Chapter 1

Combining Traffic Sign Detection with 3D Tracking Towards Better Driver Assistance

Radu Timofte[1] *, Victor Adrian Prisacariu[2] *, Luc Van Gool[1] and Ian Reid[2]
[1]*ESAT-PSI-VISICS / IBBT, Katholieke Universiteit Leuven, Belgium*
[2]*Department of Engineering Science, University of Oxford, United Kingdom*

We briefly review the advances in driver assistance systems and present a real-time version that integrates single view detection with region-based 3D tracking of traffic signs. The system has a typical pipeline: detection and recognition of traffic signs in independent frames, followed by tracking for temporal integration. The detection process finds an optimal set of candidates and is accelerated using AdaBoost cascades. A hierarchy of SVMs handles the recognition of traffic sign types. The 2D detections are then employed in simultaneous 2D segmentation and 3D pose tracking, using the known 3D model of the recognized traffic sign. Thus, we achieve not only 2D tracking of the recognized traffic signs, but we also obtain 3D pose information, which we use to establish the relevance of the traffic sign to the driver. The performance of the system is demonstrated by tracking multiple road signs in real-world scenarios.

1.1. Introduction

Traffic signs play a pivotal role in rendering traffic more efficient and safer. Unfortunately, still many accidents happen because drivers have overlooked a sign. Therefore, increasingly cars are being equipped with vision systems that detect and recognize such signs, in order to assist the driver. Moreover, public authorities are carrying out large-scale campaigns to replace older, hand-drawn maps and manually generated inventories by digital maps and GIS systems. The presence and positions of traffic signs again figure high on the list of data to be surveyed. As signs regularly change (total yearly changes in the traffic sign set are estimated to amount to 7%!), it is important to automate their detection. Otherwise, regular updates are too expensive. In such applications, it is not only important to detect the presence of traffic signs, but their actual position and orientation are crucial as well. Such factors determine whether signs are sufficiently visible and whether they are relevant to traffic participants approaching from specific directions. Thus, traffic sign analysis is not only about finding the signs in images, but also about their 3D positions and orientations, especially in an urban context.

*R. Timofte and V.A. Prisacariu contributed equally to this work.

Fig. 1.1. Importance of determining the traffic sign orientation. The no right turn sign does not point towards the car.

Most current work involves combining a detector with a Kalman filter, as in [1, 2], or with a particle filter, as in [3, 4]. These methods rely on a predictable car motion model or reliable feature detectors. For example, [1] and [5] assume the car moves in a straight line and with constant velocity. As feature descriptors, trackers usually use edges or other kinds of information extracted from the traffic sign shape. In [1] the authors explicitly model geometric properties of the traffic sign shape (i.e. a triangle-shaped sign has to be equilateral). This leads to a lack of robustness when subjected to occlusions, deformations or motion blur. In [3] the authors track circular signs and assume these have a colored border on a white interior and that clear edges can be extracted. This approach would not extend to differently shaped signs and is again vulnerable to motion blur. A solution to some of these problems is region based tracking. Regions are more stable than edges so tracking is more robust, and they are less affected by occlusions or motion blur.

To our knowledge, with notable exceptions like [3] and [6], most previous road sign work has been *2D*. That is, the position of the traffic sign was tracked in the *image* rather than in *3D space*. In [3] the authors use inertial sensors mounted on the car to help them obtain an approximation of the 3D pose of the signs with respect to the car. Unfortunately this approach fails when the traffic sign does not point towards the car, as in the case shown in figure 1.1. Here, the no right turn sign does not have the same orientation as the inertial sensor mounted on the car.

An alternative method is presented in [6], where multiple views, 3D reconstruction, and a Minimum Description Length based method are used. Although a 3D pose is recovered, it is an offline method for 3D mobile mapping purposes.

Our approach integrates a single view detection and recognition step with a multi-object, model-based, 3D tracker. This has several advantages:

(i) we are able to obtain the full 3D pose of the traffic signs in the image, accounting for the case in figure 1.1,
(ii) the tracking is region based, making it robust to motion blur and occlusions,
(iii) because our tracker processes only a small region in and around the detection we are able to achieve real time performance.

The remainder of this chapter is structured as follows: we review the state-of-the-art in Section 1.2 and present an overview of our system in Section 1.3. In Section 1.4 we detail the single view detection and recognition parts and in Section 1.5 the energy function used both in the non-maximum supression and the 3D tracking stages, themselves presented in Sections 1.6 and 1.7, respectively. Section 1.8 shows results on a large database with images and videos and we conclude in Section 1.9.

1.2. State-of-the-art

1.2.1. *Single view detection and recognition*

A traffic sign is designed to be easily recognizable using color, shape and appearance. Current state-of-the-art work considers these properties either individually or in combination. The standard pipeline starts with a detection phase for localizing candidates in the image space, followed by a recognition phase for setting the labels. The detection often employs a fast segmentation/candidate extraction step, usually done through simple methods, followed by pruning methods to reduce the false detections. The recognition phase aims at fast and robust classification of traffic signs while space/temporal integration (i.e. tracking) can help by exploiting the information from already processed frames to improve performance in the current frame.

Color is important in spotting the traffic sign in cluttered environments. Many approaches use simple crafted thresholds for each color (i.e. red, blue, yellow, black, white) in a suitable color space such as RGB, Hue-Saturation-Intensity/Value (HSI,HSV), or Luv. In [7], for each color, pairs of thresholds for Hue and Saturation channels are picked from the color distribution during training. Figure 1.3 depicts the extraction of candidates based on color thresholding. Hue is invariant to lighting. This property is exploited in [8] where a shadow-highlight invariant method is proposed. [9] also uses HSV and a dynamic pixel aggregation technique which employs dynamic thresholding to address the hue dependence on external brightness variations. The saturation and intensity values of the images are used to adapt these

thresholds. [10] uses a region growing method, which merges neighboring pixels with similar colors. Other color-based approaches train classifiers or probabilistic color models to determine the likelihood of a pixel to belong to a reference color [11].

The regular shapes shared among different traffic sign types render general shape detectors useful. A form of generalized Hough transform is used in [12] for regular polygonal shapes, while [13] uses distance transform matching and a template hierarchy to capture the variety of object shapes. AdaBoost [6, 14] or part-based models [15] are highly accurate in detection and are used in cascades for specific shape classes. In [6, 14, 16] as well as in this paper, color, shape and appearance cues are combined for detection.

Also, for recognition, there is a large variety of methods. Most machine learning techniques have also been used for classifying traffic signs. A recent classification challenge [17] on German traffic signs showed that convolutional neural networks, SVM with an intersection kernel, and Sparse Representation-based classifiers [18] are currently the top performers. The features used by the top contenders were intensity, Histogram of Oriented Gradients (HOGs) [18, 19] and their projections. Color is not a critical feature for traffic sign classification, but for discriminating between traffic signs and background.

