Published in Image Processing On Line on YYYY–MM–DD.
ISSN 2105–1232 © YYYY IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
http://dx.doi.org/10.5201/ipol.YYYY.XXXXXXXX

PREPRINT July 15, 2013

AD Census

Yohann Salaun[1] & Pascal Monasse[1]

[1] ()

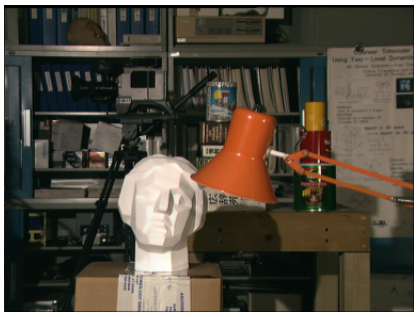
Abstract

This paper presents a local stereo matching algorithm [3] supposed to have good accuracy performances as shown by its position in Middlebury benchmark [4]. Originally published in order to be developed on graphics hardware, only the accuracy performance will be studied in this article. The algorithm bases its matching cost with a composition of the absolute difference and the Census costs. It is then aggregated in cross-based regions [7] and ends with a scanline optimization [1]. Methods are then used to detect outliers and correct them to finally give a disparity map that top performed in Middlebury benchmark.

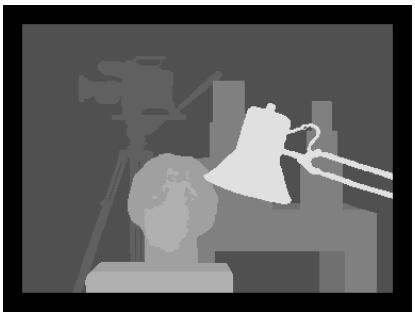
1 Overview

Stereo matching is one of the most active research areas in computer vision. Many algorithms have been proposed in the last decades and several have been presented and classified by Scharstein *et al.* [5].
The method [3] presented was aimed to be an accurate real-time method with GPU parallelization. When the article was published (August 2011), this method was the top performer of Middleburry benchmark [4] and is now (August 2013) second top performer. The aim of this article is to study the quality of the disparity map produced by this algorithm, putting aside the GPU parallelization and the time performance.

Left picture



Right picture



Real disparity

Figure 1: Tsukuba pair from the Middlebury benchmark [4] and its true disparity map.

2 Matching cost computation

A pixel of the left image I^{LEFT} and a pixel of the right one I^{RIGHT} are considered as matching points if they are on the same row and if their neighborhood intensity is about the same. In order to compare intensities, a matching cost is computed. The cost is initialized with comparison based on 2 matching candidates. Then, the cost of a picture are aggregated together in order to include the neighborhood in the comparison and thus be more robust to noise.

2.1 AD-Census Cost Measure

The cost function used in this algorithm is a combination of two other costs:

- **Absolute Difference** which computes the absolute difference of two pixel intensities.
- **Census** which computes the difference of two pixel local structures.

The absolute difference is a common matching cost, often used in disparity map computation for its good results. However it is lacking of accuracy in textureless regions. That is why the combination of two costs, supposed to improve the AD measure where its efficiency is too weak, has been used in this article.
The absolute difference cost is then given by the formula:

$$C_{AD}(p, pd) = \frac{1}{3} \sum_{i=R,G,B} |I_i^{LEFT}(p) - I_i^{RIGHT}(p+d)|$$

Here and in the whole article, p will be a pixel in the left picture and $p+d$ will be its correspondent with disparity d in the right picture.
The census matching cost compares the relative ordering of pixel intensities in a window of size 9 x 7 with respect to the central pixel intensity. It will penalize when neighbors have greater relative intensities in one picture and lower ones in the other.
The census cost formula can be given by:

$$C_{CENSUS}(p, pd) = \sum_{i=R,G,B} \sum_{q \in \text{Neighbors}(p)} \mathbb{1}_{\mathbb{R}^-} \left[\left(I_i^L(q) - I_i^L(p) \right) \left(I_i^R(q+d) - I_i^R(p+d) \right) \right]$$

where $\mathbb{1}_{\mathbb{R}^-}$ is the indicator function of $\mathbb{R}^-$.
The combination of these two costs is made with the function ρ defined by:

$$\rho(cost, \lambda) = 1 - e^{-\frac{cost}{\lambda}}$$

This function allows a better combination of the two costs by mapping their values into [0;1] where 0 stands for a perfect match and 1 for a total mismatch. Moreover, it allows us to easily define the limit between inliers and outliers separately for each cost.
The final cost measure is given by :

$$C(p, pd) = \rho(C_{AD}, \lambda_{AD}) + \rho(C_{CENSUS}, \lambda_{CENSUS})$$

The results of these different measures can be observed for the Tsukuba pair in figure 2.

