DIRAC Training Report: Single view traffic sign detection

Karel Zimmermann

July 26, 2008

1 Proposed method

We studied different object detection and recognition techniques with special attention on objects present in urban environment. Since many of these objects are meant to be very well distinguishable based on color or shape (e.g. traffic signs, road markings, touristic navigation signs, etc.) we designed a special detection pipeline which allows to detect most of these objects for very low computational cost. The pipeline which is outlined on Figure 1, consists of the following stages:

- **Segmentation** - very fast preprocessing step, where the optimal combination of simple (i.e. computationally cheap) methods adjustable by many different parameters (e.g. color thresholding or shape detection) selects thousands of possible bounding boxes where the object might be present. Segmentation is designed to have very low False Negative (FN, i.e. missed traffic signs) and reasonably high False Positives (FP, i.e. hallucinated traffic signs). An example of segmentation is a selection of connected components of a thresholded image.
- **Detection** - These bounding boxes are further verified by a binary classifier which filters out the bounding boxes containing background samples. We use cascades of Adaboost classifiers combined with Haar-like features.
- **Recognition** - Types of traffic signs within selected bounding boxes will be distinguished by a multiclass classifier, but we haven't yet worked on it.

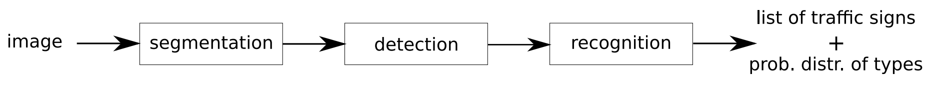

Figure 1: **Proposed pipeline** consists of segmentation, detection and recognition.

We formulate the learning of the segmentation as the Boolean Linear Optimization problem, which allows to consider various trade-offs among FN, FP and computational

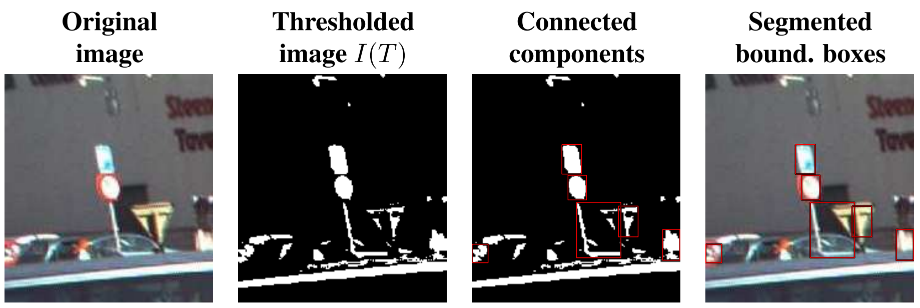

Figure 2: **Demonstration of simple segmentation** method for threshold $T = (0.5, 0.2, -0.4, 1.0)^\top$

cost. The proposed method was implemented in MATLAB environment and preliminary experiment showed that searching for the globally optimal solution is tractable in minutes if the data is preprocessed, i.e. scores of different methods on the training data set are precomputed.

2 Segmentation

The simplest (very often used) segmentation method is detection of connected component of a thresholded image, as demonstrated in Figure 2, where *thresholded image* with threshold $T = (t, a, b, c)^\top$ of a color image with color channels (I_R, I_G, I_B) is defined as follows:

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

Since there is no single threshold performing well by itself, it is necessary to combine different thresholds $\mathcal{T} = \{T_1, T_2, \dots\}$. The more thresholds is used the lower the FN but higher the FP and higher the computational cost.

2.1 Learning segmentation

Learning of segmentation is searching for $\mathcal{T} \subseteq \mathcal{P}\{\mathbb{T}\}$ (where $\mathcal{P}\{\mathbb{T}\}$ denotes power set of the set of considered thresholds $\mathbb{T}$) minimizing an error subject to some constraints. Even though the interesting learning task are NP-complete (set covering problem), we formulate them as the Boolean Linear Programming problems and experimentally show that they are tractable even for very large $\mathcal{P}(\mathbb{T})$.

The simplest learning task is searching for the optimal trade-off between FP and FN

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} \text{FP}(\mathcal{T}) + K_1 \cdot \text{FN}(\mathcal{T}),$$

where FP$(\mathcal{T})$ stands for FP of subset of selected thresholds $\mathcal{T}$ measured on a training set and FN$(\mathcal{T})$ stands for FN, respectively. K_1 is scalar value weighting FP and FN.

