2016 Fourth International Conference on 3D Vision

Fast Obstacle Detection using Sparse Edge-based Disparity Maps

Dexmont Peña [1], Alistair Sutherland [2]
Dublin City University
Glasnevin, Dublin 9, Dublin, Ireland
1: dexmont.penacarrillo2@mail.dcu.ie, 2: alistair.sutherland@dcu.ie

Abstract

This paper presents a fast approach for computing image stixels from a sparse edge-based disparity map. The use of edge-based disparity maps speeds up the computation of the stixels as only a few pixels must be processed compared to approaches which use dense disparity maps. The proposed approach produces as output the stixels in one of the views of the stereo-pair and a segmentation of the edge-points into obstacle. Additionally the proposed approach allows the identification of partially occluded objects by allowing more than one stixel per image column. The proposed approach is fast to compute with no loss on accuracy.

1. Introduction

Fast and accurate obstacle detection is a crucial task in mobile robots. For the case of autonomous cars, the immediate detection of obstacles could reduce the risk of fatal crashes or at least alleviate the effects of the impact. This task becomes even more challenging as the resources available on a mobile robot are limited.

A significant amount of research has been motivated by this challenging set-up. Badino, Franke and Pfeiffer [2] proposed the ”stixel world”, a medium level representation of the free and occupied space in front of a mobile robot. This representation has the advantages of being compact, complete, stable and robust, making it suitable for use with the limited resources of a mobile robot.

In this paper, we propose a new approach for estimating the stixel world by using sparse edge-based disparity maps as they are fast to compute and require fewer resources. These characteristics are desired in an embedded environment. Figure 1 shows an example of the input (edge-based disparities) and the output (stixels). Figure 2 shows the pipeline of our proposed approach. In addition our approach is able to preserve the semantic information from the scene and group the stixels into individual obstacles.

The paper is structured as follows: Section 2 introduces the current approaches for estimating the stixel world. Section 3 describes the method used for detecting and segmenting the obstacles. Section 4 presents the approach used for ”drawing” the stixels from the segmented edge-based disparity maps. Finally, Sections 5 and 6 evaluate the performance of our approach in the context of autonomous cars and conclude the paper.

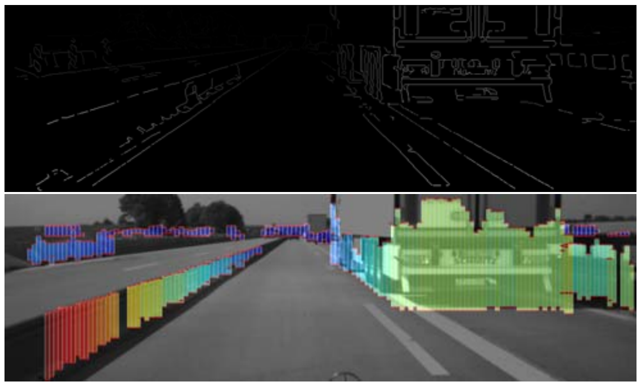
Figure 1. Sample edge-based disparity map used for input (top) and the produced stixels superimposed on one of the camera views (bottom). The stixels are colour coded according to disparity from high (red) to low (blue).

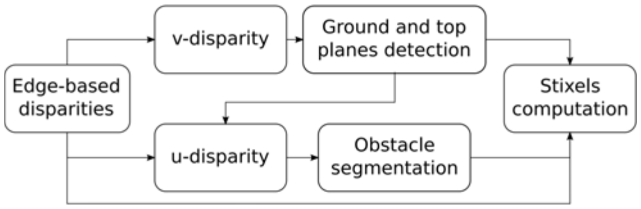

Figure 2. Pipeline of the proposed approach for computing image stixels using sparse edge-based disparities.

2. Related Work

Stixels have proven to be an efficient way to identify free and occupied space in front of a mobile ground robot [2]. A stixel is a vertical rectangle on a 2D image with an associated disparity. They are represented by a 5-tuple $s_i = (u, v, w, h, d)$ where u and v are the coordinates of the top left corner of the rectangle, h and w are the height and width and d is the disparity associated to stixel s_i.

