Del 2.1 Report on the methodology behind the specific object recognition

Del 2.2 Report on the object class recognition implementation

K.U.Leuven - PSI – 1 January, 2010

IBBT vzw, Interdisciplinair Instituut voor Breedbandtechnologie
Gaston Crommenlaan 8 (bus 102), 9050 Gent, België • E info@ibbt.be • T +32 9 331 48 00 • F +32 9 331 48 05 • www.ibbt.be



Report WP 2: Object recognition

In the following are described the methodologies behind specific object and object-class recognition. First we discuss the case of traffic sign recognition[1], then manhole recognition. Both are subject to single view processing and multi-view, 3D processing with 3D mapping purposes. We conclude by showing real-time performance for traffic sign detection and recognition along with 3D pose tracking for a driver assistance system[2].

1 Multi-view traffic sign detection, recognition, and 3D localization

Given the camera parameters from WP1, we aim at 3D mapping the traffic signs detected in any camera of the van. We do not stop at single view detection and recognition, but include 3D localisation as well. Localisation probably most resembles that of Cornelis et al. [3], who also combined explicit 3D information with 2D car and person detection in a mobile city mapping context. An important difference with this earlier work lies in the far less stringent constraints offered by the 3D scene layout (no longer objects restricted to a ground plane, see Fig. 1) and the looser spatial arrangements when looking for traffic signs. Moreover, traffic signs have been designed to come as subclasses, and different types of traffic signs within the same subclass have the same shape and colour distributions. The distinction lies in rather small details. These need to be picked up by the system. Hence, the challenge is one of detecting traffic signs irrespective of the typical problems of changing appearances and occlusions, but at the same time recognizing specific sign types based on small differences. For more details see [1].



Figure 1: 3D mapped traffic signs in a reconstructed scene

1.1 Overview of the system

We do not propose online driver assistance but an offline traffic sign mapping system performing optimization over the captured views. The considered traffic signs are those that are captured at a distance of less than 50 meters. The proposed system first processes single images independently, keeping the detection rate very high and the number of false positives (FP) reasonable. Single-view traffic sign detection in conjunction with the use of scene geometry subsequently allows for global optimization which performs 3D localisation and a refinement simultaneously. Since we deal with hundreds of thousands of high-resolution images the approach is to quickly throw out most of the background, to then invest increasing amounts of time on whatever patterns survive previous steps. We now describe each subsequent step of the single-view and multi-view processing pipelines in more detail.

The single-view detection phase consists of the following steps:

1) Candidate extraction - very fast preprocessing step, where the optimal combination of simple (i.e. computa tionally cheap), adjustable methods selects bounding boxes with possible traffic signs. This step is designed to yield very few false negatives (FN, i.e. the number of missed traffic signs), while keeping the number of false positives (FP, i.e. the number of accepted background regions) in check. This part of the pipeline is described in more detail in section 1.2, as most of the novelty in single-view processing is here.

2) Detection - Extracted candidates are verified further by a binary classifier which filters out remaining background regions. It is based on the well known Viola and Jones Discrete AdaBoost classifier [4]. The 6 Haar-like patterns used are shown in Fig. 2. Detection is performed by cascades of AdaBoost classifiers, followed by an SVM operating on normalized RGB channels, pyramids of HOGs [5] and AdaBoost-selected Haar-like features.

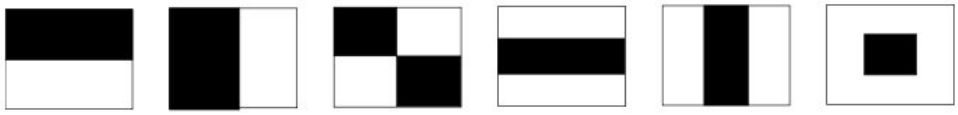


Figure 2: Haar-like patterns

3) Recognition - Six one-against-all SVM classifiers select one of the six basic traffic sign subclasses (triangle-up, triangle-down, circle-blue, circle-red, rectangle and diamond) for the different candidate traffic signs. They work on the RGB colour channels normalized by the intensity variance.

The multi-view phase consists of the following steps:

4) Multi-view hypothesis generation - We search for possible correspondences among the remaining candidates, within a predefined radius in 3D space. Every geometrically and visually consistent pair is used to create a 3D hypothesis. Geometric consistency amounts to checking the position of the backprojected 3D hypothesis against the 2D image candidates. Visual consistency gives higher weight to pairs with the same basic shape.

5) Multi-view MDL hypothesis pruning - Minimum description length is used to select the subset of 3D hypotheses which best explains the overall set of 2D candidates. A side product of the MDL optimization is quite a clean set of 2D candidates corresponding to each particular 3D hypothesis. These candidates allow for hypothesis position refinement. Usually, steps 4) and 5) are iterated. More details are given in section 1.3.