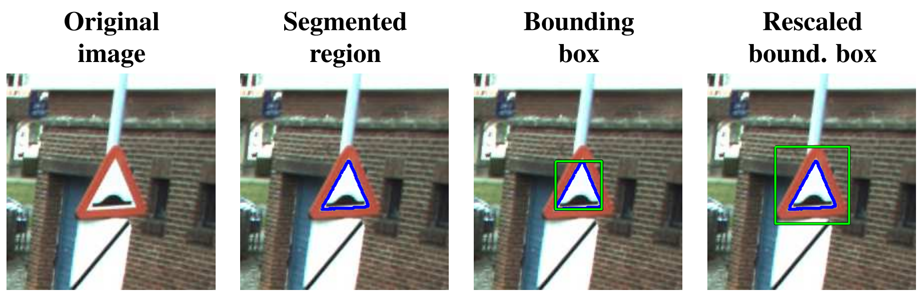

Figure 3: **Demonstration of the extended threshold.** The object is not well locally separable from background, because bricks have similar color as the red boundary. Therefore the inner white part is segmented and the received bounding is rescaled $\overline{T} = (0.1, -0.433, -0.250, 0.866, 1.6, 1.6)^\top$.

In order to avoid overfitting and also keep the segmentation reasonably fast we introduce constraint on the cardinality card$(\mathcal{T})$ of the subset of selected thresholds $\mathcal{T}$.

This could be either hard constraint card$(\mathcal{T}) < \epsilon$ or soft constraint, where $K_2 \cdot$ card$(\mathcal{T})$ become part of criterion function:

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} \ \text{FP}(\mathcal{T}) + K_1 \cdot \text{FN}(\mathcal{T}) + K_2 \cdot \text{card}(\mathcal{T}),$$

We had better results with the soft constraint, but the hard constraint is an option if the running time is an issue (e.g. real-time requirements are imposed).

Since accuracy is usually quite important we also add the term assuring that the more accurate thresholds are prefered:

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} \ \text{FP}(\mathcal{T}) + K_1 \cdot \text{FN}(\mathcal{T}) + K_2 \cdot \text{card}(\mathcal{T}) + K_3 \cdot \text{accuracy}(\mathcal{T}),$$

Scalars K_1, K_2 and K_3 are parameters of the proposed method and we estimate them by cross-validation test.

2.2 Scaling of segmented bounding boxes

Segmentation based on thresholding can perform well if and only if the object is locally threshold separable from its background. If the condition is violated the traffic sign cannot be revealed. We observed that in many traffic signs, the inner part contains some very well threshold separable regions, the segmentation of which is no longer dependent on background color. See for example Figure 3. We therefore introduce extended threshold

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

which consists of original threshold T and scaling of the segmented bounding box (s_r, s_c). Such threshold[1] can reveal traffic sign, even if the outer boundary is not well

[1]We further refer to the extended threshold as threshold.

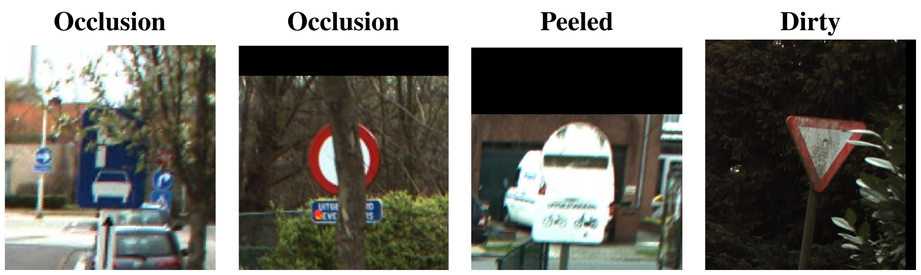

Figure 4: **Not segmentable traffic signs.** Traffic signs are not well locally separable from background.