The results of traffic sign detection and recognition thus far testify to the great difficulty of the task. In [20] the reported performance is 26% false negatives (missed traffic signs) at a level of 3 false positives (false alarms) per image. [7] applies image thresholding followed by SVM classification. Every traffic sign is detected at least twice in a total of 5000 video frames, with 22 false alarms. Detection rates per view are not given. In both works the thresholds are manually selected. In [21] the search is constrained to road borders and an overhanging strip, by using inertial sensors and estimating 3D geometry. This significantly reduces the false positives, while the false negative rate is 3.8%. These three systems were tested on highways.

The following systems have also been demonstrated off the highways. By restricting the detection to speed, stop and give-way signs, [22] reports a performance of $10^{-4} - 10^{-5}$ false positive rates for 1% false negatives, but the number of subwindows per image is not mentioned. [23] reports no false positives in a 150 minute long video, but misses 11% of all traffic signs. In [2, 24] image color thresholding and shape detection are combined, achieving 6.2% false negatives. The number of false positives is not mentioned. [25] proposed a system similar to the one in [7], where the SVM is replaced by a neural network. No quantitative results are presented.

While the majority of the previous contributions work with and report the performance on a rather small subset of sign types, our system (using settings from [6, 16]) handles 62 different types of signs. Moreover, authors usually focus on highway images, whereas our database mainly contains images from smaller roads and streets. This is a more challenging problem as signs tend to be smaller, have more often been smeared with graffiti or stickers, suffer more from occlusions, are often older, and are visible in fewer images. Also, several sign types never appear

along highways.

1.2.2. *3D Region-based Segmentation and Tracking*

Fast and accurate image segmentation and pose tracking are fundamental tasks in computer vision. Most current studies treat these two tasks independently and often use edge based segmentation, leading to an inability to deal with occlusions or motion blur. A common solution to the edge problem is the use of regions, whose shape and evolution can be effectively modelled using level-set functions [26]. The first paper to use regions in 3D pose recovery was [27]. There the authors represent the contour by a zero level-set of a 3D embedding function and evolve it in a single iterative step, in two stages: first an infinite dimensional level set energy function (with an added shape term) is minimized with respect to the segmentation, in the expectation that the contour will then match the projection of the occluding contour of the 3D object. Second, each point on the contour is back-projected to a ray (represented in Plücker coordinates), and the pose sought that best satisfies the tangency constraints that exist between the 3D object and these rays. The obvious pitfall of this method is that the evolution of the contour is not limited to the space of possible segmentations, which can make the resulting segmentations still inaccurate. The method is improved in [28] where the unconstrained contour evolution stage is removed, and the minimization takes place by evolving the contour (approximately) directly from the 3D pose parameters. The direction of contour evolution is determined by the relative foreground/background membership probabilities of each point, while the amount of evolution is apparently a "tunable" parameter.

In this work we use the variational approach of [29], where the Bibby-Reid energy function of [30] is differentiated with respect to the 6 DoF 3D pose parameters, simultaneously yielding both segmentation and pose. This method has the benefit of running in real time and of being resilient both to occlusions and to motion blur.

1.3. Overview of the System

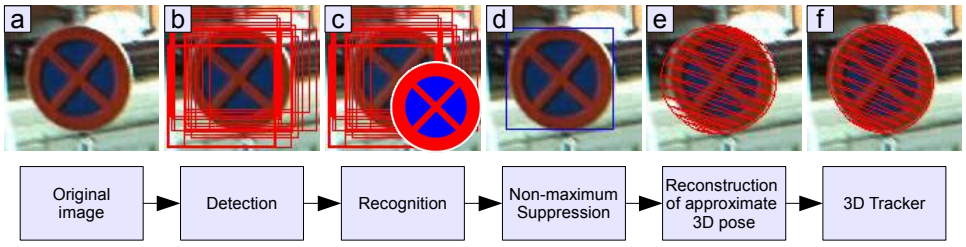

Fig. 1.2. Algorithm overview

An outline of our algorithm is shown in figure 1.2. It consists of 5 steps. First

the signs are detected using a single view detection step. Next the sign in each detection is recognized and the best detection for each object is selected. Next, for each sign in the image, an approximate 3D pose is computed, by combining a predicted 3D pose (from the previous frame, with a constant velocity motion model) with a 3D pose obtained from the detection bounding box. The detection bounding box is then converted to a 3D pose using a 4 point planar pose recovery algorithm. Finally, the 3D tracker is iterated from the approximated pose, which results in a refined 3D pose, for each object in the image. We use the object detection and recognition algorithm from [6] followed by the 3D tracking from [16].

1.4. Single-View Detection and Recognition

The single view processing starts with a fast candidate extraction process, followed by a pruning/detection process based on AdaBoost cascades, and finally after the further filtering by a background vs. traffic sign SVM classifier, the detections are assigned to the class with the highest estimated probability in the corresponding one-against-all SVM output. The candidate extraction and detection parts follow the work from [6], used also in [16]. The classification part follows settings from [18].

The simplest and most often used extraction method for traffic sign detection is extraction of connected components from a *thresholded image* [7, 25]. (see figure 1.3). The thresholded image is obtained from a color image (RGB in [6, 16] and herein), with color channels (I_R, I_G, I_B), by application of a color threshold $T = (t, a, b, c)^\top$:

$$I(T) = \begin{cases} 1 & a \cdot I_R + b \cdot I_G + c \cdot I_B \geq t \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

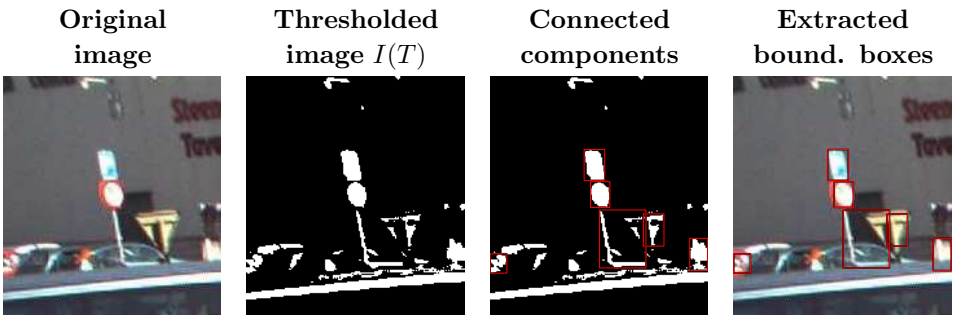

Fig. 1.3. **Color-based extraction** method for threshold $T = (0.5, 0.2, -0.4, 1.0)^\top$ [6].