6) Multi-view sign type recognition - The collected set of 2D candidates for each 3D hypothesis is classified by an SVM classifier. These classifications then jointly vote on the final type assigned to the hypothesis.

1.2 Single view candidate extraction

For step 1, we start from connected components in a thresholded image, an idea which has already been used in the literature. The principle is demonstrated in Fig. 3, where the thresholded image is obtained from a colour image by application of a colour threshold.

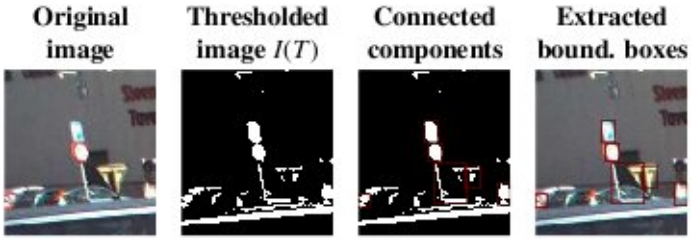


Figure 3: Colour thresholding

Since there typically is no single threshold performing well by itself, it is necessary to combine regions selected by different thresholds T = {T1 , T2 , . . . }, in the sense of adding regions (OR-ing operation). Then, regions passed on by any threshold are going to the next stage, i.e. detection. The more thresholds are used the lower FN can be made but the higher FP risks to get, and the higher the computational cost will be.

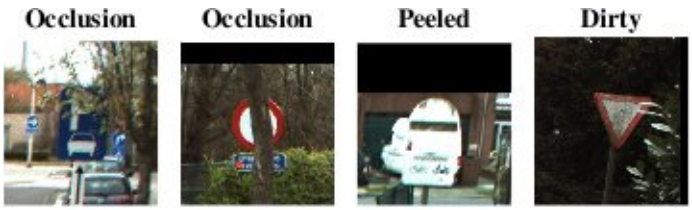


Figure 4: Not colour threshold separable traffic signs

Partially occluded, peeled or dirty traffic signs also should pass the colour test. Therefore, this cannot be made too restrictive. Examples are shown in Fig. 4. That is why we also employ shape information to further refine the candidates. Section 1.2.1 explains how the set of colour thresholds are learned and how, starting from those, the colour-based candidates are extracted. Section 1.2.2 then describes a shape-based Hough transform. This takes the borders of the colour-based candidates as input.

1.2.1 Colour-based candidate extraction

The task is to find the optimal set T of colour thresholds, given some criterion. Since for most interesting such crite ria the problem is NP-complete, we formulate our search as a Boolean Linear Programming (BLP) problem. We have experimentally found that finding the real optimum takes several hours, but that BLP, due to the sparsity of the constraints yields a viable solution within minutes.

The criterion is to search for:

T* = argmin(FP(T)+k1*FN(T)+k2*card(T)-k3*accuracy(T))

where card(T) stands for the cardinality of the set T and accuracy(T) for the average of overlap between ground truth bounding boxes with extracted bounding boxes, while k1,k2,k3 are relative weighting factors estimated by cross-validation. For the reformulation into the Boolean Linear Programming form please consult our published paper [1].

Simple colour thresholding comes out to be insufficient in practice. Therefore we introduce a couple of refinements. Many traffic signs have parts that cannot be separated from the background with any threshold. The rim of the sign is too similar in colour to the background, yet, the inner part can be separated rather easily, like in Fig. 5.

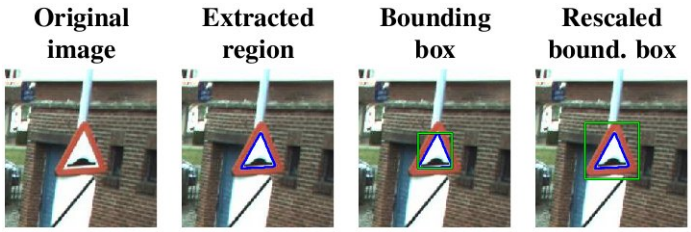


Figure 5: Extended threshold principle

This justifies introducing scaling factors to the original threshold that works for an inner part. Changing illumination poses another problem to thresholding. One could try to adapt the set of thresholds to illumination conditions, but it is better 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), see [1] for details.

1.2.2 Shape-based candidate extraction

Traffic signs have characteristic shapes. Each of the above thresholds (with scaling and TMSER extensions) let pass a series of connected components, i.e. regions. To these regions we now apply an additional filter, akin to the generalized Hough transform. The principle is outline in Fig.6

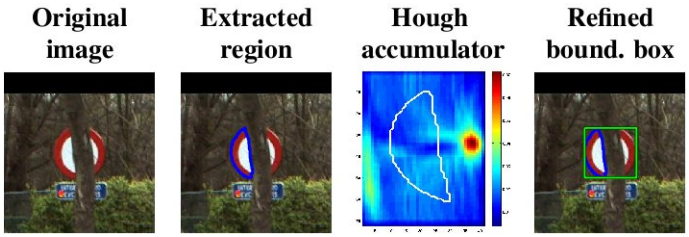


Figure 6: Shape-based extraction demonstration

In general the image shapes of the signs will be affinely transformed versions of the actual shapes. Using the generalized Hough transformation in its traditional form would require to detect every single shape in 5D (or even 6D) Hough accumulator spaces. Instead, we learn fuzzy templates which incorporate small affine transformations and shape variations and we determine explicitly only the position and sale in a 3D Hough accumulator, see [1] for details.