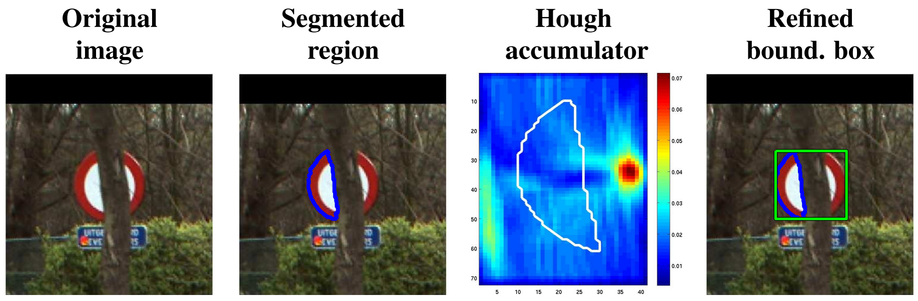

Figure 5: **Shape segmentation principle.** Traffic signs are not well locally separable from background.

locally threshold separable, as shown in Figure 3, where bricks have similar color as the red boundary of the object. Learning is then searching for the subset of 6-dimensional thresholds.

2.3 Stability of segmented regions

If the illumination of traffic sign changes, the learned threshold might not segment the traffic sign from its background with sufficient accuracy. Since it is impossible to learn that many thresholds which would cover all the illumination variations, we adjust the threshold to be *locally stable* in the sense defined by MSER. Instead of using the bounding box directly segmented by the learned threshold (t, a, b, c, s_r, s_c) , we use bounding boxes from MSERs detected within range

$$< (t-\epsilon, a, b, c, s_r, s_c); (t+\epsilon, a, b, c, s_r, s_c) >,$$

where ϵ is parameter of the method. Since MSERs are by themselves defined by stability parameter Δ, the method is parametrized by two parameters as MSER(ϵ, Δ)

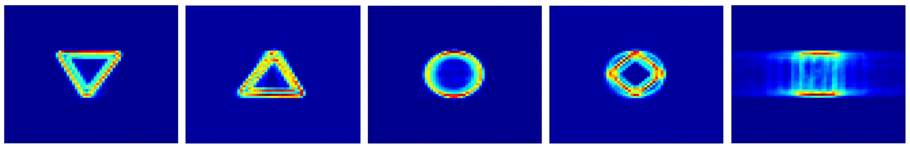

Figure 6: **Fuzzy template for separated shapes.** Triangle-up, triangle-down, circle, rhombus and rectangle. Notice that the first shape is triangle-up, but its fuzzy template correspond to its point reflection.

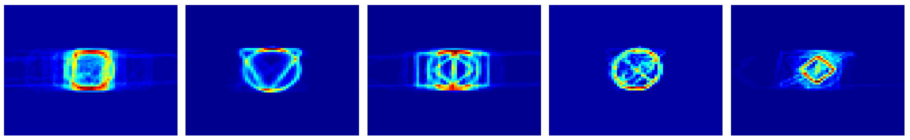

Figure 7: **Threshold specific fuzzy templates.** Selected subset $\{23, 12, 28, 32, 44\}$ from 44 fuzzy-templates.

2.4 Shape segmentation

Most of the traffic signs are segmentable by the previously described method however, there are some exceptions like partially occluded, peeled or dirty traffic signs as shown in Figure 4.

For these traffic signs we propose a shape segmentation method based on the generalized Hough transformation, which estimates known shapes in the boundary of the segmented region. Principle is outlined in Figure 5.

In general there are several different shapes under many different affine transformations. Using the Hough transformation would require to detect every single shape in 5 (or even 6) dimensional Hough accumulator. Hence, instead of using predefined set of shapes in a high dimensional accumulator, we decided to learn so-called *fuzzy templates* which incorporate small affine transformations and shape variations and explicitly model only position and scale in 3 dimensional Hough accumulator. The most straightforward fuzzy templates, are templates learned as a probability distribution of boundaries of segmented regions of predefined shapes, see Figure 6.

Such approach, however, requires to run the Hough transformation that many times as many shapes are distinguished. Since the learned thresholds are usually highly specialized for some kinds of traffic signs, we learn threshold-specific fuzzy templates, see Figure 7 for some examples. For each threshold, we first collect boundaries of segmented regions which cast correct bounding boxes. Then the scale is normalized (aspect ratio is preserved) and the probability distribution of the shapes segmented by the threshold is computed. Eventually, the fuzzy template is estimated as the point reflection of the probability distribution, because voting in the Hough accumulator requires the inverse shape. It means, that for example the second fuzzy template in Figure 7 mainly corresponds to the traffic signs which look like triangle-*up* or circle, but the fuzzy template corresponds to its point reflection, i.e. triangle-*down* and circle.

