Journal of Imaging Science and Technology® 57(6): 060501-1–060501-7, 2013.

Fast and Dense Depth Map Estimation for Stereovision Low-cost Systems

Pablo Revuelta Sanz, Belén Ruiz Mezcua, and José M. Sánchez Pena
Carlos III University of Madrid, Spanish Center for Captioning and Audiodescription (CESyA). Av. Peces Barba, 1. 28918 Leganés, Madrid, Spain
E-mail: prevuelt@ing.uc3m.es

Abstract. *Depth map extraction or estimation is very important for many computer vision applications. However, this process involves high cost. Hence, it is interesting to find new approaches, which could yield faster implementations, when very accurate results are not necessary. These new implementations can be built up for the stereovision problem, taking advantage of the specific geometrical constraints of this kind of problem. The problem is solved by searching pixel-by-pixel matching in horizontal lines. In this article, we propose a novel approach to this problem, providing measurements of the processing time and accuracy achieved, as well as a qualitative comparison with other methods available in the literature. The results obtained could reach speeds around one order of magnitude faster than those proposed in other classical implementations. Thus, the proposed algorithm can run on cheap and low-performance hardware achieving real-time processing rates. © 2013 Society for Imaging Science and Technology.*

[DOI: 10.2352/J.ImagingSci.Technol.2013.57.6.060501]

INTRODUCTION

Depth map estimation consists of extracting the distances of every single pixel of an image to the camera. A depth map of a scene can be computed by means of several approaches. This task has been solved using both passive and active methods, but we will focus only on passive methods.

We find several solutions in this approach with several cameras pointing to the scene to be computed.[1,2] Among them, we find the stereovision approach, in which two cameras point to the scene, slightly displaced.

We chose stereovision because this approach has two important advantages: the possibility of absolute depth measurements, and the fact that some geometrical assumptions can be made, simplifying the problem. These simplifications of the generalized problem allow faster implementations, as we propose in this article.

In stereovision, there is a mandatory task, referred to as *matching*, which consists of identifying the same physical points within different images.[1] With the found correspondence, the disparity between both images can be computed and, hence, the depth map can be extracted.

This approach solves the problem with four main strategies: local, cooperative, dynamic programming, and global approximations.

Received Mar. 13, 2012; accepted for publication Nov. 16, 2013; published online Jan. 3, 2014. Associate Editor: Miguel Lopez-Álvarez.
1062-3701/2014/57(6)/060501/7/$25.00

The first option takes into account only disparities within a finite window or neighborhood, which presents similar intensities in both images.[3] The value of a matching criterion (sum of absolute differences (SAD), sum of squared differences (SSD), or any other characterization of the neighborhood of a pixel) in one image is compared with the value computed in the other image for a displacing window. These windows are of $k \times k$ pixel size. Then, this sum is optimized and the best match pixel is found. Finally, the disparity is computed from the abscissa difference of matched windows.

The results of this kind of algorithm are not very accurate, as is shown, for example, in Ref. 4. The main disadvantage can be clearly seen: the number of operations needed gives a global order of the algorithm of $O(n) = N^3 \cdot k^4$ for an $N \times N$ image with windows of $k \times k$ pixels. This order is very high, and these algorithms are not very fast; the fastest one found in the literature is around 1 and 5 fps.[4]

Another possibility for local matching is implemented by means of point matching. The basic idea consists of identifying important points (relevant information) in both images. After this process, all relevant points are identified. Results of these algorithms are more accurate than those presented previously, and sometimes faster, as shown in Ref. 5. Once again, we find that these algorithms are not very fast, achieving processing times of a few seconds.[6] In the case of Lui, he gives time measures to obtain these results with a Pentium IV (2.4 GHz): 11.1 and 4.4 s for the Venus and the Tsukuba pairs, respectively. Another drawback is the necessity of interpolation, since point matching does not provide, by definition, a *dense matching*, i.e., a disparity estimate at each pixel.[7] Only matched points are measured. After that, an interpolation of the non-identified points is mandatory, slightly increasing the processing time. Another important disadvantage is the disparity computation on untextured surfaces.