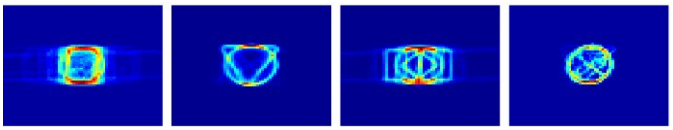


Figure 7: Threshold-specific fuzzy templates

When a boundary is extracted by a threshold, the threshold-specific fuzzy template (see Fig.7) is used to compute the Hough transformation. A 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. The role of the shape selection step mainly consists of selecting a sub-window from a colour- defined bounding box, with the right shape enclosed. We always keep the original bounding box as a separate candidate, however.

1.3 MDL 3D optimization

Single view detection and recognition is just a preprocessing stage, and the final decision results from global optimization over multiple views, based on the Minimum Description Length principle (MDL). Given the set of images, single-view detections, camera positions and calibrations, MDL searches for the smallest possible set of 3D hypotheses which sufficiently explains all detected bounding boxes. In other words, if a set of detected bounding boxes satisfies some geometrical and visual constrains, then all of these bounding boxes are explainable by one 3D traffic sign.

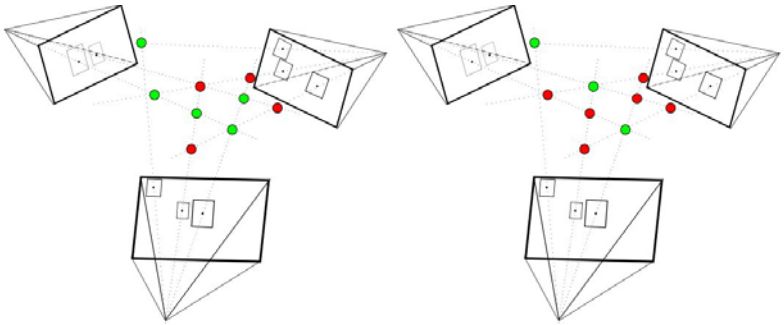


Figure 8: MDL principle - the corresponding pairs generate 3D hypotheses, from which can be picked up (green) a subset (left) or the best/smallest subset (right) in the MDL sense that explains the 2D detections

We start by generating an overcomplete set of hypotheses: For every single 2D detection we collect every geometrically and visually consistent correspondence and use this pair to generate a 3D hypothesis, see Fig. 8. Geometrical consistency means that the corresponding detection lies on the epipolar line for the camera pair. Visual consistency means that their recognized subclass types are the same. This step, of course, generates a high number of 3D hypotheses, including false positives and multiple responses for real traffic signs. The following MDL optimization selects the simplest subset which best explains 2D detections, see Fig.8, right. See [1] for more details.

	FN-TS		FN-BB		FP per
	[%]	#/1274	[%]	#/3756	2MP img
Extr (colour)[1]	0.5%	7	1.5%	58	3 442.4
Extr (colour+TMSER)[2]	0.4%	5	1.4%	53	4 008.5
Extr (colour+shape)[3]	0.2%	2	1.0%	36	6 670.3
Extr (colour+TMSER+shape)[4]	0.1%	1	1.0%	36	7 157.3
Det + Extr[1]	2.4%	31	4.9%	184	2.5
Det + Extr[4]	2.2%	28	4.3%	163	2.5

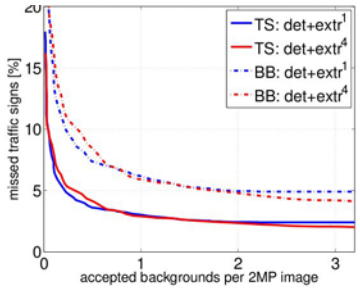


Figure 9: Summary of achieved results in single-view detection. See [1]

1.4 Results

For assessing the performance we annotated a large number of frames from a material recorded in urban environments from Flanders region, by GeoAutomation. We evaluated the system for single-view detection (see Fig.9) and recognition as well for multi-view 3D mapping (see Fig.10). More details about the ground truth data and the experiments are found in [1].