Although the term stixel was first used by Badino, Franke and Pfeifer [2], the concept of representing the occupied space by a vertical structure was previously used by Kubota, Nakano and Okamoto[12]. They modify the Disparity Space Image (DSI) to represent the matching score of a column of pixels and used the v-disparity image [13] to estimate the ground plane. Then they solve a dynamic programming (DP) problem to identify the best object-road boundary using a warping from the right to the left image and place vertical structures at the identified boundaries. Although effective, this approach requires the computation and storage of the DSI which is computationally intensive and it assumes only one vertical structure is present per image column which might not be the case for partially occluded objects.

Badino, Franke and Pfeifer [2] proposed the computation of the stixels by using occupancy grids computed from dense disparity maps. This approach additionally requires DP for estimating the height of the objects. Although this approach succeeded in representing the free and occupied space, the computation of a dense disparity map is computationally costly requiring the use of FPGAs to reach real time performance.

In order to speed up the computation of the stixels, Benenson, Timofe and Van Gool proposed an approach computing the stixels without a disparity map and used them to detect pedestrians [5]. They proposed a modified version of the v-disparity image obtained by projecting the cost volume horizontally. Then they projected the cost volume vertically and solved a DP problem to identify the road-obstacle boundary. After this they solved another DP problem to estimate the stixels height. The authors report a frame-rate of 25Hz using SIMD instructions and multiple cores on an average desktop machine. Although faster than other approaches, the SIMD architecture might not be available on an embedded system therefore reducing the frame-rate. Another disadvantage of this approach is that all the depth information is lost, therefore a separate stage would be required in order to estimate the distance of the obstacles to the camera.

Benenson, Mathias Timofte et al. proposed an approach for computing stixels and detecting pedestrians at 100Hz by using a combination of high-end CPU+GPU [4]. They modified the approach from [5] to use only 80 disparities and an obstacle height of 1.70m on 640×480 images. in a latter work, the authors showed that by modifying the parametrization of the v-disparity image and the road-obstacle boundaries they are able to obtain a frame-rate of 200Hz [3]. Although fast, the processing power requirements are hardly met by embedded systems where limited power is available. As in [5], no depth information is available, requiring an extra stage to estimate the distance from the camera to the obstacles.

Pfeiffer, Gehrig and Scheider proposed an approach for reducing the rate of false-positive stixels by using stereo confidences [15]. They used dense disparity images for the computation of the stixels and used the match confidences as outlier probabilities. Although accurate, the authors do not report any values for performance. However, the computation of dense disparity maps is known to be an intensive task.

Won, Son and Jung proposed an approach to combine the estimation of stixels with the stereo-matching process [17]. They use a plane fitting approach during the cost aggregation stage to compute the stixels. Although this method was shown to reduce the rate of false-positive stixels, no timing information was given. The use of dense disparity maps suggests that this approach is not embedded systems friendly.

3. Obstacle Segmentation

Our proposed obstacle segmentation approach takes as input edge-based disparities such as the ones shown in Figure 1. Sparse edge-based disparities are fast to compute as only a few image pixels are processed while maintaining a compact representation able to retain the image semantics [8].

The first step in the obstacle segmentation is the identification of the ground plane. This is performed by accumulating the edge-based disparities horizontally to obtain the v-disparity image as in [13]. Then the road profile is extracted using the Hough transform. In this paper the road is assumed to be flat but non-flat geometries could be processed with minimum modifications. The road profile is represented by a line with slope m and offset b on the v-disparity image as shown by a blue line in Figure 3. As only the edge-disparities are used, the required processing is reduced in comparison to using dense disparity maps.

As only the obstacles which may be in the way of our mobile robot are of interest, any point located above a top plane with height h_{max} meters is ignored. The computed camera pitch and height estimated by fitting a line to the ground profile are used to project the top plane on the v-disparity image as in [11]. Figure 3 shows an example of the detected ground plane (blue) and top plane (green) on the v-disparity image.