Under variable illumination conditions and in the presence of a cluttered/complex background an extraction method based on only a few manually selected thresholds is insufficient for extracting the traffic signs. It is necessary to

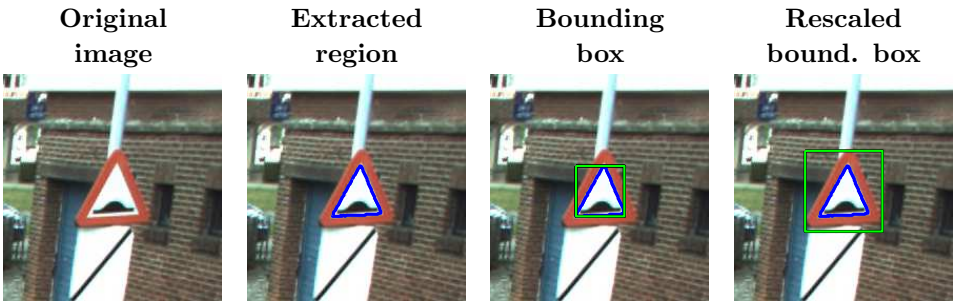

Fig. 1.4. **Extended threshold** addresses the problem of signs not well locally separable from the background. The bricks have a color similar to the red boundary, the inner white part is extracted and the resulting bounding box is rescaled $\overline{T} = (0.1, -0.433, -0.250, 0.866, 1.6, 1.6)^\top$ [6].

combine regions selected by several thresholds $\mathcal{T} = \{T_1, T_2, \dots\}$, in the sense of adding regions (OR-ing operation). The regions that pass are the input for the next stage, i.e. detection. Using more thresholds can lead to overfitting. It lowers the number of false negatives (FN), but adds computational cost and increases the number of false positives (FP).

We search for the optimal subset $\mathcal{T}$ of such thresholds starting from thousands of possible color thresholds. We formulate our search as an Integer Linear Programming (ILP) problem. ILP yields a viable solution within minutes, due to the sparsity of the constraints. Searching for a trade-off between FP and FN is the most straightforward criterion to guide this optimization. To avoid overfitting and to keep the method sufficiently fast, we additionally constrain the number of selected thresholds, the cardinality, $\mathrm{card}(\mathcal{T})$. The *accuracy*, defined as the average overlap between ground truth (annotation) bounding boxes with the extracted bounding boxes, is important for the subsequent steps in the system. We penalize inaccurate extractions:

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} \left(\mathrm{FP}(\mathcal{T}) + \kappa_1 \cdot \mathrm{FN}(\mathcal{T}) + \kappa_2 \cdot \mathrm{card}(\mathcal{T}) - \kappa_3 \cdot \mathrm{accuracy}(\mathcal{T})\right) \quad (1.2)$$

where $\mathrm{FP}(\mathcal{T})$ is the number of false positives and $\mathrm{FN}(\mathcal{T})$ is the number of false negatives measured on a training set. The weighting scalars κ_1, κ_2 and κ_3 are learned parameters estimated by cross-validation. The reformulation of problem 1.2 into an ILP form is described in [6, 14].

The contour of the traffic sign often cannot be separated from the background due to color similarity. In such cases we still have the inner contours that can be extracted by color thresholding. Such thresholding is followed by rescaling the obtained bounding box to include the whole traffic sign. See for example figure 1.4. The inner part can often define the traffic sign's outline with sufficient accuracy and we therefore introduce the extended threshold — in the sequel simply referred to as

threshold:

$$\overline{T} = (\underbrace{t, a, b, c}_{T}, s_r, s_c)^\top \tag{1.3}$$

which consists of the original threshold T and vertical and horizontal scaling factors (s_r, s_c).

One could try to adapt the set of thresholds to the illumination conditions. To add robustness to the thresholding method itself we adjust the threshold to be *locally stable* in the sense of Maximally Stable Extremal Regions (MSER) [31]. Instead of directly using the bounding box as extracted by the learned threshold (t, a, b, c, s_r, s_c), we use bounding boxes from MSERs detected within the range $[(t-\epsilon, a, b, c, s_r, s_c); (t+\epsilon, a, b, c, s_r, s_c)]$, where ϵ is a parameter of the method. This 'TMSER' method is parametrized by two parameters (ϵ, Δ), as the MSER adds a stability parameter Δ.

Dirty, peeled, partially occluded traffic signs also should pass the color test. Therefore, we also need to employ shape information in order to remain sufficiently selective. We learn fuzzy templates to incorporate small affine transformations and shape variations and we only explicitly determine the position and scale in a 3D Hough accumulator. For more details please refer to [6, 14].

The candidates extracted are verified further by a binary classifier which filters out remaining background regions. It is based on the Viola and Jones Discrete AdaBoost classifier [32]. Detection is performed by cascades of AdaBoost classifiers, followed by an SVM operating on normalized RGB channels, pyramids of HOGs [33] and AdaBoost-selected Haar-like features to prune the background further.

Finally, the detections are further classified as their specific traffic sign type. For each detection we compute pyramids of HOG features as in [34], LDA-project them, and we train one-against-all Linear SVM classifiers. The assigned traffic sign type is the one with the highest score estimated from the SVM classifier output.

1.5. Pixel-Wise Posteriors Energy Function

Before describing our non-maximum suppression and 3D tracking stages we introduce the Bibby-Reid [29, 30] pixel-wise posteriors energy function, which is the basis of both stages.

For any image I with an image domain Ω and known color statistics (foreground and background membership probabilities), the pixel-wise posteriors (PWP) energy function is a function of the contour that separates the image domain into a foreground and a background domain. The PWP energy function has a maximum when the contour best separates the two regions i.e. when all the pixels in the foreground region have a higher probability of being foreground than of being background (and viceversa). Figure 1.5 shows such an example. Here the contour is denoted with $\mathbf{C}$, the foreground region with Ω_f and the background region with Ω_b. The implicit

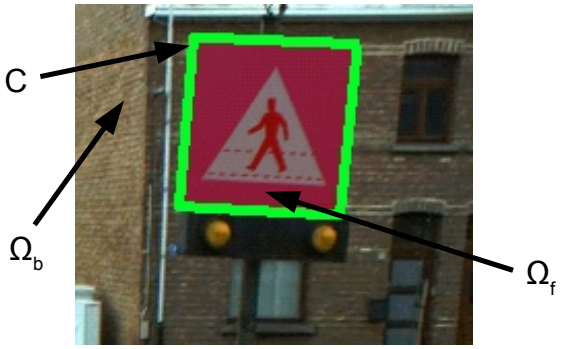

Fig. 1.5. Representation of the object showing the contour of the projection **C**, the foreground region Ω_f and the background region Ω_b

representation of the contour is used, by representing it as the zero level of a level set function [26].

Of course, the PWP energy function is not the only energy function to measure segmentation quality. It is, however, one of the best behaved ones, as demonstrated in [29, 30], having a large basin of convergence and few local minima. Furthermore, it can be used both for non-maximum suppression and for 3D tracking, as shown in the following sections.