#	No.frames/TSs	Loc.TS[1]	FP[1]	Loc.TS[2]	FP[2]	Rec.TS
1	8×3001 /78	75(96.2%)	9	74(94.9%)	5	98.7%
2	8×6201 /71	68(95.8%)	14	68(85.8%)	13	95.6%
3	8×2001 /44	41(93.2%)	5	41(93.2%)	2	100%
4	8×4001 /76	73(96.1%)	9	73(96.1%)	8	97.3%
$\sum$	8×15204 /269	257(95.6%)	37	256(95.2%)	28	97.7%

Figure 10: Summary of 3D achieved results. See [1] for more details.

A performance in 3D localisation of 95.6% with a very low number of falses is achieved. The recognition rate is around 97.7%. The average 3D mapping accuracy is of 24.54 cm.

2 Multi-view manhole detection, recognition, and 3D localization

The manholes is another class in urban environments to be 3D mapped. The detection of such objects is important also in giving landmarks/references for automatic bundle adjustment of cameras based on an existing map of manholes.

Apparently, the manholes could be treated in the same way like the traffic signs were. But the manholes are less standardized even for the same city, the manholes patterns and sizes vary largely. Also, the color is not anymore an important clue. The shape is the strongest feature, followed by the texture/pattern information. The assumption that manholes are laying on the ground plane is also helpful. You can see Fig. 11 for some manhole patterns.



Fugure 11: Manhole samples

2.1 Overview of the system

We do not propose an online system, but an offline one performing optimization over the captured views. The considered manholes are those that are laying on a 7x7 meters square on the ground plane, in front of the camera. The proposed system first processes single images independently, keeping the detection rate high and the number of false positives (FP) reasonable low. Single-view manhole detection in conjunction with the use of the scene geometry subsequently allows for global optimization which performs 3D localisation and a refinement simultaneously. Since we deal with hundreds of thousands of high-resolution images the approach is to quickly throw out most of the background, to then invest increasing amounts of time on whatever patterns survive previous steps. We now describe each subsequent step of the single-view and multi-view processing pipelines in more detail.

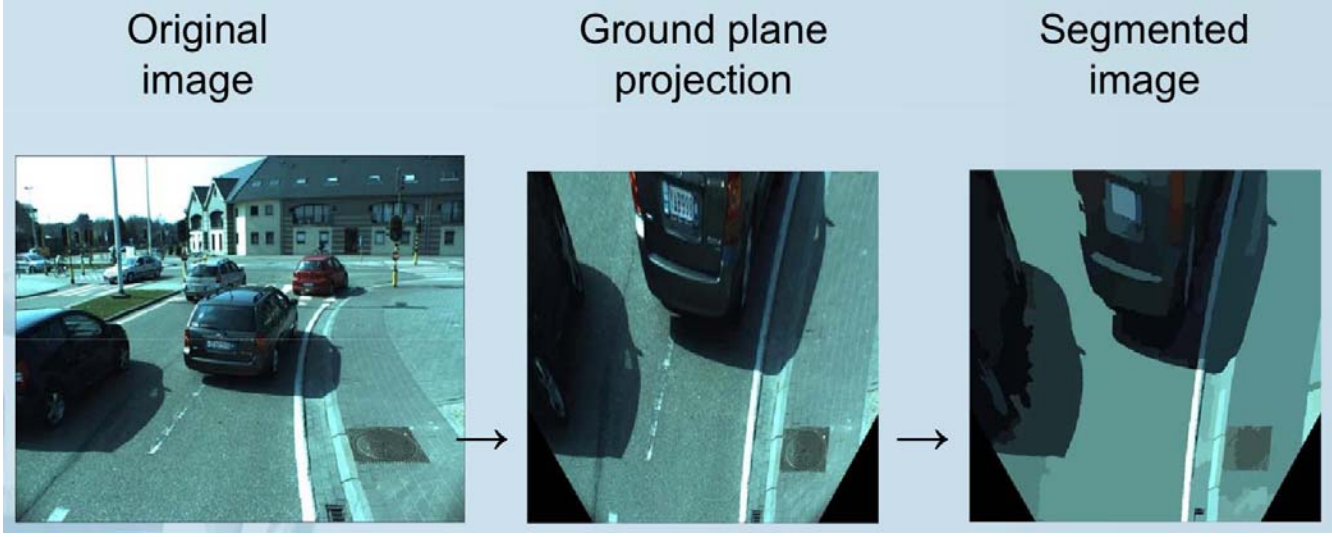


Figure 12: Ground plane projection and mean-shift segmentation

2.2 Single view processing

1) Ground plane projection - under the assumption that the manholes are laying on the ground, and the ground plane surface is not curved, we are fitting a plane beneath the wheels of the van and project the camera view on it. (see Fig. 12)