Cooperative algorithms were first proposed by Marr and Poggio,[8] and the method was implemented by trying to simulate how the human brain works. A two-dimensional neural network iterates with inhibitory and excitatory connections until a stable state is reached. Later, some other proposals in this group have been given.[9]

Global algorithms make explicit smoothness assumptions converting the problem into an optimization one. They seek a disparity assignment that minimizes a global

cost or energy function that combines data and smoothness terms.[7,10]

Another possibility of global algorithms is those of belief propagation,[11] modeling smoothness, discontinuities, and occlusions with three Markov random fields and itinerates finding the best solution of a maximum a posteriori (MAP).

Some of the best results with global strategies have been achieved with the so-called graph cuts matching. A graph cut matching extends the one-dimensional (1D) formulation of the dynamic programming approach to two dimensions, assuming a local *coherence constraint*, i.e., for each pixel, neighborhoods have similar disparity. This approximation is widely explained in Ref. 10.

A final family of global algorithms to be studied is segment-based algorithms. This group of algorithms segments the image into regions to match the regions instead of pixels. An initial pair of images is smoothed and segmented into regions (see Ref. 12 and Eq. (1) of this article). The aim of this family of algorithms addresses the problem of untextured regions:

$$\Omega = \sum_{k-1}^{K} S_k, \tag{1}$$

Ω being the whole image and S_k the kth non-overlapping region.

These algorithms have the advantage of producing a dense depth map, thus avoiding interpolation. This algorithm also performs a $k \times k$ window pre-match, and a plane fitting, producing a high computational load (and computation time of tens of seconds), and inhibiting its use in real-time applications.[12]

Combinations of segment-based and graph cuts algorithms have also been implemented.[13]

The dynamic programming strategy consists of assuming the ordering constraint as always true.[10] The scan line is assumed, then, to be horizontal and unidimensional. The independent match of horizontal lines produces horizontal "streaks". The problem with the noise sensitivity of this proposal is smoothed with vertical edges[14] or ground control points.[15]

The specific advantage of this last approach is the well-known position of cameras, which allows a simplification of the problem, as mentioned earlier. Its correlation in the matching process is the simplification of the problem from 2D to 1D matching. This is also the case of some studies available[5,16]. This idea is exploited in our proposal.

The main problem with most of the proposed approaches to the stereovision problem, even those of 1D matching, is the use of quite complex operators to find the relevant points (see Refs. 17–20, for example). Additional algorithms taking advantage of a time analysis have been proposed as well.[21,22] The calculations required for depth mapping of images have been studied in detail, and a complete review of algorithms performing this task by means of stereovision can be found in Ref. 7.

We propose in this article a novel approach to depth map estimation, sacrificing accuracy to achieve a very low complexity and, hence, high processing rates, significantly over real-time constraints.

This article is organized as follows. Following this introduction, we describe the problem and the corresponding assumptions. The same section explains in detail how our proposal works, the post-processing, and main features. The Results section gives examples of the depth maps extracted from some standard pairs of images, which will be discussed and compared with other results found in the literature in the following sections. Finally, a Conclusion section summarizes the main aspects of this study.

MATERIALS AND METHODS

As mentioned above, the matching problem applied to stereovision can be simplified if the appropriate approach is considered.

The geometrical assumptions used are explained in detail in Ref. 1, the most important one being the *fronto-parallel hypothesis*: "the retinal planes of the cameras are identical and the scene is an assembly of planes parallel to them".[1] Moreover, taking into account the idea first proposed in Ref. 23, we can arrive at the following geometrical assumption: We assume that the scan lines for point extraction and matching are horizontal. Since the *fronto-parallel hypothesis* is taken into account, and the cameras setup fits with this constraint, we can assume that every physical (and not occluded) point in one image must be found at the same height in the other image. The horizontal displacement will be used to compute the depth. The next assumption to be made is that every neighboring pixel has similar disparity, which we call the local coherence constraint.[10] Of course, this constraint is not true in edges and occluded pixels, but it allows us to perform a pixel-by-pixel matching assuming this constraint until a comparison threshold is surpassed.