Only those points which lie between the ground plane and the top plane are used for the computation of the u-disparity image. The disparities of these points are accumulated vertically as in [9] although our approach uses only edge-based disparities. After the u-disparity image has been computed, the labelling approach from [10] is used to group the edge-disparities into clusters where each cluster represents an obstacle. This labelling approach is similar to con-

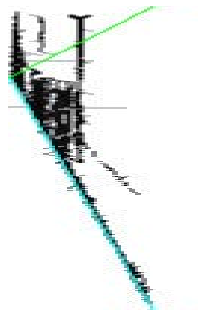
Figure 3. Sample v-disparity image computed by using the edge-based disparities. The ground plane (cyan) is identified by fitting a line using the Hough Transform. The top plane (green) is computed by using the detected ground plane and the available camera information. The shown image has been stretched for visualization purposes.

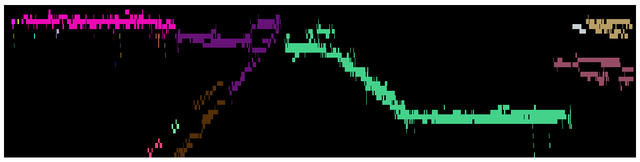
Figure 4. Sample u-disparity image with obstacle segmentation computed by using the edge-disparities lying between the ground and top planes on the v-disparity image. Each obstacle is represented by a different colour.

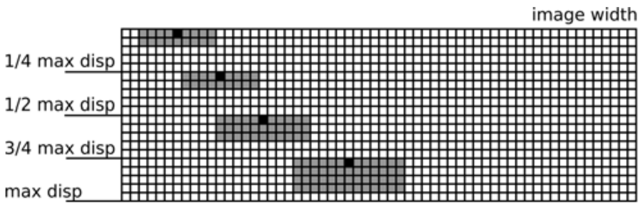

Figure 5. Custom neighbourhoods used in the connected components for grouping the obstacles in the u-disparity image. The neighbourhood is selected according to the location of the pixel being tested. Grey squares represent the pixels taken into account for the neighbourhood. The black squares represent the current pixel being tested and the white rectangles are the pixels in the u-disparity image.

nected components but custom neighbourhoods are used to determine the connectivity of a pixel. Figure 4 shows an example of the u-disparity with the resulting obstacle segmentation. Figure 5 shows the neighbourhoods used in the labelling approach.

After the u-disparity image has been computed, connected components are used with custom neighbourhoods to group the edge-disparities in clusters where each cluster represent an obstacle as it is done in [10]. Figure 5 shows the custom neighbourhoods used for the connected components. Figure 4 shows an example of the u-disparity with the resulting obstacle segmentation.

4. Stixel Computation

Once the obstacles have been segmented, the stixels are drawn on one of the 2D views from the stereo pair. Only obstacles with at least l_{min} edge-disparities are taken into account. Each segmented obstacle is processed independently by applying the following steps:

1. Divide the left view of the stereo-pair into n column-bands b_i of width w. w is the width of the stixels.
2. Identify the column-bands b_i where at least one edge-point of the obstacle lies.
3. Compute the average disparity d_i for each column-band b_i.
4. For every image column in each column-band get the top row (v^t) for the edge-points.
5. Use the ground plane to compute row v^b which corresponds to disparity d_i.
6. Create a stixel s_i for each non-empty column-band taking as disparity d_i. The u_i location of stixel s_i is set as the u location of column band b_i. The v_i location of stixel s_i is taken as the average of the top rows for the column band b_i, $v_i = \bar{v^t}$. The height h_i of the stixel s_i is computed as $h_i = v_i - v^b$.

Figure 6 shows an example of the computation of one stixel by processing the segmented edge-based disparities. The identification of stixels per obstacle allows the existence of more than one stixel per image band. This allows the representation of partially occluded obstacles in the 2D view. As only counting and averaging operations are performed for the identification of stixels the number of required computations is low resulting in a fast to compute approach. Figure 1 shows an example of the identified stixels.