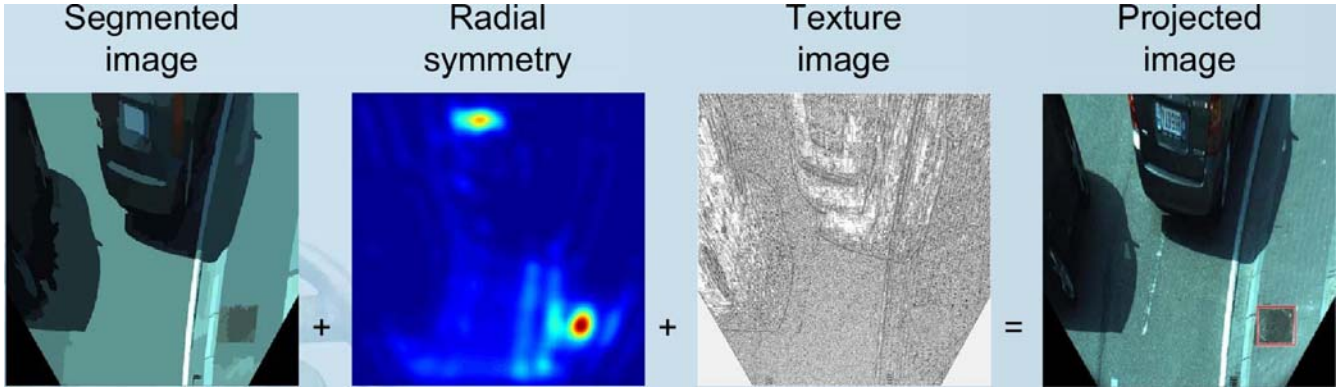


Figure 13: Detection of manholes based on segmentation, radial symmetry and texture information

2) Segmentation - fast preprocessing step, where the mean-shift color segmentation method [8] is used to reduce the potential candidates to the set of segmented regions. It is assumed that usually on one manhole distinguishable from the surrounding area will be one or more color segmented regions (see Fig. 13).

3) Detection - Local Binary Patterns (LBP) is used as basic texture descriptor[9]. Extracted candidates are

verified further by computing likelihoods as ratio between the best mutual information of their LBP histograms and trained models for manholes and that for trained models for backgrounds. Radial symmetry[10] is another assumption that helps to prune the candidates (see Fig. 13).

4) Recognition – the class/type of the manhole is assigned considering the most similar model in texture and size.

2.3 Multi-view processing

5) Grouping - For each detection, based on camera parameters and fitted ground plane, we compute the 3D position. Moved on 3D space, we group the detections based on the proximity (less than 1 meter distance) and texture likelihood. This can be solved by Minimum Description Length formulation (like in [1]). In our case, we found sufficient a simple greedy procedure, by picking first the detection with highest confidence, assigning all other detections that agrees in texture in 1 meter radius to the same cluster, and repeating the procedure with the remaining detections until all the detections are grouped. (see Fig. 14)

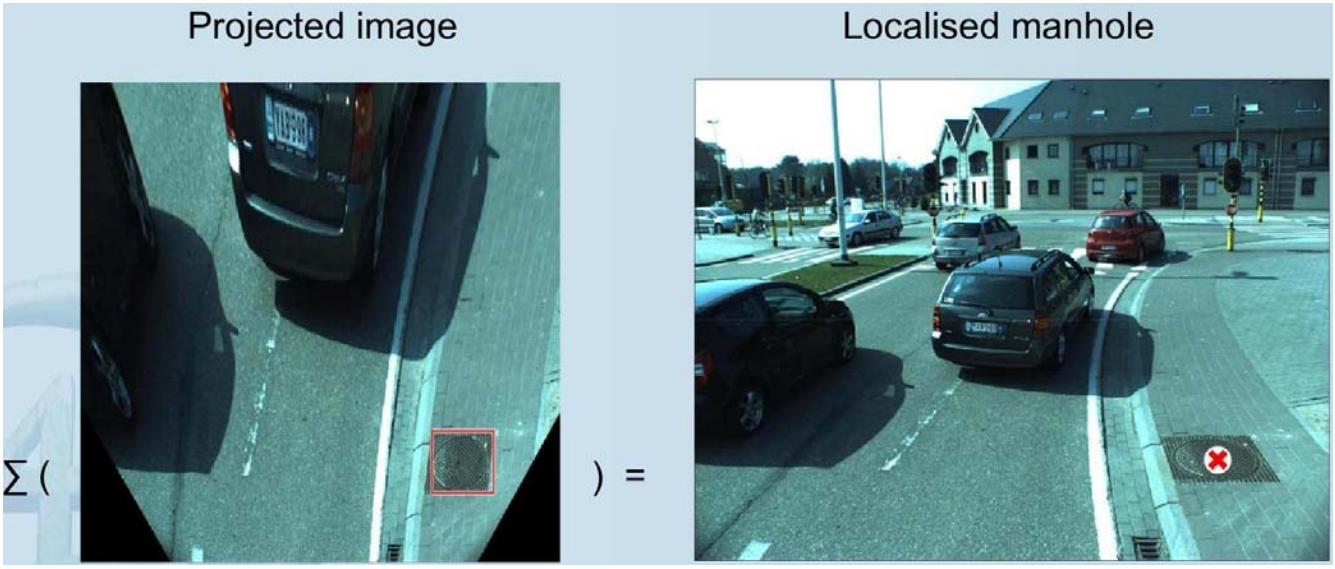


Figure 14: Manholes grouping and localisation

6) 3D accurate localisation - By running a GraphCut-based method [11] over color, texture and other visual cues, we get accurate segmentation of the potential manhole detections against the surrounding background (see Fig. 15). By using the accurate segmentations of the detections grouped into a cluster we are able to obtain through triangulation the best 3D location.

