2010 11th. Int. Conf. Control, Automation, Robotics and Vision
Singapore, 5-8th December 2010

Stereo for Robots: Quantitative Evaluation of Efficient and Low-memory Dense Stereo Algorithms

Federico Tombari
DEIS-ARCES, University of Bologna
Bologna, Italy
federico.tombari@unibo.it

Stefano Mattoccia
DEIS-ARCES, University of Bologna
Bologna, Italy
stefano.mattoccia@unibo.it

Luigi Di Stefano
DEIS-ARCES, University of Bologna
Bologna, Italy
luigi.distefano@unibo.it

***Abstract*—Despite the significant number of stereo vision algorithms proposed in literature in the last decade, most proposals are notably computationally demanding and/or memory hungry so that it is unfeasible to employ them in application scenarios requiring real-time or near real-time processing on platforms with limited resources such as embedded devices. In this paper, we have selected the subset of proposals that appears more suited to the above requirements and, since literature lacks a proper comparison between these methods, we propose a quantitative experimental evaluation aimed at highlighting the best performing approach under the two criteria of accuracy and efficiency. The evaluation is performed on a standard benchmark dataset as well as on a novel dataset, acquired by means of an active technique, characterized by realistic working conditions.**

***Index Terms*—Real-time, stereo vision, robot vision, quantitative evaluation**

I. Introduction

Stereo vision is a fundamental low-level vision task for many applications such as autonomous vehicle navigation, object manipulation and robot vision. The research activity on stereo algorithms has been particularly active in the last decade so that surveys [1], [2], [3], [4] have been proposed in order to better assess the relative performance of the various proposals. In particular, [1] proposes a taxonomy of stereo algorithms and an experimental evaluation of the whole class of stereo methods, in particular classifying them into two main categories: *local* and *global*. Differently, [2] focuses on local algorithms and presents an experimental evaluation among those methods that are suited to real-time implementation on a GPU. In addition, [3] presents a comparison between different matching functions aimed at robustness to photometric distortions and noise, while [4] proposes a classification and evaluation of cost aggregation strategies for local stereo algorithms.

Stereo applications within the aforementioned scenarios require stereo range maps to be computed at a high frame-rate, usually with real-time ($\sim$ 25 fps) or near real-time constraints ($5 \sim 10$ fps). Furthermore, given the "mobile" nature of those applications, often the processing platform is characterized by an embedded, low-power architecture with limited memory. The limited memory footprint is particularly relevant when dealing with FPGAs. Moreover, the typical embedded computing platforms deployed in these scenarios often lack of power-hungry GPUs and, sometimes, of floating-point units. Unfortunately, despite the significant number of stereo algorithms recently proposed in literature, most methods can not satisfy the efficiency and memory footprint [1], [2], [4] requirements that characterize the addressed scenarios. Hence, a first contribution of this paper is an accurate selection of stereo algorithms showing fast performance as well as limited requirements in terms of memory. Then, we propose a comparison among the selected methods, including both global and local approaches, which is lacking in literature. In particular, since we implemented all the evaluated methods following predetermined criteria and by using the same code structure, we can perform a quantitative experimental comparison aimed at highlighting the best performing algorithms not only in terms of accuracy but also of efficiency. Furthermore, in addition to executing the comparison on a standard benchmarking dataset [5] (acquired in a controlled environment with high quality cameras), we also propose a novel dataset acquired in an uncontrolled environment and with an off-the-shelf stereo camera. Thanks to the use of this dataset it was possible to evaluate the selected algorithms in more realistic working conditions.

In Section II we revise stereo literature in order to highlight the methods suited to the addressed application scenarios. The details of the proposed experimental evaluation are described in Section III, while experimental results are shown in Section IV.

II. Efficient and low-memory stereo algorithms

As introduced in Sec. I, the proposed evaluation is aimed at comparing the performance of efficient and low-memory state-of-the-art stereo algorithms, which indeed are potentially suited to execution on embedded architectures. For this reason, the selection of algorithms among the state of the art has been made keeping in mind the following constraints:

- computational efficiency: the algorithm must be particularly efficient
- limited memory footprint: the algorithm should have small memory requirements. In particular, given the current resolutions available on off-the-shelf stereo cameras (e.g. $640x480$, $1280x960$) the memory footprint should be at most of the order of $O(WD)$, with W being one side of the image(in particular, the width) and D the disparity range, so as to include algorithms suited to mapping on FPGAs.

Given a stereo pair composed of a reference image I_{ref} and of a target image I_{trg}, both of size $W \times H$, the simplest and most popular stereo algorithm (hereinafter *standard*) algorithm) computes for each point $p \in I_{ref}$ a similarity (or dissimilarity) measure over a squared window W_r of radius r centered on p and on all possible disparity candidates belonging to I_{trg} (in the following we will refer to S as the side of the window, that is $S = 2r + 1$). The disparity that maximizes (minimizes) the similarity (dissimilarity) measure is that associated with the reference point (*winner-take-all approach*). The most popular measure due to its efficiency and simplicity is the *Sum of Absolute Differences* (SAD). We will refer to this algorithm as *SAD*. In addition, we will use the SAD as the dissimilarity measure for all the evaluated algorithms. Nevertheless, since the SAD measure is not robust to photometric distortions that can arise among the two stereo images, more robust measures are usually deployed, one of them being the Normalized Cross-Correlation (NCC). In order to evaluate the performance of the two measures we also test the *standard* stereo algorithm deploying the NCC measure: we will refer to this approach as *NCC*.