When a boundary is segmented by a threshold, the threshold-specific fuzzy tem-

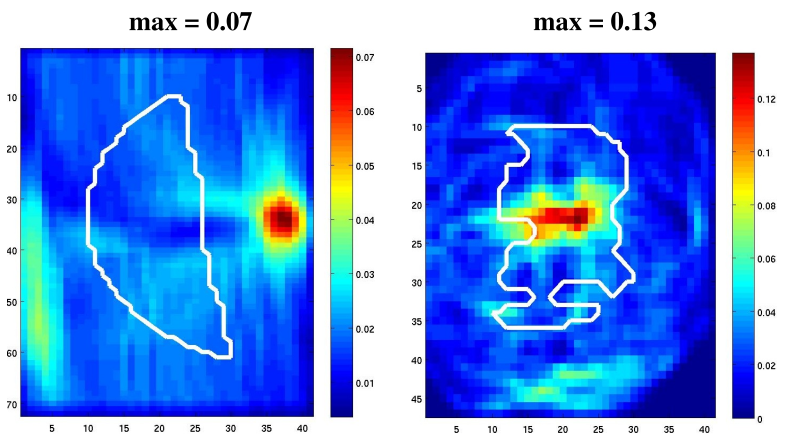

Figure 8: **Maxima in the Hough accumulator.** Maximum of the correct shape is smaller than maximum of the region with curly boundary because of fuzzyness of the template.

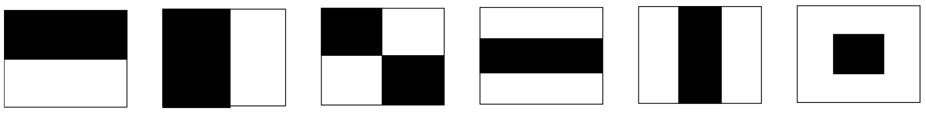

Figure 9: **Haar-like features** used in our implementation.

plate is used to compute the Hough transformation. Bounding box corresponding to the maximum in the three dimensional Hough accumulator (2 positions and 1 scale) is reported if the maximum is sufficiently high.

We observed that curly shapes, like the one depicted in Figure 8 often cast higher maxima in the Hough accumulator than the correct shape due to the fuzzyness of the template. Therefore almost every maximum have to be considered to be sufficiently high, which makes FP almost two times higher.

3 Detection

Detection is based on well known Viola and Jones' Discrete AdaBoost classifier. In our experiments, we generated in total 8.10^4 Haar-like features based on 6 Haar patterns outlined on Figure 9. These features are computed independently for different color channels. Since we have observed that the detection performs better on the HSI color space than on the RGB or on the gray-scale, the HSI with the intensity channel (I) normalized by the variance is used. We have experimentally shown that cascade of AdaBoost classifiers runs 10 times faster than single Adaboost on the same performance level and provides a better control of the overall performance.

We also trained separate classifiers for the main classes of shapes (triangle, circle, rectangle).

4 Experiments

Large dataset (including arrows texts and many sub-signs					
	FN BBs		FN TSs		FP per
	[%]	#/4333	[%]	#/1477	2MP img
Segm (thresh.)	2.6%	113	0.94%	14	4 245
Segm (thresh.+MSER1)	2.4%	106	0.88%	13	4 852
Segm (thresh.+MSER2)	2.1%	93	0.88%	13	6 714
Segm (thresh.+shape)	2.1%	92	0.74%	11	8 126
Segm (thresh.+shape+MSER1)	2.0%	87	0.67%	10	8 562
Segm (thresh.+shape+MSER2)	1.8%	78	0.67%	10	10 684
Det + Segm (thresh)	19.8%	856	11.4%	168	2.7
Det + Segm (thresh+MSER1)	18.9%	820	10.8%	159	3.2
Det + Segm (thresh+MSER2)	18.4%	796	10.7%	158	4.0
Det + Segm (thresh)^2	10.3%	446	5.1%	75	15.1

Reduced dataset (basic traffic signs)					
	FN BBs		FN TSs		FP per
	[%]	#/3756	[%]	#/1274	2MP img
Segm (thresh.)	1.4%	53	0.5%	6	4 307
Bulk + Segm (thresh)	7.4%	279	4.3%	55	2.2
Sep.casc. + Segm (thresh)	7.5%	282	4.0%	51	1.1
Bulk + Segm (thresh)^2	4.8%	182	2.6%	33	9.1

Table 1: **Summary of achieved results**. Meaning of the above used abbreviations is the following **thresh** means method described in Section 2.1+2.2, **MSER** is method from Section 2.3 and MSER1 stands for MSER(ϵ, Δ) = MSER$(0.1, 0.2)$ and MSER2 is MSER$(0.1, 0.05)$, **shape** is Section 2.4. **FN BBs** means false negative with respect to bounding boxes, **FN TSs** means false negative with respect to traffic signs. Comparison of previous dataset (without turist signs and some information tables) and reduced dataset (without arrows and all information tables and undersigns).
[2]Just another point of the detector's ROC curve.

