A Comparative Study of Stereovision Algorithms

Elena Bebeşelea-Sterp
NTT DATA ROMANIA
Sibiu, Romania

Raluca Brad
Faculty of Engineering
Lucian Blaga University of Sibiu
Sibiu, Romania

Remus Brad
Faculty of Engineering
Lucian Blaga University of Sibiu
Sibiu, Romania

***Abstract*—Stereo vision has been and continues to be one of the most researched domains of computer vision, having many applications, among them, allowing the depth extraction of a scene. This paper provides a comparative study of stereo vision and matching algorithms, used to solve the correspondence problem. The study of matching algorithms was followed by experiments on the Middlebury benchmarks. The tests focused on a comparison of 6 stereovision methods. In order to assess the performance, RMS and some statistics related were computed. In order to emphasize the advantages of each stereo algorithm considered, two-frame methods have been employed, both local and global. The experiments conducted have shown that the best results are obtained by Graph Cuts. Unfortunately, this has a higher computational cost. If high quality is not an issue in applications, local methods provide reasonable results within a much lower time-frame and offer the possibility of parallel implementations.**

Keywords*—Stereo vision; disparity; correspondence; comparative study; middlebury benchmark

I. Introduction

Stereovision is an area of computer vision focusing on the extraction of 3D information from digital images. The most researched aspect of this field is stereo matching: given two or more images as input, matching pixels have to be found across all images so that their 2D positions can be converted into 3D depths, producing as result a 3D estimation of the scene. As many other breakthrough ideas in computer science, stereovision is strongly related to a biological concept, namely stereopsis, which is the impression of depth that is perceived when a scene is viewed by someone with both eyes and normal binocular vision [1]. By aid of stereoscopic vision, we can perceive the surroundings in relation with our bodies and detect objects that are moving towards or away from us. While the entire process seems an easy task for us and other biological systems, the same does not apply to computer systems. Finding the correspondences across images is a challenging task. The earliest stereo matching algorithms were developed in the field of photogrammetry for automatically constructing topographic elevation maps from overlapping aerial images [2].

Stereo matching has been one of the most studied topics, starting with the work of D. Marr and T. Poggio [3] which focuses on human stereopsis. Lane and Thacker's study on stereo matching [4] presents a dozen algorithms from 1973 up to 1992, but no comparison between them was made. Another important study is the one done by Scharstein and Szeliski [5], in which they have compared several algorithms and their performance based on different metrics.

Stereo vision has a wide range of applications nowadays, especially domains in which realistic object models are needed. Depending on how important the processing time is, these applications can be classified into two categories: static scene description and dynamic scene description. For the first category, accuracy is of higher importance compared to the processing time. Usually, image pairs are acquired by means of a special device and reconstructed afterwards for cartography, crime scene reconstruction, car crashes scene reconstruction, 3D models for architecture. In dynamic scene description real-time processing of the data is critical. Of course, a certain level of accuracy must be fulfilled. Possible applications are obstacle (e.g. pedestrians) avoidance, autonomous navigation in which an internal map is continuously updated, height estimation, etc.

This paper aims to review different stereo methods available and compare several stereo matching algorithms across sets of images. Section 2 describes the fundamental of stereo correspondence that makes possible the depth estimation, and discuss some of the stereo methods available nowadays, classifying them into local, global and semi-global algorithms, while Section 3 analyses in more detail some referenced algorithms considered for experiments. The results are discussed in Section 4.

II. A Classification of Stereovision Algorithms

This section reviews some of the stereo correspondence algorithms that have been proposed over the last decades. These can be classified depending on multiple criteria such as the method which assign disparities to pixels, occlusion handling, color usage, matching cost computation etc.

A. Local Methods

Local methods assign disparities depending on the information provided by the neighboring pixels, usually fast and yield good results. One broad category of local methods is represented by the block matching algorithms, which try to find a correspondence for a point in the reference image by comparing a small region surrounding it with small regions in the target image. This region is reduced to a single line, called the epipolar line. Block matching algorithms are used not only in stereo vision, but also in visual tracking and video compression.

Among the first techniques which appeared is box filtering. This involves replacing each pixel of an image with the average in a box and it is an efficient general purpose tool for image processing. In [6] a procedure for mean calculation has been proposed. The advantage of box filtering is the speed, cumulating the sum of pixel values along rows and columns.

This way, the sum of a rectangle is computed in a linear time, independent of its size.

Another algorithm that achieves high speed and reasonable quality is the one proposed by Mühlmann et al. [7]. It makes use of rectified images and can be used by other, more sophisticated, algorithms that need an initial guess at the disparity map. In order to eliminate false matches it makes use of the left-right consistency check and uniqueness validation.

A new block matching algorithm has been proposed in [8] and even though the focus of this paper was on video compression, the algorithm can be applied also to stereo vision. This is based on a winner-update strategy which uses a lower bound list of the matching error to determine the temporary winner.

Efficient usage of color information can improve results, therefore, some of the best stereo correspondence algorithms use color segmentation techniques. One of the most popular color segmentation algorithms has been proposed in [9], based on the mean-shift algorithm which dates back to 1975, but extended on computer vision only later on. The algorithm proposed in [10] is one of the top performing algorithms in the Middlebury classification [11] and it makes use of this segmentation technique. This is based on inter-regional cooperative optimization. The algorithm uses regions as matching primitives. More exactly, they use color statistics of the regions and constraints on smoothness and occlusion between adjacent regions. A similar algorithm, that is currently one place above the before mentioned [10], is the algorithm proposed by Klaus et al. in [12]. Firstly, homogenous regions are extracted from the reference image, also with the method [9]. Secondly, local windows-based matching is applied using a self-adapting dissimilarity measure that combines SAD and a gradient based measure. Using the reliable correspondences, a set of disparity planes are derived.

Yoon K.J. and Kweon [13] have proposed a new window-based method that uses varying support weights of the pixels. These are adjusted in a given support-window depending on the color similarity and geometric proximity to the reference pixel in order to reduce the image ambiguity (repetitive textures, image noise) and to obtain good results also in homogeneous regions. The algorithm in [14] is a local stereo correspondence algorithm which employs segmentation cue and which has two main steps: initial matching and disparity estimation. The initial matching is performed with the contrast context histogram descriptor and two-pass cost aggregation with segmentation-based adaptive support weight. The disparity estimation has two sub-steps: narrow occlusion handling and multi-directional weighted least-squares fitting for large occlusion areas.

A novel similarity measure called DSM (Distinctive Similarity Measure) was introduced in [15] to resolve the point ambiguity problem, based on the idea that the distinctiveness is the appropriate criterion for feature selection under point ambiguity. DSM is based on the distinctiveness of image points and the dissimilarity between them, both of which are closely related to the local appearances of points. The first one is related to the probability of a mismatch and the second one to the probability of a good match.

Another class of local algorithms is represented by the gradient-based methods, also known as optical flow. In this case, the matching costs are insensitive to bias and camera noise. These methods determine the disparities between two images by formulating a differential equation relating motion and image brightness. In order to do this, the assumption is made that the image brightness of a point in the scene is constant between the two views [16].

One of the earliest methods for optical flow estimation is the one developed by Kanade and Lucas [17]. This solves the flow equations for all the pixels in the neighborhood using the least squares criterion. However it cannot provide flow information inside regions that are uniform. This method can be used in many applications of image registration, including stereo vision. More similar information can be found in [18].

A newer algorithm from the same class is the one proposed by Zhou and Boulanger [19]. This is based on relative gradients in order to eliminate the radiometric variance. Most stereo matching methods assume that the object surface is Lambertian, meaning that the color for every point in the scene will be constant in the views captured by two separate cameras. However, to most real-world objects this does not apply, reflecting light that is view dependent. The algorithm is able to deal with both view dependent and independent colors and both color and gray scale images.

Block matching and gradient-based methods are sensitive to depth discontinuities, since the region of support near a discontinuity contains points from more than one depth. These methods are also sensitive to regions of uniform texture [16]. Another class of algorithms that aims to overcome these drawbacks is represented by the feature-matching algorithms, which limit the support region to reliable features in the image. Because they produce sparse output and the demand nowadays seeks dense output, this class of algorithms is not given as much attention as in the past. An early review of such algorithms can be found in [20], proving how popular these were. Feature-matching algorithms can be divided into two subclasses: hierarchical and segmentation-based.

One example of a hierarchical feature-matching algorithm is the one in [21] which groups lines into complex structures. This method exploits four types of features: lines, vertices, edges and surfaces. As mentioned before, another approach is to first segment the images and only afterwards to match the segmented regions. Most of the existing algorithms assume that all the surfaces are parallel to the image plane, which is not actually the case. The algorithm proposed in [22] tries to solve the correspondence problem in the presence of slanted surfaces by alternating two steps: segmenting the image into non-overlapping regions, each corresponding to a different surface, and finding the affine parameters of the displacement function of each region.

B. Global Methods

Local methods are sensitive due to occlusions and uniform texture. On the other hand, global methods exploit nonlocal constraints in order to achieve more accurate results. However, accuracy comes with a trade-off, an increased computational complexity.

One of the most common global correspondence methods is based on dynamic programming. This is a fair-trade-off between the quality of the matches and the complexity of computations, which is decreased by decomposing the optimization problems into smaller and simpler sub problems. The disadvantage of DP is the possibility that local errors may be propagated along a scan line, corrupting also good matches [16]. One of the earliest algorithms which make use of dynamic programming is the one proposed by Ohta and Kanade [23]. This needs a pair of rectified images, this way, finding corresponding points can be done within the same scan lines in the left and right image, also known as the intrascan line search.

An alternative to the traditional search for global matching can be found in [24]. The authors propose a new representation of image scan lines, called intrinsic curves. An intrinsic curve is a path that a set of local image descriptors trace as an image scan line is traversed from left to right. These descriptors are defined by applying operators such as edge and/or corner operators. Another approach is to apply the DP technique to a tree structure instead of individual scan lines [25]. Because in the case of traditional DP algorithms the disparity estimates at a pixel depend only on the disparity estimates of pixels on the same scan line, but is completely independent of the disparity estimates on the other scan lines, these are not truly global optimization methods as the author states. The advantages of this approach are the following: firstly, since a tree structure is connected, the estimate of a disparity at one pixel depends on the disparity estimates at all the other pixels, making it a truly global algorithm and, secondly, a tree contains much more edges of the original grid than the collection of scan lines. Although the results obtained are in the middle range according to the classification in [11], the implementation is suitable for real-time applications.

As mentioned before, traditional approaches that use DP do not implement both horizontal and vertical continuity constraints. Methods like the one in [25] try to improve this aspect while maintaining a more than reasonable computational cost. Still, these do not fully exploit the two constraints. A solution would be to use 2D global optimization techniques like graph cuts, simulated annealing, belief propagation and others.