5. Evaluation

In order to evaluate the proposed approach to compute the stixel world the 6DVision dataset proposed by Pfeiffer, Gehrig and Schneider [15] and the AEss-Pedestrians dataset proposed by Ess, Leibe Schindler et al. [7] are used. The 6DVision dataset provides 2988 grey-scale stereo-pairs with manually annotated ground truth for images taken in good and bad weather conditions which are common in autonomous car environments. The banhof sequence from the AEss-Pedestrians dataset provides manually anotated bounding boxes for pedestrians for 1000 stereo-pairs.

In order to obtain edge-disparities two approaches are used. The first approach is a sparsification, where a dense disparity map is obtained by the Semi-Global Block Matching available in OpenCV (OpenCV-SGBM) [6]. Then the

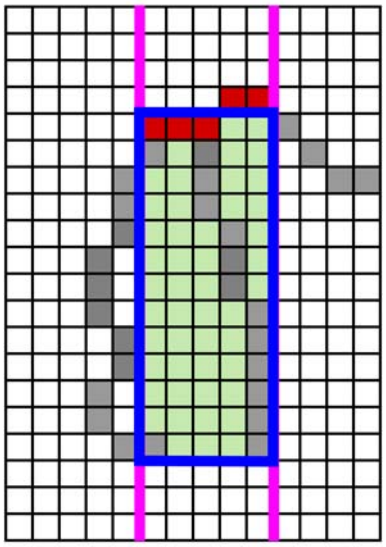
Figure 6. Pixels used for computing the stixel parameters on one of the 2D views of the stereo-pair. Pink lines represent the boundaries of the column-band being computed. Grey pixels correspond to image edges, the disparity d_i of the stixel s_i is computed by averaging the disparity of all the edge pixels in the column band. Red pixels are the top edge locations used to compute the top boundary v_t of the stixel. The bottom of the stixel is computed by using the v location on the v-disparity image corresponding to the stixel disparity d_i. The blue rectangle represents the stixel boundaries.

edges in the input are detected by Edge Drawing [1]. The obtained disparity map is dilated by a 3×3 structuring element to keep the larger disparities as in [16], only disparities at edge locations are kept. The second approach for obtaining edge-disparities is Simultaneous Edge Drawing (SED) [14] which produces fast edge-based disparities.

The Simultaneous Edge Drawing algorithm takes as input stereo-images and compute edge-disparities by matching few anchor points and propagating the disparity along the image edges. The anchor points are pixels with a high gradient across a Gaussian Scale Space. After these anchor points have been identified they are matched by using an area based approach. The matched anchors are then used as seed for a simultaneous smart routing which moves one pixel at the time across the stereo pair to propagate the disparities and produce chains of ordered 3D points corresponding to the edges in the stereo-images. The anchor point matching and the simultaneous smart routing are performed one at the time, *i.e.* anchors are matched only after the simultaneous smart routing for the previous anchor has been finished. By taking this iterative approach this algorithm is able to reduce the number of matched anchors as most of them obtain a disparity by the simultaneous smart routing.

Stage	Time (ms)	
	SED	OpenCV-SGBM
Disparity extraction	204.85	887.37
Obstacle segmentation	4.70	8.37
Stixels computation	0.17	0.59

Table 1. Average running times for each of the stages of our obstacle detection approach.

5.1. 6DVision Dataset

After the edge-disparities are computed, they are fed into our approach for computing stixels. As the labelled obstacle areas in the ground truth cover only a limited region in the image, a stixel is labelled as false positive if its disparity is larger than the disparity of the ground truth stixel located at the same column. Unfortunately during experimentation we found that the stixel disparities provided on the ground truth have a large error when they are close to the camera (large disparity) although their position in the image is right. These erroneous stixels were found to have a disparity smaller than what it should be according to the input images. In order to avoid using these erroneous disparities we only take into account ground truth stixels with disparity smaller than 32 (this value was found experimentally) and set an error threshold $t_d = 10$. The detection rate takes into account only the stixels provided in the ground truth although our approach produces many more detections. Additionally, in order to use the same amount of images (2490) as in [15] we discarded some sequences from the dataset randomly.

