Stixels estimation without depth map computation

Rodrigo Benenson, Radu Timofte and Luc Van Gool
ESAT-PSI-VISICS/IBBT, Katholieke Universiteit Leuven, Belgium
firstname.lastname@esat.kuleuven.be

Abstract

Mobile robots require object detection and classification for safe and smooth navigation. Stereo vision improves such detection by doubling the views of the scene and by giving indirect access to depth information. This depth information can also be used to reduce the set of candidate detection windows. Up to now, most algorithms compute a depth map to discard unpromising detection windows. We propose a novel approach where a stixel world model is computed directly from the stereo images, without computing an intermediate depth map. We experimentally demonstrate that such approach can considerably reduce the set of candidate detection windows at a fraction of the computation cost of previous approaches.

1. Introduction

Mobile robotics in urban environments has shown to be a challenging field of research. Arguably one of the corner stones for mobile robots is the perception problem. For navigation purposes the robot is particularly concerned with where objects may move in the future. The robot will often observe mobile objects that are not moving, e.g. pedestrians and cars waiting at a red light. These objects can only be detected by appearance, their proper detection is necessary to ensure the correctness of the world model and thus safety and proper navigation.

For navigation in dynamic environments, appearance based object detection and classification is a must.

Faster detection using stereo images The traditional use of stereo image pairs for object detection is to build a dense depth map [6, 3, 9]. This depth map is used as an additional feature for detection (improving quality) or as a method to reduce the search space (improving speed). The depth information allows the robot to rapidly rule out unlikely areas for the presence of objects, such as ground, sky or areas of the image where the depth is very irregular (vegetation). By reducing the set of candidate detection windows, the appearance based detection is accelerated. In this paper we achieve the same goal without computing the depth map image.

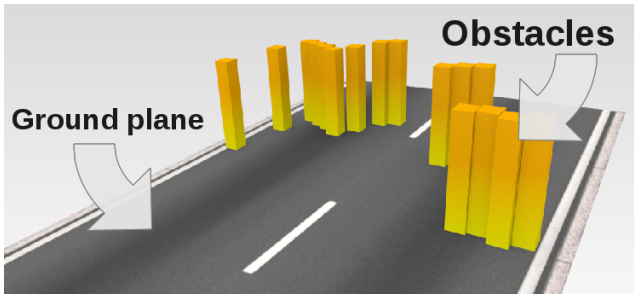


Figure 1: The stixel world is composed of the ground plane and vertical sticks describing the obstacles.

Stixel world Badino et al. introduced the notion of *stixel world* [2]. The stixel world is a particular parametrization of the world model that makes strong simplifying assumptions. It assumes that the ground is locally flat and that all objects can be described as flat "sticks" raising vertically above the ground (see figure 1). Each of these vertical sticks corresponds to a segment of a column in the image, called a *stixel*.

Obviously, the real world is not a stixel world. The stixel world may not adequately model all objects of a scene, but as long as the ground is approximately flat, it will properly model objects of interest such as pedestrians, cars, and bicycles [17, 18].

Assumptions In this paper we assume that the mobile platform acquires a stereo video stream with cameras that are parallel to the horizon, that the ground can be approximated as locally planar and that the objects of interest have a height in a known limited range (e.g. 0.5 to 3 m). We also assume that the stereo system is calibrated and that we have a rough estimate of its initial position with respect to the ground.

1.1. Contribution

To the best of our knowledge, previous works building stixel world representations are all based on an initial dense depth map estimation. This depth map contains much more information than the stixel world representation, but is consequently slower to compute and risks to be a costly detour.

Independent of future hardware improvements, algorithmic speed improvements are always welcome because they leave more computational resources for the higher level tasks of the robot. There is constant pressure to speed up the object detection module.

In this paper we show that it is possible to compute a stixel world estimate *directly* without computing an intermediary depth map. By skipping this step, we obtain a stixel world estimate faster. Having a faster stixel world estimate, in turn, allows us to quickly discard large image portions, lowering the latency of object detection (see figure 2).