Graph-cut, also known as min-cut, is an algorithm that finds a globally optimal segmentation solution. Having a graph that can be partitioned into two disjoint sets by simply removing edges connecting the two sub-graphs we can compute the degree of dissimilarity between the two sub-graphs as the total weight of the edges that have been removed. One of the most cited works is the one in [26]. The authors propose an expansion move and swap algorithms that can simultaneously modify labels of large pixel sets.

The algorithm in [27] uses the graph cuts method in conjunction with color segmentation. The authors use the mean-shift algorithm to decompose the image into homogenous regions, based on the fact that large disparity discontinuities only occur on the boundaries of homogenous segments. Afterwards, the disparity plane estimation takes place. Finally, the graph-cut technique is applied to approximate the optimal solution of the energy function. The graph-nodes represent here the homogenous segments, and not the pixels as in most of the approaches.

Another approach is to use belief propagation. Sun et al. [28] formulate the stereo matching problem as a Markov network and solve it using Bayesian BP (Belief Propagation). According to the authors, the Bayesian approach, which tries to solve the stereo correspondence problem by finding a best guess, has many advantages. It can encode several prior constraints such as spatial smoothness, uniqueness and the ordering constraint and it can handle the uncertainties. Also, Bayesian methods model discontinuities and occlusions. Bayesian methods can be classified into two categories, depending on the computational model: DP-based and MRF-based (Markov Random Fields). In this paper, three MRF are used, modeling the spatial interaction with the aid of a smooth field for depth/disparity, a line process for depth discontinuity and a binary process for occlusion. Belief propagation is used to obtain the maximum a posteriori model in order to enhance the stereo results obtained.

Nonlinear diffusion is another class of global methods. One of the main problems in the stereo correspondence problem is finding the optimal window sizes for comparing the image blocks. If the window is too small, a wrong match might be found. On the other hand, if the region is too big, it can no longer be matched entirely because of problems such as occlusion and foreshortening. The paper in [29] does not use fixed-sized windows, but introduces some novel algorithms which find the best support region based on iteratively diffusing support at different disparity hypotheses and which are an alternative to the adaptive windows. One of these algorithms is the membrane algorithm. This sums the neighboring matching costs iteratively and uses an additional term to prevent the support region from growing indefinitely. When a local stopping condition is used, the authors have seen that the algorithm behaves similar to the adaptive window algorithms. Another algorithm proposed is derived from the Bayesian model of stereo matching and results in a nonlinear diffusion process, having an increased performance compared to the standard diffusion.

Beside the above mentioned global methods many other ideas exist [30]. For example, there is a class of methods that do not seek explicit correspondences, used mainly when reconstructing a 3D object from different views of the scene. These model the scene using different objective functions. Furthermore, some algorithms make use of wavelets. Based on the continuous wavelet transform, such a method extracts the redundant information and uses it for the matching process.

C. Semi-Global Methods

Local methods try to find optimal disparities for small image regions which can lead to discontinuities between different regions in the image. On the other hand, global methods try to optimize all the disparities at once, which can offer better results, but at higher computational costs. More recently, a third class of algorithms has been developed, namely the semi-global class, which tries to incorporate the advantage of both groups.

The first semi-global algorithm has been developed by Hirschmüller [31] and used since then in real-time stereo vision and intelligent vehicles applications. In this implementation, mutual information was used as matching cost, because it is insensitive to radiometric differences and models very well image noise. Radiometric differences occur because of the different camera characteristics (vignette effect, different exposure time etc.) and different scene properties (non-lambertian reflection, changes in the light source etc.). Using mutual information, the global radiometric difference is modeled in a joint histogram of corresponding intensities. The matching is done pixel-wise based on the mutual information and approximation of a global smoothness constraint. Beside the good results delivered fast, this approach has the advantage of occlusion detection and sub-pixel accuracy in determining the disparities. Post-processing is possible afterwards in order to clean up the disparity image.

Another algorithm that aims to preserve depth discontinuities and to give good results in low-textured regions is the one in [14]. The authors have proposed three solutions for improving the sub-pixel accuracy. Firstly, they show the benefits of evaluating the disparity space at fractional level. Secondly, they introduce a new constraint, called the gravitational constraint. This assumes that the disparity levels are sorted in the vertical direction and it helps global algorithms to reduce the false matches. Finally, they propose a new algorithm that enforces smoothness at a sub-pixel level.

A more recent approach and an optimization to the original algorithm are proposed in [32]. The original algorithm favors regions of constant disparities because of the two penalties applied by the objective function. This way, a large amount of errors is caused. In order to obtain better results, the authors propose an extension to the algorithm's parameterization, by using individual penalties, depending on the path orientations and on intensity gradients. Furthermore, because the results obtained from one path can be better than the ones obtained by another path with a different orientation, they have introduced weights for each path orientation. Last but not least, the authors extend the original adaptation of penalty depending on the intensity gradient to a more general approach. Due to the high number of parameters, they need to be automatically tuned in order to find the best configuration.

III. The Algorithms Considered in Our Study

This section describes in detail several stereo vision algorithms with the aid of which the correspondence problem is solved. The comparative results obtained by them on the Middlebury benchmark are discussed in the next section.

A. The Problem of Finding Pixel Correspondence

With regard to software implementations, there are two main categories of algorithms that solve the correspondence problem: those that produce sparse output, and the ones that produce dense output. The first category is also known as the feature-based algorithms and they find correspondences by matching sparse sets of image features like edges, corners, line or curve segments, all of which are robust against change of perspective. These methods are inspired from human vision studies and have been very popular at the beginnings of stereo vision, having the great advantage of being feasible for implementation on the hardware available at that time. Investigating only a small subset of pixels, such algorithms are very fast. Moreover, the accuracy is very good, limiting the results to matches with very high certainty. This happens for example in applications where the illumination can vary significantly and edges are the only stable features. The main disadvantage is the sparse output which cannot be always of use in applications. However, sparse 3D reconstructions can be later interpolated.

Even if we gain computational time and accuracy, a lot of time is invested in finding a feature extractor that performs well. One of the most successful and applied methods is using the SIFT (Scale Invariant Feature Transform) detector proposed in [33]. This algorithm extracts interesting points of an object in order to provide a feature description of the object during a training phase and uses them afterwards when recognizing the object in a new image, by comparing each feature from the new image to the features stored in the database based on the Euclidian distance of their feature vectors. Location, scale and orientation are a few of the descriptors that are used for finding the best match.

In order to overcome the disadvantage of the feature-based methods, dense correspondence algorithms have appeared, especially because nowadays computational resources are no longer a big issue. Many contemporary applications such as image-based rendering and modeling demand such a dense output and this approach is more challenging than the previous one, having to solve the correspondence problem in case of image sensor noise, perspective distortions, textureless regions, repetitive structures and textures, reflections, occlusions, photometric variations [34].

In [5], a taxonomy has been proposed for the dense stereo correspondence algorithms. According to the authors, a large set of existing algorithms can be easily constructed from a set of building blocks, steps that all these algorithms perform, as in Fig. 1. Matching cost computation is the first step in extracting the disparity map and it quantifies the similarity between pixels in the reference and target images. In general, the more similar the pixels are, the lower is the value of the matching cost.

The simplest method relies on pixel color information, for example, SSD (Sum of Squared Differences) [35] and SAD (Sum of Absolute Differences) [36]. These techniques are used also when matching objects from consecutive frames in video processing, but under the names of MSE (Mean Squared Error) and MAD (Mean Absolute Difference).

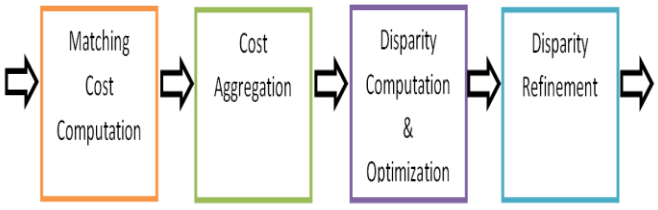

Fig. 1. Disparity map extraction flow.

In both cases, for every pixel in the reference image we search for a match on the epipolar line in the target image. Usually, only a maximum disparity is allowed so the search occurs in a defined window. For every possible pair, a cost is computed using (1) in case of SSD and (2) in case of SAD.

$$f(x,y,d) = \sum (I_L(x,y) - I_R(x,y-d))^2 \quad (1)$$

$$f(x,y,d) = \sum \left| I_L(x,y) - I_R(x,y-d) \right| \quad (2)$$

Some local algorithms combine matching cost computation and aggregation and use a matching cost that is based on a support region. NCC (Normalized Cross-Correlation) (equation 3) is such an algorithm and behaves similar to SSD, but is more complex, involving more mathematic operations. Due to the fact that in many image-processing applications the brightness of the image can vary due to lightning and exposure, the image must be firstly normalized.

$$f(x,y,d) = \frac{\sum I_L(x,y) I_R(x,y-d)}{\sqrt{\sum {I_L}^2(x,y) \sum {I_R}^2(x,y-d)}} \quad (3)$$

After computing the cost for each pixel, one way to choose the best corresponding pixel would be to choose the pixel for which the cost has the smallest value, approach also known as WTA (Winner Takes All). However, the resulting disparity map is very noisy. An alternative would be to compare small patches of pixels, instead of individual pixels.

As mentioned before, in order to overcome the bad result produced by comparing single pixels, a matching window around the pixel of interest is used instead. Increasing the window size reduces the noise effect in the disparity map.

However, this also results in increasing the computational time and choosing the correct window size is just another problem to address. Among the techniques used are SAD and SSD, which are applied also to single pixels, but the difference is that *f*(*x, y, d*) is computed now over an area and summed afterwards. Very similar is STAD (Sum of Truncated Absolute Differences), defined by (4), where *T* is the disparity threshold.

$$f(x,y,d) = \sum \min \left\{ \left| I_L(x,y) - I_R(x,y-d) \right|, T \right\} \quad (4)$$

It can be seen from Fig. 2 that matching cost aggregation has produced better results, but they are still far from being optimal because there exist some problems with fixed windows. Firstly, such a method assumes that the depth is constant within the window. Usually this is not the case due to depth discontinuities and slanted/non-planar surfaces. Secondly, many images contain repetitive textures or uniform areas. In these cases, there are many "weak" minima of the matching cost. Last but not least, sometimes the window is larger than the structure. Even so, because its simplicity, fast execution time and low memory requirements, SAD followed by a WTA approach is often chosen in implementations. It can be run real-time on standard processors (SIMD) and hardware implementations like FPGA, consuming a limited amount of power.