The parameters used for our approach are kept constant through all the experiments. The minimum height of an obstacle is set to $t_h = 0.2$ m. The height of the top plane is set to $h_{max} = 2.5$ m. The minimum number of edge-disparities on an obstacle to be taken into account is set to $l_{min} = 15$. These parameters were found experimentally to provide a good performance in our approach. The stixel width is set according to the dataset as $w = 7$ such as the camera parameters. The parameters for SED are taken from [14]. The parameters for OpenCV-SGBM are left as their default values.

An average laptop with an Intel i7-2675 @ 2.2GHz with 8GB RAM was used for testing. No parallel processing or SIMD intrinsics are used in our implementation. Table 1 shows the average running times for each of the stages of our approach.

Table 2 compares our evaluation with the results from [15]. It can be seen that the number of false positives in our approach is larger than that in [15] but the detection rate is almost the same. The increased number of false positives on our approach are created by wrong disparities or errors on the fitting of the ground plane, both could be avoided by the use of tracking through the sequence. In contrast with the

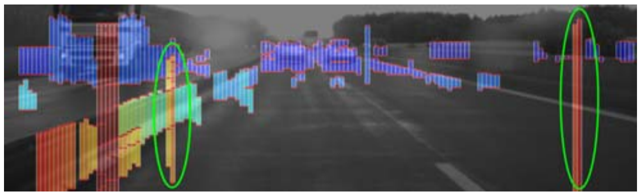
Figure 7. Example of the false positive stixels. The stixels in the green circle are produced by vertical line with a wrong disparity. This kind of artefacts can be identified by tracking.

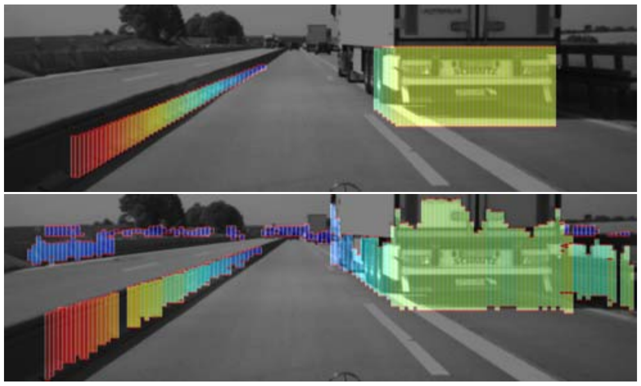
Figure 8. As it can be seen, the ground truth (top) provided in the dataset [15] labels only a small region in the source image. In contrast our approach (bottom) provides stixels for the whole image.

Method	#fp	#fwfp	det. rate
Confident Stixels [15]	107	45	0.802
Edge-Stixels-SED	375	1432	0.802
Edge-Stixels-SGBM	1487	17401	0.89

Table 2. Results of evaluating our approach on the dataset used provided in [15]. #fp is the number of false-positives, #fwfp is the number of frames with false positives, det. rate is the detection rate. The results for Confident Stixels are taken from [15].

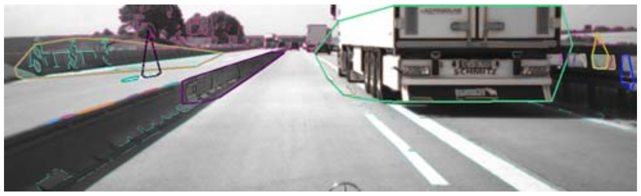
Figure 9. Obstacle segmentation produced by our approach as an intermediate step.

approach in [15] our approach does not require the use of FPGA which would increase the complexity of an embedded system. Additionally our approach provides a segmentation of the detected obstacles as it can be seen in Figure 9. Figure 8 shows a comparison of the stixels provided by the ground truth and the stixels detected by our approach.