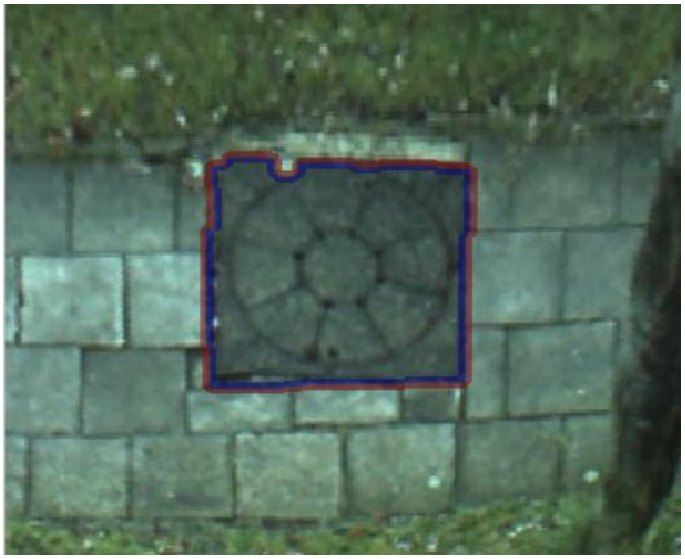


Figure 15: Example of GraphCut refined segmentation for a manhole

7) Voting – for each group is assumed a unique class which is determined through a voting process.

2.4 Results

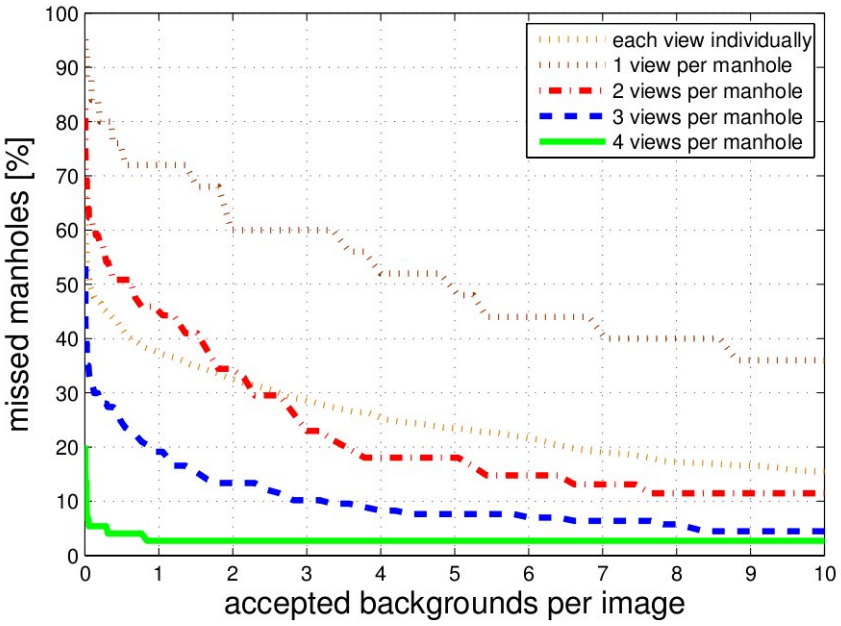


Figure 16: Manhole evaluation

For evaluation we are using 317 manhole annotations and 270 non-manholes images in testing set. Detection rate increases with the number of views available for each manhole (see Fig.16). Single-view detection rate is about 41% for very few false positives/alarms. In multi-view evaluation we achieve 97% manhole detection rate, with very few false positives. In some preliminary experiments we were able to obtain localization accuracies at pixel-level in images and at cm-level in 3D space.

With our Matlab/mex scripts one frame is processed in several seconds.

3 Integrating object detection with 3D tracking towards a better driver assistance system

Accurate 3D pose is required to establish if a traffic sign is relevant to the driver in a driver assistance system scenario. We propose a real-time system that integrates single view detection with region-based 3D tracking of road signs. The optimal set of candidate detections is found, followed by AdaBoost cascades and SVMs. The 2D detections are then employed in simultaneous 2D segmentation and 3D pose tracking, using the known 3D model of the recognised traffic sign. Future work could add person and card 3D detection and tracking to create a complete driver assistance system.