The PWP energy function is:

$$E(\Phi) = -\sum_{x\in\Omega} \log\Big(H_e(\Phi)P_f + \big(1 - H_e(\Phi)P_b\big)\Big) \tag{1.4}$$

Here H_e is the smooth Heaviside function, x is the pixel in the image, Φ the level set embedding function and:

$$P_f = \frac{P(y_i|M_f)}{\eta_f P(y_i|M_f) + \eta_b P(y_i|M_b)}, \quad P_b = \frac{P(y_i|M_b)}{\eta_f P(y_i|M_f) + \eta_b P(y_i|M_b)} \tag{1.5}$$

with η_f the number of foreground pixels, η_b the number of background pixels, y_i the color of the i-th pixel, $P(y_i|M_f)$ the foreground model over pixel values y and $P(y_i|M_b)$ the background model. We use RGB images and our models are histograms with 32 bins for each channel, which are updated online, allowing for variations in illumination.

1.6. Non-Maximum Suppression

At this stage several detections might be available for each object, as shown in figure 1.6. We need to select only a single, best segmentation, process known as non-maximum suppression. There are several methods for doing this. For example [19] uses mean-shift to search in a 3D scale–location space.

We on the other hand, we have the advantage of knowing color models for the foreground and for the background. This means that, for all the detected

Fig. 1.6. Non-maximum suppression example results

bounding boxes, we can select the one which best segments the traffic sign from the background. To do this we evaluate the Bibby-Reid energy function described above for all detected bounding boxes and select the one with the highest color matching score.

With b as the bounding box we write:

$$P(b) = \prod_{x \in \Omega} \Big(H_e(a(x)) P_f + \big(1 - H_e(a(x))\big) P_b \Big) \tag{1.6}$$

where we replace the level set function Φ with a function $a(x)$ indicating membership

of pixel x to the inside or outside of the bounding box:

$$a(x) = \begin{cases} 1 & x \text{ inside the bounding box} \\ 0 & \text{otherwise} \end{cases} \tag{1.7}$$

Obviously, when an object is detected for the first time, we have no color model yet. In this case we choose the detection with the highest SVM score, and initialize the two color models $P(y_i|M_f)$ and $P(y_i|M_b)$.

1.7. Approximate 3D Pose Reconstruction and 3D Tracking

At this stage of the algorithm we have a single, correct, bounding box for every traffic sign in the image and we need to track their 3D position (rotation and translation). The core of our tracking procedure is the PWP3D algorithm [29]. It assumes a known 3D model and a calibrated camera and it maximizes the above-mentioned pixel-wise posteriors energy function with respect to the pose of the known model. This optimization is done by computing the derivatives of the energy function with respect to the pose parameters and using gradient descent. Note that convergence is not guaranteed within the permitted number of iterations, but in our testing we noticed that an average of 15 iterations is enough for a satisfactory result.

Following [29], we can write:

$$\frac{\partial E}{\partial \lambda_i} = \sum_{x \in \Omega} \frac{P_f - P_b}{H_e(\Phi) P_f + \big(1 - H_e(\Phi)\big) P_b} \frac{\partial H_e(\Phi)}{\partial \lambda_i} \tag{1.8}$$

where λ_i are the pose parameters (7 in total). In this work, similar to [29], we use quaternions to describe rotation.

$$\frac{\partial H_e\big(\Phi(x,y)\big)}{\partial \lambda_i} = \frac{\partial H_e}{\partial \Phi} \left(\frac{\partial \Phi}{\partial x} \frac{\partial x}{\partial \lambda_i} + \frac{\partial \Phi}{\partial y} \frac{\partial y}{\partial \lambda_i} \right) = \delta_e(\Phi) \begin{bmatrix} \frac{\partial \Phi}{\partial x} & \frac{\partial \Phi}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial \lambda_i} \\ \\ \frac{\partial y}{\partial \lambda_i} \end{bmatrix} \tag{1.9}$$

where δ_e is the Dirac delta function.

At each iteration of the algorithm Φ is recomputed as the signed-distance function of the projected contour. The partial differentials of the level set function with respect to pixel position, $\frac{\partial \Phi}{\partial x}$ and $\frac{\partial \Phi}{\partial y}$, are computed using centered finite differences.

Every 2D point on the contour of the projection of the 3D model has at least one corresponding 3D point $\mathbf{X}$, for which:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -f_u \frac{X}{Z} - u_0 \\ \\ -f_v \frac{Y}{Z} - v_0 \end{bmatrix} \tag{1.10}$$

Therefore:

$$\frac{\partial x}{\partial \lambda_i} = -f_u \frac{\partial}{\partial \lambda_i}\frac{X}{Z} = -f_u \frac{1}{Z^2}\left(Z\frac{\partial X}{\partial \lambda_i} - X\frac{\partial Z}{\partial \lambda_i}\right) \tag{1.11}$$

Similar for y.

Continuing, each point $\mathbf{X}$ in camera coordinates, has a corresponding 3D point $\mathbf{X_0}$ in object coordinates, so:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{R} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \\ 0 \end{bmatrix} \tag{1.12}$$

where $\mathbf{R}$ is the rotation matrix and is a function of the rotation quaternion and t_x, t_y and t_y are the translation parameters. $\frac{\partial X}{\partial \lambda_i}, \frac{\partial Y}{\partial \lambda_i}$ and $\frac{\partial Z}{\partial \lambda_i}$ follow trivially. For more details the reader is refereed to [29].

There are two issues with the above formulation which need to be addressed in order for us to apply this tracker to our problem. The PWP3D tracker needs an initial 3D pose and values for the foreground/background membership probabilities. Also, the PWP3D tracker can lose tracking, so at each new frame, the 2D detections need to be converted to (approximate) 3D poses to be combined with the 3D tracker. These two problems are addressed in the following sections.

1.7.1. *Initialization*

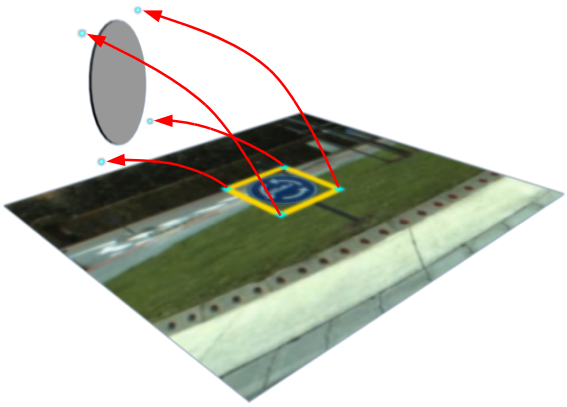

Fig. 1.7. 3D–2D point correspondences

Traffic signs can be approximated as planar objects, which means we can use any one of several planar pose recovery algorithms currently available to convert the 2D bounding box surrounding the detection to a 3D pose. We use the currently state-of-the-art algorithm introduced in [35].