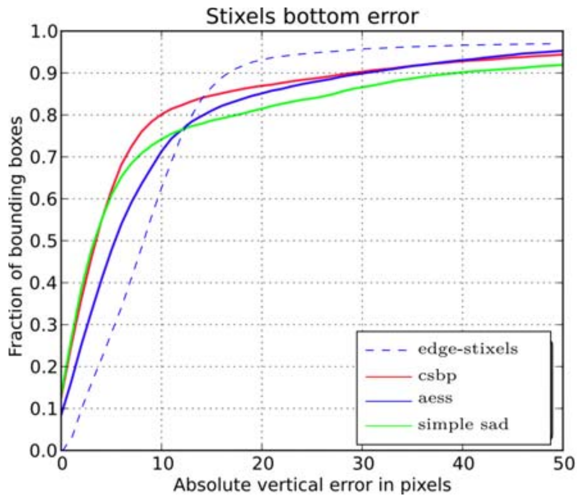

Figure 10. Recall for detection of bounding boxes using different error thresholds for the bottom boundary of the bounding boxes. Figure taken from [5] with our results (edge-stixels) overlapped. The recall of our Edge-Stixels goes higher at a smaller error threshold.

5.2. AEss-pedestrians

Due to the limitations on the 6DVision dataset, our approach is evaluated on a pedestrian dataset as in [5]. As the AEss-Pedestrians dataset [7] does not provide stixel information and instead it provides bounding boxes for pedestrians identified manually, the following approach is used: obstacles are marked as detected if the bottom of a stixel is found to coincide with the bottom of a ground-truth bounding box measured at the middle.

The parameters used for our approach are kept constant through all the experiments. The minimum height of an obstacle is set to $t_h = 0.1$ m as this value was found to reduce inaccuracies on the ground plane detection. The height of the top plane is set to $h_{max} = 1.8$ m as in [5]. The minimum number of edge-disparities on an obstacle to be taken into account is set to $l_{min} = 20$ as it provided a good accuracy without sacrificing recall. The stixel width is set as $w = 7$. The parameters for SED are taken from [14].

Figure 10 shows a comparison of the recall of our approach and the results from [5]. It can be seen that our approach is able to obtain a higher recall rate at a lower error level. This is a desired quality as it means that our stixels are closer to the ground truth in comparison with the approach from [5].

Figure 11 shows an example of wrong stixels detected by the approach in [5] but properly detected by our approach. As the approach in [5] does not support multiple obstacles on the same column, the detection of zero height stixels results in non-detected obstacles. The Figure shows that by applying a minimum height to the obstacles these zero-height obstacles are avoided. Even if present, our ap-

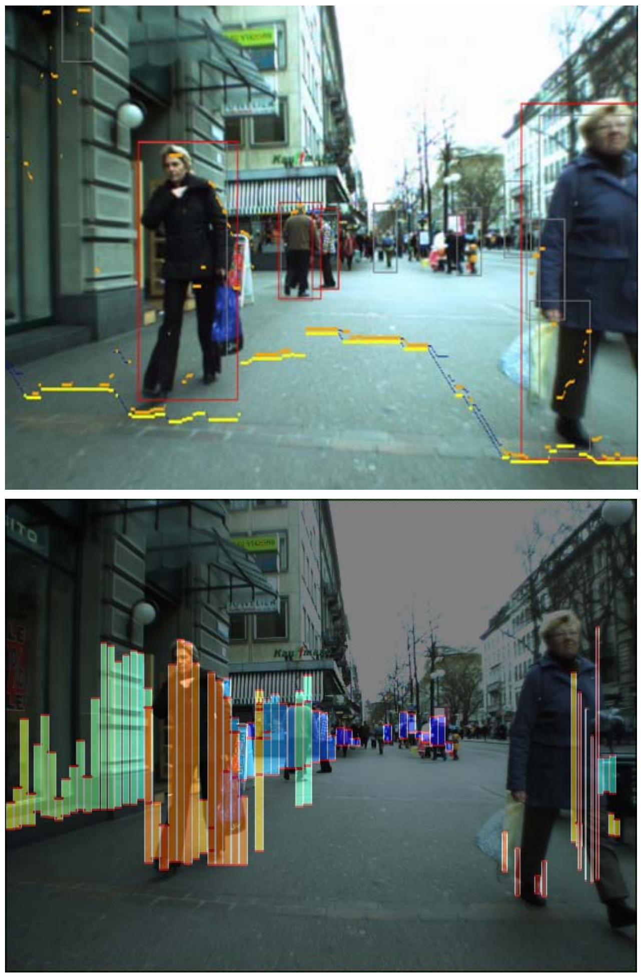
Figure 11. Erroneous stixels detected by [5] are correctly identified by our approach. The top image is taken from [5], it shows erroneous stixels with zero-height. The bottom image shows that no zero-height stixels are found in our approach.

proach is able to detect more than one obstacle per column therefore true obstacles are still recovered.