4.1 Ground truth data

Ground truth data so far provided by GeoAutomation consists of 7356 labeled images (in total 11219 bounding boxes), which correspond to 2459 poles. These images contain 115 out of 210 defined traffic signs. Since some of the defined traffic signs are combinations of basic traffic signs (e.g. "triangle with undersign" is defined as a special traffic sign, or array of traffic signs), the experiment was conducted only on selected subset of 90 basic traffic signs out of provided 115 types. More complex traffic signs will be detected as a spatial constellation of these basic traffic signs. In addition to the positive images, we also obtained 2974 non traffic signs images to make the extraction of negative examples easier.

Since the visual appearance of information tables, text undersigns and arrows is not well guaranteed we also perform experiment on *reduced dataset*, which contains 76 types out of 115 provided types.

4.2 Evaluation of the proposed method

Many different experiments with different numbers of features and different settings of segmentation methods were conducted. We present the best so far achieved results learned on 840 traffic signs, i.e. 4261 bounding boxes and on 1477 testing traffic signs, 4333 bounding boxes, respectively. Error is evaluated with respect to both bounding boxes (BBs) and traffic signs (TSs). Since the traffic sign is usually present on several views (usually 3 views per traffic sign are available), the error with respect to traffic signs is significantly lower.

We define *coverage-ratio* between two bounding boxes as ratio between their intersection and union. The error in Table 1 is evaluated with respect to the predefined accuracy. FN is reported whenever none of the segmented and/or detected bounding boxes does reach *coverage-ratio* ≥ 0.65, which approximately corresponds to the shift of 20×20 bounding box by 2 pixels in both directions.

Note that the testing data contains very challenging traffic signs, for example the smallest traffic sign is 11×10. Approximately 25% of not-segmented bounding boxes were smaller than 17×17, most of the others was either taken from oblique angles and/or significantly visually corrupted. Some of the examples of not segmented traffic signs are demonstrated in Figure 10. Traffic signs, which are not locally threshold separable but which were segmented based on their shape are shown in Figure 10.

5 Summary and conclusions

1. Error on the reduced dataset is 2.5x smaller.
2. Separate cascade is used only on reduced dataset, which also makes the error 2x smaller.

$\Rightarrow$ (1)+(2) gives 5x smaller error, then on the original dataset.

3. We can have for example:

$$FN_{TS} = 4.0\%, FN_{BB} = 7.5\%, FP = 1.1/\text{ 2Mpxl image}$$
$$FN_{TS} = 2.6\%, FN_{BB} = 4.8\%, FP = 9.1/\text{ 2Mpxl image}$$

Note, that for some traffic signs only small number of samples was provided. In this case the traffic sign is present in training data but missing in testing data (in order to avoid testing on training meterial). This means that results are not evaluated on rare traffic signs, if more data are provided results might get little bit worse.