The *standard* algorithm is very efficient: in particular, thanks to the use of incremental techniques such as Box Filtering [6] and of SIMD optimization, implementations of this method have been made that can run in real-time, such as e.g. the stereo algorithm included in the OpenCV library [7]. Nevertheless, due to its inherent simplicity, this algorithm can not deal satisfactorily with situations such as low-textured areas, depth borders, occluded areas. Several approaches have been proposed with the aim of improving the performance of the *standard* algorithm. As already mentioned, according to [1] they can be divided between *local* and *global* methods.

As for the first category, most state-of-the-art local methods aim at achieving improved accuracy by means of a variable support [4], i.e. by adaptively aggregating supporting pixels based on the local neighborhood instead of employing a fixed, squared window as in the *standard* approach. As for the state-of-the-art approaches belonging to this category, and as shown in [4], the most efficient algorithms are those that select, for each reference pixel, the support within a set of rectangular windows, or as a sum of several squared window. Approaches based on unconstrained support shapes [8], [9] or adaptive weights [10], [11], [12], though yielding generally higher accuracy, are far from achieving real-time or near-real-time performance. Some of these approaches are also particularly memory-demanding [4] compared to the aforementioned constraints. Based on these considerations, we have included in our comparison the following methods as representative of local approaches :

- *Shiftable Window* (SW): this approach, derived from the method proposed in [13], considers a set of possible local supports as all squared windows W_r centered over the horizontal range $[x - r; x + r]$, x being the horizontal coordinate of the current reference point p. The current support is selected as the window yielding the lowest SAD score among the considered set.
- *Three Windows* (3W): this approach, derived from the method proposed in [14], [15], selects the current support as the squared window yielding the lowest SAD score among three different horizontal position: $x, x - r, x + r$.
- *Multiple Windows* (MW): this approach, derived from the method proposed in [16], defines the current support as the squared window centered on x plus the squared window yielding the lower SAD among that centered on $x - r$ and that centered on $x + r$.

As it can be inferred from their description, both SW and MW represent simplified approaches compared to their respective original versions [13], [16]. This is to allow for an implementation that, at the same time, could be efficient (i.e. deploying incremental techniques [6]) and could deal with the constraint of $O(WD)$ in terms of memory requirements.

For what concerns the category of global methods, most solutions rely on formulations based on a global energy term that includes a smoothness constraint for better handling low-textured areas. Typical algorithms employed to determine approximate minima of the energy term are Graph Cuts [17] and Belief Propagation [18]. Though approaches aimed at decreasing the computational burden of global methods have been proposed [19], they are still far from achieving near-real-time efficiency; in addition they have a notable memory footprint (on the order of $O(WHD)$). Instead, approaches based on Scanline Optimization (SO) and Dynamic Programming (DP) trade off accuracy for efficiency by minimizing a global energy term over image scanlines. For this reason, we have included in our comparison the standard DP algorithm [1]. In addition, we also included a two-pass SO algorithm similar to that proposed in [20], which is based on the minimization of an aggregated cost computed by applying SO over two scanlines, one horizontal from left to right, and the other one horizontal gain but from right to left. The use of more sophisticated schemes based on DP and SO (e.g. [21], [22]) has not been included given their more expensive memory requirements (of the order of $O(WHD)$).

An additional set of methods is represented by those deploying an adaptive support to compute a local aggregation cost in conjunction with a global approach embedding a smoothness constraint to improve the local cost [23], [20], [11], [24]. Even though the majority of these methods is particularly expensive in terms of computations and/or memory requirements [23], [20], [11], we have included two methods derived from [24]. The first, referred to as AWDP, deploys, in the local aggregation step, a simplification of method proposed in [10] by computing the cost only over vertical columns [24], then it deploys DP to the purpose of energy minimization. The second, referred to as AWSO, uses the same aggregated cost function as AWDP and employs a two-pass SO instead of DP for energy minimization.

Overall, we have selected 9 methods to be evaluated in the proposed comparison. All implementations have been made in C using the same code structure in order to conduct a fair comparison also in terms of measured efficiency. In particular, the computation of the local cost (SAD, NCC) for all methods

Alg.	Nops	Memory
SAD	WHD	WD
NCC	WHD	W(D+2)
SW	2WHD	2WD
3W	2WHD	2WD
MW	2WHD	2WD
DP	2WHD	4WD
SO	3WHD	4WD
AWSO	WHD(S+2)	3WD
AWDP	WHD(S+1)	3WD

TABLE I
NUMBER OF ELEMENTARY OPERATIONS AND MEMORY REQUIREMENTS OF THE EVALUATED ALGORITHMS.

but AWSO and AWDP is performed over a squared window by means of incremental techniques, namely by means of a running Box-Filtering approach [6] that has a memory footprint of the order of $O(WD)$. As for AWSO and AWDP, even though the box-filtering approach can not be deployed for the computation of the local cost, Look-Up Tables are deployed to re-use calculations and thus speed-up the computation. Moreover, for fairness of comparison we have not included additional optimization such as SIMD optimization or GPU porting.