A final hypothesis is assumed: both images (and, hence, cameras) have the same photogrammetric parameters, so no brightness correction (or white balance if color images are used) is performed. This is a key point of the algorithm when working in real-life situations, but given that in this study we will use standard stereo pairs of images, the assumption is fulfilled.

In this scenario, we propose the following solution to depth map estimation.

(1) Finding a first relevant point for each scan line by means of a horizontal and unidimensional gradient operator.

(2) Finding the corresponding point in the other image with the same operation.

(3) Trying to match pixel by pixel, defining an acceptance threshold and allowing some outliers (to reduce the effect of impulsive noise).

The proposed algorithm works with rectified gray-scale images.

The first task to be implemented is the relevant point extraction. This is done with a horizontal differential mask of size 1×3, with the following structure: $\{-1, 0, 1\}$. The horizontal size of the window allows one to reduce the effect of the Gaussian blur. The convolution over this mask is compared with a threshold to decide whether it is a relevant starting point or not.

Every comparison is performed with the sum of absolute difference approach shown in Eq. (2) for two arbitrary values V_1 and V_2, and compared with a pre-defined threshold T:

$$abs(V_1 - V_2) < T \tag{2}$$

Once two corresponding pixels are found in one line, pixel-by-pixel matching is implemented for each line i, with L and R referring to the left and right images, j being the column number and $point(x, y)_X$ a specific point in the left (if $X = \mathrm{L}$) or right (if $X = \mathrm{R}$) image, at the coordinates x and y:

```
for j_L = 1:N                                   //Being N the number of columns
   if SAD (mask·window_L(i,j_L)) < TH_1 then    //A relevant point found in left image
A:  for j_R = 1:N                               //Then, we scan the right image
       if point(i, j_R)_R == point(i, j_L)_L then   //looking for the same point in both images
         while j_R < N do
           if outliers < MAX_OUTLIERS then      //The number of outliers is 4
             if SAD (point_R, point_L) < TH_2 then  //Matching condition
               depth (i,j_L) = j_L- j_R;        //Match implemented point-by-point
               j_L++, j_R++;                    //both indexes increase (so we compare the
                                                //next couple of points
             endif;
             outliers++;                        //"OUTMAX" outliers (non-matching pixels)
                                                //are allowed
             j_L++;                             //It might be an occlusion, so only left index
                                                //is increased
           endif;
           j_R++;                               //It might be an occlusion in the other image,
                                                //so only the right index is increased
           goto A;                              //Once some index has been increased, new
                                                //point-by-point matching is tried.
         endwhile;
       endif;
     endfor;
   endif;
endfor;
```

In this scheme, the fourth line comparison is done by comparing values of the examined pixels in both images, as well as those of the sides and those at the bottom, i.e., for a point (i, j), points $(i, j - 1)$, $(i, j + 1)$, and $(i + 1, j)$ are also compared in both images to minimize the error of taking wrong matches. This decision makes this first disparity measurement stronger. Although arbitrary, it is as simple as possible to allow high speed rates without threatening the accuracy.

The seventh line comparison is a single value difference of both points. No mask is used in this step, strongly improving the performance of the algorithm.

Given that we are comparing points which are, theoretically, the same (i.e. have the same brightness assuming the equal photogrammetric parameters), any difference should be interpreted as a discontinuity. However this does not work. Small differences appear in any pair of real images, and the algorithm must present some degree of tolerance. This is implemented with TH_2 and the outliers. TH_2 is the maximum allowed difference in the brightness of two points to be treated as the same one. However, salt and pepper noise may overpass this value, although it is not an occlusion or a disparity edge. This is the reason why there is a number of outliers allowed. With this tolerance, the algorithm ignores some pixels even though they are not the same. If there is indeed a discontinuity, after four pixels the algorithm understands that it is a discontinuity, and looks for the new disparity value. This flexibility makes the algorithm accurate enough with respect to the time performance.

