambiguous regions to process by the classifier in the next layer. This recursive process is further continued for the increasingly "hard" regions and only the regions successfully passing the last layer are retained. The AdaBoost algorithm [7] is used to train the classifier in each layer and the expected performance specifications of the cascade are given as the training parameters. For example, the boosted classifier in each layer might be set to grow until it can correctly classify 99% of the true positives from the previous layer and not less than 50% of the previous layer's false positives. The third parameter, maximum overall false positive rate of the cascade is provided to determine when to stop the training process. Robustness of the cascade setup in combination with using simple Haar wavelet filters underlying each weak classifier makes the cascade relatively inexpensive in terms of the computation involved.

Although there is a common agreement on the usefulness of the rejection cascade for general object detection, this approach has also many disadvantages. As all discriminative methods, to achieve a good generalization performance, the classifier must see in the training stage a sufficiently large number of object examples describing the distribution of the target class appearance. Therefore, it may be insufficiently discriminative if the intra-class variability is too high and the number of training images available is small. On the other hand, if this number is large, the training process may become extremely lengthy. In addition, many cascade implementation details are technically challenging. For example, it is unclear how to efficiently generate negative images to populate the training pool of the classifiers located deep in the cascade, say, at the nth level. An overall false positive rate of the cascade up to the level $n-1$ might already be very low. This implies that random selection of each background region from the image not containing the target object might require many repetitions (until this region is classified as false positive by the so-far built cascade) and hence can be extremely time-consuming.

The second detection technique we evaluate is based on the Hough transform (HT) [18]. The purpose of this method is to find the imperfect instances of objects within a certain class of shapes by a voting procedure carried out in a parameter space.

The simpler the parametric description of a shape, the more suitable this approach is in real-time vision. In our case, most of the popular road signs are either circles or equiangular polygons: equilateral triangles, squares, or octagons (STOP sign), depending on the country. To detect circular structures in an image, a well-known circular Hough transform can be used, which involves voting in a three-parameter space. For regular polygons a generalized method has been proposed by Barnes et al. [3]. A desirable property of these HT variants is their accuracy and tolerance to noise and partial occlusions. Among major disadvantages is their sensitivity to the quality of the input edge map, which in turn depends on the external factors, such as scene illumination.

Both techniques are known to suffer from the problem of producing multiple, mostly redundant, positive hypotheses around the true target instances. As processing of each such hypothesis separately is impractical, the output of an over-sensitive detector is typically subject to some sort of postprocessing intended to produce a single, accurately fit shape per instance. Below, we propose such a postprocessing technique based on the mean shift clustering.

2.3 Confidence-weighted mean shift refinement

Accuracy of an over-sensitive detector that produces redundant positive hypotheses around the true object candidates must necessarily be improved to make it useful for real-time operation. One possible way of doing that is to consider the detector's response space a probability distribution with modes to be found. The Mean Shift algorithm [19] is a well-established kernel density estimation technique that can be used to find the modes of the underlying distribution. However, the original mean shift formulation does not account for the possibly varying relevance of the input points. Below, we propose a simple modification, called *Confidence-Weighted Mean Shift*, which alleviates this shortcoming by incorporating the confidence of the detector's responses into the mode finding procedure. It is shown that such a refinement procedure can be applied to the output of any detector that yields a soft decision or can be modified to do so.

We first characterize each positive hypothesis of the detector with a vector, $\mathbf{x}_j = [x_j, y_j, s_j]$, encoding the object's centroid position and its scale. In addition, $\mathbf{x}_j$ is assigned a confidence value, q_j, which is related to the soft response of the detector. In the case of a single boosted classifier in each layer of the Haar cascade, such a confidence measure can naturally be related to the distance of the response from the linear decision boundary:

$$q_j = q(\mathbf{x}_j) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}_j), \tag{4}$$

where $h_t(\mathbf{x}_j)$ denote the weak classifier responses, $\alpha_t = \log(\frac{1-e_t}{e_t})$, and e_t are the error rates of the weak classifiers. In the case of an entire cascade, the confidence formula can no longer be treated as a distance from the decision boundary, which is now non-linear. However, it can be approximated by a sum of $q_j^{(k)}$ terms over all K cascade layers, taking the modified thresholds t_k in each layer into account:

$$q_j = \sum_{k=1}^{K} q_j^{(k)} = \sum_{k=1}^{K} \sum_{t=1}^{T_k} (\alpha_t h_t(\mathbf{x}_j) - t_k). \tag{5}$$

In the case of a Hough detector, the confidence of each above-threshold circle picked from the accumulator array can simply be measured with the normalized number of votes cast for this circle. In general, confidence q_j can be expressed with any quantity that evaluates to a numerical, comparable value, and is indicative of the likelihood of the target object's presence in a given position and scale.

Assuming that $f(\mathbf{x})$ is the underlying distribution of $\mathbf{x}$, the mean shift algorithm iteratively finds the stationary points of the estimated density via alternate computation of the mean-shift vector, and translation of the current kernel window by this vector, until convergence (for details refer to [19]). Our modified mean-shift vector is made sensitive to the confidence of the input points in the following way:

$$\mathbf{m}_{h,G} = \frac{\sum_{j=1}^{n} \mathbf{x}_j q_j g \left\| \frac{\mathbf{x}-\mathbf{x}_j}{h} \right\|^2}{\sum_{j=1}^{n} q_j g \left\| \frac{\mathbf{x}-\mathbf{x}_j}{h} \right\|^2} - \mathbf{x}, \tag{6}$$

where $g(\cdot)$ is the underlying gradient density estimator and h is the bandwidth parameter determining the scale of the estimated density. Incorporating the confidence terms q_j in (6) is equivalent to amplifying the density gradients pointing towards the more reliably detected circle locations. The found modes of $f(\mathbf{x})$ correspond to the new road sign candidates which we need to track in the consecutive frames of the input video.

3 Tracking

To recognize traffic signs from a moving vehicle, it is crucial to have a view-independent object detector. Training such a detector directly exhibits serious difficulties as it requires feature descriptors to be both: highly discriminative and pose-invariant. Our method of solving such a detection problem follows a different strategy and has been shown successful in several studies, e.g. [20,21]. Instead of devising a pose-independent feature representation of the target, we learn an application-specific motion model from the random affine transformations applied to the full-face view of a detected sign. This model is learned via regression using

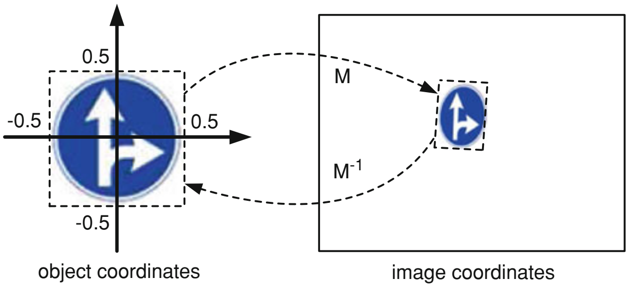


Fig. 3 Affine transformation matrix and its inverse

the Lie algebra of the motion group, and encodes the correlations between a unique feature representation of a sign and the affine transformations it is subject to while being approached by a camera. In Sect. 3.1, we provide the theoretical foundations of our regression tracking algorithm. Section 3.2 describes a concrete realization of this method.

3.1 Tracking as a regression problem

Let $\mathbf{M}$ be an affine matrix that transforms a unit square at the origin in the object coordinates to the affine region enclosing the target object in the image coordinates:

$$\mathbf{M} = \begin{pmatrix} \mathbf{A} & \mathbf{t} \\ 0 & 1 \end{pmatrix}, \tag{7}$$

where $\mathbf{A}$ is a 2×2 nonsingular matrix and $\mathbf{t} \in \mathbb{R}^2$. Let $\mathbf{M}^{-1}$ be an inverse transform that maps the object region from image coordinates back to the object coordinates, as shown in Fig. 3. Our goal is to estimate the transformation matrix $\mathbf{M}_t$ at time t, given the observed images up to that point, $I_{0,\dots,t}$, and the initial transformation $\mathbf{M}_0$. $\mathbf{M}_t$ is modeled recursively:

$$\mathbf{M}_t = \mathbf{M}_{t-1} \Delta\mathbf{M}_t, \tag{8}$$

which means that it is sufficient to estimate only the increment $\Delta\mathbf{M}_t$ corresponding to the motion of the target from time $t-1$ to t in object coordinates. It is determined by the regression function f:

$$\Delta\mathbf{M}_t = f\left(\mathbf{o}_t(\mathbf{M}_{t-1}^{-1})\right), \tag{9}$$

where $\mathbf{o}_t(\mathbf{M}_{t-1}^{-1})$ denotes an image descriptor applied to the previously observed image, after mapping it to the unit rectangle.