Table I summarizes the computational complexity and memory requirements of the evaluated algorithms. In particular, it shows the number of elementary operations and the memory footprint of each algorithm. As for the latter aspect, non-predominant memory terms (e.g. sized as D or less) have been neglected. As it can be seen from the table, the most efficient algorithms appear to be SAD and NCC, while the less efficient AWSO and AWDP. This will be confirmed by the results concerning measured execution times shown in Section IV, with the exception of the NCC algorithm, since Table I does not take into consideration the fact that this algorithm is mostly characterized by floating point operations (including multiplications). Furthermore, from the Table it can observed that the most efficient algorithm in terms of memory footprint is the SAD algorithm, while the most memory-expensive ones are SO and DP.

III. THE PROPOSED EVALUATION FRAMEWORK

In this section we describe the proposed performance evaluation framework. In particular, we present datasets and performance metrics and discuss parameter selection.

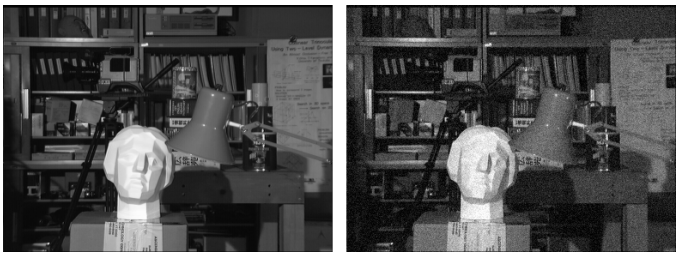

Fig. 1. Left image of *Tsukuba* before (left) and after addition of Gaussian noise for the creation of *Noise* dataset.

A. Dataset

The stereo algorithms are evaluated on 3 grayscale datasets. The first, recalled as *Middlebury*, refers to the standard stereo benchmarking website [5] and it is composed by the 4 stereo pairs, available online, named *Tsukuba*, *Venus*, *Cones*, *Teddy*. Since different masks to evaluate the error over different areas of the image are available in this dataset, we have performed our test using the *N.O.* masks (*Non-occluded*, i.e. occluded areas are not evaluated) and the *DISC* mask(*Discontinuities* masks, i.e. only regions around depth borders and without occlusions are evaluated). The mask including also occluded regions (i.e., *ALL*) has not been used since none of the considered algorithms handles explicitly occluded regions. The second dataset is composed by the same 4 stereo-pairs of dataset *Middlebury* with the addition of synthetic Gaussian noise with $\sigma = 1$, and it is recalled as *Noise*. Fig. 1 shows the left image of stereo-pair *Tsukuba* before and after the addition of Gaussian noise.

It is worth noting that the Middlebury dataset includes images acquired under particularly good illumination conditions and with high-quality camera devices: this results in particularly high image quality. Nevertheless, we think that it is important to evaluate the stereo algorithms also under more realistic working conditions. For this reason, we have created a third stereo dataset made out 6 scenes of different objects on a table which have been acquired in our laboratory by means of an off-the-shelf stereo camera and under standard, indoor lighting. We recall it as *Lab*. The images in this dataset have typical characteristics of data of real application scenarios such as uncontrolled ambient light, photometric distortions and artifacts, small defocus in the lenses, non-perfect rectification. For this reason, and only for this specific dataset, all images are pre-processed by means of a common procedure in stereo, that is, subtracting to each pixel the mean value over a x square window centered on the pixel in order to improve the performance of the algorithms (that otherwise would be quite poor). Ground-truth for this dataset has been obtained by means of a technique known as Spacetime Stereo (STS) [25], [26], i.e. for each scene, a sequence of texture-augmented images was acquired by means of a standard light projector and then processed as described in [25], [26]. Despite the highly accurate disparity maps yielded by the STS approach, to avoid possible disparity mistakes due to occluded regions and residual photometric distortions a Left-Right consistency check [27] was applied and a binary mask, analogous to those present in the Middlebury dataset, was produced to discriminate those disparities that survived the consistency check. Differently than in the *Middlebury* and *Noise* datasets, in this case we use only this mask since it is not straightforward to explicitly determine occluded regions and discontinuity regions within these stereo images. Nevertheless, given the properties of the Left-Right consistency check, we can derive that the attained mask is analogous to the *N.O.* mask of the other datasets, for it tends to filter out mainly the pixels belonging to occluded regions. An example of the *Lab* dataset is depicted in Fig. 2, which shows, for two stereo-pairs belonging to the dataset, respectively the