AD only

Census only

Left picture

AD Census

Figure 2: The AD measure is very noisy because the comparison window is too small but the edges are sharper whereas the Census measure has the opposite characteristics. The AD Census is the compromise of the 2 previous measures.

2.2 Cost Aggregation

Once the cost is initialized with AD-Census as explained above, it is aggregated in windows in order to increase matching accuracy. The method used is inspired by the one proposed by *Zhang et al.* [7]. The aggregation windows are computed independently for each pixel and depends on the local structure of their neighborhoods. The aim of these adaptive windows is to consider only the pixels that have similar intensities and thus that should belong to the same structures. This way, occlusion issues and fattening effects should be less existing.

We then define, for each pixel p, upward, downard, leftward and rightward borders pixel q with the following algorithm. The border pixel q has to be the last pixel in a given direction that is such that:

1. $D_C(q,p) < \tau_1$ and $D_C(q, q_{predecessor}) < \tau_1$

2. $D_S(q,p) < L_1$

3. $D_C(q,p) < \tau_2 < \tau_1$ if $L_2 < D_S(q,p) < L_1$

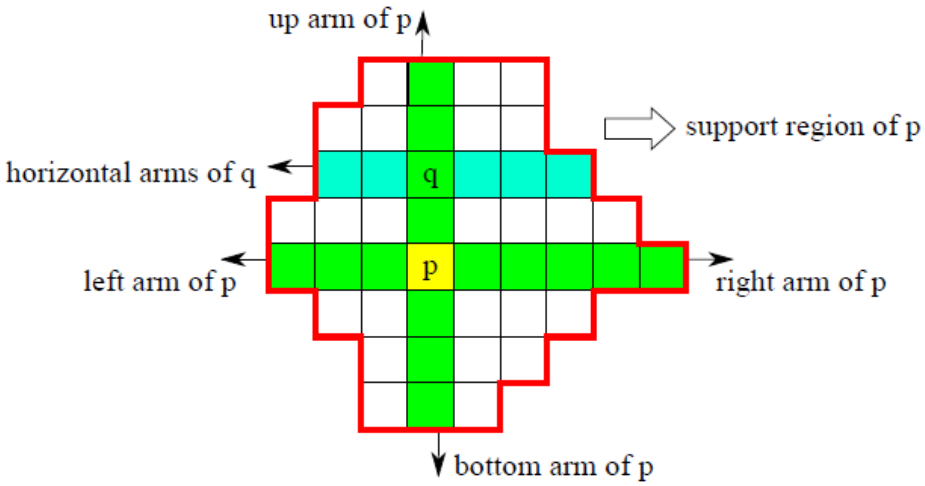


Figure 3: The adaptive windows only keeps pixel that have strong enough color similarities with the pixel in its center. It also has a maximum size which avoid the case of huge windows.

where $D_C(p,q) = \max_{i \in R,G,B} |I_i(p) - I_i(q)|$ is the color distance between 2 pixels and $D_S(p,q) = |p-q|$ is the spatial distance between 2 pixels.
The first condition ensures that pixels which belong to the same window have enough color similarity and that this similarity is kept from the center of the window to the edge. The second defines a limit size for the window in order to avoid too large windows that would not perform better but would cost more computation time. The last condition reduce the maximal size of the window when the region is too textured.
With the borders defined, the cost can be aggregated either horizontally first or vertically first. This way, two adaptive windows are defined depending on the order of aggregation. In order to get a stable cost volume, [3] advises us to compute the agregation 4 times, twice horizontally first and twice vertically first.
The effect of the aggregation can be observed on figure 5.

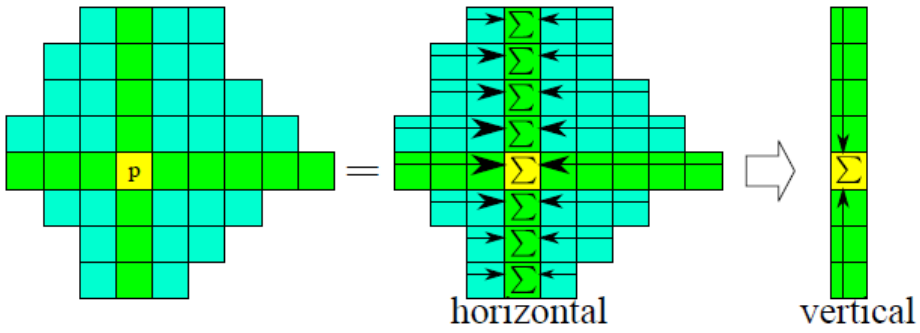


Figure 4: The aggregation is done one direction after the other with the corresponding border pixels.