The core idea lies on the realization that stereo matching is inherently ambiguous. Image areas with low horizontal gradient will yield ambiguous matches, which translate in ambiguous depth estimates. All stereo methods providing a dense depth map will either use smoothing constraints or prior knowledge. In our case, the stixel world itself defines the prior knowledge and smoothing constraints. As such, it should be possible to directly estimate this reduced representation, without spending time on a more complex model based on different constraints and assumptions.

1.2. Related work

While the term "stixel" appeared recently, the idea of using very simplified world models has been visited many times. Fifteen years ago, stereo vision was already used to estimate ground parameters and to detect the presence of obstacles on the ground [8]. Such systems then evolved to allow for free space estimation [16] and, ten years ago, to have rough object detection [7]. These initial approaches were further improved to handle non-flat ground [14] and non-frontal objects [15, 11, 19].

More recently, the idea of computing free space using dynamic programming was introduced [1, 13] and extended to estimate both the free space and the height of the frontmost obstacles [2].

With the exception of [13], all methods mentioned use (dense or sparse) depth map computation as an initial step. In a sense our work can be seen as an extension of [13], by adding the object height estimation and by removing the need for strong oriented gradients; or as an optimization of [2], where we remove the need for a full depth map to estimate the stixel world.

The general approach of using dynamic programming over column-wise matching cost also relates to the work of Cornelis et al. [4], where it was used to estimate the buildings of the street, instead of the objects in the street.

2. Stixel estimation

The proposed stixel estimation method has four steps. From the input rectified stereo images a pixel-wise cost volume is computed (§2.1). This cost volume is used to estimate the ground plane (§2.2), which in turn is used to estimate the stixel disparities (§2.3). Finally, the stixel disparity estimates are used to also estimate the stixel heights (§2.4).

For simplicity's sake, we describe the method using stixels of one pixel width. A stixel is estimated for each image column u. Using broader stixels would reduce the overall computational cost by averaging columns of data immediately after the cost volume computation, but will also reduce the spatial resolution of the stixels. In this paper we do not explore this quality versus speed trade-off.

2.1. Cost volume computation

Given a pair of rectified stereo images, we compute a matching cost volume: for every pixel in the left image and for every disparity value, we compute the cost as the vanilla sum of absolute differences over the RGB colour channels (using a window of size 1 pixel). Kubota et al. [13] used orientation based matching to improve robustness, assuming that the images contain strong directional gradients. This seems too restrictive as it precludes handling textures with spots instead of lines (see the ground in figure 5).

The computed values are stored in memory as $c_m(u, v, d)$ ("matching cost") where u, v, d indicate the horizontal axis, vertical axis and disparity respectively. Since the operation is done pixel-wise, it is embarrassingly parallel and thus very fast to compute. Most (if not all) stereo matching methods include an equivalent step with similar or higher cost, which is usually followed by a smoothing process (which dominates the computation time) [10].

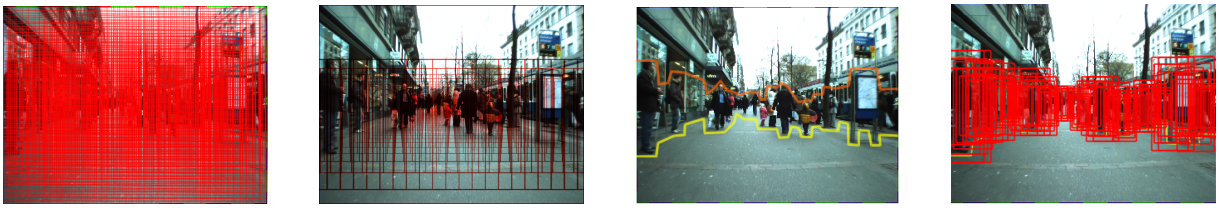
2.2. Ground plane estimation

The ground plane is estimated using the v-disparity method [14], but without computing a depth map. We directly project the cost volume along the horizontal axis (u axis) to create a "v-disparity" image. Each pixel in this image contains the summed cost of every pixel along the v-disparity unidimensional slice. The ground plane parameters are then obtained by robustly fitting a line on the v-disparity image.

For stixel world estimation we only need a bijective relation $f_{ground} : U \times V \mapsto D$ where for any pixel position $(u, v) \in U \times V$ we obtain the corresponding ground disparity $d \in D$ (and vice versa). Given our assumptions, this mapping takes the form $f_{ground} : V \mapsto D$. More complex ground models could also be used [12].