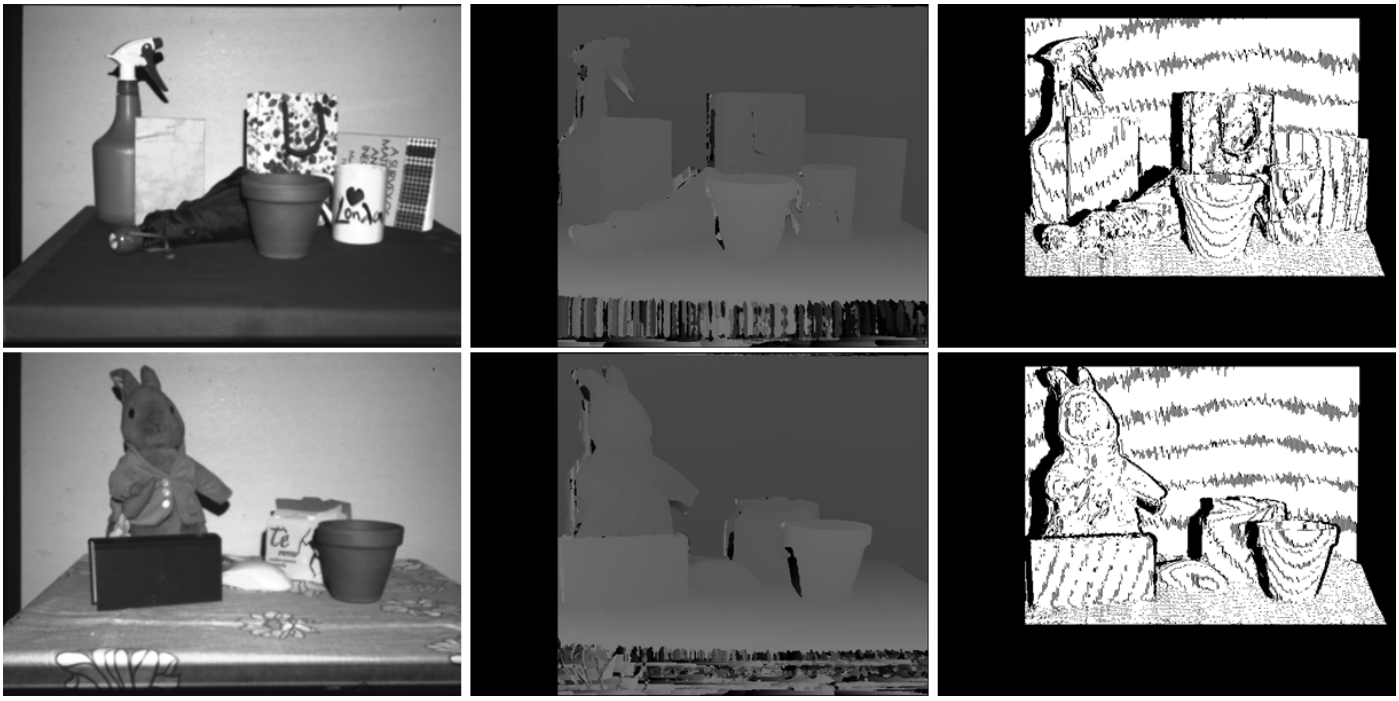

Fig. 2. Two examples from the dataset *Lab*: from left to right, reference image, ground-truth map, binary mask.

reference image, the ground-truth map yielded by STS and the binary mask obtained by means of the Left-Right consistency check. As for the binary masks, it is worth observing the black regions (masked pixels) at occluded regions. This dataset is made publicly available online through our website [1], so that it can be used for further experimental evaluation by the stereo community.

The disparity range D for datasets *Middlebury* and *Noise* it is set to the default values specified in [5], while it is 128 for dataset *Lab*.

B. Performance Metrics

The proposed evaluation aims at comparing the selected stereo algorithms in terms of accuracy and computational efficiency. As for the first criterion, we use a standard methodology to evaluate the accuracy of stereo algorithms [5], i.e. we compute the $\%$ of correct disparities - within a given binary mask - by comparing the disparity map yielded by an algorithm to the ground truth disparity map. The error threshold to determine if a disparity value is correct is set to 1. Also, since all methods compute a local cost over rectangular windows and we do not explicitly handle image border areas, the disparity map (and hence accuracy) is not computed at image bordering regions.

As for computational efficiency, we compare algorithm based on the Million of Disparities computed per second (MDS):

$$MDS = \frac{W_{eff} \cdot H_{eff} \cdot D}{T} \qquad (1)$$

[1] http://vision.deis.unibo.it/fede/ds-stereo-lab.html

where W_{eff}, H_{eff} are the effective width and height of the current stereo-pair (i.e. excluding image bordering regions) and T is the measured execution time on a given stereo-pair. All tests have been executed on an AMD Turion X2 CPU running at 1.6 GHz and with 1 GB RAM. Algorithms are ran multiple times (i.e. 20) then the median execution time is extracted in order to produce more stable MDS evaluations.

Algorithm	Parameters
SAD	r
NCC	r
SW	r
3W	r
MW	r
DP	r, s
SO	r, s
AWSO	r, s, γ_c, γ_s
AWDP	r, s, γ_c, γ_s

TABLE II
TUNED PARAMETERS FOR EACH ALGORITHM.