Table I. Performance of the algorithm for different number of scan lines in the Tsukuba pair of images. Threshold for the error estimation is 2.

Number of scan lines (%)	Time (ms)	fps	Non-occluded Errors (%)	All (%)	Disc (%)
(c) 100	17.8	56	8.34	8.81	27.2
(d) 50	8.8	114	7.45	7.93	29.0

We have, then, the possibility of scanning or not every line of the original pair of images. When not every row of the image is scanned, the algorithm deals with subimages of $L \times M$, L being the number of scan lines. Thus, the final image is a stretched version of the estimated depth map and must be resized to its original size $N \times M$. This allows managing a simplified version of the images, where any other processing will be much lighter than in the original size image.

We find three degrees of freedom in this algorithm:

- percentage of horizontal lines scanned;
- threshold for pixel acceptance as non-outlier;
- number of allowed outliers.

The number of scanned lines can vary, taking all of them or a subset. This possibility opens the door to a better optimization between accuracy and processing speed, finding a specific agreement between these two factors depending on the final application.

The threshold tolerance to accept or not a new pixel (the SAD between this and the previous one) is an important parameter in the algorithm speed, as will be shown in the Results section. Moreover, this parameter is related to the number of accepted outliers before searching a new matching condition.

The main post-process implemented over the stretched image in our proposal is a vertical median filter, processing the estimated stretched depth map before resizing it, when not every line has been examined.

RESULTS

The algorithm was implemented in C using the OpenCV library over a 1.6 GHz PC. It was tested over standard sets of images used in stereovision from the Middlebury test bed.[24]

On processing the Tsukuba pair of images, we obtained the results shown in Figure 1 for 100 and 50% of scanned lines, for a threshold of 8, and three allowed outliers.

As explained earlier, the results sacrifice accuracy to achieve high processing speed.

The time delay caused by the algorithm shown in Table I considers the scanning, the matching, and the post-processing median algorithm with 3×3 window.

Table II. Performance of the algorithm for different number of scan lines in the Teddy pair of images.

Number of scan lines (%)	Time (ms)	fps	Non-occluded errors (%)	All (%)	Disc (%)
(c) 50	26.3	38	23.9	28.1	36.1
(d) 100	52.6	19	23.1	27.4	31.7

Table III. Performance of the algorithm for different number of scan lines in the Venus pair of images.

Number of scan lines (%)	Time (ms)	fps	Non-occluded Errors (%)	All (%)	Disc (%)
(c) 50	17.8	56	9.69	10.1	29.6
(d) 100	35.7	28	9.77	10.2	30.5

In the table, *All* means all pixels, and *Disc* only pixels in discontinuities. These results are obtained automatically from the Middlebury algorithm tester.

The image obtained with half of the scan lines has been interpolated (each line is doubled) to achieve the original size and allow an automatic comparison with the depth truth.

As usually happens in the dynamic programming approach, "streak" lines appear, which cannot be completely removed by the median vertical filter without important information loss.

Figure 2 shows the results for the Teddy pair of images. The size of each image is 375 × 450 pixels, and the median filter size used in the post-processing is 5 × 5.

The time analysis is given in Table II.

Finally, the pair of Venus images was processed, and the results are shown in Figure 3. These images have a resolution of 383 × 434 pixels, and a median filter of 5 × 5 was implemented before presenting.

The time analysis is given in Table III.

It is important to note that the last two examples are very rich in texture, where algorithms dealing with edges or region growing used to have several problems.

DISCUSSION

Nowadays, the proposed algorithms achieves accurate depth map estimations investing a huge amount of computational resources and time.

Figure 4 presents some results in (color-based) depth map estimations from the same pairs of images of (a) a real-time correlation-based algorithm from Ref. 4, (b) a segment-based algorithm from Ref. 13, and (c) also a segment-based algorithm from Ref. 25. These algorithms will be used for time performance comparisons.