This algorithm requires (at least) 4 3D–2D point correspondences and works iteratively, minimizing a sum of squared object-space collinearity errors. We use the 4 2D corners of the 2D detection bounding box and relate them to the 4 corners of the bounding box enclosing the 3D model of object, as depicted in figure 1.7.

An example result is depicted in figure 1.2e.

1.7.2. *Detector–Tracker Integration*

The PWP3D tracker will often lose its track if there is no overlap between the positions of the tracked object at consecutive frames. We therefore use the 3D poses recovered from the detection bounding boxes (as presented in the previous subsection), combined with a constant velocity motion model, to re-localize the 3D tracker.

One common solution is to use a Kalman filter for a purely statistical fusion of the tracking data with the approximate poses from the detector. A Kalman filter represents all measurements, system state and noise as multivariate Gaussian distributions. In contrast, in this work we first compute an approximate 3D pose for each 3D sign at the current frame, by combining the detection with a motion model. This serves as an initialization for the PWP3D pose optimization described above. By iterating the tracker rather than doing a purely statistical data fusion we make no pretense on the type of these probability distributions.

We define a constant velocity motion model:

$$v^i_{t_k} = t_{k-1} - t_{k-2}, \quad v^i_{r_k} = r_{k-1} r^{-1}_{k-2} \tag{1.13}$$

where k is the current frame, $k-1$ is the previous frame, t is the translation and r is the rotation quaternion from the tracker.

There is also a velocity given by the 2D detections:

$$v^{ii}_{t_k} = u_k - u_{k-1}, \quad v^{ii}_{r_k} = p_k p^{-1}_{k-1} \tag{1.14}$$

where u is the translation and p is the rotation quaternion, obtained from the detector by using the 4 point planar pose recovery algorithm.

When the object is initially detected it is most likely far from the camera, so its rotational motion is less predictable, making the chance of lack of overlap higher. In this case detections should be trusted more. When the object is close to the camera, rotational motion is more predictable so the chance for lack of overlap is smaller. In this case the motion model should be trusted more.

The predicted pose for the current frame becomes:

$$t_k = t_{k-1} + \alpha v^i_{t_k} + \beta v^{ii}_{t_k}, \quad r_k = r_{k-1} q_\alpha v^i_{r_k} q_\beta v^{ii}_{r_k} \tag{1.15}$$

where k is the current frame, $k-1$ is the previous frame, t is the translation from the tracker and u is the translation from the detector. The variables α, β, q_α and q_β are dependent on the distance between the object and the camera. α and q_α are inversely proportional to this distance while β and q_β are proportional to it.

Finally the predicted pose (t_k, r_k) is refined using the tracker.

Table 1.1. Summary of achieved results in single-view detection.

	FN-TS		FN-BB		FP per
	[%]	#/859	[%]	#/2571	2MP img
Extr1 (color)	0.7%	6	1.1%	29	3 281.8
Extr2 (color+TMSER)	0.7%	6	1.1%	28	3 741.7
Extr3 (color+shape)	0.5%	4	0.7%	17	5 206.2
Extr4 (color+TMSER+shape)	0.5%	4	0.7%	17	5 822.0
Det + Extr1	2.3%	20	4.0%	103	2
Det + Extr4	1.9%	16	3.2%	82	2

Extr marks candidate extraction setting. **color** means extended color thresholds, **TMSER** stands for TMSER(ϵ, Δ) = TMSER$(0.1, 0.2)$, **shape** is for fuzzy template methods. **FN-BB** means false negative with respect to bounding boxes, **FN-TS** means false negative with respect to traffic signs.

1.8. Experiments

We report results for different parts of our system: detection and recognition, and 3D tracking. We are using the BelgiumTS[a] database as in our previous work [6, 14, 16]. It contains 13444 traffic sign (TS) annotations in 9006 still images, corresponding to 4565 physically distinct traffic signs visible at less than 50 meters from the camera. BelgiumTS has 3 main subparts: "Training", "2D Testing", and "3D Testing", as well as a classification subset, BelgiumTSC [18].

1.8.1. *Detection Results*

The Training part of BelgiumTS is used for learning the suitable candidate extraction methods as well as for training the AdaBoost cascades and the SVM classifiers, while the 2D Testing part is used to assess the performance. The method has only been trained for 62 traffic signs classes. In 2D Testing the number of used annotations is 2571, corresponding to 859 physically distinct traffic signs. The output of the cascades is processed further by an SVM classifier that uses Haar-like features, pyramids of HOGs and pixels in RGB space.

The detection and extraction errors (Table 1.1) are evaluated according to two criteria: either demanding detection every time a sign appears (FN-BB), or only demanding it is detected at least once (FN-TS). On average, a sign is visible in about 3 views. *Coverage* is defined as the ratio between intersection and the union of two areas. A detection is successful if *coverage* ≥ 0.65 when compared to the annotation.

Table 1.1 shows results of both the candidate extraction (still with an appreciable number of FP, see first four rows) and the final detection (i.e. candidate extraction followed by AdaBoost detector and SVM, see last two rows). The shape extraction significantly increases the number of false positives (see for example 4th row in the table). The reason is that we keep both the original color bounding boxes and add all bounding boxes that reflect a good shape match. Combined extraction lowers

[a]BelgiumTS is available at: `http://homes.esat.kuleuven.be/~rtimofte/`

FN, however. Here we use the **Det+Extr1** setting.

We compare the pipeline outlined so far with a sliding window approach and a part-based model detector. For the sliding window, we train Discrete AdaBoost cascades directly on sampled subwindows from the Training data, instead of the extracted candidates as for **Det+Extr1**. For details please refer to [14]. The number of processed windows per 1628×1236 image is higher than 12 million. Figure 1.8 shows the complementarity of the sliding window approach to our **Det+Extr1** pipeline. While the individual pipelines have a comparable performance, the combination improves over both. Figure 1.9 shows cases that could be detected by one

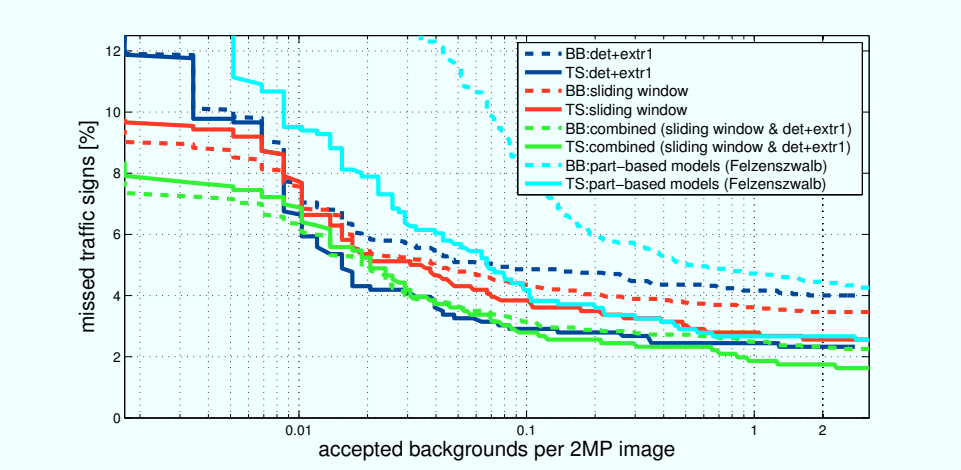

Fig. 1.8. **Comparison with state-of-the-art methods.** Detection plots for every time a sign appears (**BB**-bounding box level), or demanding it is detected at least once (**TS**-traffic sign level).

pipeline but not the other. In our single-core / single-thread implementations, the **Det+Extr1** pipeline is about 50 times faster than the sliding window pipeline.