C. Parameter selection

We have performed a different tuning of the algorithms' parameters for each dataset, for a total of 3 different tuning procedures. Parameters were selected as those maximizing the accuracy of each algorithm: this is due to the fact that, given the current implementations of the algorithms (i.e. the use of incremental techniques) accuracy is more dependent on parameter values than efficiency. Also, since, as discussed, bordering regions are excluded from disparity computation, and in order not to reduce too much the area of the final

	MID-N.O.		MID-DISC		NOISE-N.O.		NOISE-DISC		LAB		AVG.	
	ACC	MDS	ACC	MDS	ACC	MDS	ACC	MDS	ACC	MDS	ACC	MDS
AWSO	95.1	2.9	84.0	2.9	87.3	2.9	67.7	3.0	82.3	3.5	83.3	3.0
AWDP	95.4	3.6	84.4	3.6	87.3	3.1	68.3	2.8	80.5	4.1	83.2	3.4
DP	94.2	30.8	82.0	31.1	84.7	31.7	65.4	31.1	82.2	31.4	81.7	31.2
SO	94.2	19.7	82.2	19.7	85.9	21.9	61.4	21.5	82.4	22.3	81.2	21.0
SW	90.5	9.1	74.9	9.1	76.5	8.6	65.5	8.5	66.5	8.8	74.8	8.8
3W	90.4	45.7	74.4	45.7	76.4	37.1	65.2	42.4	66.8	47.3	74.6	43.7
MW	89.8	49.9	67.9	50.9	78.4	49.0	61.5	48.9	69.4	53.7	73.4	50.5
SAD	88.6	68.9	63.6	68.5	75.8	58.7	58.9	66.9	64.4	75.3	70.3	67.7
NCC	89.8	19.7	63.7	19.7	74.4	18.8	57.0	20.1	65.9	21.5	70.1	20.0

TABLE III
VALUES OF ACCURACY (%) AND MDS YIELDED BY THE EVALUATED ALGORITHMS AVERAGED OVER ALL STEREO-PAIR OF EACH DATASET. THE OVERALL VALUES OF ACCURACY (%) AND MDS AVERAGED OVER ALL DATASETS ARE SHOWN IN THE LAST TWO COLUMNS.

	MID-N.O.		MID-DISC		NOISE-N.O.		NOISE-DISC		LAB		AVG.		AVG.
	ACC	MDS	ACC	MDS	ACC	MDS	ACC	MDS	ACC	MDS	ACC	MDS	RANK
DP	3	4	4	4	4	4	4	4	3	4	3.6	4.0	3.8
MW	8	2	7	2	5	2	6	2	5	2	6.2	2.0	4.1
SO	4	6	3	5	3	5	7	5	1	5	3.6	5.2	4.4
3W	6	3	6	3	7	3	5	3	6	3	6.0	3.0	4.5
SAD	9	1	9	1	8	1	8	1	9	1	8.6	1.0	4.8
AWDP	1	8	1	8	2	8	2	8	4	8	2.0	8.0	5.0
AWSO	2	9	2	9	1	9	1	9	2	9	1.6	9.0	5.3
SW	5	7	5	7	6	7	3	7	7	7	5.2	7.0	6.1
NCC	7	5	8	6	9	6	9	6	8	6	8.2	5.8	7.0

TABLE IV
RANKINGS CONCERNING THE VALUES OF ACCURACY (%) AND MDS AVERAGED OVER EACH DATASET. THE AVERAGE RANKINGS AVERAGED OVER ALL DATASETS ARE SHOWN IN THE THIRD-LAST AND SECOND-LAST COLUMN. THE LAST COLUMN SHOWS THE OVERALL MIXED ACCURACY-MDS RANKING.

disparity map, we have imposed an upper bound on the radius of the rectangular window deployed by each algorithm for the computation of the local cost (i.e. $r_{max} = 8$). This also allows us to determine a fixed area on which to compute the disparity map that does not depend on the particular radius value deployed by each algorithm.

The tuned parameters of each algorithm are reported in Table II. In the table, r represents for all algorithms the radius of the squared window on which the local cost is computed, with the exception of AWSO and AWDP for which it represents the size of the column on which the local cost is computed; s is the smoothness penalty deployed by algorithms embedding the smoothness constraint and γ_c, γ_s are respectively the color and spatial bandwidths [10] used by AWSO and AWDP.

IV. EXPERIMENTAL RESULTS

This section presents the results concerning the proposed experimental evaluation. Table III shows, for each dataset the % Accuracy and the MDS averaged over all stereo-pairs belonging to that dataset. In particular, for both *Middlebury* and *Noise* the table shows the results concerning the 2 considered binary masks (*N.O.* and *DISC*). In addition, it also shows, in the last two columns, the values of Accuracy and MDS averaged over all datasets. Besides, following the methodology adopted by the standard benchmarking website for stereo algorithms [5], Table IV shows, for each dataset, the ranking of each algorithm according to the % Accuracy and MDS metrics. Also in this case, the third-last and second-last columns report the overall Accuracy and MDS rankings averaged over all datasets. Finally, inspired by the methodology proposed in [4], the last column of the table highlights the trade-off ranking obtained by averaging the overall Accuracy and the overall MDS rankings.