The quality of these depth map estimations is higher than that proposed in this article, but, regarding the time performance, we find interesting results. Hirschmüller et al. obtain this result over a 450 MHz processor at 4.7 fps (notice that the image size is 240 × 320 in this setup).[4] Hong and Chen get the image in Fig. 4(b) with a 2.4 GHz PC after 3 s (image size 288 × 384). In the case of Klaus et al., the computation time required is higher than 14 s on a 2.21 GHz machine[25] (the size is not specified in this study). We can easily see the improvement in terms of performance of our algorithm, since our results for the same image achieve a frame rate of 56 fps in slower processors and the best case (except the case of Hirschmüller et al.). This is the new deal achieved between accuracy and time and, hence, the main contribution of our proposal.

Regarding the complexity of the algorithm, we can appreciate linearity in the computation time as a function of the number of lines. Moreover, the order of the algorithm we present and analyze in this article is $O(M \cdot L)$, M being the number of columns in the image and L the number of lines to be scanned. No dependency of the height of the image was found. Just for comparison purposes, we can take the example given in Ref. 26. The algorithm proposed in this article matches points for fingerprint identification. It has a complexity of $O(n^2k^2 \log n)$, k being the operator size and n the size of the image.

This essential difference is due to the highly specific approach adopted in this study: we do not deal with rotation or scale-variant images, as is done in Ref. 26. We can assume that the epipolar lines are horizontal and identical in both images and, hence, we do not have to search the feature points in a 2D space. This constraint reduces the complexity of the problem to one dimension. We can exploit all these specifications of a specific stereovision problem to adjust the searching and matching algorithm to make it as simple as possible. Then, its complexity falls down dramatically, and although the final results are not very accurate (it depends on the final application, as said), we find it interesting as a new way to solve the stereo matching problem.

These constraints are taken into account in the presented dynamic processing algorithms with temporal comparison. In the case of Ref. 22, the time analysis with a dual core 2.1 GHz PC and images of 512 × 512 pixels reports delays between 104 and 188 ms. The main reason for the increment of the processing time is the use of more complex operators (as the Canny edge) and, of course, the time comparison between consecutive frames.

We can clearly see the arrangement between complexity and accuracy. In fact, the goodness of each algorithm depends on the final application of the algorithm. What we have searched is a light way to estimate the depth map when accuracy is not critical.

Finally, we found some unexpected results: the half-line definition depth estimation had, in some cases, lower errors than the full-line definition depth estimations. As can be appreciated from Fig. 1 (the one with largest differences concerning this issue), image (c) (full-line definition) has more streaky lines than image (d) (half-line definition, which is smoother, as happens with every other half-line definition depth estimation). This result is arbitrary: for the same couple of contiguous lines, one algorithm may produce an important error (creating a streak) in one of them and no error in the other one. The half-line process will just process