2.3. Stixels distance estimation

A projection of the cost volume along the horizontal axis provided the ground estimate. In this section a projection



(a) Dense set of detection windows

(b) Reduced set of windows using a ground plane estimate

(c) Ideal stixel estimations (same color code as figure 5)

(d) Reduced set of candidate windows using stixels

Figure 2: This paper aims at providing a faster use of stereo images to reduce the set of candidate detection windows. The stixel world representation allows to properly estimate the candidate windows for the frontmost objects, as illustrated here.

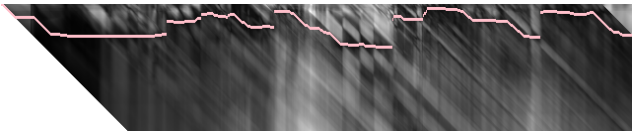


Figure 3: Example c_s with estimated $d_s^*(u)$ in pink.

along the vertical (v axis) will provide an estimate of the stixels depth. Following the approach of Kubota et al. [13], the disparity (and thus the depth) of each stixel is estimated using 2D dynamic programming over a data term c_s and a smoothness term s_s, as explained below.

The goal is to find the optimal disparities for the stixels

$$d_s^*(u) = \underset{d(u)}{\operatorname{argmin}} \sum_u c_s\left(u, d(u)\right) + \sum_{u_a, u_b} s_s\left(d(u_a), d(u_b)\right) \tag{1}$$

where u_a, u_b are neighbours ($|u_a - u_b| = 1$).

2.3.1 Data term

For each u coordinate and possible d disparity value, we calculate the evidence supporting the presence of a stixel in the left image by computing the cost $c_s(u, d)$ ("stixel cost"). The lower the cost the more likely that a stixel is present at position (u, d).

The cost c_s is the result of summing two costs: $c_o(u, d)$ ("object cost"), the cost of a vertical object being present, and $c_g(u, d)$ ("ground cost"), the cost of a supporting ground being present (see figure 3).

$$c_s(u, d) = c_o(u, d) + c_g(u, d) \tag{2}$$

Using $v(d) = f_{ground}^{-1}(d)$ we can map each (u, d) coordinate to a point $(u, v(d))$ in the image plane. Using the camera calibration and the ground estimate we can also estimate the vertical position of a point above the ground for a given height $v(h, d)$. We will use the assumed minimum height of objects $\check{h}_o$ to search for the depth of the stixels.

We can now define c_o and c_g as

$$\begin{aligned} c_o(u, d) &= \sum_{v=v(\check{h}_o, d)}^{v(d)} c_m(u, v, d) \quad , \\ c_g(u, d) &= \sum_{v=v(d)}^{|V|} c_m(u, v, f_{ground}(v)) \end{aligned} \tag{3}$$

where $|V|$ indicates the number of rows in the image and the smallest v is at the top of the image.

2.3.2 Smoothness term

Not all the pixels in the right image are visible in the left image. Some of the objects visible in the right image are occluded in the left image by nearer objects (and vice versa). Since the stixels are estimated from stereo data, it is expected that some of them correspond to occluded areas.

When analysing the left image, this occlusion constraint invalidates any stixel behind the "one disparity less per pixel to the left" line [13]. We construct the smoothness term s_s ("stixel smoothness") accordingly,

$$s_s\left(d_a, d_b\right) = \begin{cases} \infty & \text{if} \quad d_a < d_b - 1 \\ c_o\left(u_a, d_a\right) & \text{if} \quad d_a = d_b - 1 \\ 0 & \text{if} \quad d_a > d_b - 1 \end{cases} \tag{4}$$

where $d_a = d(u_a)$, $d_b = d(u_b)$, and u_a is one pixel to the left of u_b. The case $s_s = \infty$ ensures that no stixel distance estimate will violate the occlusion constraint.

2.3.3 Dynamic programming

Given the data term c_s and the smoothness term s_s, the optimal depth for each stixel, $d_s^*(u)$ ("stixel disparity"), is computed by solving a standard minimizing 2D dynamic program in the u-disparity domain, as in [13].