The regression function $f : \mathbb{R}^m \longmapsto A(2)$ is an affine matrix-valued function, where $A(2)$ denotes a two-dimensional affine transformation. When multiplied on the left by the previous-frame motion matrix, function f gives an accurate pose estimate of the target in the current frame. To learn its parameters, it is necessary to know the initial pose of an object, $\mathbf{M}_0$, and the image I_0 at time t_0. Training examples are generated as pairs $(\mathbf{o}_0^i, \Delta\mathbf{M}_i)$, where $\Delta\mathbf{M}_i$ are random deformation matrices around identity and $\mathbf{o}_0^i = \mathbf{o}_0(\Delta\mathbf{M}_i^{-1}\mathbf{M}_0^{-1})$.

The optimal parameters of f are derived on the grounds of the Lie group theory [22] by minimizing the sum of the squared distances between the pairs of motion matrices: estimated $f(\mathbf{o}_0^i)$ and known $\Delta\mathbf{M}_i$. The affine motion matrices can be considered points on the Lie group with a structure of a six-dimensional differentiable manifold given by (7). An adequate measure of distance between two motion matrices treated as points on the manifold is the minimum length of a curve connecting these points, called geodesic. It is given by:

$$\rho(\mathbf{M}_1, \mathbf{M}_2) = \| \log(\mathbf{M}_1^{-1}\mathbf{M}_2) \|. \tag{10}$$

With (10), the measure of the error to minimize becomes:

$$J = \sum_{i=1}^{n} \rho(f(\mathbf{o}_0^i), \Delta\mathbf{M}_i)^2. \tag{11}$$

Tuzel et al. [21] show that if two vectors $\mathbf{m}_1$ and $\mathbf{m}_2$ can be expanded into motion matrices $\mathbf{M}_1$ and $\mathbf{M}_2$, respectively, then the first-order approximation to the geodesic distance between them is:

$$\rho(\mathbf{M}_1, \mathbf{M}_2) = \|\mathbf{m}_2 - \mathbf{m}_1\|. \tag{12}$$

Therefore, selecting $d = 6$ orthonormal bases on the Lie algebra, the error function in (11) can be computed as a sum of the squared Euclidean distances between the vectors $\log(f(\mathbf{o}_0^i))$ and $\log(\Delta\mathbf{M}_i)$, i.e.:

$$J = \sum_{i=1}^{n} \| \log(f(\mathbf{o}_0^i)) - \log(\Delta\mathbf{M}_i) \|^2. \tag{13}$$

Details of how the above error function is minimized can be found in [21].

3.2 Tracker architecture

The regression tracker introduced in Sect. 3.1 is utilized in our traffic sign recognition system as shown in Fig. 4. Once a candidate sign has been detected for the first time, a new

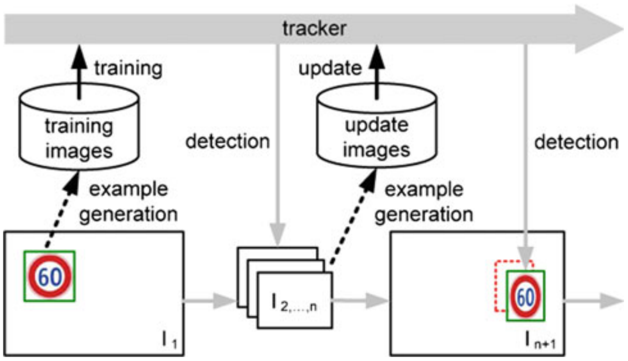


Fig. 4 Operation of a road sign tracker over time. The period between the initial candidate detection and the first tracker update is depicted

tracker is initialized with the region corresponding to the bounding rectangle of the found circle instance, assuming no distortion.[1] At this point a small number of random deformations are generated from the observed image and used for instant training. A map of 6 × 6 regularly spaced 6-bin gradient orientation histograms is used as an object descriptor. The trained tracker is employed to detect the sign in the subsequent frames, each being used to generate and enqueue m new random deformations.

In a realistic traffic scenario the scene is often difficult and changes fast. Therefore, the accuracy of the tracker is likely to deteriorate very quickly. It is a result of the cumulated reconstruction errors caused by: (1) contaminating the object coordinate images with the unwanted background fragments, and (2) changing appearance of the target. To deal with this problem, we update the tracking function after every f_U frames by re-training it on the collected portion of $f_U \times m$ random training transformations. This update process is carried out in a similar way as the initial training, i.e. by minimizing the sum of the squared geodesic distances between the estimated and the known motion matrices, but another constraint is introduced on the difference between the current and the previous regression coefficients (refer to [21] for more details). The updated tracker is used to re-estimate the pose of the observed sign and the space is allocated for a new portion of random transformations.

Finally, the track is assumed to be lost when the sign either gets out of the field of view or when the normalized cross-correlation between its current image in object coordinates and the same image recorded at the last update drops below a predefined threshold. The latter condition prevents the track from running out of control due to the errors cumulated in the regression function.

4 Recognition

Recognition of traffic signs is a hard multi-class problem with an additional difficulty caused by the fact of certain signs being very similar to one another, e.g. speed limits. The approach we have adopted in this work is centered around the concept of trainable similarity that can be inferred from the pairs of examples. When the similarity between any two images can be estimated, any multi-class classification problem can be solved by comparing the similarities between the unknown example and each class's prototype. A tested example belongs to the class of the prototype to which it is the most similar. For robust similarity assessment we use a novel variant of AdaBoost algorithm, called *SimBoost*. It is derived in Sect. 4.1. In Sect. 4.2, we outline how the classifier trained via SimBoost is used to recognize objects in image sequences.

4.1 SimBoost algorithm

Formally, our classifier, $F(\mathbf{x})$, is designed to recognize only two classes: "same" and "different", and is trained using pairs of images, i.e. $\mathbf{x} = (I_1, I_2)$. The pairs representing the same type of sign are labeled $y = 1$ (positive), and the pairs representing two different types are labeled $y = -1$ (negative). A real-valued discriminant function F is learned using a modified AdaBoost algorithm [7] which we call *SimBoost*. We define F as a sum of image features f_j:

$$F(I_1, I_2) = \sum_{j=1}^{N} f_j(I_1, I_2). \tag{14}$$

Each feature evaluates to:

$$f_j(I_1, I_2) = \begin{cases} \alpha & \text{if } d(\phi_j(I_1), \phi_j(I_2)) < t_j \\ \beta & \text{otherwise} \end{cases}, \tag{15}$$

where ϕ_j is a filter defined over a chosen class of image descriptors, d is a generic distance metric that makes sense for such descriptors, and t_j is a feature threshold. In other words, each feature quantifies a local similarity between the input images and responds to this similarity according to whether or not it is sufficient to consider the images as representing the same class.

Let $h(\mathbf{x}) = h_j(I_1, I_2) = d(\phi_j(I_1), \phi_j(I_2))$. Let us also denote by W_+^+ the total weight of these positive examples that are labeled positive by a given weak classifier (true positives), and by W_+^- the total weight of those that are labeled negative (false negatives). By analogy, let W_-^- and W_-^+ be the total weights of true negatives and false positives, respectively. In other words:

$$\begin{aligned} W_+^+ &= \sum_{\substack{k:\ y_k = 1 \\ \wedge\, h_j(\mathbf{x}_k) < t_j}} w_k \qquad & W_+^- &= \sum_{\substack{k:\ y_k = 1 \\ \wedge\, h_j(\mathbf{x}_k) \ge t_j}} w_k \\ W_-^+ &= \sum_{\substack{k:\ y_k = -1 \\ \wedge\, h_j(\mathbf{x}_k) < t_j}} w_k \qquad & W_-^- &= \sum_{\substack{k:\ y_k = -1 \\ \wedge\, h_j(\mathbf{x}_k) \ge t_j}} w_k \end{aligned} . \tag{16}$$

In each boosting round, the filter ϕ_j and the threshold t_j are selected so as to minimize the weighted error of the training examples:

$$e_j = \sum_{\substack{k:\ y_k = 1 \\ \wedge\, h_j(\mathbf{x}_k) \ge t_j}} w_k + \sum_{\substack{k:\ y_k = -1 \\ \wedge\, h_j(\mathbf{x}_k) < t_j}} w_k = W_+^- + W_-^+. \tag{17}$$

[1] This assumption is valid as road signs are detected for the first time at a considerable distance from the camera, where this distance is much greater than the distance of the sign from the camera's optical axis.