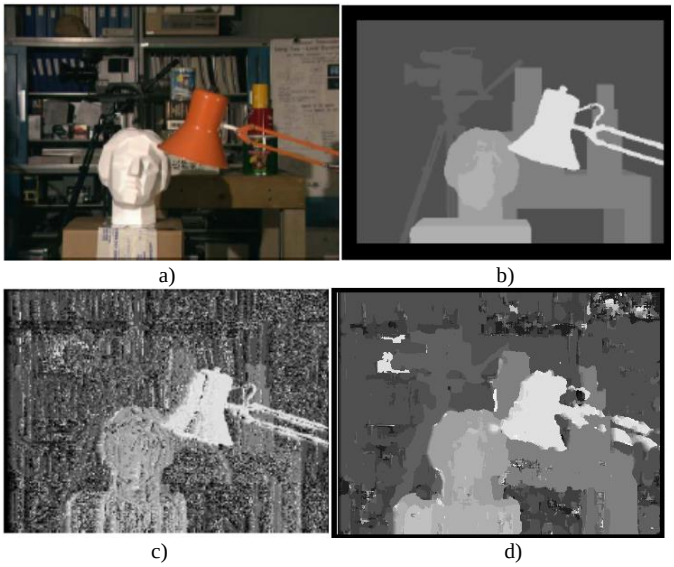

Fig. 2. Single pixel vs. matching window. a) left camera image; b) optimal result; c) single pixel comparison; d) matching window.

The next step tries to find the best disparity assignment and we can distinguish two main classes of methods: local and global ones.

In local methods, the most important steps are matching cost computation and cost aggregation. Choosing the disparity is a trivial task, namely finding the one associated with the minimum cost. As mentioned before, this is WTA optimization applied to each pixel. The disadvantage of these methods is that a uniqueness match is not enforced both directions: the reference image has a single match in the target image, but pixels in the target image can correspond to multiple pixels in the reference image.

On the other hand, in global methods the focus is put on this step. Sometimes, the cost aggregation step is skipped. In many methods, the aim is finding a disparity function that minimizes a global energy. Usually, the global energy function has two terms like in (5).

$$E(d) = E_{data}(d) + E_{smooth}(d) \quad (5)$$

The first term measures how well the disparity function *d* agrees with the input image pair in terms of overall matching cost [5]. The cost functions used can be pixel-based or, more effective, support aggregation strategies. The second term encodes assumptions (of continuity) of the scene made by the algorithm. Large disparity variations are penalized and are allowed only at depth borders. Finding the best assignment that minimizes the energy function is an NP-hard problem. In order to make this computationally possible, often only differences between neighboring pixels' disparities are computed.

A different class of global optimization algorithms is based on dynamic programming, which can find the global minimum for independent scan lines in polynomial time [5]. Complex problems are solved by breaking them down into sub-problems, each of which is solved only once. This technique has the advantage of enforcing the ordering constraint (e.g. the pixels must be in same order both in reference and target image) and

being accurate at depth borders and uniform regions. DP algorithms find the optimal disparity solution for a scan line in two steps: forward pass and backward pass. The forward pass, the minimum sum of aggregation, matching cost and smoothness penalty for each disparity candidate pixel along the scan line is searched. At each iteration, the winner is then summed with the first term of (5), resulting in an aggregation cost. In the backward pass, the optimal disparity map scan line solution is obtained iteratively backwards.

Cooperative algorithms are another class of optimization algorithms and are among the earliest methods proposed [37]. They perform local computations, but use nonlinear operations that result in an overall behavior similar to global optimization algorithms [5]. In some implementations it is possible to state the global function to be minimized. The authors have formulated two assumptions on which the algorithm is based: uniqueness, allowing only one match along each view direction, and continuity, meaning that neighboring matches should have similar disparity values. These constraints define a complicated error-measure which is minimized through cooperative network dynamics [6].

The last step from the stereo matching workflow is the disparity refinement, which is an optional one, used for improving the matching quality. In this pass, outliers produced by the previous steps, are identified and corrected. Also, because the disparity maps are usually computed at pixel level, more accurate disparity assignments (computed at sub-pixel level) would be desirable.

B. Shiftable Window SSD

Shiftable window SSD is a classic stereo vision algorithm which belongs to the area-based correlation category. It follows four steps, in which each area-based algorithm is divided: matching cost computation, cost aggregation, disparity computation and the optional disparity refinement.

The basic idea is matching the intensity values within windows between the stereo image pair. The algorithm needs as input a pair of rectified images. For every pixel in the reference image, a correspondent in the target image is computed by comparing a square window centered on this pixel against windows of the same size centered at points on the corresponding scan line (epipolar line) in the target image. The number of candidate points in the second image is given by the maximum disparity value, which typically lies between 10 and 20. SSD is used as a measure of dissimilarity between the windows. The point for which the surrounding window has yielded the minimum sum of squared intensity values will be chosen as the best match and the offset between its location and the location of the reference point will be stored as the disparity at that location [38].

As mentioned in the previous sections, the main issue in window-based methods is choosing the optimal window size. If the value is too small, wrong matches are likely to occur due to noise and ambiguities. However, the object shape will be preserved. On the other hand, if the window is too large, it will reduce the number of wrong matches, but at the same time it will blur the object boundaries. In Fig. 3, the results have been obtained by using various window sizes.

a) b) c)

Fig. 3. Disparity maps obtained by applying shiftable window SSD [39]. a) window size = 3; b) window size = 11; c) window Size = 25.

The shiftable window algorithm can be implemented with other matching costs, for example SAD or NCC instead of the presented SSD. The SSD method has a higher computational cost compared to SAD, involving numerous multiplication operations.

C. Dynamic Programming

Dynamic programming is a technique that is used not only in the stereo correspondence problem, but also in fields very different with regard to computer vision, such as mathematics, economics and bioinformatics. The basic idea is to decompose a problem into a set of sub problems, such that, given a solution to the sub problems, the solution to the original problem can be quickly computed and the sub problems can be solved recursively in terms of each other. Because the solution to a sub problem is used later on, multiple times, we need to store the solutions in order to avoid recomputing them.

DP can be thought of as a method for filling in a table. At each position a value is found, corresponding to a sub problem we need to solve. DP algorithm iterates over the table entries and computes for each location a value that is based on the previously seen entries. The relationship between the current entry and the previous ones is usually given in a simple, recursive manner, but at times can be more complex. Generally, it provides better results than the area-based methods and it is even faster than these.

In the stereo correspondence problem, DP helps to find a path through the image which provides the best match. For each pair of corresponding scan lines, a cost matrix of all pair wise matching costs is built. The goal is to find the minimizing path through this matrix. Fig. 4 depicts how such a matrix looks like.

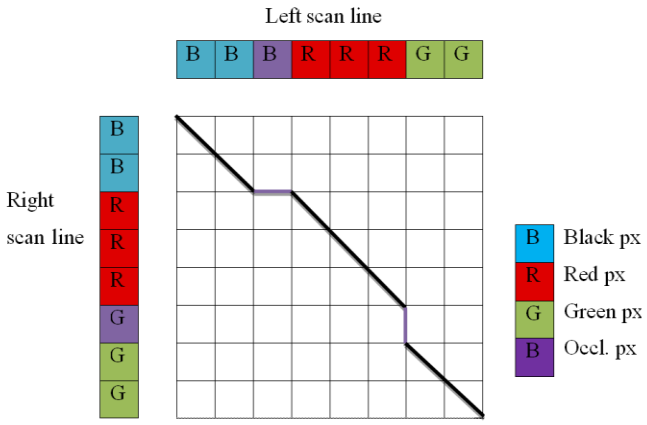

Fig. 4. Stereo matching using DP.

Let's assume that we have two scan lines, with black, red and green pixels which need to be matched. Also, we have 1 left occluded pixel and 1 right occluded pixel. Any path starting from the top left of the matrix and ending at the bottom right of the matrix represents a specific correspondence of pixels in the left scan line to pixels in the right scan line. In this example, the best possible path assumes that the first 2 pixels correspond and then there is a left occlusion. The next 3 pixels correspond again, followed by a right occlusion and another match of 2 pixels.

We can observe that at each location there are 3 possible directions:

- diagonal: match
- right: left occlusion
- down: right occlusion

Usually, when searching for a match, only a limited range of disparities is considered which will lead to fewer paths. A path is constituted by a succession of values on the matrix. The value of an arbitrary point in the matrix represents the value of getting there. As seen in (6), for a point $P(x, y)$, if we have a match, we will pay no penalty, but add a matching cost to the previous, precomputed value for the previous point, $P(x–1, y\text{-}1)$, from which we arrived by following the diagonal direction. This is the case for the first 2 pixels in Fig. 4, for example. On the other hand, in case of no match, we will pay an additional cost, the occlusion cost, which will add up to the precomputed values for the previous points, depending on the type of occlusions: $P(x–1, y)$ in case of left occlusion and $P(x, y\text{-}1)$ in case of right occlusion.

$$P(x,y)=\min\begin{cases}\text{match cost}\,P(x,y)+P(x-1,y-1)\\ \text{occlusioncost}+P(x-1,y)\\ \text{occlusioncost}+P(x,y-1)\end{cases} \quad (6)$$

In the end, we will obtain the correspondence cost. This is the first part of a DP algorithm. However, once found the best cost, we need to reconstruct the optimal path using backtracking, which is the second part of the algorithm. The disadvantage of DP is that local errors might be propagated.

Optionally interpolation can be used. Such a method is Birchfield-Tomasi interpolation [22], which computes for every pixel the minimum and maximum values in the two half-intervals before and after it. These values are further used when computing the matching cost (SAD, SSD etc.).

D. Scan Line Optimization

Scan line optimization (SO) is like DP, a global optimization technique which optimizes one scan line at a time. However, unlike DP, SO is an asymmetric method and does not make use of constraints such as ordering and visibility constraints. What is more, there is no penalty cost for occlusions. At each point, a disparity value is assigned so that the overall cost along the scan line is minimized. This way, a global minimum on each scan-line is obtained, but no inter-scan line consistency, resulting in a horizontal streaking effect in the resulting disparity map.

SO is very similar to another global technique, Graph Cuts, which is presented in the next section. The difference is that vertical smoothness is not taken into consideration. However, it does use the horizontal smoothness terms, without which the algorithm would be just WTA optimization.

The first step of this algorithm is computing the DSI. This means that for every pixel in every scan line a cost is computed for a maximum number of disparities, ranging from 0 (the case in which the corresponding pixels have the same location) to disparity max (the maximum possible offset of two corresponding pixels). The cost is computed in this version of the algorithm using SAD, but any other matching cost could be used. The aggregation step is skipped. Afterwards, for each point, the best disparity is chosen using, in this case, the scan line optimization technique.

E. Simulated Annealing

Simulated annealing (SA) is a global method which performs standard moves, meaning that only one pixel changes at a time. This is why in many stereo vision implementations it is shown to be the slowest optimization techniques. Such a disadvantage is solved by the algorithm presented in the next section, which changes a whole group of pixels in one move. However, SA is still used occasionally, especially in highly connected and highly non-submodular graphs.