2.3 Scanline Optimization

For better matching results, a second aggregating step is then computed. It follows the multi-direction scanline optimizer developed by *Hirschmuller* in [1]. The idea is to refine the cost along

one direction by comparing the cost of the neighboring pixels and disparities.
Four scanline optimizations are thus processed independently, leftward, rightward, upward and downward. For a given direction r, the cost is updated from the first column/line to the last with the following formula:

$$\begin{aligned} C_r(p,d) &= C(p,d) - \min_k C_r(p-r,k) \\ &\quad + \min\Big(C_r(p,d), C_r(p-r,d+1)+P_1, C_r(p-r,d-1)+P_1, \min_k C_r(p-r,k)+P_2\Big) \end{aligned}$$

where $p-r$ is the previous pixel along the direction r.
$P_1 < P_2$ are parameters that penalize the disparity change between neighboring pixels. They depends on $D_1 = D_C(p, p-r)$ in the left picture and $D_2 = D_C(pd, pd-r)$ in the right picture. They are symmetrically set by the formulas:

1. $P_1 = \Pi_1, \quad P_2 = \Pi_2$, if $D_1 < \tau_{SO}$ and $D_2 < \tau_{SO}$
2. $P_1 = \frac{\Pi_1}{4}, \quad P_2 = \frac{\Pi_2}{4}$, if $D_1 > \tau_{SO}$ and $D_2 < \tau_{SO}$
3. $P_1 = \frac{\Pi_1}{4}, \quad P_2 = \frac{\Pi_2}{4}$, if $D_1 < \tau_{SO}$ and $D_2 > \tau_{SO}$
4. $P_1 = \frac{\Pi_1}{10}, \quad P_2 = \frac{\Pi_2}{10}$, if $D_1 > \tau_{SO}$ and $D_2 > \tau_{SO}$

where $\Pi_1, \quad \Pi_2$ and τ_{SO} are parameters.

Finally, the scanline optimization cost is the average of the 4 directional costs computed previously:

$$C(p,d) = \frac{1}{4}\sum_r C_r(p,d)$$

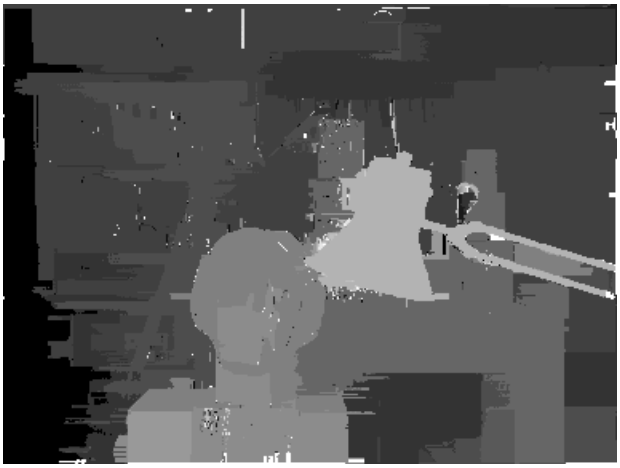
3 Disparity Refinement

The aim of this part is to find the outliers in the disparity maps of the right and the left picture. Once they are found, their intensities are replaced by new one computed with the information found on the neighboring inliers. The original article [3] proposes many methods in order to find outliers and correct their values.
The first one is the left-right consistency check that is commonly used for refinement. It consists of declaring outliers all the pixels whose disparity is not consistent between left and right picture:

$$p \text{ is declared outlier } \Leftrightarrow \text{disparity}(p) \neq \text{disparity}(p+d)$$

However, even if [3] does not precise it, we consider that pixels that checks the equality above with an error of ± 1 are not considered as outliers. In fact, as the disparity is an integer, many good matches would be declared as outliers.
The other methods of refinement presented by [3] were not enough detailed so we used another algorithm implemented in [6] and inspired by [2]. The main idea of this filling algorithm is to consider that the rejected pixels are mainly occluded pixels. Thus, the algorithm looks for the nearest non occluded pixel on the same line and assign its disparity. Then, in order to avoid streak-line artifacts, this line-oriented procedure is followed by a smoothness filtering using bilateral filter.

without aggregation



with aggregation

Left picture

with aggregation and scanline optimization

Figure 5: The aggregation has a sort of denoising effect but creates some artifacts due to the fattening effect. The scanline optimization does not seem to be very effective.