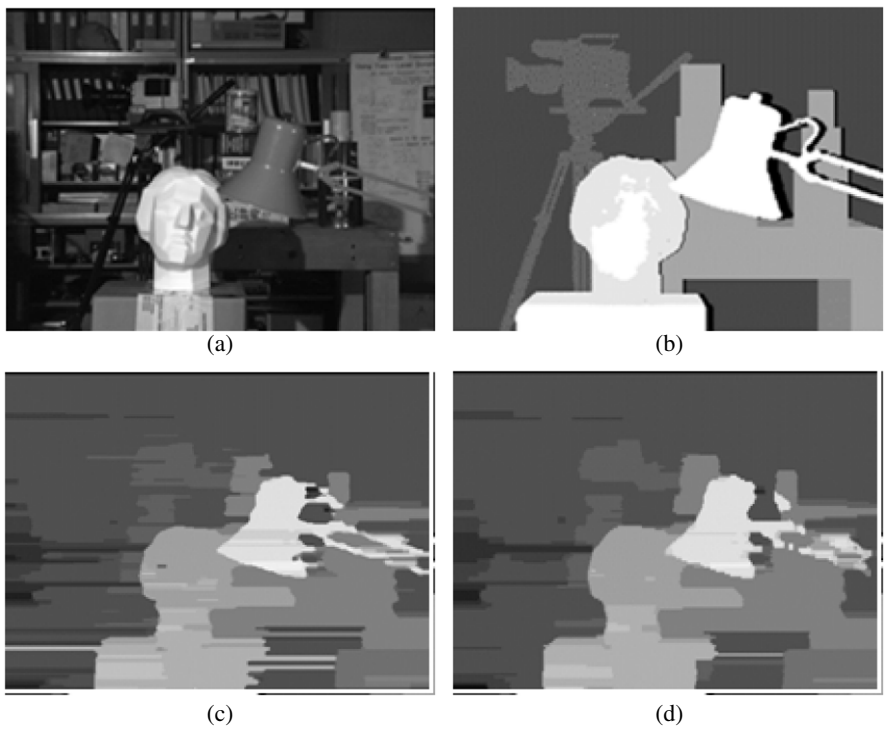

Figure 1. (a) Original Tsukuba left image and (b) ground truth, and depth map estimation with (c) 100% and (d) 50% of scan lines.

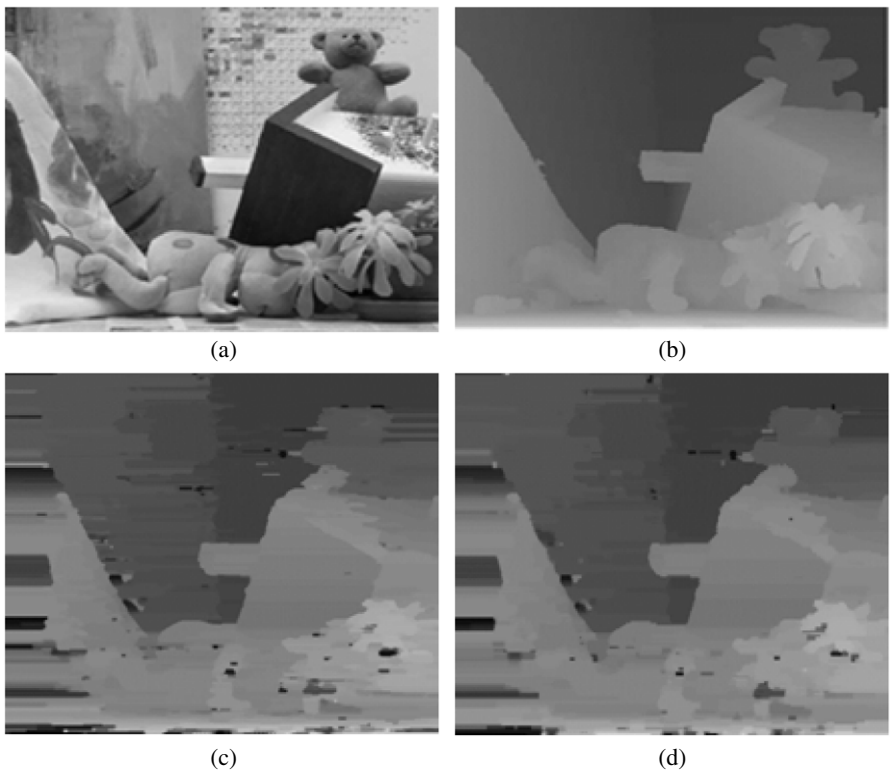

Figure 2. (a) Original Teddy left image and (b) true depth map. Teddy depth map estimation with (c) 187 and (d) 375 scan lines.

one of them, and it may take the line producing errors (and so, when interpolating, would produce the double of error pixels) or the best one (with no error generation). We can affirm that it is not a theoretical effect of the algorithm but