SA is a random search function that effectively approximates the global optimum solution. This is inspired by the process in metallurgy in which a metal alloy is heated to a very high temperature. The atoms are left at this temperature long enough to reach thermal equilibrium. Afterwards, the metal is gradually cooled on a very specific and gradual schedule. As they cool, the atoms settle into an optimal crystalline structure. This process improves the cold-working properties of the metal.

One main characteristic of this algorithm is the iterative improvement, which means that it goes through a several number cycles, each of them trying to improve the current solution. It uses local random search, instead of sampling through the entire sample space. Also, SA is an algorithm that explores the solution space, especially early in the search, but as the search progresses, coming closer to the global optimum; it becomes greedy, accepting only improvements of the solution.

SA starts at a high temperature so that a large part of the state space is explored randomly, and cools gradually the temperature to find a suitable local minimum. There are several variants of this algorithm. For example, in the Metropolis variant, downhill steps are always taken, and uphill steps are only sometimes (randomly) taken in order to avoid being stuck in a local minima. Another variant is the Gibbs sampler. Instead of choosing the lowest energy for the variable being updated, it chooses among several possible states: either a new, random state (disparity), either one of the possible disparities at a given pixel is chosen. As mentioned above, the algorithm starts at a high temperature and then this is gradually decreased. The algorithm is terminated when a maximum number of iteration is reached.

F. Graph-Cut Optimization

The graph-cut algorithm (GC) belongs to the global class and it is one of the best algorithms available. It produces very accurate results and, therefore, it is a very popular one. However, this happens at an increased computational cost. As in the case of other global techniques, we are interested in minimizing an energy function. In the case of GC, we have a label set, namely the disparities and a set of pixels. The goal of the algorithm is to find a labeling f which minimizes some energy. Min-cut/max-flow algorithms which come from combinatorial optimization have been proven to minimize such energy functions in computer vision. Stereo matching and image segmentation are such applications.

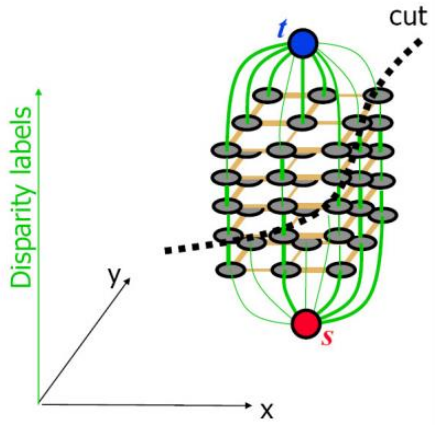

Fig. 5. S-T graph cut [40].

A graph is a model that looks like a network and is built from a number of nodes and a number of edges that connects them. A graph is called a directed graph if the edges have a direction associated with them. Very often, each of these oriented edges are associated a cost so that choosing one path or the other results in different costs. Sometimes, a graph contains a number of additional special nodes, called terminals. For this algorithm, graphs with two terminals are considered, the source *s* and the sink *t*. With respect to the stereo vision application of graph theory, the terminal nodes correspond to the disparities that will be assigned to pixels. There are two types of edges in the graph: *n*-links and *t*-links. The first category is used to connect neighboring pixels, representing the neighborhood system in the image. The cost associated to these represents the penalty for the pixel discontinuity. The second category connects pixels with the terminals, labels (more exactly, disparities). In this case, the cost of such links corresponds to the penalty for assigning a certain label to a pixel.

In a two terminal graph, a cut is a partitioning of the graph nodes into two disjoint subsets, *S* and *T*, such that the source *s* is in *S* and the sink *t* is in *T* [41]. This is also called *s*-*t* graph cut and can be 2D or 3D, as illustrated in Fig. 5. A cut has an associated cost, the sum of the edges that are eliminated when partitioning the graph into two sub-graphs.

a) b) c)

Fig. 6. SO vs. GC. a) original image; b) SO result; c) GC result [42].

One fundamental problem in combinational optimization is the minimum s/t cut, sometimes also called the maximum flow problem, because of their equivalence. We can think of it in the following way: maximum flow is the maximum "amount of water" that can be sent from the source to the sink by interpreting graph edges as directed "pipes" with capacities equal to edge weights [41]. The goal is to delete enough edges so that each pixel is connected to exactly one label node, while finding the global minimum of energy function. As mentioned above, the cost for a cut has two terms, the image consistency (horizontal edges) and the spatial consistency (vertical edges) like in (7). Because of these two costs, the streaking effect from SO is no longer visible, as shown in Fig. 6.

$$E(d) = \sum \left| I(p) - I(p + d_p) \right| + \sum w_{pq} \left| d_p - d_q \right| \qquad (7)$$

The authors of [42] have developed two new algorithms for energy optimization, which uses graph-cuts iteratively. These generate a labeling representing the global minimum of energy with respect to two types of moves: α-expansion and α-β-swap. Through such moves a large number of pixels change their labels to α or β simultaneously. The main purpose of these algorithms is to compute swaps or expansions until convergence.

G. Bayesian Diffusion

In the previous sections the main focus of the algorithms lied in the optimization step. However, in this section we will focus on the aggregation step. In [29] the authors have presented a Bayesian model with non-Gaussian statistics to handle gross errors and discontinuities in the surface.

The Bayesian model presented consists of two parts. Firstly, there is a prior model which uses MRF to encode preferences for smooth surfaces. Such a model uses the Gibbs distribution in (8), where *d* is the vector of all disparities and *Zp* is a normalizing factor.

$$p_p(d) = \frac{1}{Z_p} \exp\left(-E_p(d)\right) \qquad (8)$$

$$E_p(d) = \sum_{c \in C} E_c(d) \qquad (9)$$

Secondly, is the measurement model, which is based on the intensity differences between the left and right images.

IV. Experimental Results

This section presents the data sets used for experiments, with the main characteristics, as well as the quality metrics used to assess the performance of the algorithms tested. Furthermore, a series of experiments are described, comparing different matching cost techniques, from both the quality and time-related performance point of view. Also, an overall comparison of the algorithms presented in the previous section is described.

A. Data Set

The experiments presented below are run on four image sets, also known as the 2001 stereo data sets [5]. Two of the data sets, *Sawtooth* and *Venus*, are directly acquired by the authors of the paper mentioned. They each consist of a sequence of 9 images, taken at equally-spaced intervals by a camera mounted on a horizontal translation stage, along with a ground truth disparity map. The original images are sampled down to a resolution which is four times less and cropped in order to normalize the motion of the background objects to only a few pixels by frame. The content is made up of piecewise planar objects such as posters or paintings, some with cut-out edges. The other two sets are *Tsukuba*, from the University of Tsukuba, and *Map*, whose ground-truth disparity map has been computed the same way as for the first two sets.

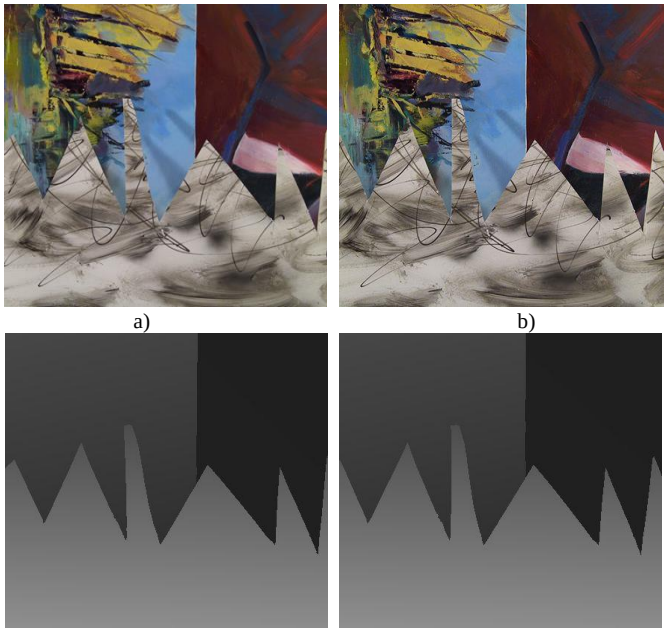

Fig. 7. Sawtooth data set. a) image 2 and b) image 6; c) ground-truth disparity maps for image 2 and d) image 6.

Finding a proper data set on which the correspondence algorithms can be tested is a difficult challenge. On one hand, there are the synthetic images which have been largely used, but because they often contain scenes that are not very complex, made up of simple geometric objects and textures, they do not produce very relevant results. On the other hand, images obtained from real cameras are difficult to convert into working sets because of issues such as aliasing, misalignment, lens aberrations, gain and bias, etc. but they model the real-world better. However, such images contaminated with noise are too difficult to be solved. For each of the images, a border of 10 pixels is excluded to avoid the border effect. For the *Tsukuba* image data set, this border is 18 pixels because no ground-truth disparity values are provided.

The *Sawtooth* data set contains both grayscale and color textures with sloping walls. It contains 9 images (numbered from 0 to 8) and 2 ground-truth disparity maps for images 2 and 6, scaled by a factor of 8 (Fig. 7). This means that for a value of 80 in the ground-truth disparity map for image 2, the corresponding pixel in image 6 is 10 pixels to the left. The images for which no ground-truth disparity map is given are only used in case the prediction error is measured. Image size: height 380, width 434 and disparity: minimum 3.875, maximum 17.875.

The *Venus* data set contains color textures. It is actually a superposition of several sloping planes: newspaper, painting, illustration. Just like the *Sawtooth* data set, it is made up of 9 images and 2 ground-truth disparity maps, for images 2 and 6 (Fig. 8). Image size: height 383, width 434 and disparity: minimum 3, maximum 19.75.

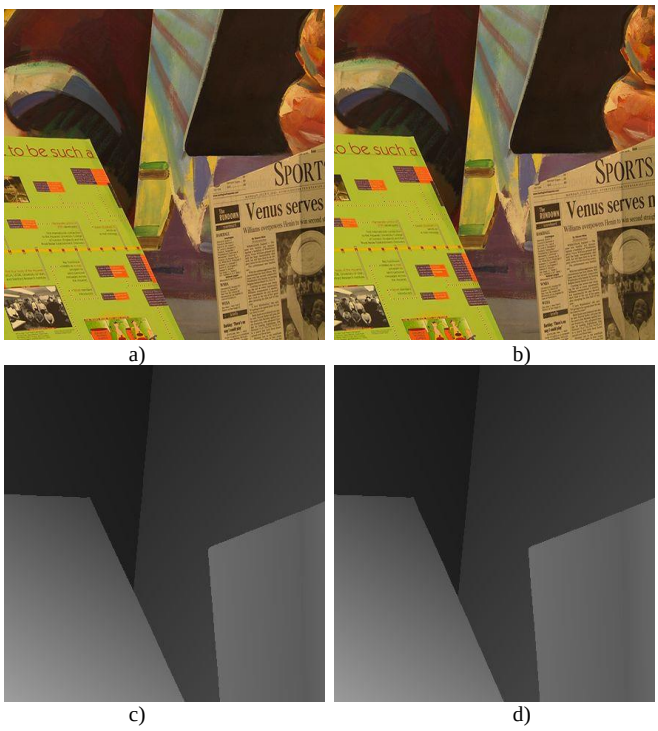

Fig. 8. Venus data set; a) image 2 and b) image 6; c) ground-truth disparity maps for image 2 and d) image 6.