Using the u-disparity boundary $d_s^*(u)$ (see figure 3) and the ground plane model f_{ground}, we can compute the u-v boundary $v_{bottom}^*(u)$ that represents the bottom of each

stixel, as shown in the figure 5. Stixels with a disparity moving along the line of $s_s = \infty$ are labelled as occluded.

2.4. Stixels height estimation

To estimate the depth of the stixels we used the object minimal height $\check{h}_o$ assumption. To estimate the actual height of each stixel, we will estimate the likelihood that each pixel above the ground belongs to the estimated stixel disparity $d_s^*(u)$. These estimates are provided by the membership function $m(u, v)$.

2.4.1 Membership function

Badino et al. [2] use the input depth map to compute a membership function based on the distance between the pixel-wise disparities and the stixel disparities. We show here how to compute a similar membership function, without estimating pixel-wise disparities.

If a pixel (u, v) in the image belongs to a given disparity d_a we expect the cost function around $c_m(u, v, d_a)$ to be a local minima in the disparity dimension. If the true disparity at (u, v) is far from d_a, then the surroundings of $c_m(u, v, d_a)$ will not resemble a local minima. Our membership function $m(u, v)$ is a local measure of how much $c_m(u, v, d_s^*(u))$ resembles a local minima or not.

This value is fast to compute since we need to visit only a small fraction of the cost volume c_m. All pixels below the estimated u-v boundary $v^*_{bottom}(u)$ can be skipped. All pixels above the maximum expected height of the objects can be skipped as well. All columns corresponding to occluded stixels can be skipped. For the pixels that we do visit, only a few values around $c_m(u, v, d_s^*(u))$ need to be visited.

Our membership function $m(u, v)$ is defined as

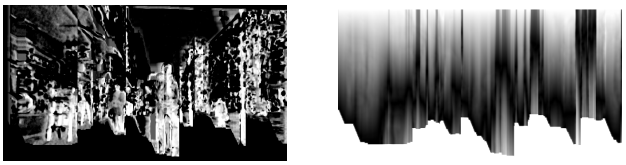
$$m(u, v) = 2 \cdot (\max(0, m_1(u, v)) - 0.5) \tag{5}$$

$$m_1(u, v) = \sum_{d \in N(d_s^*(u))} \frac{m_2\left(\tilde{c}_m(u, v, d), \tilde{c}_m(u, v, d_s^*(u))\right)}{|N(d_s^*(u))|} \tag{6}$$

$$m_2(c, c^*) = \begin{cases} +\max(|c-c^*|, \Delta_{max})/\Delta_{max} & \text{if} \quad c > c^* \\ -\max(|c-c^*|, \Delta_{max})/\Delta_{max} & \text{otherwise} \end{cases} \tag{7}$$

where $N(d_a)$ indicates a small neighbourhood around d_a (e.g. ± 10 pixels), $|N(d_a)|$ indicates the number of elements in $N(d_a)$, Δ_{max} is a small constant ($\Delta_{max} = 10$ in our implementation) and $\tilde{c}_m$ is the cost value after applying a 5×5 mean filter.

As defined in equation 5, $m(u, v) = 1$ means full membership, -1 means no membership, 0 indicates no contribution.



(a) Example of $m(u, v)$ (b) Corresponding c_h

Figure 4: Examples of the cost matrices used to estimate the stixels height (frame 7).

The traditional approach for height estimation would require visiting the complete cost volume $c_m(u, v, d)$ in order to compute a full depth map. On the other hand, our approach requires to visit less than 15% of the cost volume, with a corresponding reduction in the computation cost.

2.4.2 Data term

The membership function $m(u, v)$ is then converted into a height cost $c_h(u, v)$ (see figure 4):

$$\begin{aligned} c_h(u, v) \quad = \quad & \sum_{w=v^*_{bottom}(u)}^{v} |m(u, w) - 1| \\ & + \sum_{w=v}^{v(\hat{h}_o, d_s^*(u))} |m(u, w) + 1| \end{aligned} \tag{8}$$

where $v(\hat{h}_o, d_s^*(u))$ indicates the row of the maximum height considered for an object.