3.1 Overview of the system

An outline of our algorithm is shown in Fig. 17. It consists of two phases: first the the single view detector is ran on the image and the best detection for each object is selected. The second part of the algorithm begins by predicting the 3D pose at the current frame, based on the 2D detection and a constant velocity motion model. The 2D detection bounding box is converted to a 3D pose using a 4 point planar pose recovery algorithm. The 3D tracker is then used to refine the 3D pose, for each object in the image. If the detection corresponds to a traffic sign the approximate 3D pose will be its initialisation.

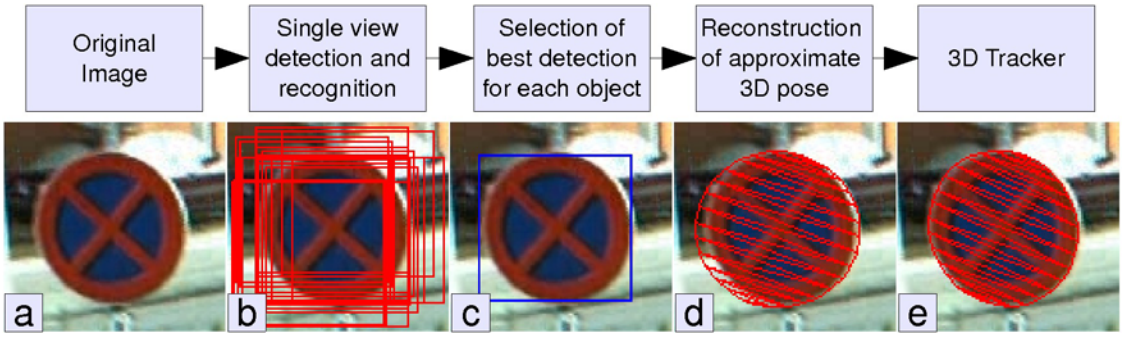


Figure 17: System pipeline

3.2 Single view processing

When a new frame is available the first step is to use the object detection and recognition algorithm presented in [1]. The single view phase consists of the following steps:

1) Candidate extraction - very fast preprocessing step, where the optimal combination of simple (i.e. computationally cheap), adjustable extraction methods select bounding boxes with possible traffic signs. This step requires an automatic offline learning stage, where an optimal subset of those extraction methods is learnt to yield very few false negatives, while keeping the number of false positives in check.

2) Detection - Extracted candidates are verified further by a binary classifier which filters out remaining background regions. It is based on the Viola and Jones [4] Discrete AdaBoost classifier. Detection is performed by cascades of AdaBoost classifiers, followed by an SVM operating on normalised RGB channels, pyramids of HOGs [5] and AdaBoost-selected Haar-like features.

3) Recognition - An hierarchy of SVM classifiers splits the detections in six basic traffic sign subclasses (triangle-up, triangle-down, circle-blue, circle-red, rectangle and diamond) and then another hierarchy of SVM classifiers (for each subclass) assigns the traffic sign type for the different candidate detections.

At this stage several detections might be available for each object, as seen in Fig. 17b. In [6] non-maximum suppression is accomplished via mean shift weighted by SVM score, for a variety of starting points. Instead, we perform non-maximum suppression based on the colour segmentation score, see [2].

3.3 3D tracking

The core of our 3D pose tracking is the PWP3D algorithm [2]. It assumes a known 3D model, a calibrated camera and known foreground/background region statistics. A level set embedding function is built from the contour of the projection of the model and the posterior per-pixel foreground/background probability is maximised as a function of pose.

The PWP3D tracker needs an initial 3D pose and values for the foreground/background membership probabilities. Also, at each new frame, the 2D detections need to be converted to (approximate) 3D poses to be combined with the 3D tracker. In our case the objects can be approximated with their planar counterparts, which means we can use any one of several planar pose recovery algorithms currently available to convert the 2D bounding box to a 3D pose. We use a current state-of-the-art algorithm introduced in [7]. This algorithm requires (at least) 4 3D-2D point correspondences. We use the 4 corners of the detection bounding box as the 2D points and relate them to the 4 corners of the bounding box enclosing the 3D model of object. An example result is depicted in Fig. 17d.

At each new frame, for all previously known objects, the new 3D poses (obtained from the 2D bounding boxes) must be integrated with the tracker. To do this we define a constant velocity motion model (see [2] for more information) and the predicted pose is refined using the tracker. The tracker could have been used alone to obtain the pose changes from consecutive frames but this would have led to longer processing times and would have increased the chance of loosing tracking.

We could have used 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. By iterating the tracker rather than doing a purely statistical data fusion we make no pretence on the type of these probability distribution.

3.4 Results

We tested our algorithm with a multitude of traffic signs types and shapes (the annotated data used also in [1] and [2]). In Figure 18 (top) we show our system tracking a pedestrian crossing sign and obtaining a reasonably accurate pose even when the object is far from the camera. It would also be possible to calculate the distance between the sign and the car, which could then be used to automatically adjust the speed of the car.