The *Map* data set contains grayscale images, representing a map on top of other maps, with different textures (Fig. 9). It contains non-repetitive textures and very small occlusions, which means that all algorithms will perform well on this data set. Unlike the previous stereo sets, it contains only 2 images and 2 ground-truth disparity maps. Image size: height 216, width 284 and disparity: minimum 4.375, maximum 28.125.

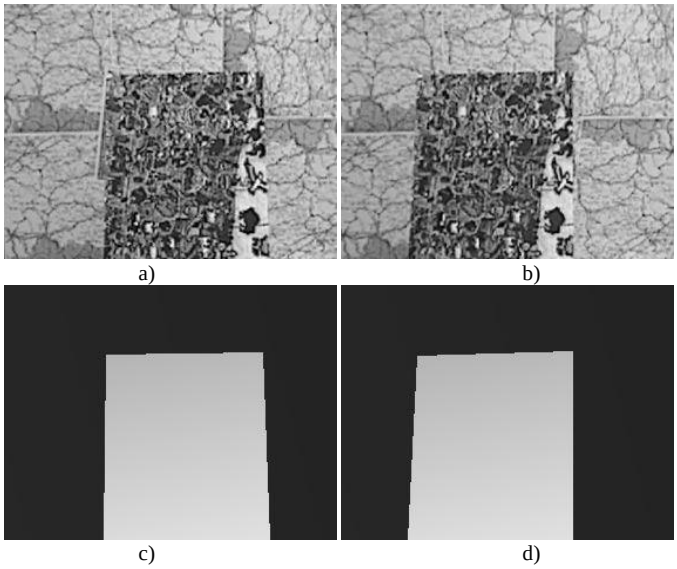

Fig. 9. Map data set; a) image 0 and b) image 1; c) ground-truth disparity maps for image 0 and d) image 1.

The *Tsukuba* data set contains images that represent a real scene (Fig. 10). It is made up of several distinct colored layers, one on top of the other. Almost all algorithms have problems in finding the lamps' arms because they try to provide smooth disparities. It contains 5 images and 1 ground truth disparity map. Image size: height 288, width 384 and disparity: minimum 0, maximum 14.

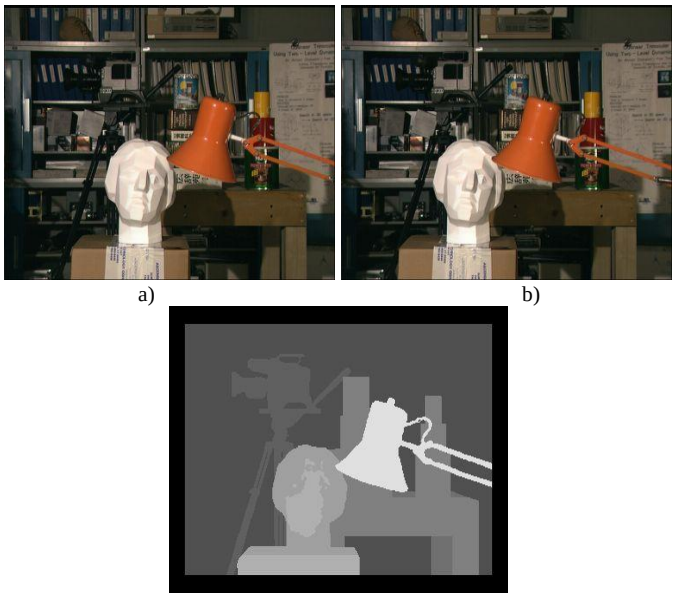

Fig. 10. Tsukuba data set; a) image 4 and b) image 5; c) ground-truth disparity map.

B. Quality Metrics

In order to assess the performance of the stereo correspondence algorithms we need some quality metrics. There exist two approaches in this direction. Firstly, some statistics can be computed based on the results obtained and the ground truth disparity map that already exists in the data set. The second approach is to assess the quality of the algorithm by using the entire image set and the resulted disparity map in order to predict the appearance of other views. This can be done by either forward wrapping the reference image by the computed disparity map to a different view, or by inverse wrapping a new view by the computed disparity map to generate a stabilized image and compare it against the reference image [5].

For a more clear and quantitative image of the algorithms the first approach will be used. These statistics can compute the RMS (root mean square) or several percentages of pixels that are labeled with bad disparities. The RMS is a statistical measure widely used for comparing obtained results against some ground data. In our case, it is measured in disparity units between the obtained disparity map and the ground-truth disparity map existing in the data set. This is computed with (10), were N represents the total number of pixels, $d_C(x, y)$ represents the computed disparity for the pixel at location (x, y) and $d_T(x, y)$ the ground-truth disparity.

$$RMS = \frac{1}{N}\sum_{(x,y)}\left|d_C(x,y) - d_T(x,y)\right|^2 \quad (10)$$

In addition to RMS, the percentage of bad-matching pixels can be computed using (11), where T denotes a threshold, the acceptable disparity error for counting the bad-matching pixels.

$$B = \frac{1}{N}\sum_{(x,y)}\left(\left|d_C(x,y) - d_T(x,y)\right| > T\right) \quad (11)$$

Depending on the regions investigated we can segment the image in three regions. Firstly, there are the textureless regions, where the difference in intensity gradient is below a certain threshold. Secondly, occluded regions are the ones which exist only in the reference image. Depth discontinuity regions are the ones that contain pixels with a significant difference in the neighboring disparities. Computing such percentages of bad-matching pixels gives us a more relevant image of the overall performance of an algorithm. For example, according to [5], a stereo correspondence algorithm is said to perform reasonably well if the percentage of non-occluded bad-matching pixels is less than 10%. Because of the 10% poorly matched image region, the RMS is contaminated by the large disparity errors, even if the 90% rest of the image can have no errors at all. RMS measure is to be used when the bad pixel percentage is very low.

In order to assess the performance of the algorithms at least two stereo images and a corresponding ground-truth disparity map are needed by the program. The quality metrics to be computed have been described above, but in the following experiments only some of the regions will be considered: textureless, depth-discontinuity, non-occluded. Also, an overall performance was referred. Besides the percentage of bad pixels, the RMS was also displayed graphically.

C. Results for Matching Costs

This experiment aims to assess the performance of matching costs using a local algorithm which actually translates to the traditional SSD/SAD algorithm presented above, but with a fixed square window of 9 pixels.

Fig. 11. Experiment 1 workflow.

Fig. 11 shows the blocks of a stereo-correspondence algorithm with regard to this experiment. The images are not preprocessed beforehand. The matching costs tested are SAD and SSD by themselves or in combination with the previously presented Birchfield-Tomasi interpolation. Also, for each of the matching costs truncation has been applied with values 1, 2, 5, 10, 20 and 50. This plays a role in the step in which the matching costs are computed. The minimum between the computed difference and the truncation value is chosen to be stored the cost matrix. It is to be noted that if SSD is used, the truncation value is squared.

Three specific categories of metrics have been taken in consideration for this experiment: the non-occluded pixels, the textureless pixels and the pixels that are near depth-discontinuities. An overall percentage of bad pixels were also computed. The experiment has been run on all four data sets presented in the previous section and the results are presented in Fig. 12. Besides the percentage of bad pixels, also the RMS measure is graphically displayed for all regions in Fig. 13.

As an overall conclusion we can state that there is very little difference between the performance of SSD and SAD matching costs. Generally, SAD tends to have better results for larger truncation values (e.g. 50, 20) and SSD for lower truncation values (e.g. 5, 10). Truncation values such as 1 and 2 should not be taken into account here, because the errors are very large in these cases. Good results are obtained for values greater than 5 and less than 50, usually around the value of 20. The greatest impact of truncation is definitely on the regions with depth-discontinuities, which contain pixels that correspond to the background and pixels that correspond to the foreground. Truncating the matching cost helps to limit the influence of bad matches.

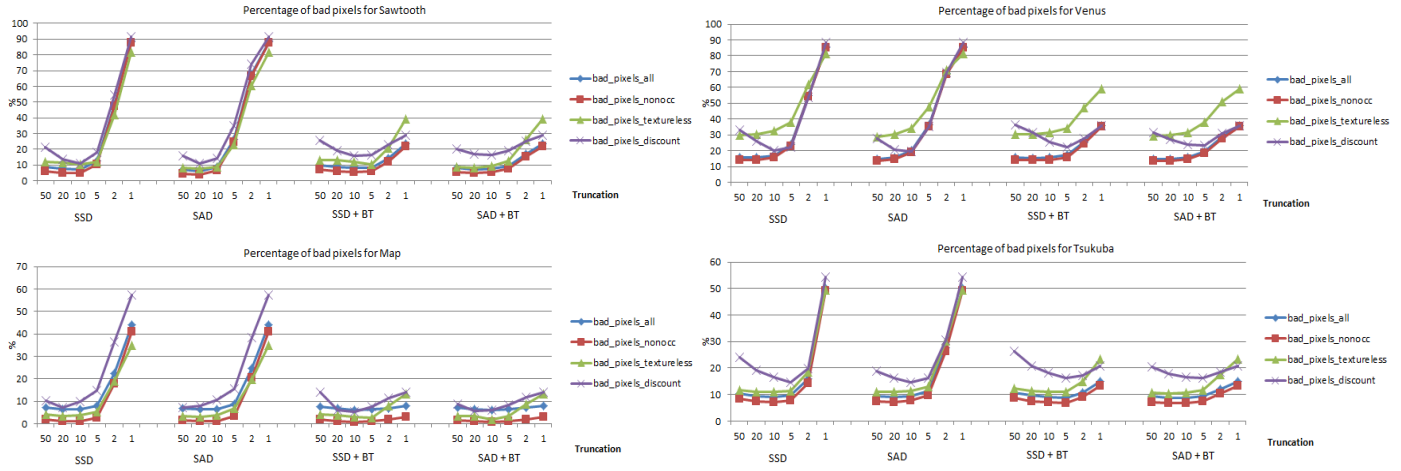

Fig. 12. Bad-matching pixel percentage for experiment 1.