4 Results

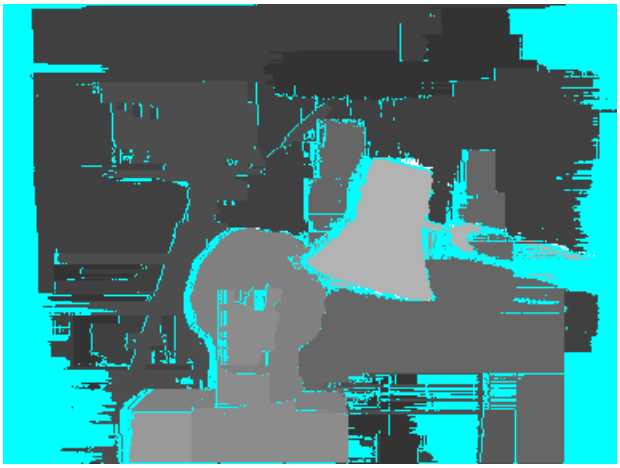
The default parameters are the same than the one presented in [3] and are summed up in table 1.

AD Census			Cost Aggregation					Scanline Optimization		
λ_{AD}	λ_{CENSUS}	Census window	L_1	L_2	τ_1	τ_2	Iterations	Π_1	Π_2	τ_{SO}
10	30	9×7	34	17	20	6	H-V-H-V	1	3	15

Table 1: Default parameters used for the algorithm.
H (resp. V) is for a horizontally-first (resp. vertically-first) cost aggregation.

The algorithm has run on the Middlebury database in order to be compared with the results obtained in [3]. Even though the method used for refinement is totally different between our algorithm and the method presented in [3], some artifacts that should disappear thanks to the cost computation processus can be seen in our final disparity map.

disparity map



outliers detection

Left picture

outliers filled

Figure 6: The scanline optimization does not seem to be very effective.

4.1 Census

TO CHANGE

In [3], the AD Census measure is presented as a method that can find the good disparity in textureless region while the AD measure alone cannot. In fact, our results show that the Census measure is better than the AD measure in textureless region. However, the AD-Census measure as presented in [3] and implemented in our algorithm is less efficient than Census alone in such regions. One solution could be that the coefficients used to run our algorithm and supposed to be the same as [3] are not strictly the same. Assigning a larger weight for Census measure could then maybe solve this issue.

4.2 Scanline Optimization

No real results are shown in [3] about the before and after of the scanline optimization. However, when running our algorithm, no meaningful difference can be observed. In fact, some noise disappear but also appear on different places. Once again, it could be a difference between the parameters used to run the algorithm though we used the same presented in the article.

left picture | results obtained in [3] | our results without outliers filling | census and aggregation only | ad census and aggregation only

Figure 7: Comparison of the different measures in textureless region.

before | after

Figure 8: Comparison between before and after scanline optimization.

5 Conclusion

Finally, the article [3] giving no sufficient details, the algorithm that we implemented is too different from the original one that topped the Middlebury benchmark. Thus, the quality of the disparity map obtained by our algorithm cannot really be a proof that [3] works or does not work well. However, the results presented in the previous part seem to contradict a bit its good performances and the choices made for the algorithm.
The work to be done from now would be to suppose that the parameters given are not necessarily the best (even for the Middlebury benchmark only) Then we could try with many different parameters and see if their choices really influence the results.
What can be said about [3] is that an algorithm has been implemented and perform very well for the Cones, Teddy, Tsukuba and Venus pictures. However, we do not really know how far this algorithm is from the article and thus its implementation cannot be performed without the authors' participation. Because of this difference, we cannot say if it outperforms only Middlebury benchmark or if it is really one of the current state of the art method about disparity computation.

References

[1] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):328–341, February 2008.

[2] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):504–511, 2013.

[3] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang. On Building an Accurate Stereo Matching System on Graphics Hardware. In *Third Workshop on GPUs for Computer Vision*, November 2011.

[4] D. Scharstein and R. Szeliski. Middlebury benchmark. *vision.middlebury.edu/stereo/*.