Secondly, the optimal values of α and β are found based on the Schapire and Singer's criterion [23] of minimizing:

$$Z = \sum_{k=1}^{M} w_k e^{-y_k f(\mathbf{x}_k)}, \tag{18}$$

where M is the total number of training examples. First, the sum is split as follows:

$$\begin{aligned} Z &= \sum_{k:\, y_k=1} w_k e^{-f(x_k)} + \sum_{k:\, y_k=-1} w_k e^{f(x_k)} \\ &= \sum_{\substack{k:\ y_k = 1 \\ \wedge\, h_j(\mathbf{x}_k) < t_j}} w_k e^{-\alpha} + \sum_{\substack{k:\ y_k = 1 \\ \wedge\, h_j(\mathbf{x}_k) \geq t_j}} w_k e^{-\beta} \\ &\quad + \sum_{\substack{k:\ y_k = -1 \\ \wedge\, h_j(\mathbf{x}_k) < t_j}} w_k e^{\alpha} + \sum_{\substack{k:\ y_k = -1 \\ \wedge\, h_j(\mathbf{x}_k) \geq t_j}} w_k e^{\beta} \\ &= W_+^+ e^{-\alpha} + W_+^- e^{-\beta} + W_-^+ e^{\alpha} + W_-^- e^{\beta}. \end{aligned} \tag{19}$$

Taking partial derivatives of Z with respect to α and β and setting each to zero determine the optimal values of each parameter to be set in a given boosting round:

$$\alpha = \frac{1}{2} \log\left(\frac{W_+^+}{W_-^+}\right) \quad \beta = \frac{1}{2} \log\left(\frac{W_+^-}{W_-^-}\right). \tag{20}$$

The other steps of the SimBoost procedure are similar to those known from the classical AdaBoost algorithm. Specifically, weights of the input image pairs are updated after each boosting round using the well-known exponential loss function:

$$w_k^{(t+1)} = w_k^{(t)} e^{-f_j(\mathbf{x}_k) y_k}, \tag{21}$$

where f_j denotes the feature selected in the current round. Weights of all training examples are then normalized. One difficulty is related to the size of the input pairs space. Note that the total number of possible input pairs for K classes, each containing N images, is $\frac{1}{2}(NK - 1)NK$, while the total number of "same" pairs is only $\frac{1}{2}K(N - 1)N$. Using all possible image pairs for training would make this process intractable, which calls for sampling. Second, the large quantitative imbalance between the number of positive and negative pairs implies that random sampling is inappropriate because it would carry a risk of very few positive pairs being selected. Enforcing more "same" than "different" pairs could, on the other hand, bias the classifier.

We propose the following solution to the above problem. First, the cumulative example weight is defined:

$$c_j = \sum_{k=1}^{j} w_k. \tag{22}$$

Given the total of M examples and the target sample size S, cumulative weights $c_1, \ldots, c_M$ are computed. Then, S times a random number, r, is generated on the interval $[0, c_M]$. The example $\mathbf{x}_j$ with index satisfying $c_j < r < c_{j+1}$ is assigned weight equal to r and put in the target sample. This is equivalent to choosing it multiple times, each with weight 1.

A classifier trained using the SimBoost algorithm yields a decision which is a linear combination of the weak classifiers' responses:

$$l(I_1, I_2) = \text{sign } F(I_1, I_2) = \text{sign}\left(\sum_{j=1}^{N} f_j(I_1, I_2)\right). \tag{23}$$

4.2 Temporal classification

In order to be able to use the binary classifier discussed in Sect. 4.1 for solving a multi-class problem, the classifier's response must be made soft. This can be done in a straightforward way by omitting the sign in the right-hand side expression of Eq. (23), i.e. considering the bare value of function F. This value can be treated as a degree of similarity of the two input images. Let $p_1, \ldots, p_K$ be the prototype images of K targeted classes. If one of the images passed on input of our road sign classifier, say I_1, is a prototype of known class k ($I_1 = p_k$), the classifier assigns such a label to the other, unknown image, that satisfies:

$$l(I) = \arg\max_k F(p_k, I). \tag{24}$$

In other words, $l(I)$ is determined from the prototype to which the tested image is the most similar.

To classify a sequence of images, $I_{1,\ldots,T}$, the maximum rule in (24) is applied to the sum of $F(p_k, I_t)$ terms over all images $I_t, t = 1, \ldots, T$. Each I_t denotes a reconstructed full-face image of a sign obtained by applying the inverse of the transformation matrix $\mathbf{M}_t$ to the frame at time t. In addition, the contribution of the most recent observations is emphasized to reflect the fact that the image of a sign becomes generally clearer as the vehicle approaches the target. The ultimate classifier's decision at time T is determined from:

$$l(I_{1,\ldots,T}) = \arg\max_k \sum_{t=1}^{T} q(t) F(p_k, I_t), \tag{25}$$

where $q(t) = b^{T-t}$, $b \in (0, 1]$, is a relevance of the observation I_t.

5 Experimental results

In this section, we present the experimental evaluation of the proposed road sign detection, tracking and recognition algorithms. Each of the three core modules of the system are first

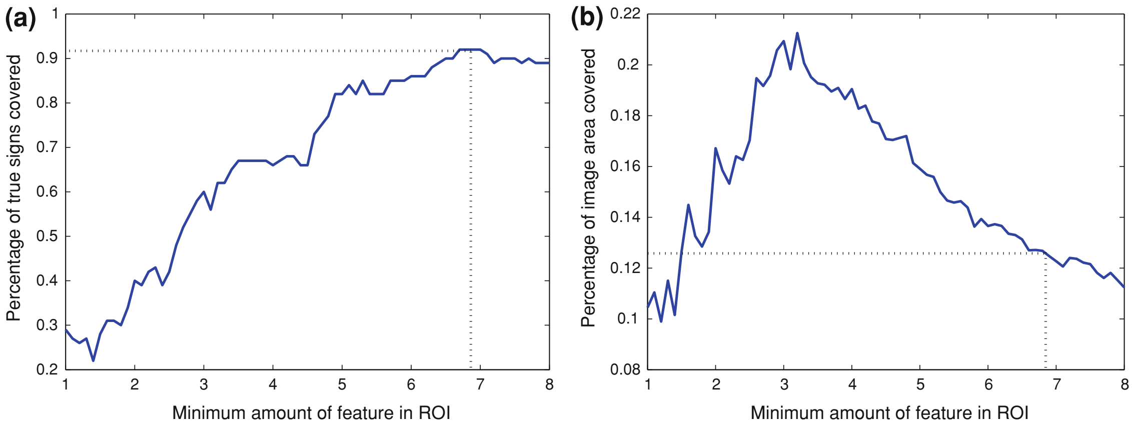


Fig. 5 Determination of threshold $t_{\min}$: **a** the optimal value, $t^{\star}_{\min} \approx 6.9$ corresponds to the maximum percentage of true signs covered by the found RoIs as a function of $t_{\min}$, **b** relationship between the percentage of image area covered and the same values of $t_{\min}$ as in the left plot. $t^{\star}_{\min}$ is marked with dotted lines in both plots

tested as stand-alone components. The quad-tree ROI finding, the two considered detection methods and the detection refinement algorithm are evaluated on the static road sign images in Sect. 5.1. The better-performing refined detector is chosen to be incorporated into the prototype system and the justification of this choice is provided. In Sect. 5.2, we concentrate on the regression tracker and estimate its capability of modeling the affine distortions that the traffic signs are subject to while being approached by a car-mounted camera. A set of synthetic image sequences are generated to facilitate this experiment. Performance of a classifier trained via SimBoost is measured in Sect. 5.3, again using the static images of traffic signs. The feature representation guaranteeing the most correct assessment of the similarity between the two images is determined based on the obtained experimental results. Finally, in Sect. 5.4, we assemble the entire sign detection, tracking, and recognition system and test it on a number of real-life video sequences captured from a moving vehicle.

5.1 Evaluation of road sign detectors

The proposed focus operator is not expected to yield only true road sign regions as this is not possible based on such simple evidence as cumulative color gradient. In cluttered urban scenes this algorithm is likely to produce multiple false RoIs or single overly large RoIs. However, at this stage of the processing it is the most essential to capture as many true positives as possible for given input data, even at the cost of detecting and further analyzing the uninformative fragments of the image too. In practice, if the threshold $t_{\min}$ is set correctly, which is done based on a number of training images with ground truth positions and scales of signs available, the algorithm captures a vast majority of signs in the incoming video, but the reduction of the computation involved is still huge.

In order to quantify the capability of our focus operator of localizing road signs, we have performed an experiment involving realistic traffic images captured with a wide-angle in-vehicle camera in crowded street scenes. The total of 70 high-resolution images depicting 100 road signs were used, each of dimensions 1,920 × 1,088 pixels. In order to find the optimal threshold $t_{\min}$, our algorithm was run for the increasing value of this threshold on these 70 images and the percentage of true signs covered by the produced RoIs was maximized. In Fig. 5 we have illustrated this quantity, as well as the percentage of the total image area covered by the found RoIs, as functions of $t_{\min}$.

The obtained results demonstrate the usefulness of our search region reduction algorithm. While it captures a vast majority of traffic signs in the scene, the average area of the image to be analyzed is only a small fraction of the entire image's area, which dramatically reduces computation. The signs not covered by the found interest regions in this experiment were of very low figure-background contrast and as such could not be captured by the actual detector anyway. It means that there is practically no performance decrease related to using the attention operator.