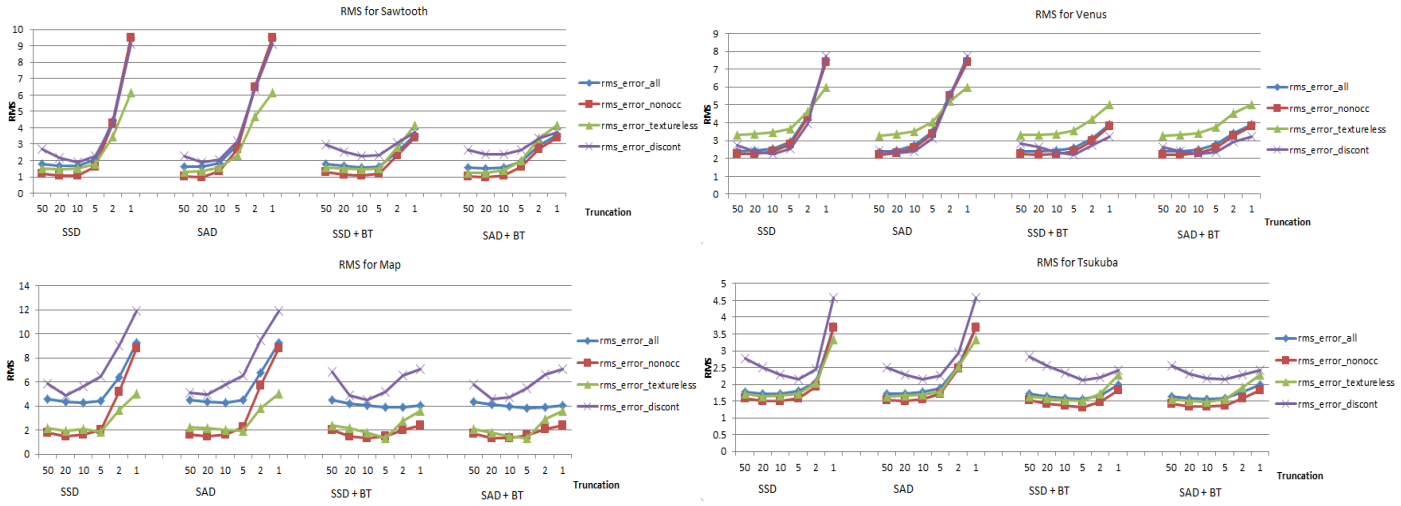

Fig. 13. RMS results for experiment 1.

Birchfield and Tomasi's interpolation improves the results for small truncation values, but does not improve them in case of large truncation values. On the contrary, in the latter case it tends to raise the errors for truncation values such as 20 and 50. This happens across all stereo image pairs. It can be seen that the RMS obtained by the matching costs follows the same trend as in the case of bad-matching pixel percentages.

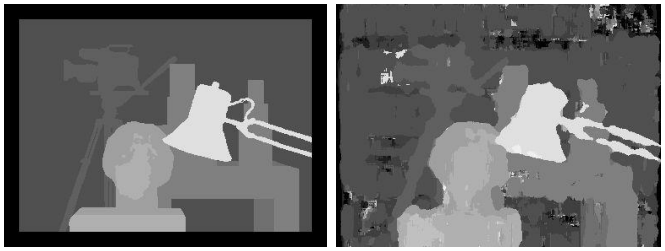

Fig. 14. Ground-truth vs. obtained disparity map for *Tsukuba* data set.

In Fig. 14, the ground-truth and the best disparity map for *Tsukuba* data set are displayed. These were obtained with SAD and Birchfield-Tomasi matching cost, with a truncation value of 10.

D. Results for Shiftable Windows

This experiment aims to assess the performance of shiftable windows and is very similar to experiment 1 (Fig. 15). In fact, it does the same operations as in experiment 1, but uses a 9x9 min-filter, which is equivalent with the shiftable windows aggregation. SAD/SSD matching costs are combined with Birchfield-Tomasi interpolation and truncation values.

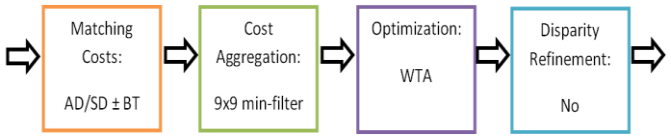

Fig. 15. Experiment 2 workflow.

This experiment has been run on all four data sets and the same statistics as the ones computed for experiment 1 are presented in Fig. 16 and 17. Trying all *nxn* shifted windows around a pixel is equivalent to applying a box-filter and a min-filter of the same size and it is not expensive from the computational point of view [5].

Just like in experiment 1, there is little difference between the AD and SD matching costs. What is interesting to observe is that for all experiments, the larger the truncation value, the smaller the error. Using no truncation at all yields the best results. The most obvious decrease in the error along with increasing the truncation value can be seen for the pixels that are near depth-discontinuities. This is very helpful, because we have already seen that choosing the best truncation value is a difficult task, depending very much on the data set. Instead, shiftable windows (box filter and min-filter aggregation) can be used for very good results, avoiding outliers. Birchifeld-Tomasi interpolation is helpful in reducing very large errors, which occur for low truncation values, but does not bring any significant improvement for reasonable errors, which are obtained for larger truncation values.

Just like in experiment 1, there is little difference between the AD and SD matching costs. What is interesting to observe is that for all experiments, the larger the truncation value, the smaller the error. Using no truncation at all yields the best results. The most obvious decrease in the error along with increasing the truncation value can be seen for the pixels that are near depth-discontinuities. This is very helpful, because we have already seen that choosing the best truncation value is a difficult task, depending very much on the data set. Instead, shiftable windows (box filter and min-filter aggregation) can be used for very good results, avoiding outliers. Birchifeld-Tomasi interpolation is helpful in reducing very large errors, which occur for low truncation values, but does not bring any significant improvement for reasonable errors, which are obtained for larger truncation values.

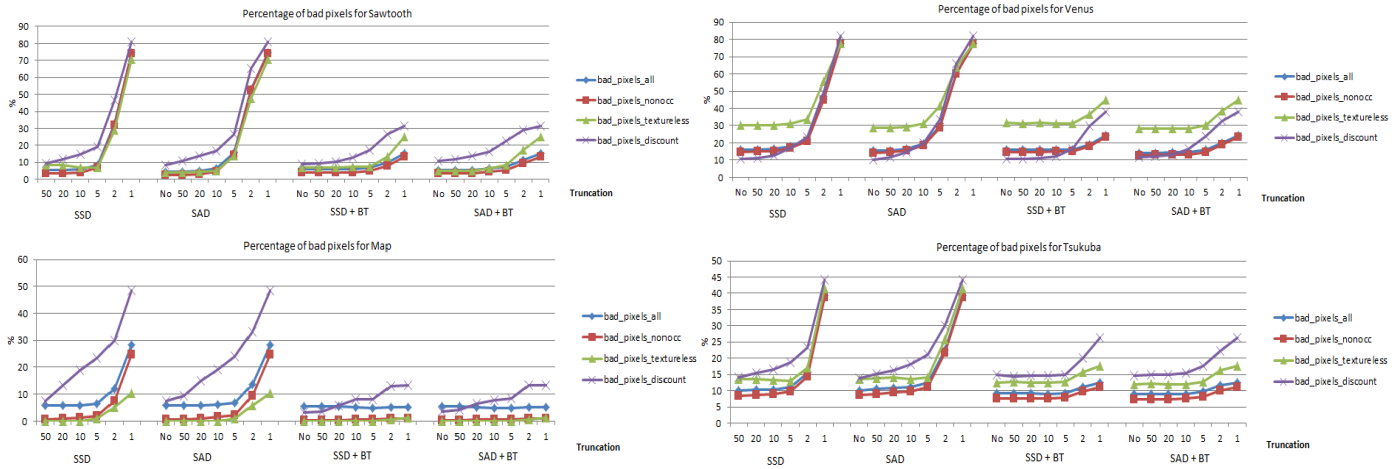

Fig. 16. Bad-matching pixel percentage for experiment 2.

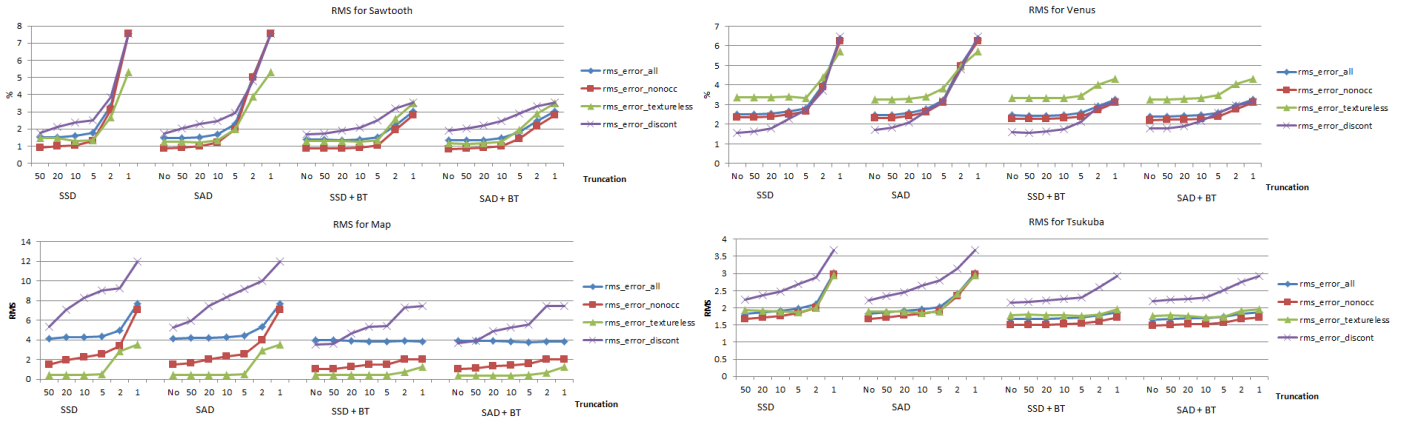

Fig. 17. RMS for experiment 2.

In Fig. 18, the best resulting *Tsukuba* disparity map was displayed, along with the ground-truth disparity map. The lowest percentage of bad-matching pixels on this data set was obtained for SAD with BT, with no truncation: 7.176 %. The best percentage on this data set in experiment 1 was 6.8003 %.

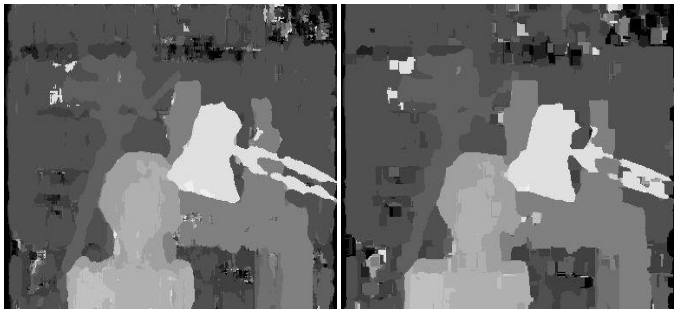

Fig. 18. Experiment 1 vs. experiment 2 results on *Tsukuba* data set.