[5] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 2002.

[6] P. Tan and P. Monasse. Stereo disparity through cost aggregation with guided filter. *IPOL*, 2013.

[7] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Trans. Cir. and Sys. for Video Technol.*, 19:1073–1079, July 2009.

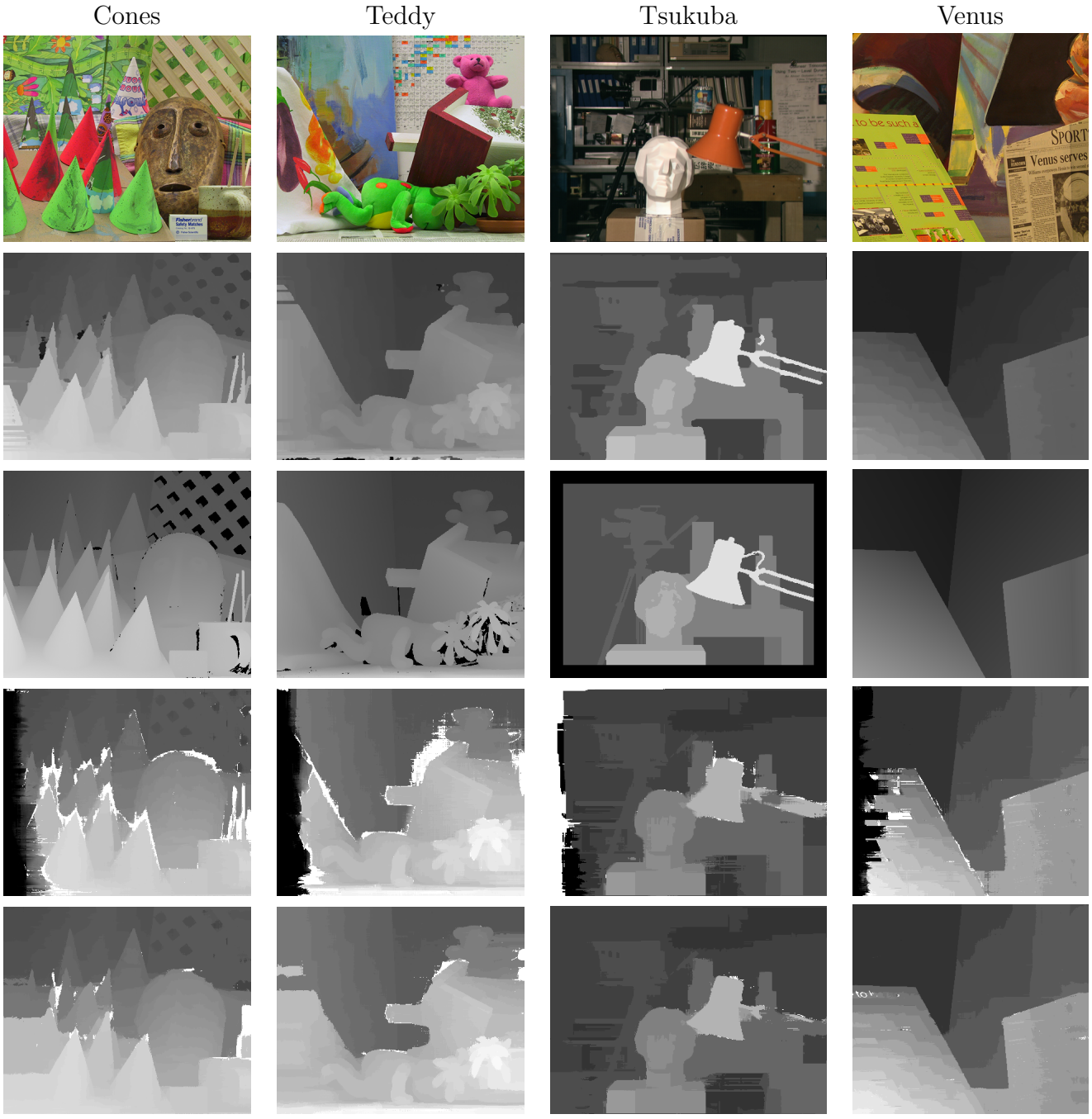


Figure 9: The 4 pictures used for the ranking in middlebury benchmark. First line shows the left picture of each pair, second line the results obtained by [3], third line the truth presented in [4] and the last two show the results obtained with our algorithm (before and after outliers correction).