By analyzing the % Accuracy values and the associated rankings reported by the two tables, we can see that global methods deploying a smoothness constraint (SO, DP) as well as variable support-based local methods (3W, SW, MW) are all able to improve the performance of the standard algorithm (SAD), even if a different, more robust local measure is used (NCC). In particular, global methods perform overall better than those based on a variable support, with the exception of dataset *Noise*, *DISC* mask, where DP, SW and 3W outperform SO and MW. In addition to these considerations, it is also clear that the use of a variable support together with a global approach yields the best performance over the *Middlebury* and the *Noise* datasets. Instead, for what concern dataset *Lab*, the use of a local support only slightly improves the performance compared to standard global approaches, for methods AWSO and AWDP yielding performance similar to SO and DP. Overall, the most accurate algorithms are AWDP and AWSO, followed by DP and SO. As for the NCC algorithm, it is clear that the use of a more robust matching measure does not pay in terms of improved accuracy, not even on the *Lab* dataset given the use of the pre-processing stage. Additional experimental results, not shown here for lack of space, showed instead that without the pre-processing stage the % Accuracy

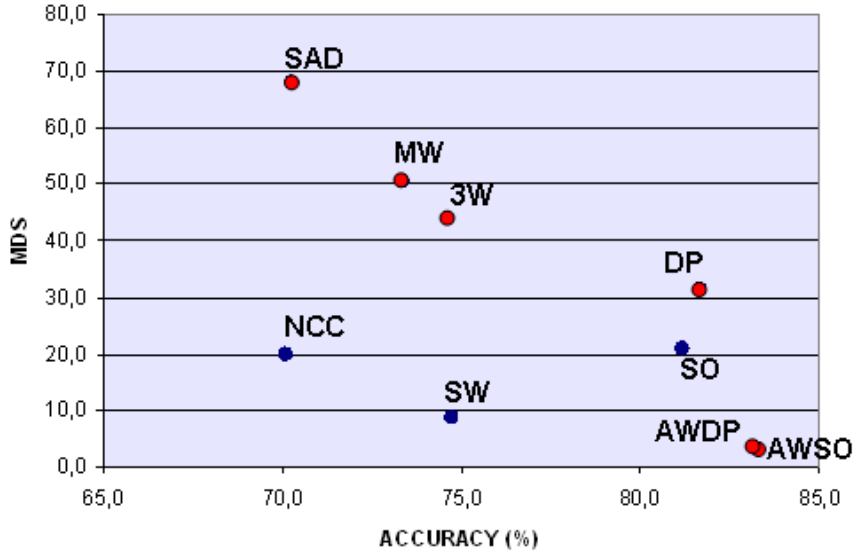

Fig. 3. Performance of the evaluated algorithms under both criteria: Accuracy (along x-axis) and MDS (along y-axis). *Pareto* optima are marked with a red circle.

yielded by NCC would be almost 2.5 times higher than that achieved by the SAD algorithm. Finally, for what concerns the various datasets, the most evident benefits brought in by the evaluated methods with regards to the standard approach is on dataset *Middlebury*, *DISC* mask, and on dataset *Lab*: e.g., the improvement in $\%$ Accuracy brought in by AWSO with regards to SAD is, respectively, 20.4% and 17.9%. Also, it is worth pointing out that the algorithms perform overall significantly worse on the image data acquired in the more realistic conditions created in our lab than on the standard benchmark dataset: this is particularly evident if we compare the accuracy reported by the evaluated algorithms on *Middlebury* dataset, *N.O.* mask with those reported on *Lab* dataset. The results also show that the images belonging to the *Lab* dataset are more challenging than those of the Middlebury website corrupted by artificial noise (see the results for *Noise* dataset, *N.O.* mask).

As far as computational efficiency is concerned, Table III shows a dramatic difference - i.e. one order of magnitude - in the measured MDS values between the fastest algorithms (SAD, 3W, MW) and the less efficient ones (AWSO, AWDP, SW). As for variable support-based local algorithms, there is a notable difference in efficiency between 3W and MW compared to SW, the former running approximatively one third slower than the SAD algorithm, the latter being almost 10 times slower. As for global methods, DP and SO run overall respectively 2.4 and 3.4 times slower than the SAD algorithm, DP being approximately 1.5 times faster than SO. This is due to the higher number of operations involved in the double scan performed by SO compared to the single scan carried out by DP. Finally, the use of an adaptive weight-based aggregation cost in methods AWSO and AWDP notably slows down the execution time of these algorithms, rendering them the slowest ones (i.e. approximately 20 times slower than SAD). Finally, the use of a more robust matching measure (NCC) significantly affects efficiency, yielding an algorithm that runs more than 3 times slower than SAD, particularly due to the floating-point multiplications involved in the computation of the NCC term. Overall, by looking at the last column in Table IV we can observe that the algorithm maximizing the accuracy-speed trade-off is DP, followed by MW and SO. The worst algorithms under this criterion are SW and NCC.