In order to measure the capability of capturing traffic signs by the detectors outlined in Sect. 2.2, we first tested the Haar cascade and the Hough circle detector without considering the video context. The test image sequences we possess were acquired in the urban areas in Japan, where most of the traffic signs captured were circular. We have, therefore, concentrated on this particular type of sign. We ran each detector in the small regions of the input images around the known ground truth sign locations. Specifically, the size of each analysis region was set to 3 × diameter of a sign located in

Fig. 6 Example images used in the experimental evaluation of the traffic sign detectors

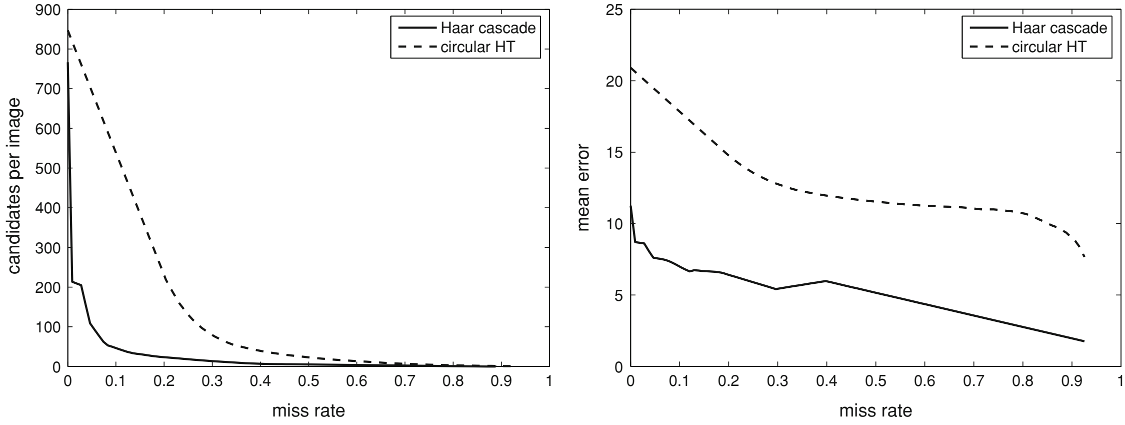


Fig. 7 Relationship between the mean number of candidates per image detected and the miss rate (*left*), and between the mean distance of the detected candidates from the ground truth circles and the miss rate (*right*). Plots are extrapolated outside of the common miss rate range

the region's center. A few example regions used are shown in Fig. 6. The experiment was performed using a total of 8,175 challenging images representing 14 different sign classes and repeated for (1) varying threshold of the classifier in the last cascade layer, and (2) varying threshold in the Hough vote space. To increase the discriminative power of both detectors, we transformed each input image using the color filters (3). In the case of Hough detector, the color-specific edge maps were computed and the HT was run on each of them, pixel-by-pixel. When evaluating the Haar cascade, filters (3), along with a gray-scale transformation, were used to parameterize the Haar wavelets, as proposed by Bahlmann et al. [6]. In order to train the classifier, a set of another 4,218 images was first clustered to reduce the intra-class variability. Then, a separate cascade was trained for each cluster. The test images were scanned by the cascaded classifiers with a 2-pixel step to reduce computation.

For each image a ground truth center position and the radius of a sign was given by a triple: (x_c, y_c, r). Quantities measured were: (1) mean number of candidates per image detected, (2) mean distance between a detected circle and the ground truth circle expressed with a Euclidean metric over the above-mentioned triples, and (3) miss rate, i.e. the percentage of images where no sign was detected. Relationship between the miss rate and the two other quantities is illustrated in Fig. 7. The experiment showed that the Haar cascade is a more accurate road sign detector than the circular Hough transform in the entire range of practically useful operating points. However, this advantage was achieved at the cost of more computation. While the average processing time of a single image was approximately 10 ms for a Hough detector, this time increased to over 20 ms for a Haar cascade.[2] The difference in the accuracy of both detectors can partly be attributed to the nature of voting in Hough space. As it is generally unknown whether the road sign is darker or lighter than the background, the votes coming from the contour pixels are cast on both sides of the circle. Sometimes the number of votes cumulated outside the true sign may be sufficiently high to produce false candidates that are adjacent to it. Besides, the circular Hough transform is relatively insensitive to scale when the input image contains thick edges. In that case it often yields above-threshold responses for a whole range of radii. Regardless of the results of this comparison, both techniques appeared to be impractical when used alone, i.e. without an appropriate postprocessing of the detector's responses.

We have repeated the above experiment, but applying the proposed *Confidence-Weighted Mean Shift* refinement algorithm to the output generated by each detector. Obtained results are shown in Fig. 8. It can be noticed that for both detectors and for significant miss rates, the mean number of detected candidates per image roughly corresponds to the percentage of the images where any candidate was detected. This implies that the proposed detection refinement scheme most likely collapses the multiple positive responses of the detector into a single candidate, which is an intended outcome. The mean error of both detectors is lower with the

[2] For a pixel-by-pixel scanning, the cascaded classifier was approximately 7 times slower than the Hough detector.

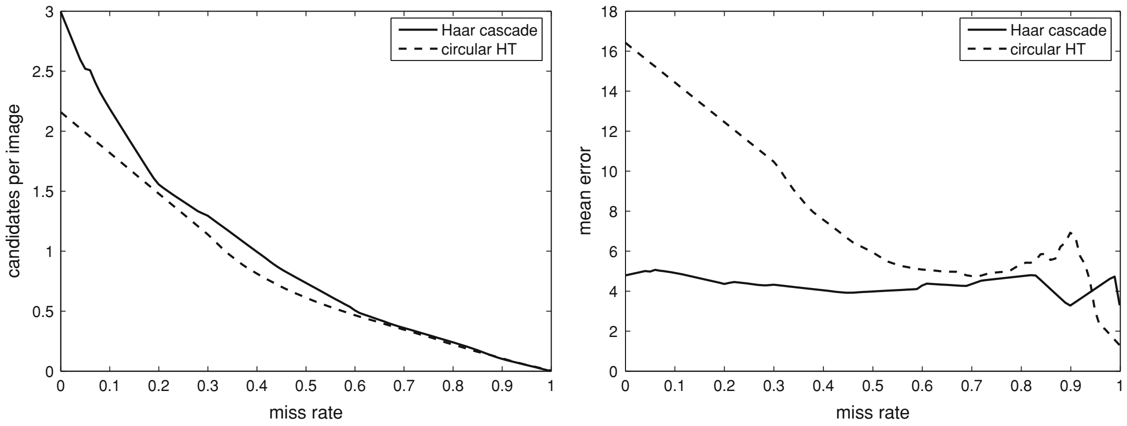


Fig. 8 Relationship between the mean number of candidates per image detected and the miss rate (*left*), and between the mean distance of the detected candidates from the ground truth circles and the miss rate (*right*). The above results were obtained using the same detectors as in Fig. 7, but after the *Confidence-Weighted Mean Shift* clustering

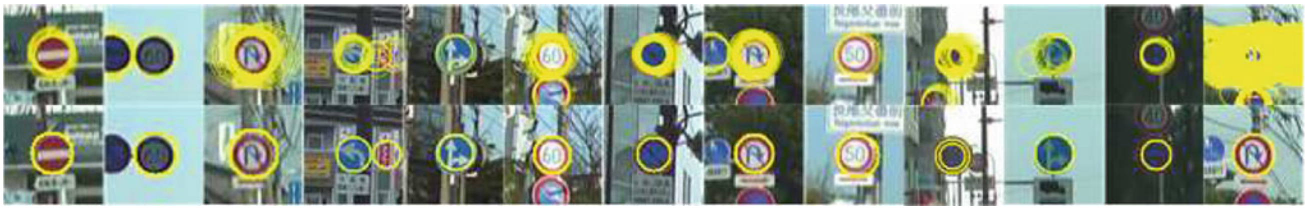


Fig. 9 Output of the Hough circle detector before (*upper row*) and after (*lower row*) applying the refinement procedure. The transparency of the detected circles in the upper row images correspond to their confidence expressed with the scaled number of votes picked from the Hough voting space

refinement procedure enabled, with the Haar cascade still being more accurate. The postprocessing step increased the average processing time of a single image by less than 5 ms for both detectors. Figure 9 illustrates example output of the HT detector before and after applying the *Confidence-Weighted Mean Shift* refinement algorithm. Overall, although slightly higher detection accuracy of the Haar cascade was observed in the above experiment, due to its higher computational complexity and very long training process, we decided to adopt the HT-based method for further experiments presented in this paper.