2.4.3 Smoothness term

The height cost $c_h(u, v)$ is the data term for a 2D dynamic programming formulation similar to §2.3.3, but with different data and smoothness terms. The smoothness term is:

$$\begin{aligned} s_h(u_a, v_a, u_b, v_b) \quad = \quad & k_1 \cdot |v_a - v_b| \cdot \\ & \max\left(0, 1 - \tfrac{z(d_s^*(u_a)) - z(d_s^*(u_b))}{\Delta z_2}\right) \end{aligned} \tag{9}$$

where $|u_a - u_b| = 1$, k_1 is a scaling factor that penalizes top shapes that are non horizontals (set to 1 in our experiments), and Δz_2 indicates how much near stixels should influence each other given their depth (set to 3 m in our experiments).

As a post-processing step, if the estimated stixel height is too far from the expected one we consider it erroneous and reset it to the expected height. In our experiments we used an expected object height of 1.8 m and a 20 pixels margin.

3. Results

3.1. Evaluation dataset

It has already been shown that, when using an intermediate dense depth map, the stixel world representation can accurately represent the obstacles present in the scene [18]. Our aim is slightly different. Since we focus on object detection, we evaluate our algorithm not based on metric ground truth but rather on manually annotated bounding boxes around pedestrians. We expect the calculated stixels to be tightly bounded by such boxes. To our knowledge this is the first time that stixel world estimations are evaluated in the context of object detection.

We use the publicly available dataset from Ess et al. [6] to evaluate our method. The "Bahnhof" sequence contains ~ 1000 stereo frames of crowded sidewalks in Zürich. The dataset provides ~ 7400 bounding boxes for all pedestrians of at least 40 pixels in height (up to~ 25 meters away) appearing in each left frame of the sequence . The dataset also provides the internal camera calibration, the stereo rig calibration and a rough estimate of its position with respect to the ground. Each frame contains 640×480 pixels.

In figure 5, we present some qualitative results of success and failure cases.

3.2. Stixels estimation error

For a quantitative evaluation of the stixel estimates we measure the error between the bottom and top of the stixel at the centre of each annotated bounding box. Stixels estimated as occluded are also considered in the evaluation.

With this evaluation we are interested in answering two questions: It is better to estimate the height of the pedestrians than assuming a fixed height? How does our height estimation method compare to using a full depth map?

To answer these questions we estimate the stixels based on three methods: `ours`, `simple sad` and `csbp`.

`ours`: stixels estimation without depth map, as described in section 2.

`simple sad`: a depth map is estimated using winner-take-all, sum of absolute differences over 9×9 pixels. The u-disparity, and v-disparity images are computed using the depth map, and the stixels height is estimated using the approach of Badino et al. [2]. This method serves as a baseline since it resembles the internals of our algorithm, and it is among the simplest (and fastest) stereo algorithms.

`csbp`: a depth map is estimated using fast variant of the hierarchical belief propagation approach [20]. This method is expected to provide better depth maps at an increased computational cost (see figure 6). We use the original implementation provided by the authors.

All parameters are kept fix between the three methods (we experimented adjusting the parameters for the depth maps cases, but results where slightly worse). For each method we also compare estimating the height from the data (`estimated` case), or simply using a fixed stixel height of 1.8 m, based on the expected object height (`fixed` case). The results are presented in figure 7.