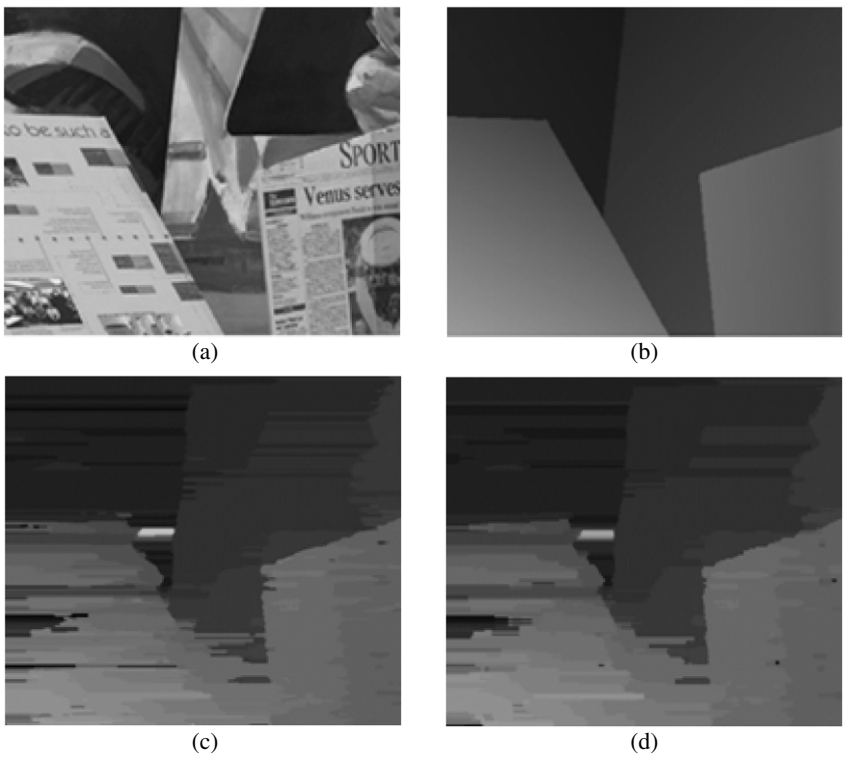

Figure 3. (a) Original Venus left image and (b) true depth map. Venus depth map estimation with (c) 191 and (d) 383 scan lines.

Figure 4. (a) Real-time correlation-based result taken from Ref. 4, (b) segment-based result from Ref. 13, and (c) another segment-based result from Ref. 25.

a random effect of the lines selected. Thus, it should not be taken as relevant (but the time required in each case is).

CONCLUSIONS

Depth map estimation is a complex computation task. Because of this, we have to find a balance between time and accuracy. In this scenario, we have proposed a fast algorithm to compute dense depth maps that work between 6 (worst case) to 100 times faster than more accurate options found in the literature, achieving rates from 20 to hundreds of fps in general-purpose PCs depending on the required definition.

To get that result, we have approximated the general matching problem by means of several geometrical constraints, which allow us to simplify the problem to that of scanning horizontal lines implementing a 1D differential filter to find relevant points at the beginning of each line.

Another interesting result found was the linearity of the order of the proposed algorithm, which depends on the number of lines to be scanned and the width of the image, since the scanning lines are horizontal.

Some problems in repetitive patterns and untextured areas were found, which have to be treated separately in further studies.

ACKNOWLEDGMENT

We would like to acknowledge the student grant offered by the Universidad Carlos III de Madrid and Spanish Center for Subtitling and Audiodescription (CESyA), which has allowed this research work to be performed.

REFERENCES

[1] J.-P. Pons and R. Keriven, "Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score," Int. J. Comput. Vision **72**, 179 (2007).