5.2 Evaluation of the regressor tracker

We have conducted a separate experiment aimed at evaluating the ability of our road sign tracker to retrieve the full-face view of a sign under affine transformations. This experiment was done in the following way. Six synthetic image sequences were prepared using the OpenGL framework [24]. In each sequence a template image of one sign is shown in an empty 3D scene. The consecutive images depict the sign getting closer to the virtual camera and hence increasingly distorted. This simulates a realistic scenario of a car approaching a road sign mounted on the side of the road or above the road lane. The rendered scenes were deliberately constructed without any background and with constant illumination so as to minimize the effects of possible contamination of the image regions enclosing the target and to ensure its consistent appearance. For each image sequence the refined circular Hough detector was set to capture the circle instances of radius in between 12 and 24 pixels. The tracker was triggered at the time of initial detection of a sign by the HT and updated every 15 frames. Upon the initial detection, the nearly undistorted image of a sign in gray scale was recorded to serve as a reference image.

Robustness of the on-line learned tracking function to the affine distortions was measured by recording a normalized cross-correlation (NCC) between the reconstructed full-face view of a sign in each frame and the reference image. The changes of this correlation over time for all six sequences are shown in Fig. 10.[3] In each plot the behavior of NCC for a 6D regression function encoding all six 2D affine transform parameters is compared to the behavior of NCC observed using three other trackers. These are: (1) a 4D regression function encoding only two rotation-shift parameters and both translation parameters, (2) a 3D regression function encoding only one isotropic scaling-rotation parameter and

[3] The sequences used in this experiment are provided in the supplementary material accompanying this paper.

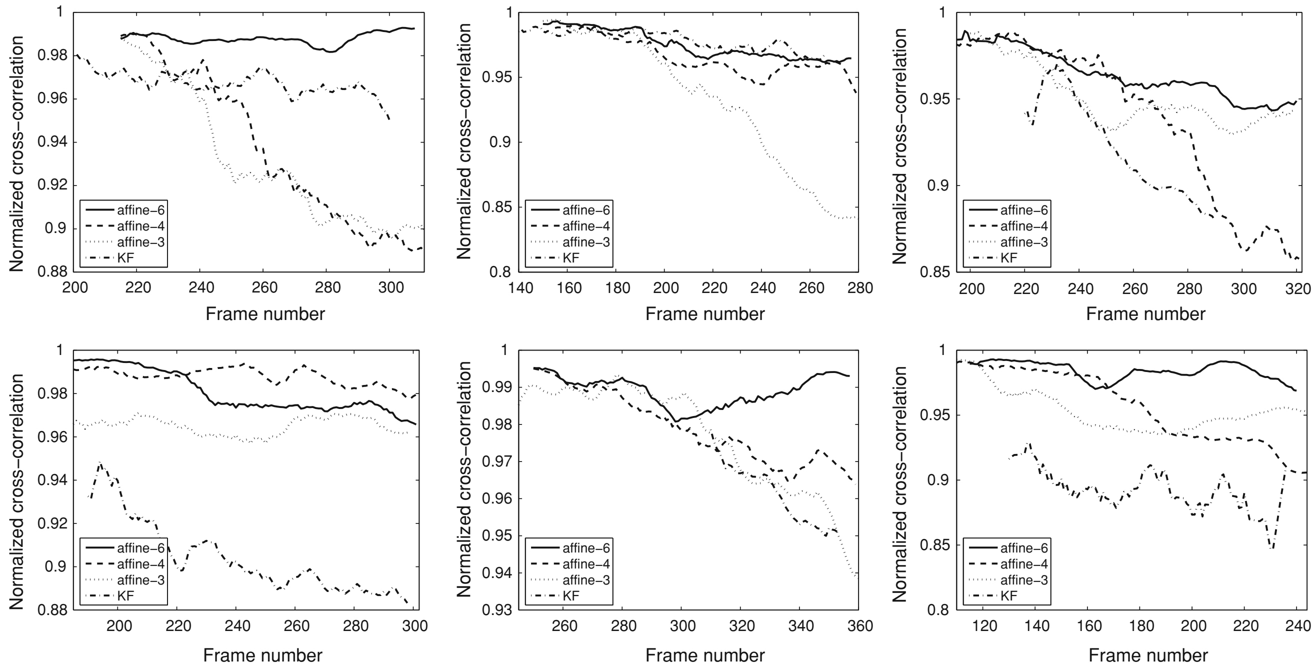


Fig. 10 Normalized cross-correlation (NCC) between the reference image recorded at the time of initial sign detection and the reconstructed full-face view of a sign in each subsequent frame of the input sequences. Each sequence was generated in a synthetic empty 3D scene and simulates what is typically observed from a vehicle approaching a traffic sign

both translation parameters, and (3) a simple tracker which makes independent circle detections in each frame, but uses a Kalman filter (KF) [25] to predict the position and scale of a sign. During the on-line training of the regression trackers, all non-translation parameters were randomly generated within the range $[-0.2, 0.2]$ and the translation parameters were randomly generated within the range $[-0.4, 0.4]$.

Based on the results of the above experiment, we conclude that learning the motion model based on the Lie algebra enables construction of a robust object tracker which is invariant to the affine transformations. In Fig. 10, the 6D affine tracker outperforms the two other regression trackers and the KF-based tracker, which do not model the full structure of the motion. The correlation between the original frontal view of a sign and a view inferred from the current transformation parameter estimates remains high for the entire duration of the sequences. In the case of the 4D and 3D affine trackers, as well as the KF-based tracker, this correlation drops more quickly, particularly in the second part of each sequence. In addition, the behavior of the KF-based tracker is less stable, as no temporal dependency between the consecutive frame observations is modeled. In other words, as long as the sign remains relatively unaffected by the affine distortion, all methods provide a satisfactorily accurate track of the target. However, when the sign gets closer to the camera and thus becomes more substantially distorted in the image plane, only the fully-affine regression tracker remains able to restore the full-face view of the target with low error. From the point of view of the entire system this is a particularly useful property because the most informative frames of the input video, when the appearance of a sign is the least ambiguous, can be efficiently used for recognition.

On the implementation side, the main complexity of the tracker is in its periodic batch re-training which involves HOG feature extraction from a candidate sign's region and several matrix operations (multiplications, inverses, additions and transposes) per image, where the largest matrices are of size $216 \times n$ and $n \leq 60$ is the total number of random deformation matrices in a training portion. With the core code implemented based on the OpenCV library [26] and using a modern PC, the regressor training/update run at frame rate. Pose estimation in each frame based on the Eqs. (8) and (9) requires, apart from feature extraction, only several matrix multiplications.[4] It is therefore much faster than the training/update step of the algorithm and no more expensive than other tracking methods, such as Kalman Filter [25].

5.3 Evaluation of SimBoost

Performance of the road sign classifier trained with the SimBoost algorithm introduced in Sect. 4.1 has been estimated

[4] Multiplication in (9) is repeated several times for better accuracy. For details refer to [21].

Table 1 Image descriptors and the associated distance metrics used in the experimental evaluation of the traffic sign classifier trained using the SimBoost algorithm

Image feature	Description	Associated distance metric
Color-parametrized Haar wavelet [6]	Rectangular filters shown in Fig. 11, parametrized with color, as described in Sect. 5.1. Only the filters of scale $w, h = \{4, 8\}$px, shifted by $\frac{1}{4}w$, $\frac{1}{4}h$ along each dimension were used, where by scale we refer to the width and height of a single rectangular block of a filter	$d(\phi_j(i_1), \phi_j(i_2)) = \lvert v_1 - v_2 \rvert$, where $v_1 = \phi_j(i_1)$, $v_2 = \phi_j(i_2)$
Histogram of oriented gradients (HOG) [27]	6-bin gradient orientation histograms computed at all possible image regions satisfying: $w, h = \{10, 15, 20\}$px, $d_x = \frac{1}{2}w$, $d_y = \frac{1}{2}h$, where w, h are the width and height of the analysis region, and d_x, d_y are the shifts along each axis	$d(\phi_j(i_1), \phi_j(i_2)) = \sqrt{\sum_{k=1}^{n}(v_{1,k} - v_{2,k})^2}$, where $\mathbf{v}_1 = \phi_j(i_1)$, $\mathbf{v}_2 = \phi_j(i_2)$, $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^n$, and n is the number of histogram bins
Region covariance [28]	4×4 covariance matrices encoding x and y coordinates and the first-order image derivatives. Only the regions of scale $w, h = \{10, 15, 20\}$px, shifted by $\frac{1}{2}w$, $\frac{1}{2}h$ along each dimension were considered	$d(\phi_j(i_1), \phi_j(i_2)) = \sqrt{\sum_{k=1}^{n} \ln^2 \lambda_k(\mathbf{C}_1, \mathbf{C}_2)}$, where $\{\lambda_k(\mathbf{C}_1, \mathbf{C}_2)\}_{k=1,\ldots,n}$ are the generalized eigenvalues of $\mathbf{C_1}$ and $\mathbf{C_2}$, computed from $\lambda_k \mathbf{C}_1 \mathbf{x}_k = \mathbf{C}_2 \mathbf{x}_k$

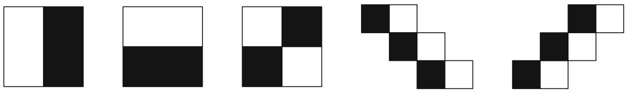


Fig. 11 Haar wavelet features used in the experimental evaluation of the traffic sign classifier trained using the SimBoost algorithm

using the similar dataset as the one used in Sect. 5.1. 7,757 static images of 14 circular Japanese road signs were extracted from the test video sequences such that each sign filled the entire image, and used to train a 100-feature classifier. Another 8,434 images were used for testing. The quality and the illumination in all images varied significantly. When constructing the test input pairs, the prototype images of each class were chosen randomly out of all images available for this class. Exploiting flexibility of the local distance formulation in (15), three different image descriptors and the associated distance metrics were used within the SimBoost framework to populate the pool of input features. They are listed in Table 1.