E. Results for Aggregation Methods

Unlike the previous experiments, experiment 3 focuses on the aggregation step (Fig. 19). Three methods are used: square windows, shiftable windows and diffusion. For the square windows, the algorithm is in fact SAD. Several window sizes are used, starting from 3 x 3 and ending with 29 x 29. The shiftable windows algorithm is reproduced by using a box filter followed by a min-filter. Several window sizes are also used here, just like for the square windows. Last but not least, regular diffusion is tested. In this case, we no longer need a window size. The algorithm is controlled by the number of iterations, which ranges from 10 to 150, but which has the same meaning, controlling the extent of the aggregation.

Regarding the other steps, AD is used for computing the matching costs. The aggregation is followed again by a WTA optimization and no disparity refinement is done. The images were not preprocessed.

Fig. 19. Experiment 3 workflow.

Regular diffusion is an alternative to using fixed windows, easier than the Bayesian model which is controlled by much more parameters. This technique aggregates support with only a weighted support function, for example a Gaussian. Four neighbors of the pixel are used when computing the energy and lambda, which controls the speed of the diffusion. In this experiment, lambda was chosen 0.15. According to [29], to ensure convergence, lambda needs to be < 0.25.

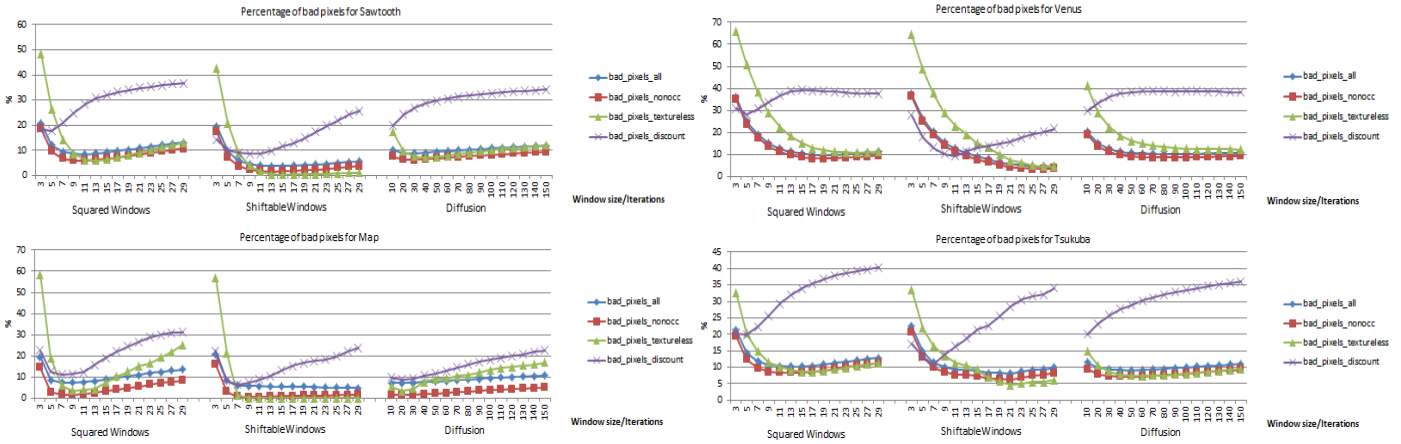

Fig. 20. Bad-matching pixel percentage for experiment 3.

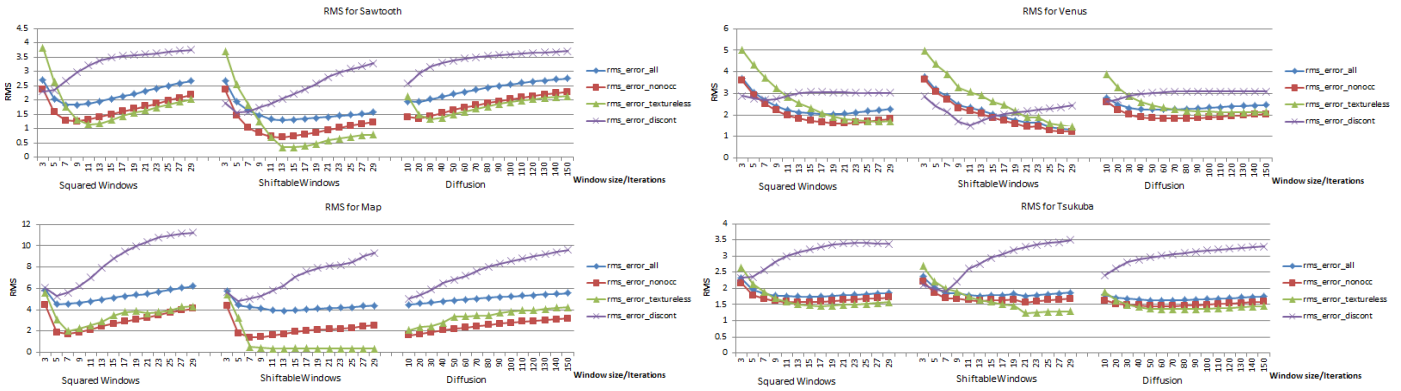

Fig. 21. RMS for experiment 3.

In Fig. 20 and 21, the results of this experiment were displayed. The most interesting observation is that the curves shows opposite trends for the textureless pixels and the ones located near depth-discontinuities. The larger the window sizes for aggregation, the smaller errors in the textureless areas. On the contrary, in the depth-discontinuities regions, the errors increase with the size of the aggregation extent.

This is the fundamental drawback of local algorithms: choosing the window size proves to be a difficult task, depending on the image sets and the regions they contain. The local methods assume that all pixels in one window share the same characteristics, namely disparities. However, often, especially in highly textured images, such windows contain mixed pixels. Some belong to the background and some belong to the foreground. The aggregated cost of such a window can only take one direction and this is done depending on how much horizontal texture exists in the regions near a depth-discontinuity. This problem is also known as fattening effect, which characterizes the local methods. This means that the center of a window inherits the disparity of the pixels with a stronger texture. In case of strong depth-discontinuities, this is called foreground fattening effect. The fattening effect, as stated by its name, makes the objects in the blocks look larger (fatter) than in reality. This is not very obvious in 2D, but becomes very noticeable in 3D depth reconstruction, creating unrealistic models.

In Fig. 22, it can be observed the ground-truth disparity map and the obtained disparity map with the squared windows and shiftable windows methods, having window size 29. The squared windows technique makes the foreground object visibly larger than in reality, while the shiftable windows technique recovers the original proportions pretty well.

a) b) c)

Fig. 22. Fattening effect on *Map* data set. a) ground-truth; b) square windows; c) shiftable windows disparity maps.

If we look at the graphics we can see that the shiftable windows algorithm yields the best results from all the three techniques, especially in the case of depth-discontinuity regions. Shiftable windows method is the simplest variation of the adaptive window techniques, which try to eliminate the fattening effect. This is followed by the diffusion model. Although it is the simplest method from the three, the square windows has the poorest results and choosing the right window size has even more impact than in the case of the shiftable windows.

F. Optimization Methods

Experiment 4 focuses on the next step of a stereo correspondence algorithm, the optimization (Fig. 23). Four methods are compared: dynamic programming, scanline optimization, graph cuts and simulated annealing. For the dynamic programming, three variants have been tested, depending on the occlusion cost: 20, 50 or 80. The experiment aims to assess the effect of the smoothness parameter on the four techniques above mentioned. Therefore, the smoothness parameter ranges from 5 to 1000. Also, for SA a number of 500 iterations have been chosen. The images are not preprocessed beforehand. As matching cost, AD is used. No aggregation and no disparity refinement are used.

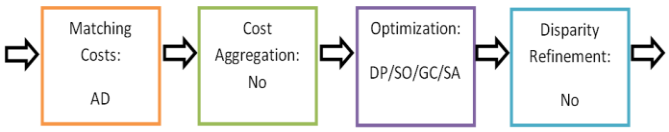

Fig. 23. Experiment 4 workflow.

The experiment has been run on all data sets and the findings are displayed graphically in Fig. 24 and 25. It can be seen that among the four techniques tested, GC clearly yields the best results in a consistent manner. Unfortunately, among these methods, GC and SA are much slower. The best results are seen especially in textureless and depth-discontinuities regions. The other three techniques do not display large differences in performance. Sometimes SO is slightly better, like in the case of *Map* and *Tsukuba* data sets, but, as it will be seen in the next experiment, will cause a streaking effect in the disparity maps obtained, because it ignores the vertical smoothness term. DP with an occlusion cost of 20 has the smallest errors.

Comparing the results of different smoothness parameters, it is interesting to see that errors are not monotonically growing or decreasing, especially in the case of SO and GC. For very small smoothness values, the errors are large and the same happens also for very large values. Therefore, choosing the right value has a great impact on the results. These are also influenced by the data set. For example, in the case of an image with very few objects, such as *Map*, the optimal value is quite high, around 500. On the other hand, in images with many objects located at different depths, the best results are obtained for small smoothness values, between 20 and 200.

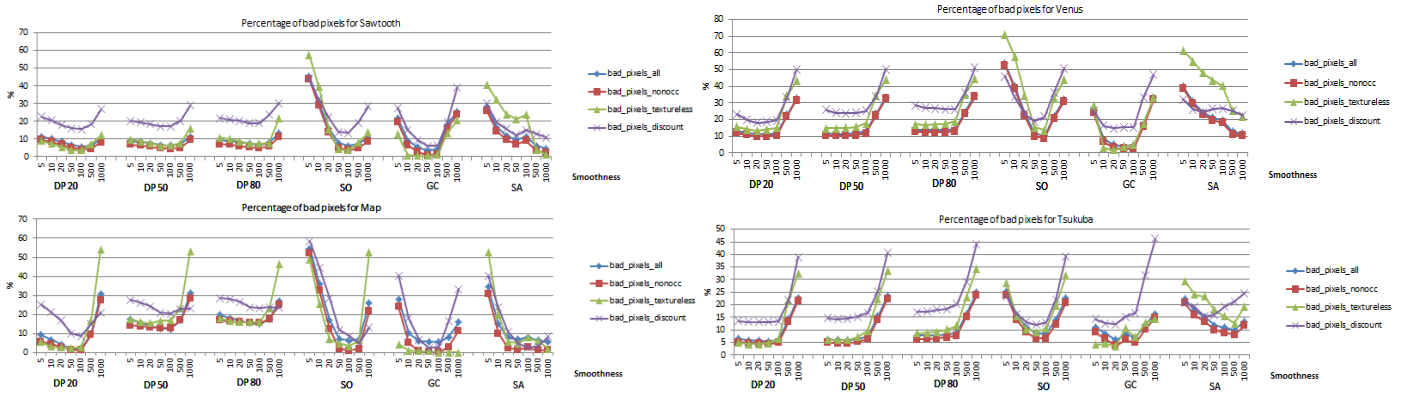

Fig. 24. Bad-matching pixel percentage for experiment 4.