[2] H. K. I. Kim, K. Kogure, and K. Sohn, "A real-time 3D modeling system using multiple stereo cameras for free-viewpoint video generation," LNCS Image Anal. Recog. **4142**, 237 (2006).
[3] M. S. Islam and L. Kitchen, "Nonlinear similarity based image matching," Int. Federation Inform. Process. **228**, 401 (2004).
[4] H. Hirschmüller, P. R. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," J. Comput. Vision **47**, 229 (2002).
[5] B. Liu, H.-B. Gao, and Q. Zhang, "Research of correspondence points matching on binocular stereo vision measurement system based on wavelet," *Int'l Conf. on Machine Learning and Cybernetics, 2006*, **3687**, (2006).
[6] J. C. Kim, K. M. Lee, B. T. Choi, and S. U. Lee, "A dense stereo matching using two-pass dynamic programming with generalized ground control points," *IEEE Computer Society Conf. on Comput. Vision Pattern Recognit, 2005 (CVPR 2005)*, **2**, 1075 (2005).
[7] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," Int. J. Comput. Vision **47**, 7 (2002).
[8] H. Marr and T. Poggio, "Cooperative computation of stereo disparity," Science **194**, 283 (1976).
[9] H. Mayer, "Analysis of means to improve cooperative disparity estimation", *ISPRS Conf. on Photogrammetric Image Analysis* XXXIV (2003).
[10] J. Käck, "Robust stereo correspondence using graph cuts," Master Thesis (2004).
[11] J. Sun, H.-Y. Shum, and N.-N. Zheng, "Stereo matching using belief propagation," *European Conf. on Comput. Vision*, **510**, (2002).
[12] M. Bleyer and M. Gelautz, "A layered stereo matching algorithm using image segmentation and global visibility constraints," J. Photogrammetry & Remote Sensing **59**, 128 (2005).
[13] L. Hong and G. Chen, "Segment-based stereo matching using graph cuts," *Proc. 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 2004 (CVPR 2004)*, **1**, I-74 (2004).
[14] Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic programming," IEEE Trans. Pattern Anal. Mach. Intell. **7**, 139 (1985).
[15] A. Bobick and S. Intille, "Large occlusion stereo," Int. J. Comput. Vision **33**, 181 (1999).
[16] Z. Wang and Y. Quan, "An improved method for feature point matching in 3D reconstruction," *International Symposium on Information Science and Engineering, 2008 (ISISE '08)*, **1**, 159 (2008).
[17] K. Li, S. Wang, M. Yuan, and N. Chen, "Scale invariant control points based stereo matching for dynamic programming," *The Ninth Int'l Conf. on Electronic Measurement & Instruments (ICEMI'2009)* (2009), pp. 3–769.
[18] J. Zhao, H.-J. Zhou, and G.-Z. Men, "A method of sift feature points matching for image mosaic," *International Conference on Machine Learning and Cybernetics, 2009*, **4**, 2353 (2009).
[19] F. Tiesheng, N. Bing, W. Qingsong, W. Tao, and Q. Dapeng, "Novel stereo matching method on multi-scale Harris corner points," *International Symposium on Computer Science and Computational Technology, 2008 (ISCSCT'08)***1**, 167 (2008).
[20] S. Bereg, N. Mutsanas, and A. Wolff, "Matching points with rectangles and squares," Comput. Geometry **42**, 93 (2009).
[21] M. El Ansari, S. Mousset, and A. Bensrhair, "Temporal consistent real-time stereo for intelligent vehicles," Pattern Recognit. Lett. (2010).
[22] A. Mazoul, M. El Ansari, K. Zebbara, and G. Bebis, "Fast spatio-temporal stereo for intelligent transportation systems," Pattern Anal. Appl. (2012).
[23] Y. Zhang and Y. J. Gerbrands, "Method for matching general stereo planar curves," Image and Vision Computing **13**, 645 (1995).
[24] D. Scharstein, Middlebury Database, http://vision.middlebury.edu/stereo/ (2010).
[25] A. Klaus, M. Sormann, and K. Kraner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," *18th International Conference on Pattern Recognition, 2006 (ICPR 2006)* 15 (2006).
[26] A. Bishnu, S. Das, S. C. Nandy, and B. B. Bhattacharya, "Simple algorithms for partial point set pattern matching under rigid motion," Pattern Recognit **39**, 1662 (2006).