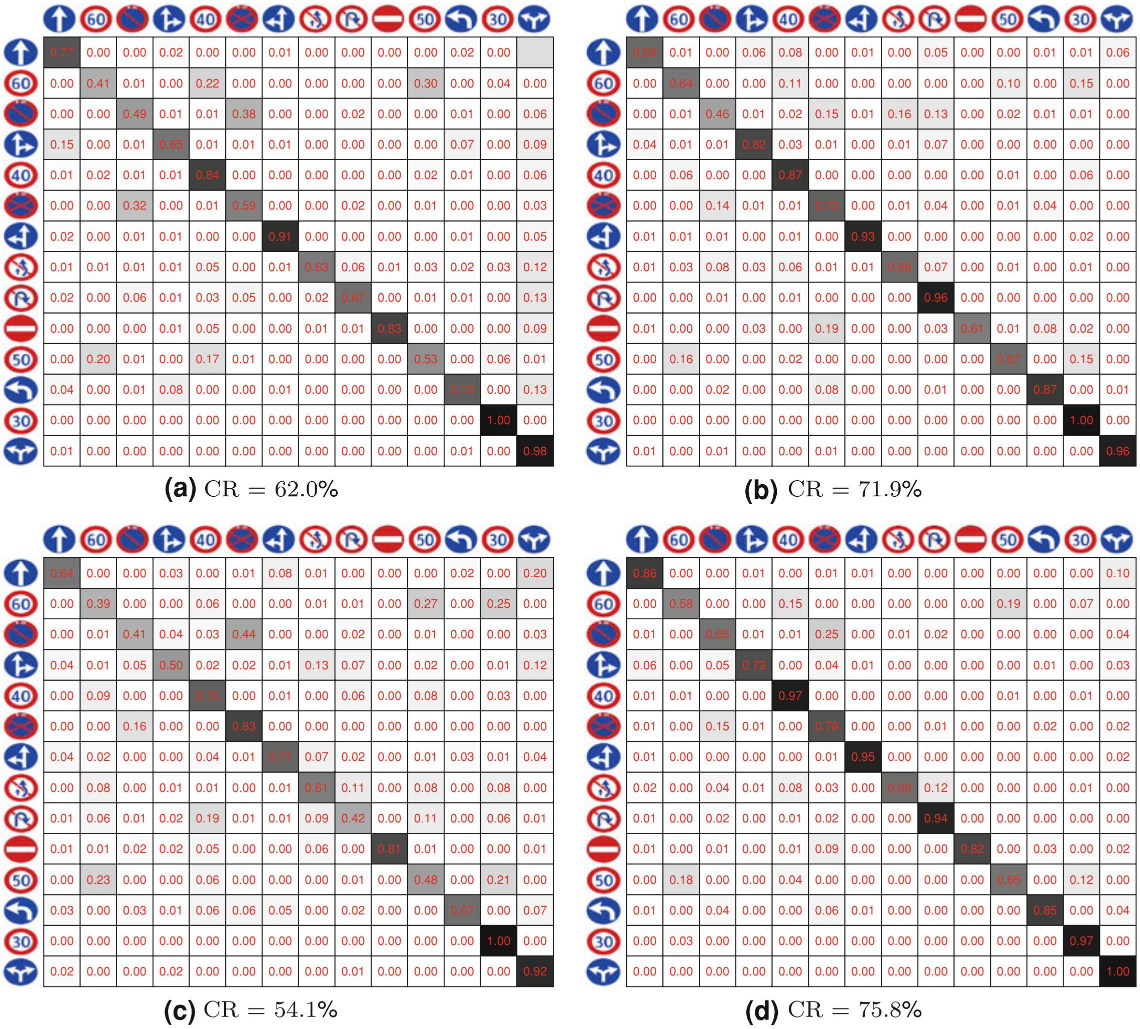
Results of the experiment are shown in the confusion matrices in Fig. 12. As seen, the histograms of oriented gradients and the color-parametrized Haar wavelet filters are the most useful image descriptors for classification of the traffic signs. Interestingly, both types of features carry only partly overlapping pieces of discriminative information. In the SimBoost framework these different nature cues can easily be intermixed. The classifier trained with both types of descriptors available in the input feature pool achieved a superior correct classification rate of nearly 76%. In Fig. 13 we have visualized the first 10 features selected by SimBoost for this best-performing classifier. It should be noted that a significant speedup of the classifier can be achieved if all used feature descriptors are color-aware. In such a case the set of possible classes can be limited to only those that share the characteristic rim color that was used as a primary cue at the detection stage.

5.4 Performance of the entire system

To evaluate the proposed traffic sign detection, tracking and recognition algorithms altogether, we built a prototype system incorporating all three components with their found optimal settings. A demo application was implemented in C++ and part of the computationally demanding image processing operations were handled by the OpenCV library [26]. The system allows manual modification of several parameters, among others the frequency of detection,[5] the scale of the signs to be detected, and the frequency of the tracker update. A list of major parameters is given in Table 2.

We have obtained a number of realistic video sequences to test an overall performance of the system. Each sequence was captured with a front-looking wide-angle camera mounted on board of a vehicle, in various, usually crowded street scenes in Japan. The illumination of the scene was roughly constant in all test videos. In system runtime, a 720 × 540 pixels portion of the scene was cropped from the upper-central region of each frame of the input video, and further downscaled by 50%. The range of radii of the circles captured by the detector was set to 12–24 pixels and the tracker updated itself every $f_U = 15$ frames, generating $m = 4$ new random affine transformations in each frame. During the on-line training of the regression tracker, the affine matrix parameters

[5] Exploration of the entire scene in search of the new road sign candidates in each frame of the input video is unnecessary and can be performed every k frames without the increase in the miss rate.



	↑	60	⦸	↱	40	⊗	↰	⊘	⊘	⛔	50	↶	30	⇄
↑	0.77	0.00	0.00	0.02	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.02	0.00	
60	0.00	0.41	0.01	0.00	0.22	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.04	0.00
⦸	0.00	0.00	0.49	0.01	0.01	0.38	0.00	0.00	0.02	0.00	0.00	0.01	0.00	0.06
↱	0.15	0.00	0.01	0.65	0.01	0.01	0.01	0.00	0.00	0.00	0.00	0.07	0.00	0.09
40	0.01	0.02	0.01	0.01	0.84	0.00	0.00	0.00	0.00	0.00	0.02	0.01	0.00	0.06
⊗	0.00	0.00	0.32	0.00	0.01	0.59	0.00	0.00	0.02	0.00	0.01	0.00	0.00	0.03
↰	0.02	0.00	0.01	0.01	0.00	0.00	0.91	0.00	0.00	0.00	0.00	0.01	0.00	0.05
⊘	0.01	0.01	0.01	0.01	0.05	0.00	0.01	0.63	0.06	0.01	0.03	0.02	0.03	0.12
⊘	0.02	0.00	0.06	0.01	0.03	0.05	0.00	0.02	0.67	0.00	0.01	0.01	0.00	0.13
⛔	0.00	0.00	0.00	0.01	0.05	0.00	0.00	0.01	0.01	0.83	0.00	0.00	0.00	0.09
50	0.00	0.20	0.01	0.00	0.17	0.01	0.00	0.00	0.00	0.00	0.53	0.00	0.06	0.01
↶	0.04	0.00	0.01	0.08	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.73	0.00	0.13
30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
⇄	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.98