We also compare with the state-of-the-art generic object class detector of Felzenszwalb et al. [15], the top performer in the PASCAL VOC Challenge 2009 [36]. We use the scripts from the authors and we train on the Training part of BelgiumTS a model with 5 components which correspond to the basic shapes of the traffic signs. The poorer performance of the part-based detector when compared with our specialized systems (see figure 1.8) is believed to be caused by the fact that this approach is a generic one and only works on HOG features. However, the part-based model detector is also still far from realtime performance.

1.8.2. *Recognition Results*

We evaluate several settings for traffic sign recognition on BelgiumTSC [18], the classification subset of BelgiumTS database which contains 62 different sign types, 4591 training samples and 2534 testing samples. We use raw gray scale pixel val-

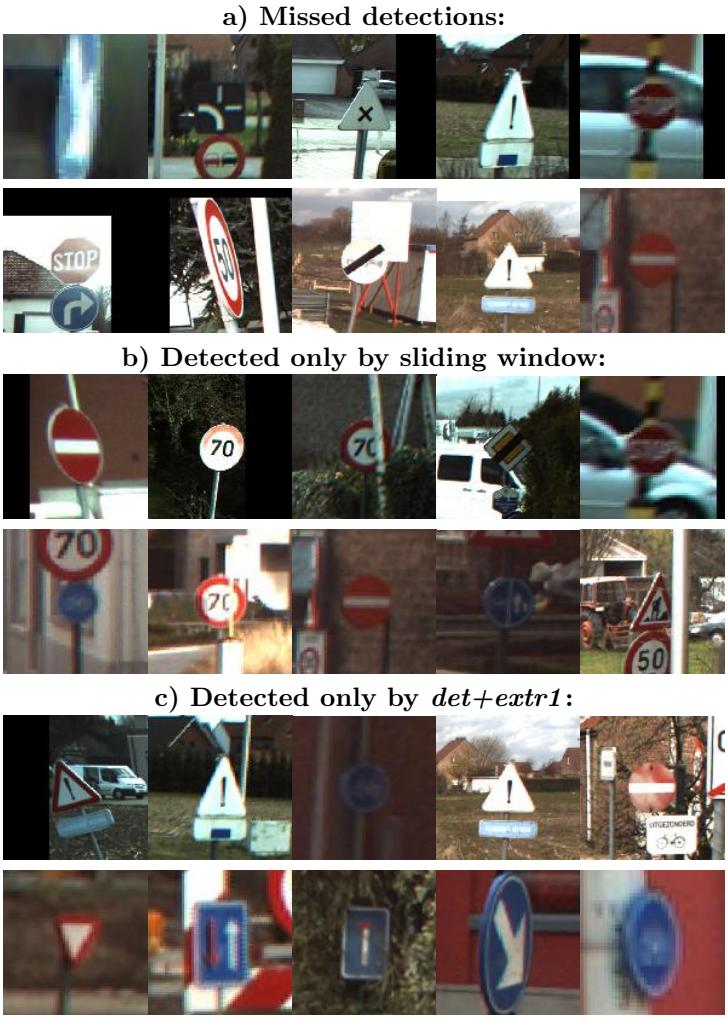

Fig. 1.9. **Complementarity of sliding window and the proposed approach.** Shown are samples where one method fails but the other one is successful (**a,b**), at the same threshold level.

ues — intensity (I) and pyramids of histogram of oriented gradients (PHOG) as were used in [18], where it was shown that for classification purposes the color information is not essential when the samples are known to be traffic signs. The most discriminative features of the traffic sign are the inner pattern and the shape. This is of course an explicit design goal of traffic signs. We investigate the supervised dimensionality reduction based on Linear Discriminant Analysis (LDA) with regularization [37]. The features are l_2-norm normalized to sum to 1. The LDA regularization parameter is 0.1.

Table 1.2 depicts the performance achieved using the Sparse Representation-

based Classifier (SRC), Linear kernel SVM (LSVM), Intersection Kernel SVM (IKSVM), Polynomial Kernel SVM (POLYSVM) and Radial Basis Function SVM (RBFSVM). The classifiers and their settings are as in [18]. PHOG features are more discriminative than the raw intensity values, but this comes with an increase in time for computing the features and training the classifiers. While provided here, SRC is not applicable for our task as it is 10 to 100 times slower than the other SVM classifiers when run for test. The PHOG features with IKSVM is the winning setting, achieving 97.79% on the BelgiumTSC data. Working on LDA projected features speeds up the training and the testing. SRC benefits greatly from the discriminatively projected features. We use here LSVM+LDA+PHOG because of the high speed and performance.

Table 1.2. Recognition rates on the BelgiumTSC database.

Features	I	PHOG	LDA+I	LDA+PHOG
SRC [%]	85.04	92.34	**94.32**	**97.36**
LSVM [%]	90.69	96.68	91.28	96.96
IKSVM [%]	**91.36**	**97.79**	91.67	96.80
POLYSVM [%]	89.94	96.61	91.87	96.80
RBFSVM [%]	90.41	96.57	91.99	97.16

1.8.3. *3D Tracking Results*

We begin by showing qualitative examples. In figure 1.10 we show our system tracking a pedestrian crossing sign and obtaining an accurate pose even when the object is far from the camera. Similarly, in figure 1.11 we show our algorithm successfully tracking a give way sign. An application which comes to mind is to calculate the distance between each sign and the car, to then automatically adjust the speed of the car. Figure 1.12 shows our system tracking multiple round traffic signs. Currently we estimate each pose independently, though of course relative to the camera each sign undergoes the same rigid transformation. This coupling would results in improved performance.

As mentioned several times in this paper, our system is able to deal with occlusions. Figure 1.13 shows our system successfully tracking a parking sign, in spite of it being occluded by a tree.