Figure 18: Filmstrip showing 5 frames from a video tracking a pedestrian crossing sign (top) and tracking multiple traffic signs (bottom)

In Figure 18 (bottom) we show our system tracking multiple objects. At present the pose for each sign is estimated independently, though of course relative to the camera each sign undergoes the same rigid transformation. We might expect improved performance when we introduce this coupling.

In Figure 19 we compare the performance of our system, tracking a single sign over a distance of 70m (or 70 frames at 36 km/h), with and without the 3D tracker. The car was moving in a straight line, with constant speed. Translation should therefore change linearly, while rotation should remain constant. Though there is little choice between using the 3D tracker and using just the 4 point planar pose recovery algorithm with regard to translation the tracker must be used to reliably obtain 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 type, orientations and colours, with or without motion blur and occlusions. Our CPU implementation of the detection/recognition phase runs at around 20fps on 640x480 images while GPU based tracker needs up to 20ms per object (on a 640x480 image).

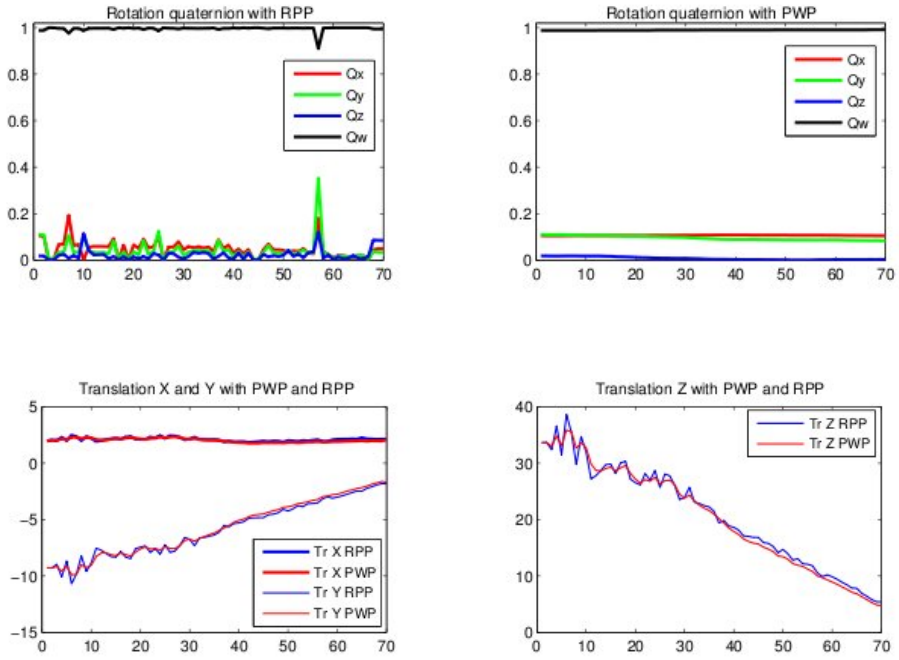


Figure 19: System performance while tracking a sign over 70m, with just the 4 point pose recovery (PPP) and with the tracker (PWP)

References

[1] R.Timofte, K.Zimmermann, L.Van Gool, Multi-view traffic sign detection, recognition, and 3D localisation, WACV,2009

[2] V.A.Prisacariu, R.Timofte, K.Zimmermann, I.Reid, L.Van Gool, Integrating object detection with 3D tracking towards a better driver assistance system, ICPR,2010

[3] N.Cornelis, B.Leibe, K.Cornelis, L.Van Gool, 3D urban scene modeling integrating recognition and reconstruction. IJCV, 78(2-3):121-141,2008

[4] P.Viola, M.Jones, Robust real-time face detection. In IJCV, volume 2, pages 747-757, 2001

[5] A.Bosch, A.Zisserman, X.Munoz, Representing shape with a spatial pyramid kernel, 6th ACM International Conference on Image and Video Retrieval, pp 401-408, NY, USA, 2007

[6] N.Dalal, B.Triggs, Histograms of orientated gradients for human detection, CVPR, 2005

[7] G.Schweighofer, A.Pinz, Robust pose estimation from a planar target, PAMI, 28:2024-2030, 2006

[8] D.Comaniciu, P.Meer, Mean shift: A robust approach toward feature space analysis, PAMI, 2002

[9] T.Ojala et al, Multiresolution gray-scale and rotation invariant texture classification with Local Binary Patterns, PAMI, 2002

[10] G.Loy, A. Zelinsky, Fast Radial Symmetry for detecting points of interest, PAMI, 2003

[11] B.Fulkerson, A. Vedaldi, S.Soatto, Class segmentation and object localization with superpixel neighborhoods, ICCV, 2009