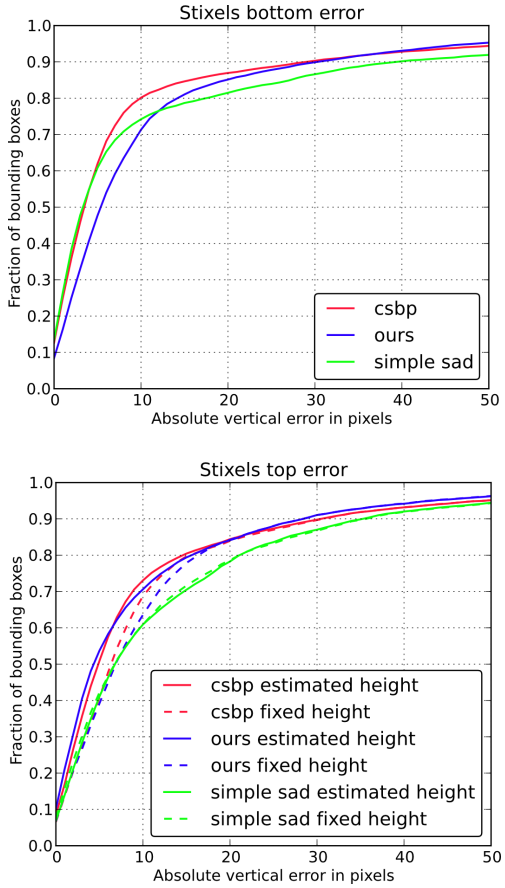


Figure 7: Count of bounding boxes with bottom or top stixel absolute error lower than a given value. Evaluation over 999 frames of the Bahnhof sequence. Stixels are estimated using the proposed method (`ours`), and two full depth map estimation methods (`csbp` and `simple sad`). Plot is limited to the range $(0,\ 50)$ pixels.

For both bottom and top errors, irrespective of the method, the fraction of covered bounding boxes reaches $\sim 90\%$ at around 30 pixels of absolute error. The remaining 10% correspond to cases with high error. These include pedestrians that are highly occluded, or who cannot be properly represented by the stixel world model, such as the one appearing "below the arm" of someone else (violating the non-transparency assumption of the stixels), see figure 5.

Overall the results of figure 7 show that having a better depth estimate improves the stixel estimation, that estimating the height is better than using a fixed value, and that our proposed stixel estimation without depth map provides

(a) Frame 7 (b) (c)

(d) Wrong depth, due to wrong ground plane (e) Wrong height due to ambiguous disparities (f) Wrong height due to reflections

Figure 5: Examples of good results (a,b,c) and failure cases (d,e,f). Red boxes indicate ground truth, the grey boxes the raw results from the detector, and the white boxes the detections validated by the stixels estimate. The lower yellow line and the upper orange line indicate the stixels bottom and top respectively. Dark blue pixels indicate occluded stixels.

(a) Full depth map estimated using `simple sad` (see §3.2) (b) Full depth map estimated using `csbp` (see §3.2) (c) Cost matrix $m(u, v)$ built from the stereo image (see §2.4)

Figure 6: The stixels height can be estimated using a full depth map or directly estimating the cost matrix (i.e. without any depth map computation). All images computed over frame 7.

comparable results to having a good depth map estimate.

For the bottom error `csbp` is better than `ours`, but the difference fades out as the absolute error margin reaches 30 pixels. This is the margin we use for objects detection (see figure 8). For the top error `ours` provides comparable results.

3.3. Pedestrian detection rates

The aim of this paper is to show that stereo information can be used to accelerate object detection, without needing to compute a depth map first.

In figure 8, we present the detection rates obtained by a standard HOG + linear SVM pedestrian detector [5], its raw results, the results filtered using the ground plane estimate, and the results filtered using the estimated stixels. For each detection we take the middle stixel and check the top and bottom margin. We additionally re-weight the detections score based on the distance to the ground plane/stixel estimates.

It can be seen in figure 8 that, in the low false positives

per image area, using detections based on the stixel estimates provides better results than the raw detector, despite using a search space orders of magnitude smaller. The stixel results also compare favourably to using the ground plane constraint only.

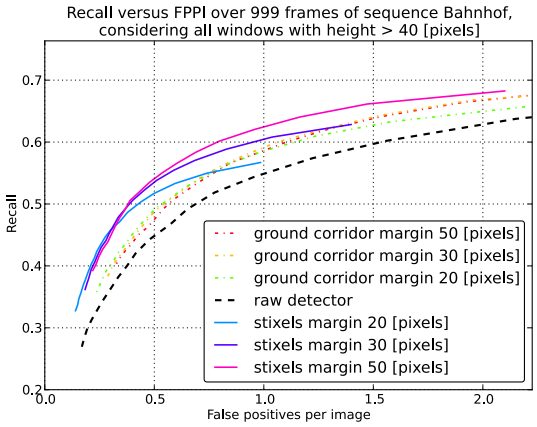


Figure 8: Detection rates obtained from a HOG detector, filtered using stixels (`estimated` case), the ground plane constraints only (`ground corridor`), and the raw detector only.