Fig. 1.10. Filmstrip showing frames from a video tracking a pedestrian crossing sign

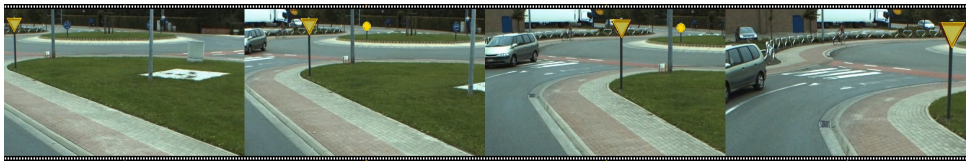

Fig. 1.11. Filmstrip showing frames from a video tracking a triangular sign

Fig. 1.12. Filmstrip showing frames from a video tracking multiple round signs

Fig. 1.13. Filmstrip showing frames from a video tracking a parking sign, in spite of it being occluded by a tree

We compared the performance of our system by tracking a single sign over a distance of 60m (or 60 frames at 36km/s), with and without the 3D tracker. The car was moving along a straight line, with constant speed. Translation should therefore change linearly, while rotation should remain constant. Figure 1.14 shows the results for a circular sign, figure 1.15 for a rectangular sign and figure 1.16 for a triangular sign. At higher distances the object shrinks to only a few tens of pixels, so detection and tracking are prone to errors. Still, the 3D tracker is able to provide smoother values for the translation. Though there may be little difference between using the 3D tracker and using just the 4 point planar pose recovery algorithm with regard to the translation, the tracker must be used to reliably obtain the rotation.

A video processed by our system is available at `http://homes.esat.kuleuven.be/~rtimofte`. It shows our system tracking multiple objects of different types, orientations and colors, with or without motion blur and occlusions. Our CPU implementation of the detection phase runs at around 20fps on 640×480 images while the GPU based tracker needs up to 20ms per object (on a 640×480 image).

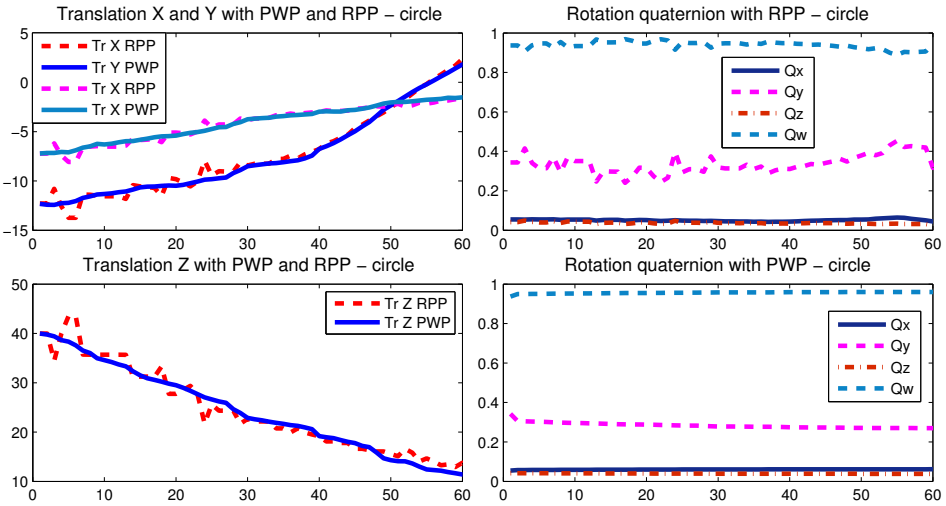

Fig. 1.14. System performance while tracking a circle shaped sign over 60m, with just the 4 point pose recovery (RPP) and with the tracker (PWP)

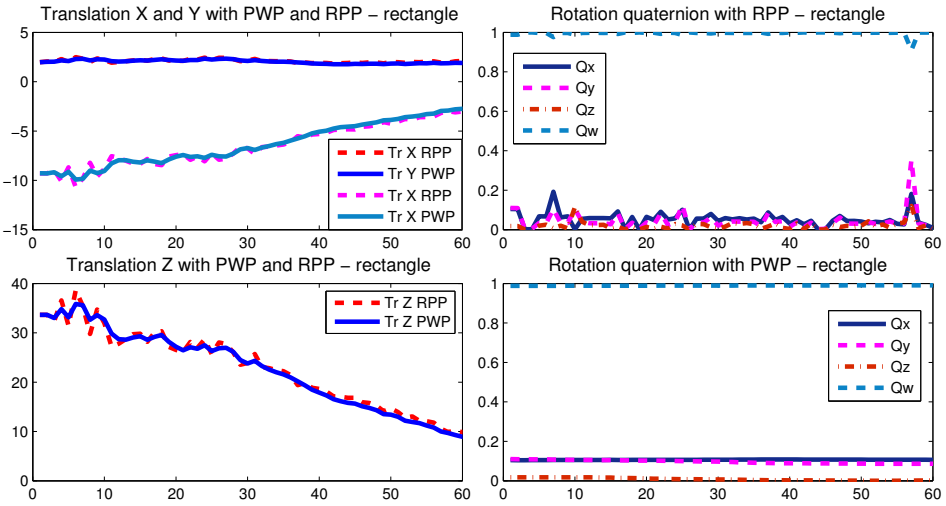

Fig. 1.15. System performance while tracking a rectangle shaped sign over 60m, with just the 4 point pose recovery (RPP) and with the tracker (PWP)

1.9. Conclusions

In this chapter we reviewed the advances in driver assistance with respect to traffic sign recognition and proposed a system that can track multiple traffic signs in 3D, from a single view. By integrating accurate detections with 3D region based tracking

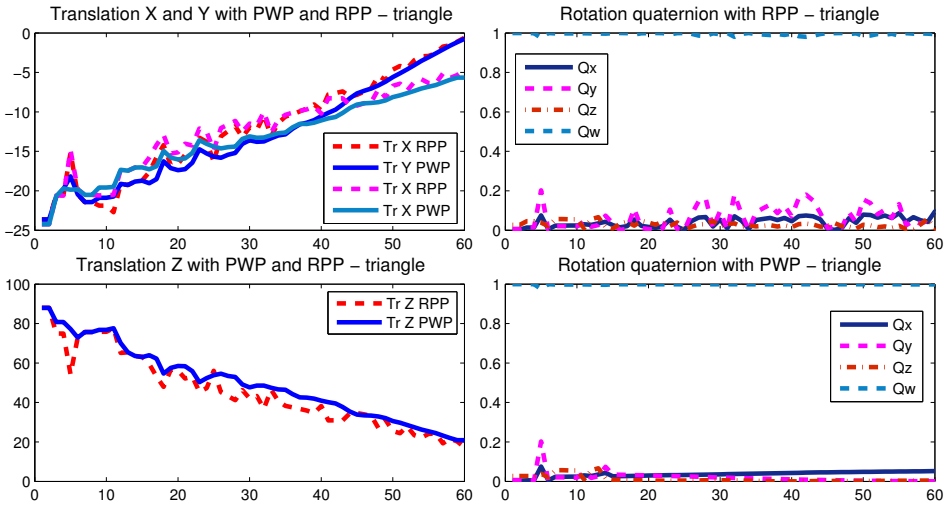

Fig. 1.16. System performance while tracking a triangle shaped sign over 60m, with just the 4 point pose recovery (RPP) and with the tracker (PWP)

our system is robust to motion blur and occlusions, while still running in real time. Such a system would require more, e.g. person and car 3D detection and tracking, etc.