(a) CR $= 62.0\%$

	↑	60	⦸	↱	40	⊗	↰	⊘	⊘	⛔	50	↶	30	⇄
↑	0.69	0.01	0.00	0.06	0.08	0.00	0.01	0.00	0.05	0.00	0.00	0.01	0.01	0.06
60	0.00	0.64	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.15	0.00
⦸	0.00	0.01	0.46	0.01	0.02	0.15	0.01	0.16	0.13	0.00	0.02	0.01	0.01	0.00
↱	0.04	0.01	0.01	0.82	0.03	0.01	0.00	0.01	0.07	0.00	0.00	0.00	0.00	0.00
40	0.00	0.06	0.00	0.00	0.87	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.06	0.00
⊗	0.00	0.00	0.14	0.01	0.01	0.73	0.00	0.01	0.04	0.00	0.01	0.04	0.00	0.00
↰	0.01	0.01	0.01	0.00	0.01	0.00	0.93	0.00	0.00	0.00	0.00	0.00	0.02	0.00
⊘	0.01	0.03	0.08	0.03	0.06	0.01	0.01	0.68	0.07	0.00	0.01	0.00	0.01	0.00
⊘	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.96	0.00	0.00	0.00	0.00	0.00
⛔	0.01	0.00	0.00	0.03	0.00	0.19	0.00	0.00	0.03	0.61	0.01	0.08	0.02	0.00
50	0.00	0.16	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.67	0.00	0.15	0.00
↶	0.00	0.00	0.02	0.00	0.00	0.08	0.00	0.00	0.01	0.00	0.00	0.87	0.00	0.01
30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
⇄	0.01	0.00	0.01	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.96

(b) CR $= 71.9\%$

	↑	60	⦸	↱	40	⊗	↰	⊘	⊘	⛔	50	↶	30	⇄
↑	0.64	0.00	0.00	0.03	0.00	0.01	0.08	0.01	0.00	0.00	0.00	0.02	0.00	0.20
60	0.00	0.39	0.00	0.00	0.06	0.00	0.00	0.01	0.01	0.00	0.27	0.00	0.25	0.00
⦸	0.00	0.01	0.41	0.04	0.03	0.44	0.00	0.00	0.02	0.00	0.01	0.00	0.00	0.03
↱	0.04	0.01	0.05	0.50	0.02	0.02	0.01	0.13	0.07	0.00	0.02	0.00	0.01	0.12
40	0.00	0.09	0.00	0.00	0.72	0.00	0.01	0.00	0.06	0.00	0.08	0.00	0.03	0.00
⊗	0.00	0.00	0.16	0.00	0.00	0.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
↰	0.04	0.02	0.00	0.00	0.04	0.01	0.71	0.07	0.02	0.00	0.01	0.03	0.01	0.04
⊘	0.00	0.08	0.00	0.01	0.01	0.00	0.01	0.61	0.11	0.00	0.08	0.00	0.08	0.00
⊘	0.01	0.06	0.01	0.02	0.19	0.01	0.01	0.09	0.42	0.00	0.11	0.00	0.06	0.01
⛔	0.01	0.01	0.02	0.02	0.05	0.00	0.00	0.06	0.00	0.81	0.01	0.00	0.00	0.01
50	0.00	0.23	0.00	0.00	0.06	0.00	0.00	0.00	0.01	0.00	0.48	0.00	0.21	0.00
↶	0.03	0.00	0.03	0.01	0.06	0.06	0.05	0.00	0.02	0.00	0.00	0.67	0.00	0.07
30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
⇄	0.02	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.92

(c) CR $= 54.1\%$

	↑	60	⦸	↱	40	⊗	↰	⊘	⊘	⛔	50	↶	30	⇄
↑	0.86	0.00	0.00	0.01	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.10
60	0.00	0.58	0.00	0.00	0.15	0.00	0.00	0.00	0.00	0.00	0.19	0.00	0.07	0.00
⦸	0.01	0.00	0.65	0.01	0.01	0.25	0.00	0.01	0.02	0.00	0.00	0.00	0.00	0.04
↱	0.06	0.00	0.05	0.79	0.00	0.04	0.01	0.00	0.00	0.00	0.00	0.01	0.00	0.03
40	0.01	0.01	0.00	0.00	0.97	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00
⊗	0.01	0.00	0.15	0.01	0.00	0.78	0.00	0.00	0.01	0.00	0.00	0.02	0.00	0.02
↰	0.01	0.00	0.00	0.00	0.00	0.01	0.95	0.00	0.00	0.00	0.00	0.00	0.00	0.02
⊘	0.02	0.00	0.04	0.01	0.08	0.03	0.00	0.68	0.12	0.00	0.00	0.00	0.01	0.00
⊘	0.01	0.00	0.02	0.00	0.01	0.02	0.00	0.00	0.94	0.00	0.00	0.00	0.00	0.00
⛔	0.01	0.00	0.01	0.00	0.01	0.09	0.00	0.00	0.00	0.82	0.00	0.03	0.00	0.02
50	0.00	0.18	0.00	0.00	0.04	0.00	0.00	0.00	0.00	0.00	0.65	0.00	0.12	0.00
↶	0.01	0.00	0.04	0.00	0.00	0.06	0.01	0.00	0.00	0.00	0.00	0.85	0.00	0.04
30	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.97	0.00
⇄	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

(d) CR $= 75.8\%$

Fig. 12 Classification accuracy of a 100-feature classifier trained using the SimBoost algorithm and different image descriptors: **a** color-parametrized Haar wavelets [6], **b** histograms of oriented gradients (HOG), **c** 4 × 4 covariance matrices encoding x and y coordinates and the first-order image derivatives [28], **d** Haar and HOG features jointly

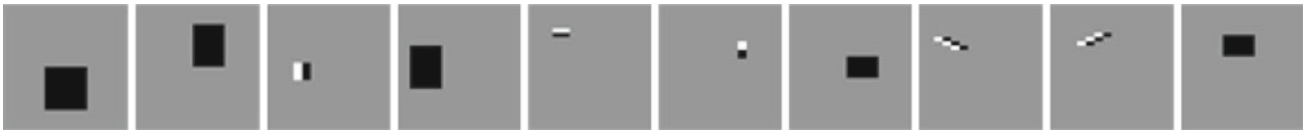


Fig. 13 10 best features selected by the SimBoost algorithm while training the 14-class road sign classifier using jointly the color-parametrized Haar wavelet filters and the histograms of oriented gradients. Both kinds of image descriptors are present

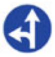
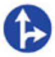
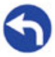
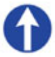
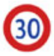
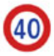
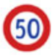
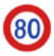
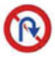
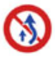
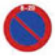
were generated randomly within the same ranges as defined in the experiment from Sect. 5.2, i.e. $[-0.2, 0.2]$ for all non-translation parameters, and $[-0.4, 0.4]$ for both translation parameters. Table 3 illustrates the numbers of traffic signs of each class that occurred in the videos and were detected, together with the numbers of these signs that were correctly classified.

As seen, an overall error rate of the classifier did not exceed 15%. Misclassifications were mainly caused by the motion blur erasing the relevant image gradients, and by the cumulated reconstruction errors of the tracker. These errors can partly be attributed to the background fragments which contaminate the corners of the regions enclosing the target circular signs. Regarding the other system components, the refined Hough circle detector appeared to be relatively accurate and resistant to clutter. Overall, it missed 14 true signs, mostly due to the insufficient figure-background contrast, and yielded fewer than ten false sign candi-

Table 2 Parameters of our traffic recognition system

Parameter	Description	Default value
Detection frequency f_D	The number of frames between the time points when the sign detector is run	$f_D = 5$
Sign's radius range $[r_{\min}, r_{\max}]$	Range of radii (in pixels) of circles to be captured by the detector	$r_{\min} = 12$, $r_{\max} = 24$
Minimum amount of feature in ROI t_{ROI}	The minimum cumulative value of feature (color-specific gradient magnitude in our case) contained in a region to be considered ROI. It determines the sensitivity of the attention operator and the depth of the quad-tree recursion	$t_{\mathrm{ROI}} = 6.9$
Gradient magnitude threshold t_G	Minimum color-specific gradient magnitude of a pixel to consider it edge. Hough-style voting is run within each found ROI only from the edge pixels	$t_G = 0.1$
Hough voting threshold t_V	Minimum cumulative vote accumulated for a given point in the parameter space of the regular polygon detector to consider it a centroid of a potential sign	$t_V = 0.5$
Candidate sign descriptor's dimensionality $N \times N \times M$	Dimensionality of the image descriptor calculated around the candidate sign's region, where N is the number of spatial bins per axis and M is the number of histogram bins. It is an argument to the tracker's regression function	$N = 6$, $M = 6$
Tracker's update frequency f_U	The number of frames between the time points when the tracker is re-trained	$f_U = 15$
Per-frame tracker's training portion size m	The number of random affine deformation matrices generated per frame. The total number of such matrices used to re-train the tracker is equal to $f_U \times m$	$m = 4$
Similarity function's complexity c_S	The number of weak classifiers incorporated in the global sign similarity function. It also determines the number of boosting rounds and hence affects the classifier training time	$c_S = 100$
Temporal weight base b	Base of the exponent used in Eq. (25). It determines how much relative importance is attached by the classifier to the most recent observations	$b = 0.8$

Table 3 Classification rates obtained in the dynamic experiment

					
7/7	4/5	1/1	4/5	1/2	10/10
					
6/9	2/2	3/3	9/10	26/31	

The numbers of correctly classified signs of each class are given against the total numbers of such signs detected in the input sequences

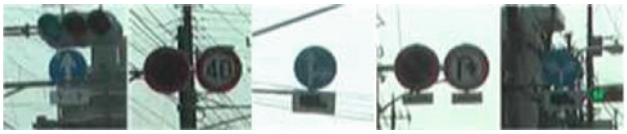


Fig. 14 Examples of road signs the refined Hough detector could not capture

dates. Figure 14 shows several examples of signs our detector was not able to capture. Finally, the tracker demonstrated its ability to rapidly correct small affine sign distortions, which enabled real-time system operation. Example videos demonstrating this ability are available at: http://aruta.pl/MVA2009/.