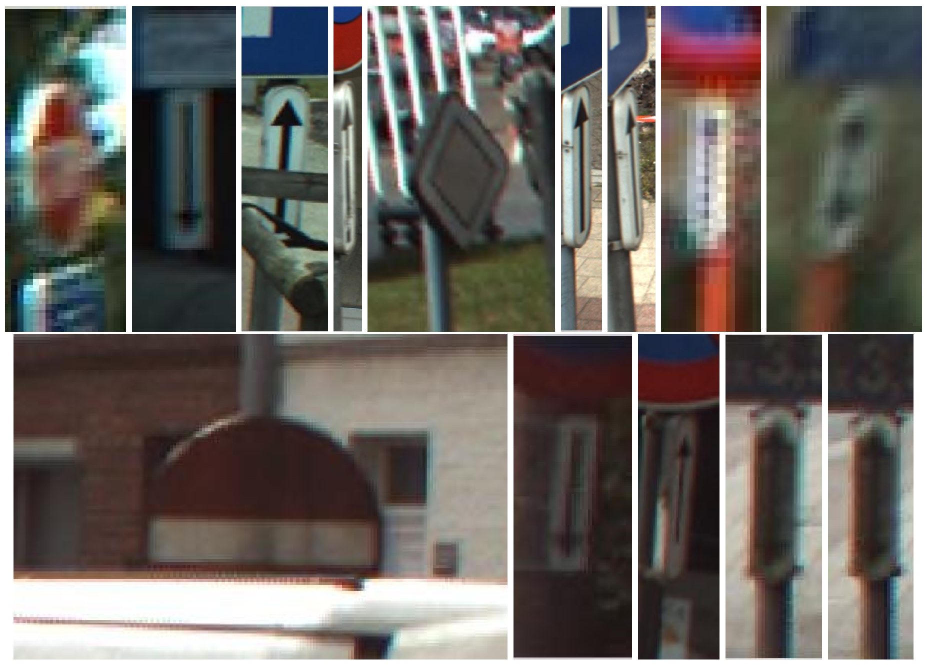
Figure 10: **Examples of not segmented traffic signs**

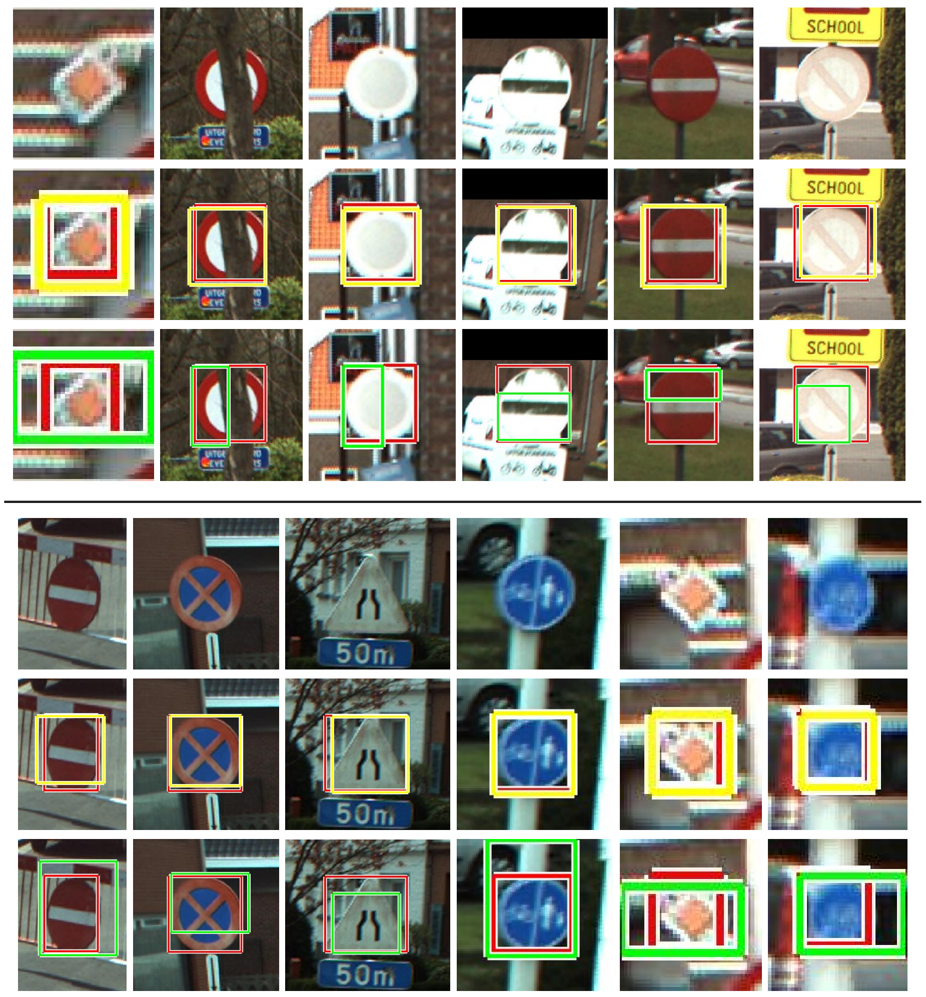

Figure 11: **Shape segmentable but threshold inseparable traffic signs** - ground truth is delineated by red rectangle, the best shape-based segmentation is in yellow and the best threshold-based segmentation is in green.