Acknowledgments

This work was supported by EC FP7-EUROPA and Flemish IBBT-ISBO projects, in continuation of research started under IBBT-URBAN project, and by EPSRC through a DTA grant. The authors thank GeoAutomation for providing the images.

References

[1] C.-Y. Fang, S.-W. Chen, and C.-S. Fuh, Road-sign detection and tracking, *Vehicular Technology, IEEE Transactions on.* **52**, 1329–1341, (2003).
[2] A. Ruta, Y. Li, and X. Liu, Real-time traffic sign recognition from video by class-specific discriminative features, *Pattern Recognition.* **43**, 416–430, (2010).
[3] M. Meuter, A. Kummert, and S. Muller-Schneiders. 3d traffic sign tracking using a particle filter. In *Intelligent Transportation Systems*, pp. 168–173.
[4] L. D. Lopez and O. Fuentes. Color-based road sign detection and tracking. In *ICIAR*, pp. 1138–1147, (2007).
[5] G. Piccioli, E. De Micheli, P. Parodi, and M. Campani. Robust road sign detection and recognition from image sequences. In *Intelligent Vehicles Symposium*, (1994).
[6] R. Timofte, K. Zimmermann, and L. Van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. In *WACV*, Snowbird, Utah, USA, (2009).
[7] S. Maldonado, S. Lafuente, P. Gil, H. Gómez, and F. López, Road-sign detection and

recognition based on support vector machines, *IEEE Trans. Intelligent Transportation Systems, pp 264–278*. **8**(2), (2007).

[8] H. Fleyeh, S. Gilani, and M. Dougherty. Road sign detection and recognition using a fuzzy ARTMAP: A case study swedish speed-limit signs". In *10th IASTED International Conference on Artificial Intelligence and Soft Computing*, (2006).

[9] S. Vitabile, G. Pollaccia, G. Pilato, and F. Sorbello. Road signs recognition using a dynamic pixel aggregation technique in the hsv color space. In *ICIAP*, (2001).

[10] M. Lalonde and Y. Li. Road sign recognition - survey of the state of art. Technical Report CRIM-IIT-95/09-35, (1995).

[11] D. Kellmeyer and H. Zwahlen. Detection of highway warning signs in natural video images using color image processing and neural networks. In *ICNN*, (1994).

[12] G. Loy and N. Barnes. Fast shape-based road sign detection for a driver assistance system. In *IROS*, vol. 1, pp. 70–75, (2004).

[13] D. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *ICCV*, pp. 87–93, (1999).

[14] R. Timofte, K. Zimmermann, and L. Van Gool, Multi-view traffic sign detection, recognition, and 3d localisation, *submitted to Machine Vision and Applications*. (2011).

[15] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2010).

[16] V. A. Prisacariu, R. Timofte, K. Zimmermann, I. Reid, and L. V. Gool, Integrating object detection with 3d tracking towards a better driver assistance system, *ICPR*. pp. 3344–3347, (2010). doi: http://doi.ieeecomputersociety.org/10.1109/ICPR.2010.816.

[17] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *submitted to International Joint Conference on Neural Networks*, (2011).

[18] R. Timofte and L. Van Gool. Fast approaches to large-scale classification. In *submitted to International Joint Conference on Neural Networks*, (2011).

[19] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR 2005*, vol. 1, pp. 886–893, (2005).

[20] S. Lafuente, P. Gil, R. Maldonado, F. López, and S. Maldonado. Traffic sign shape classification evaluation i: Svm using distance to borders. In *IEEE Intelligent Vehicles Symposium, pp 654–658*, (2005).

[21] C. Nunn, A. Kummert, and S. Muller-Schneiders. A novel region of interest selection approach for traffic sign recognition based on 3d modelling. In *IEEE Intelligent Vehicles Symposium, pp 654–658*, (2008).

[22] N. Pettersson, L. Petersson, and L. Andersson. The histogram feature - a resource-efficient weak classifier. In *IEEE Intelligent Vehicles Symposium, pp 678 - 683*, (2008).

[23] F. Moutarde, A. Bargeton, A. Herbin, and L. Chanussot. Robust on-vehicle real-time visual detection of american and european speed limit signs, with a modular traffic signs recognition system. In *IEEE Intelligent Vehicles Symposium*, (2007).

[24] A. Ruta, Y. Li, and X. Liu. Towards real-time traffic sign recognition by class-specific discriminative features. In *British Machine Vision Conference*, (2007).

[25] A. Broggi, P. Cerri, P. Medici, P. Porta, and G. Ghisio. Real time road signs recognition. In *Intelligent Vehicles Symposium, 2007 IEEE*, pp. 981–986 (June, 2007).

[26] S. J. Osher and R. P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. (Springer, October 2002). ISBN 0387954821.

[27] B. Rosenhahn, T. Brox, and J. Weickert, Three-dimensional shape knowledge for joint image segmentation and pose tracking, *International Journal of Computer Vision*. **73**

(3), 243–262, (2007).
[28] C. Schmaltz, B. Rosenhahn, T. Brox, D. Cremers, J. Weickert, L. Wietzke, and G. Sommer. Region-based pose tracking. In *Proc. 3rd Iberian Conference on Pattern Recognition and Image Analysis* (June, 2007).
[29] V. Prisacariu and I. Reid. Pwp3d: Real-time segmentation and tracking of 3d objects. In *Proceedings of the 20th British Machine Vision Conference* (September, 2009).
[30] C. Bibby and I. Reid. Robust real-time visual tracking using pixel-wise posteriors. In *ECCV 2008*, pp. 831–844, (2008).
[31] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, pp. 384–393, (2002).
[32] P. Viola and M. Jones. Robust real-time face detection. In *IEEE International Conference on Computer Vision*, vol. 2, pp. 747–757, (2001).
[33] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pp. 401–408, (2007).
[34] S. Maji and J. Malik. Fast and accurate digit classification. Technical Report UCB/EECS-2009-159, EECS Department, University of California, Berkeley, (2009).
[35] C. ping Lu, G. D. Hager, I. C. Society, and E. Mjolsness, Fast and globally convergent pose estimation from video images, *T-PAMI*. **22**, 610–622, (2000).
[36] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html.
[37] D. Cai, X. He, and J. Han, Srda: An efficient algorithm for large-scale discriminant analysis, *IEEE Trans. on Knowl. and Data Eng.* **20**(1), 1–12, (2008).