3.4. Failure cases

In figure 5 we present some representative failure cases. These are rare events, that show the limitations of the approach. Most observed failures fall into two categories:

1) Ground plane estimation failures. Any error in the ground plane estimation will have a direct impact on the quality of the stixel estimates (see figure 5d). In this paper we use a simplistic ground estimation, its robustness can certainly be improved.

2) The pixel matching is uninformative. The proposed method is based on matching costs. When these are ambiguous (black area in figure 5e) or inaccurate (reflections in figure 5f) the height estimation will be inaccurate. We have noticed that the disparity estimation is less sensitive to this effect (which is more critical for the robot navigation than the height). These disturbances could be mitigated by improving the matching cost, and by better exploiting the information available in the image.

3.5. Computational cost

3.5.1 Stixel estimation versus full depth map

From a computational perspective the algorithm described in section 2 requires:

1) Computing the pixel-wise cost volume c_m.

2) Generating the v-disparity image and detecting the ground plane on it.

3) Generating the cost matrices c_o and c_g.

4) Generating the height cost matrix c_h.

5) Solving two 2D dynamic programming problems, one in the $U \times D$ domain and a second one in a subset of the $U \times V$ (image) domain.

Most stereo algorithms that provide "good enough" results are based on some variant of belief propagation. The simplest of them will compute c_m and run multiple 2D dynamic programs over different image slices, a number comparable to the rows in the image [10]. In that perspective, steps 1 and 5 account for only a very small fraction (less than 10%) of such stereo algorithms. On the other hand, step 4 corresponds to less than 15% of a simple winner-take-all sum of absolute differences stereo algorithm, possibly the fastest stereo algorithm available.

Since previous stixel world estimation methods are based on depth map estimation and our approach cuts the cost of this step without increasing posterior computations, for equivalently efficient implementations, our algorithm is guaranteed to be significantly faster.

Our initial, fully multi-threaded, naive implementation runs at about ~ 2 Hz. By making a better use of the CPU capabilities, we achieve a significant speed improvement. There are three main differences between our optimized and naive version.

1) The memory access patterns are optimized (transposing the images when necessary), and the allocated memory is reduced to minimal bits representations (e.g. int16 versus float32).

2) SIMD instructions are used explicitly for the sum of absolute differences operations.

3) The summed costs are computed directly. This is the key difference that, combined with the above points, gives the main speed-up.
Instead of computing (and storing) the cost volume c_m, and then computing the u axis sum (ground estimate), the v axis sum (distance estimate) or the small windows sums (height estimate); the sums are computed directly by iterating over the images. In theory, this means recomputing three times the cost volume, but in practice, given the architecture of modern CPUs, this is significantly faster.

When running on a 2009 desktop machine (Intel Core i7 860@2.80GHz, 4 cores with hyper-threading), averaged over the Bahnhof sequence, we obtain the following speed results:

~ 300 Hz for ground plane estimation (see §2.2).

~ 95 Hz for ground plane + stixels distance estimation (`fixed` case) (see §2.3).

~ 25 Hz for ground plane + stixels distance + stixels height estimation (`estimated` case) (see §2.4).

When properly optimized, the proposed approach is capable of reaching real-time speed processing on CPU only. We leave the GPU fully available for the appearance based

object detection (or any other desired processing).

For comparison, the original stixel world paper [2] uses a FPGA+CPU combo that runs at $\sim$ 15 Hz (40 + 30 ms, from images input to stixels estimate). Also, the SIMD optimized, multi-threaded CPU implementation of stereo block matching in OpenCv 2.2 (default parameters) runs at $\sim$ 20 Hz on our evaluation machine.

3.5.2 Detection search space reduction

Assuming an image of 640×480 pixels, that the detection windows are positioned in steps of 8 pixels (vertically and horizontally), and that 16 different scales are used; a full search would require considering 75000 detection windows.

Using the ground plane information allows one to fix the scale (assuming constant ratio) and the search area (assuming an horizon at the middle of the image), leading to a reduction of the number of detection windows to $\sim$ 2500 windows. Using the stixel estimates with a margin of $\pm$30 pixels this number is further reduced to $\sim$ 650 windows (see figure 2), about *one fourth* of when using the ground estimate only.