In Fig. 3 we also present the obtained experimental results by plotting the overall values of $\%$ Accuracy and MDS as a two-dimensional scatter graph, with each scatter dot representing a different algorithm. Moreover, we also highlight with red dots those algorithms that represent Pareto-optima with regards to the set of evaluated algorithms. Overall, we can point out that, as intuitively expected, the algorithms performing more favorably in terms of accuracy-efficiency trade-off are also those representing the Pareto-optima in the Figure. It is also important to note that overall, while the difference between the slowest algorithms and the most efficient ones is around one order of magnitude, the most accurate ones are approximately 1.2 times more accurate compared to the less accurate ones. This means that, generally speaking, to increase the performance of stereo algorithms in terms of accuracy we have to really pay a lot in terms of efficiency. For example, DP is almost 1.2 times more accurate than the SAD algorithm, but also 2.2 times slower. Analogously, AWSO is 1.3 times more accurate than the SAD algorithm, but 22.6 times slower.

Finally, we show some qualitative results by displaying, in Fig. 4, the disparity map yielded by each algorithm on Scene $\#1$of dataset *Lab*. From the Figure it is possible to assess the improved accuracy yielded by methods DP, SO,

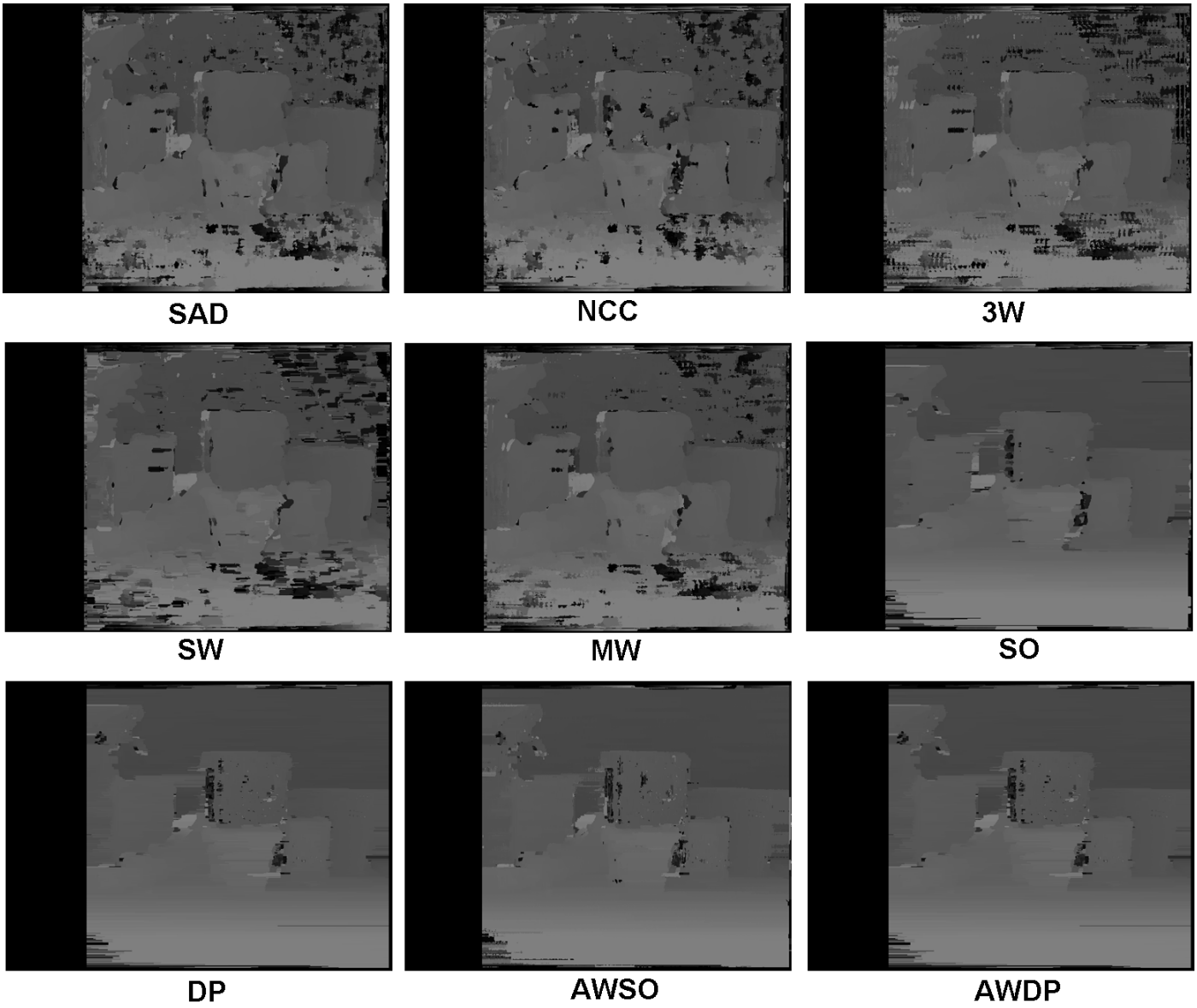

Fig. 4. Disparity maps yielded by the evaluated algorithms on Scene #1 of dataset *Lab*. Ground-truth and reference image are shown in Fig. 2

AWSO, AWDP compared to others, especially for what concerns handling of low-textured regions. Nevertheless, typical "streaking" effects due to the use of the smoothness penalty across scanlines are also evident in the disparity maps obtained by of these global methods.