6 Conclusions

In this study we have presented a comprehensive approach to detection, tracking and recognition of traffic signs from a moving vehicle. Our system is comprised of three components. The detector utilizes a state-of-the-art object detection technique, but features a *Confidence-Weighted Mean Shift* mode-finding algorithm to improve its accuracy and cope with multiple redundant hypotheses in the detector's response space. The main contribution of our work are the novel tracking and recognition algorithms. The proposed tracker models the motion of the target through an instance-specific tracking function. It encodes correlations between the unique feature representation of a candidate sign and the affine distortions it is subject to while being approached by the camera. Based on the Lie group theory such a tracking function can be learned and updated instantly from random transformations applied to the image of the target in known pose. A detected and tracked sign candidate is classified by maximizing its similarity to the class's prototype image. This similarity is estimated by a linear combination of local image descriptor differences and is learned from image pairs using a novel variant of AdaBoost algorithm, called *SimBoost*.

The proposed algorithms have been evaluated in a number of experiments involving static road sign images, synthetic image sequences, and real-life video captured with a car-mounted camera. The first experiment was aimed at evaluation of the detection refinement algorithm with two different object detection techniques and identifying the best-performing refined detector. The second experiment was intended to demonstrate the ability of the tracker to model the affine motion of the signs and reconstruct their frontal views under significant viewpoint changes. In the third experiment, we estimated the error rate of a classifier trained with different low-level image descriptors using the SimBoost algorithm. Based on the comparison of the obtained classification rates, we determined the most discriminative feature representation of the traffic signs. The overall performance of the system was measured based on the prototype C++ implementation and using realistic traffic video. The obtained results prove the efficiency of the presented algorithms and show that our approach could have good prospects for application on board of intelligent vehicles.

References

1. Piccioli, G., De Micheli, E., Parodi, P., Campani, M.: A robust method for road sign detection and recognition. Image Vis. Comput. **14**(3), 209–223 (1996)
2. de la Escalera, A., Moreno, L.E., Salichs, M.A., Armingol, J.M.: Road traffic sign detection and classification. IEEE Trans. Indus. Electron. **44**(6), 848–859 (1997)
3. Barnes, N., Loy, G., Shaw, D., Robles-Kelly, A.: Regular polygon detection. In: Proc. of the 10th IEEE International Conference on Computer Vision, vol.1, pp.778–785 (2005)
4. Ruta, A., Li, Y., Liu, X.: Towards real-time traffic sign recognition by class-specific discriminative features. In: Proc. of the 18th British Machine Vision Conference, vol. 1, pp. 399–408 (2007)
5. Fang, C-Y., Chen, S-W., Fuh, C-S.: Road-sign detection and tracking. IEEE Trans. Veh. Technol. **52**(5), 1329–1341 (2003)
6. Bahlmann, C., Zhu, Y., Ramesh, V., Pellkofer, M., Koehler, T.: A system for traffic sign detection, tracking and recognition using color, shape, and motion information. In: Proc. of the IEEE Intelligent Vehicles Symposium, pp. 255–260 (2005)
7. Freund, Y., Schapire, R.E.: A short introduction to boosting. J. Jpn. Soc. Artif. Intell. **14**(5), 771–780 (1999)
8. Viola, P., Jones, M.: Robust real-time face detection. Int. J. Comput. Vis. **57**(2), 137–154 (2004)
9. Miura, J., Kanda T., Shirai Y.: An active vision system for real-time traffic sign recognition. In: Proc. of the IEEE Conference on Intelligent Transportation Systems, pp. 52–57 (2000)
10. Kang, D.S., Griswold, N.C., Kehtarnavaz, N.: An invariant traffic sign recognition system based on sequential color processing and geometrical transformation. In: Proc. of the IEEE Southwest Symposium on Image Analysis and Interpretation, pp. 88–93 (1994)
11. Aoyagi, Y., Asakura, T.: A study on traffic sign recognition in scene image using genetic algorithms and neural networks. In: Proc. of the 22nd IEEE International Conference on Industrial Electronics, Control, and Instrumentation, vol.3, pp. 1838–1843 (1996)
12. Douville, P.: Real-time classification of traffic signs. Real Time Imaging (9) **6**(3), 185–193 (2000)
13. de la Escalera, A., Armingol, J.M., Mata, M.: Traffic sign recognition and analysis for intelligent vehicles. Image Vis. Comput. **21**(3), 247–258 (2003)
14. Nguwi, Y.-Y., Kouzani, A.Z.: Detection and classification of road signs in natural environments. Neural Comput. Appl. **17**(3), 265–289 (2008)
15. Paclík, P., Novovicova, J., Pudil, P., Somol, P.: Road sign classification using the Laplace Kernel Classifier. Pattern Recognit. Lett. **21**(13–14), 1165–1173 (2000)
16. Gao, X.W., Podladchikova, L., Shaposhnikov, D., Hong, K., Shevtsova, N.: Recognition of traffic signs based on their colour and shape features extracted using human vision models. J. Vis. Commun. Image Represent. **17**(4), 675–685 (2006)
17. Paclík, P., Novovicová, J., Duin, R.P.W.: Building road-sign classifiers using a trainable similarity measure. IEEE Trans. Intell. Transport. Syst. **7**(3), 309–321 (2006)
18. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (1972)
19. Comaniciu, D., Meer, P.: Mean shift: a robust approach towards feature space analysis. IEEE Trans. Pattern Anal. Machine Intell. **24**(5), 603–619 (2002)
20. Bayro-Corrochano, E., Ortegón-Aguilar, J.: Lie algebra approach for tracking and 3D motion estimation using monocular vision. Image Vis. Comput. **25**(6), 907–921 (2007)
21. Tuzel, O., Porikli, F., Meer, P.: Learning on Lie groups for invariant detection and tracking. In: Proc. of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
22. Rossmann, W.: Lie Groups: An Introduction Through Linear Groups. Oxford University Press, New York (2002)
23. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Mach. Learn. **37**(3), 297–336 (1999)
24. Open Graphics Library. http://www.opengl.org/
25. Kalman, R.E.: New approach to linear filtering and prediction problems. Trans. ASME J. Basic Eng. **82**(D), 35–45 (1960)
26. Open Computer Vision Library, http://sourceforge.net/projects/opencvlibrary/
27. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. of the 2005 IEEE International Conference on Computer Vision and Pattern Recognition vol. 1, pp. 886–893 (2005)
28. Tuzel, O., Porikli, F., Meer, P.: Region covariance: A fast descriptor for detection and classification. In Proc. of the 9th European Conference on Computer Vision, pp. 589–600 (2006)

Author biographies

Andrzej Ruta received his M.Sc. in computer science from the AGH University of Science and Technology, Krakow, Poland. In 2009, he defended his Ph.D. thesis at the School of Information Systems, Computing and Mathematics at Brunel University, Uxbridge, United Kingdom, where he was a member of the Intelligent Data Analysis Group and worked as a Teaching Assistant. His doctoral research concerned visual object detection and recognition from video input. In June 2009, he re-joined the AGH University as a lecturer. His current research interests include intelligent data analysis, object detection, tracking and recognition, visual driver assistance, data mining and nature-inspired computing.

Fatih Porikli received his Ph.D. degree from the Polytechnic University, Brooklyn, NY, USA in 2002. He worked for AT&T Research Labs, Holmdel, NJ, USA (1997) and Hughes Research Labs, Malibu, CA, USA (1999). In 2000, he joined Mitsubishi Electric Research Labs, Cambridge, MA, USA, where he is currently a senior principal scientist. His research interests are in the areas of video processing, computer vision, aerial image processing, 3-D depth estimation, texture segmentation, robust optimization, network traffic management, multi-camera systems, data mining, and digital signal filtering.

Yongmin Li received his M.E. and B.E. from Tsinghua University, Beijing, P.R. China and the Ph.D. degree from Queen Mary, University of London, London, United Kingdom. He is a Senior Lecturer at the School of Information Systems, Computing and Mathematics, Brunel University, Uxbridge, United Kingdom. Before joining Brunel University, he worked as a research scientist in the British Telecom Laboratories, Ipswich, United Kingdom. His research interests cover the areas of computer vision, image processing, video analysis, machine learning and pattern recognition. Dr. Li is a senior member of the IEEE.