4. Conclusion and future work

Robot navigation requires object detection. We have shown that the stixel world can be used to reduce the set of candidate detection windows.

We have proposed a new algorithm to estimate the stixel world without having to compute a depth map at all. This new method provides an algorithmic speed improvement over state of the art, enabling real time performance on standard computers ($\sim$ 25 Hz in our CPU implementation). A quantitative evaluation has shown that the number of candidate detection windows is reduced to a fourth of what would be needed when using ground estimation only, while slightly improving the precision-recall curve.

We are currently developing an appearance based objects detector specifically designed to be coupled with the stixel world estimate. We hope to be able to exploit the height estimation or the membership function $m(u, v)$ to further improve detection rates.

The proposed algorithm can also be used in robots with stereo cameras and horizontal laser scanners. In this setup, the laser would provide the stixel depth, and our algorithm would provide the height estimation. We are interested in exploring this configuration for robotic applications.

Acknowledgement The authors would like to thank Kristof Overdulve for his inputs. This work has been partly supported by the Toyota Motor Corporation and the EU project EUROPA (FP7-231888).

References

[1] H. Badino, U. Franke, and R. Mester. Free space computation using stochastic occupancy grids and dynamic programming. In *ICCV, Workshop on Dynamical Vision*, 2007. 2

[2] H. Badino, U. Franke, and D. Pfeiffer. The stixel world - a compact medium level representation of the 3d-world. In *DAGM*, 2009. 1, 2, 4, 5, 8

[3] M. Bajracharya, B. Moghaddam, A. Howard, S. Brennan, and L. H. Matthies. A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. *IJRR*, 28:1466–1485, 2009. 1

[4] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool. 3d city modeling using cognitive loops. In *Video Proceedings for CVPR*, June 2006. 2

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, CA, USA, 2005. 6

[6] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*. IEEE Press, June 2008. 1, 5

[7] U. Franke and A. Joos. Real-time stereo vision for urban traffic scene understanding. In *IVS*, 2000. 2

[8] U. Franke and I. Kutzbach. Fast stereo based object detection for stop and go traffic. In *IVS*, 1996. 2

[9] H. Hattori, A. Seki, M. Nishiyama, and T. Watanabe. Stereo-based pedestrian detection using multiple patterns. In *BMVC*, 2009. 1

[10] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *CVPR*, volume 2, pages 807–814, CA, USA, 2005. 2, 7

[11] Z. Hu, F. Lamosa, and K. Uchimura. A complete u-v-disparity study for stereovision based 3d driving environment analysis. In *3DIM*, 2005. 2

[12] C. Keller, D. Fernandez, and D. Gavrila. Dense stereo-based roi generation for pedestrian detection. In *DAGM*, 2009. 2

[13] S. Kubota, T. Nakano, and Y. Okamoto. A global optimization algorithm for real-time on-board stereo obstacle detection systems. In *IVS*, Turkey, June 2007. 2, 3

[14] R. Labayrade, D. Aubert, and J.-P. Tarel. Real time obstacle detection on non flat road geometry through 'v-disparity' representation. In *IVS*, 2002. 2

[15] S. Nedevschi, R. Danescu, D. Frentiu, T. Marita, F. Oniga, C. Pocol, R. Schmidt, and T. Graf. High accuracy stereo vision system for far distance obstacle detection. In *IVS*, 2004. 2

[16] M. Okutomi and S. Noguchi. Extraction of road region using stereo images. In *ICPR*, volume 1, pages 853–856, 1998. 2

[17] D. Pfeiffer and U. Franke. Efficient representation of traffic scenes by means of dynamic stixels. In *IVS*, 2010. 1

[18] D. Pfeiffer, S. Moralez, A. Barth, and U. Franke. Ground truth evaluation of the stixel representation using laser scanners. In *ITSC*, September 2010. 1, 5

[19] A. Seki and M. Okutomi. Robust obstacle detection in general road environment based on road extraction and pose estimation. In *IVS*, 2006. 2

[20] Q. Yang, L. Wang, and N. Ahuja. A constant-space belief propagation algorithm for stereo matching. In *CVPR*, 2010. 5