6. Conclusions

In this paper we presented a new fast approach for identifying stixels in outdoor scenarios by using edge-based disparities obtaining a recall comparable with state-of-the-art approaches. Our proposed approach has low computational requirements making it compatible with embedded systems, while additionally providing obstacle segmentation and geometry information by the use of edges. As we found that the disparities provided in the ground truth are not accurate further research aims to the creation of accurate ground truth. Additionally, we aim to implement it on an embedded system such as the Raspberry Pi which lacks the performance from a laptop machine and using tracking to have an estimate of the possible trajectories of the obstacles.

Acknowledgements

This research has been funded by the Irish Research Council under the EMBARK initiative, application No RS/2012/489.

References

[1] C. Akinlar and C. Topal. Edpf: a Real-Time Parameter-Free Edge Segment Detector With a False Detection Control. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(01):1255002, feb 2012. 4

[2] H. Badino, U. Franke, and D. Pfeiffer. The Stixel World - A Compact Medium Level Representation of the 3D-World. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5748 LNCS, pages 51–60. 2009. 1, 2

[3] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Fast Stixel Computation for Fast Pedestrian Detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7585 LNCS, pages 11–20. 2012. 2

[4] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Pedestrian detection at 100 frames per second. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2903–2910. IEEE, jun 2012. 2

[5] R. Benenson, R. Timofte, and L. Van Gool. Stixels estimation without depth map computation. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2010–2017. IEEE, nov 2011. 2, 5, 6

[6] G. Bradski. The OpenCV Library. *Dr Dobbs Journal of Software Tools*, 25:120–125, 2000. 3

[7] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008. 3, 5

[8] R. Fabbri and B. Kimia. 3D curve sketch: Flexible curve-based stereo reconstruction and calibration. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1538–1545. IEEE, jun 2010. 2

[9] Z. Hu and K. Uchimura. U-V-disparity: an efficient algorithm for stereovision based scene analysis. In *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, volume 2005, pages 48–54. IEEE, 2005. 2

[10] A. Iloie, I. Giosan, and S. Nedevschi. UV disparity based obstacle detection and pedestrian classification in urban traffic scenarios. In *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 119–125. IEEE, sep 2014. 2, 3

[11] S. Kramm and A. Bensrhair. Obstacle detection using sparse stereovision and clustering techniques. In *2012 IEEE Intelligent Vehicles Symposium*, pages 760–765. IEEE, jun 2012. 2

[12] S. Kubota, T. Nakano, and Y. Okamoto. A Global Optimization Algorithm for Real-Time On-Board Stereo Obstacle Detection Systems. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 7–12. IEEE, jun 2007. 2

[13] R. Labayrade, D. Aubert, and J.-P. Tarel. Real time obstacle detection in stereovision on non flat road geometry through ”v-disparity” representation. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 646–651. IEEE, 2002. 2

[14] D. Pena and A. Sutherland. Disparity Estimation by Simultaneous Edge Drawing. In *ACCV Workshop on 3D modelling and applications*, 2016. 4, 5

[15] D. Pfeiffer, S. Gehrig, and N. Schneider. Exploiting the Power of Stereo Confidences. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 297–304. IEEE, jun 2013. 2, 3, 4, 5

[16] J. Witt and U. Weltin. Robust Real-Time Stereo Edge Matching by Confidence-Based Refinement. In C.-Y. Su, S. Rakheja, and H. Liu, editors, *Intelligent Robotics and Applications*, volume 7508 of *Lecture Notes in Computer Science*, pages 512–522. Springer Berlin Heidelberg, 2012. 4

[17] K. H. Won, J. Son, and S. K. Jung. Stixels estimation through stereo matching of road scenes. In *Proceedings of the 2014 Conference on Research in Adaptive and Convergent Systems - RACS ’14*, pages 116–120, New York, New York, USA, 2014. ACM Press. 2