V. Conclusions

We have presented an experimental evaluation of a set of stereo algorithms that have been selected based on their being efficient and with low memory requirements. The comparison, carried out on a standard benchmarking dataset as well as on a novel, more challenging dataset characterized by particularly realistic working conditions, has highlighted that global algorithms such as SO and DP allow improved accuracy but run overall slower than variable-support local algorithms such as 3W and MW. The use of the proposed dataset also highlighted how overall performance of stereo algorithms deteriorate under more realistic working conditions. Another consideration is that at the state of the art, relatively small improvements in accuracy generally require notable additional computational costs. As for future work, we plan to extend our evaluation to the case of texture-augmented stereo-pair, which is a popular solution adopted to cope with the presence of large low-textured regions in images.

References

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. Jour. Computer Vision*, vol. 47, no. 1/2/3, pp. 7–42, 2002.

[2] M. Gong, R. Yang, W. Liang, and M. Gong, "A performance study on different cost aggregation approaches used in real-time stereo matching," *Int. Journal Computer Vision*, vol. 75, no. 2, pp. 283–296, 2007.

[3] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR 2007)*, vol. 1, 2007, pp. 1–8.
[4] F. Tombari, S. Mattoccia, L. Di Stefano, and E. Addimanda, "Classification and evaluation of cost aggregation methods for stereo correspondence," in *Proc. CVPR*, 2008.
[5] "Middlebury stereo evaluation," http://vision.middlebury.edu/stereo.
[6] M. Mc Donnel, "Box-filtering techniques," *Computer Graphics and Image Processing*, vol. 17, pp. 65–70, 1981.
[7] "Opencv library," http://opencv.willowgarage.com.
[8] Y. Boykov, O. Veksler, and R. Zabih, "A variable window approach to early vision," *IEEE Trans. PAMI*, vol. 20, no. 12, pp. 1283–1294, 1998.
[9] M. Gerrits and P. Bekaert, "Local stereo matching with segmentation-based outlier rejection," in *Proc. Canadian Conf. on Computer and Robot Vision (CRV 2006)*, 2006, pp. 66–66.
[10] K. Yoon and I. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Trans. PAMI*, vol. 28, no. 4, pp. 650–656, 2006.
[11] S. Mattoccia, F. Tombari, and L. Di Stefano, "Stereo vision enabling precise border localization within a scanline optimization framework," in *Proc. Asian Conf. on Computer Vision (ACCV 2007)*, 2007, pp. 517–527.
[12] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann, "Local stereo matching using geodesic support weights," in *Proc. Int. Conf. on Image Processing (ICIP 09)*, 2009.
[13] A. Bobick and S. Intille, "Large occlusion stereo," *Int. Journal Computer Vision*, vol. 33, no. 3, pp. 181–200, 1999.
[14] J. Zhao and J. Katupitiya, "A fast stereo vision algorithm with improved performance at object borders," in *Proc. Conf. on Int. Robots and Systems (IROS)*, 2006, pp. 5209–5214.
[15] L. Sorgi and A. Neri, "Bidirectional dynamic programming for stereo matching," in *Proc. Int. Conf. on Image Processing (ICIP)*, 2006, pp. 1013–1016.
[16] H. Hirschmuller, P. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *Int. Journ. of Computer Vision*, vol. 47, pp. 1–3, 2002.
[17] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions via graph cuts," in *Proc. Int. Conf. Computer Vision (ICCV 2001)*, vol. 2, 2001, pp. 508–515.
[18] J. Sun, H. Shum, and N. Zheng, "Stereo matching using belief propagation," *IEEE Trans. PAMI*, vol. 25, no. 7, pp. 787–800, 2003.
[19] P. Felzenszwalb and D. Huttenlocher, "Efficient belief propagation for early vision," in *Proc. IEEE CVPR*, vol. 1, 2004, pp. 261–268.
[20] J. Kim, K. Lee, B. Choi, and S. Lee, "A dense stereo matching using two-pass dynamic programming with generalized ground control points," in *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR 2005)*, 2005, pp. 1075–1082.
[21] G. M. Bleyer, M, "Simple but effective tree structures for dynamic programming-based stereo matching," in *Proc. Int. Conf. on Computer Vision Theory and Applications (VISAPP)*, vol. 2, 2008.
[22] H. Hirschmuller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *Proc. Conf. on Computer Vision and Pattern recognition (CVPR 2005)*, vol. 2, 2005, pp. 807–814.
[23] K. Yoon and I. Kweon, "Stereo matching with symmetric cost functions," in *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR 2006)*, vol. 2, 2006, pp. 2371 – 2377.
[24] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High-quality real-time stereo using adaptive cost aggregation and dynamic programming," in *Proc. 3rd Int. Symposium 3D Data Processing, Visualization and Transmission (3DPVT'06)*, 2006, pp. 798–805.
[25] J. Davis, D. Nehab, R. Ramamoorthi, and S. Rusinkiewicz, "Spacetime stereo: a unifying framework dor depth from triangulation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, February 2005.
[26] L. Zhang, B. Curless, and S. Seitz, "Spacetime stereo: shape recovery for dynamic scenes," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2003.
[27] P. Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features," *Machine Vision and Applications*, vol. 6, no. 1, pp. 35–49, 1993.