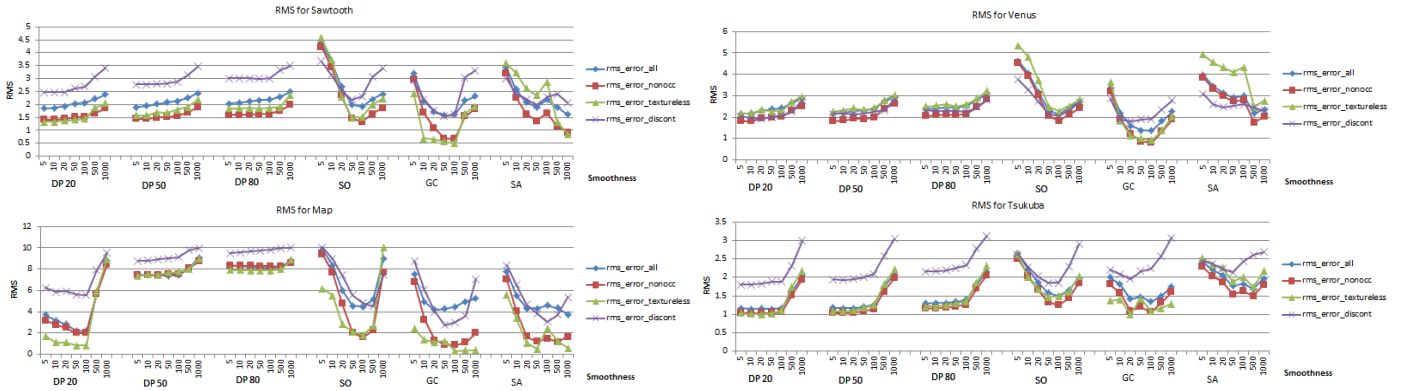

Fig. 25. RMS results for experiment 4.

G. Overall Comparison

We have compared the performance of the 6 algorithms described above, among them one being a local algorithm and the other five, global algorithms. For each, the parameters used in the tests are listed in Table 1.

Table 2 displays the results obtained by these algorithms with the parameters above applied to each image. Of course, these results would have been better if they would have been fine-tuned for each set of images, but a general configuration should be found. From all of the metrics discussed, the percentage and ranking of non-occluded bad-matching pixels is listed.

From Table 2 and from the results of the above sub-sections, it can be seen that the best overall performance is obtained by global optimization algorithms. In particular, the best method is definitely GC, on which there is large focus nowadays. SA solves the same optimization problem, but generally does not yield very good results. The execution time for SA is way higher than the one for GC, so the latter algorithm is preferred.

The diffusion-based method, Bayesian diffusion, performs well, following GC. In the case of *Map* image data set, it even outperforms this method. This happens because these images contain more noise than the others, which leads to bad results in case of algorithms that depend very much on internal parameter settings. On the other hand, in case of Tsukuba, which is a more complex data set, with many objects at different depths, the Bayes diffusion algorithm has poor results.

SSD, which is the only local algorithm in the experiment, yields reasonable results, especially if we are to think at the complexity of the algorithm and execution times, which make it a good choice if the results do not need to be of very high accuracy. As do all algorithms, it does not perform well in regions with depth-discontinuities.

Lastly, but not least, the scanline algorithms DP and SO perform less well, having the poorest results of them all, with the exception of *Tsukuba* image set, were they recover the shape of the objects pretty well, and the lack of inter-scanline consistency raise the percentages of bad-matching pixels.

TABLE I. PARAMETERS FOR THE 6 ALGORITHMS TESTED

	SSD	DP	SO	GC	SA	Bayes
Matching cost	SD	AD	AD	AD	AD	AD
Match function	No	No	No	No	No	No
Truncation	No	Yes	Yes	Yes	Yes	No
Birchfield-Tomasi						
Aggregation						
Window size	21	-	-	-	-	-
Min filter window size	21	-	-	-	-	-
No. of iteration steps	1	-	-	-	-	1000
mu parameter for Bayesian diffusion	-	-	-	-	-	0.5
sigma parameter for Bayesian diffusion	-	-	-	-	-	0.4
epsilon parameter for Bayesian diffusion	-	-	-	-	-	0.01
scale for Bayesian diffusion	-	-	-	-	-	0.01
Optimization						
Optimization function	WTA	DP	SO	GC	SA	Bayes
Smoothness	-	20	50	20	20	-
Occlusion cost	-	20	-	-	-	-
Start temperature	-	-	-	-	50	-
Maximum number of optimization iterations	-	-	-	-	1000	-
Threshold for intensity gradient	-	8	8	8	-	-
Smoothness penalty if gradient is too small	-	4	2	2	-	-

TABLE II. RANKING AND PERCENTAGE OF BAD-MATCHING PIXELS FOR THE 6 ALGORITHMS TESTED

	Sawtooth	Venus	Map	Tsukuba
SSD	(3) 2.2051	(2) 3.7441	(4) 0.6638	(3) 5.3337
DP	(5) 4.3094	(5) 8.8257	(5) 0.7748	(2) 4.5472
SO	(6) 4.4275	(4) 7.7776	(6) 1.1016	(4) 6.0406
GC	(1) 1.4049	(1) 1.8324	(2) 0.3103	(1) 1.9184
SA	(4) 3.5237	(6) 9.1775	(3) 0.5878	(6) 7.6125
Bayes	(2) 1.4533	(3) 4.0044	(1) 0.1973	(5) 6.4924

V. CONCLUSIONS

In this paper, the domain of stereo vision, one of the most studied fields of computer vision of the last decades, has been explored. The hardest part of stereo vision is finding the right match for every point in the image, also known as the stereo correspondence problem and solved by the stereo matching algorithms. The building blocks of every such algorithm have been presented: matching cost computation, cost aggregation, disparity computation and optimization and disparity refinement. Additional preprocessing and postprocessing can also be done. The focus in this paper was put mainly on the first three building blocks, as disparity refinement is rather optional. The results obtained by the algorithms are more than often sufficient for applications such as tracking or robot navigation.

During the years, many techniques have been proposed, some novel or refinements of existing ones. A literature overview has been presented for both local and global algorithms and also for the newly introduced semi-global category, presenting techniques from the most basic ones, such as SAD/SSD, up to the state-of the art in the domain, GC.

As mentioned before, most of the existing algorithms can be split in several building blocks. In order to obtain the best performance, each of these components should be optimized. The first experiments have tried to display different techniques for each individual step, showing how choosing parameters influences the results. Unfortunately, sometimes there are so many parameters, that it is hard to find the optimal combination. Furthermore, the image data sets are comprised of different textures, occluded/non-occluded regions, different color information, all of which make a general optimal configuration very hard to find. Some algorithms perform better than the others depending on the input information.

The literature probably contains a great number of algorithms variations, such as a full comparison of stereo matching algorithms would have been practically impossible. Instead, a few diverse algorithms have been presented, focusing on their core ideas. A number of experiments have been conducted, comparing them from the quality point of view (RMS and statistics on bad-matching pixels). If we do not want a very accurate disparity map in our application, the local methods such as the SSD algorithm should be sufficient. If, on the other hand, quality is very important, GC is one of the best techniques nowadays. What more, it has been shown that SA, which solves almost the same optimization problem as GC, has the highest computational costs and the results are in many cases far from the best. Of course, its performance depends very much on the internal parameters such as the start temperature and number of iterations. However, increasing the number of iterations will also increase the computational time, so the focus should be moved rather to GC methods.

The quality of matching could be studied not only by computing the RMS and statistics on bad-matching pixels, but also the prediction error. Another idea would be to find a solution to adapt parameters of algorithms to the content of the images automatically. As we have seen, a small smoothness value is needed when the image is complex, containing many objects, but a larger value can be used of the image is not so complex. Because GC has obtained good results, this method can be further studied for improvements, along with other state of the art techniques such as tree filtering, for example. Last but not least, besides the two-frame approach, the area of multi-view stereo can be investigated. Using multiple images will reduce the matching ambiguity, leading to quality improvements.

REFERENCES

[1] Stereopsis, Available from http://en.wikipedia.org/wiki/Stereopsis, Accessed: 15/04/2017

[2] R. Szekely, Computer Vision: Algorithms and Applications, Springer, 2010

[3] D. Marr, T. Poggio, “A Computational Theory of Human Stereo Vision”, Proceedings of the Royal Society of London. Series B, Biological Sciences, vol. 204, no. 1156, pp. 301-328, May 1979

[4] R. A. Lane, N. A. Thacker, Stereo vision research: An algorithm survey, Technical Report 94/16, University of Sheffield, 1994

[5] D. Scharstein, R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms”, International Journal of Computer Vision, vol. 47, pp. 7-42, June 2002

[6] M.J. McDonnell, “Box filtering techniques”, Computer Graphics and Image Processing, vol. 17, pp. 65–70, 1981

[7] K. Muhlmann, D. Maier, J. Hesser, R. Manner, "Calculating dense disparity maps from color stereo images, an efficient implementation", International Journal of Computer Vision, vol. 47, no.1-3, pp.79-88, 2002

[8] Y.S. Chen, Y.P. Hung, C.S. Fuh, "Fast Block Matching Algorithm Based on the Winner-Update Strategy", IEEE Transactions on Image Processing, vol. 10, pp. 1212-1222, August 2001

[9] D. Comaniciu, P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp.603-619, May 2002

[10] Z.F. Wang, Z.G. Zheng, "A Region Based Stereo Matching Algorithm Using Cooperative Optimization", IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, 2008.

[11] Stereo, Available from http://vision.middlebury.edu/stereo, Accessed: 16/04/2017

[12] A. Klaus, M. Sormann, K. Kaner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure", ICPR, vol. 3, pp. 15-18, 2006

[13] K.J. Yoon, I.S. Kweon, "Adaptive Support-Weight Approach for Correspondence Search", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 4, April 2006

[14] S.K. Gehrig, U. Franke, "Improving stereo sub-pixel accuracy for long range stereo", IEEE International Conference on Computer Vision, pp. 1-7, 2007

[15] K.J. Yoon, I.S. Kweon, "Stereo Matching with the Distinctive Similarity Measure", ICCV 2007, pp.1-7, October 2007

[16] M.Z. Brown, D. Burschka, G.D. Hager, "Advances in Computational Stereo", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 8, pp. 993-1008, August 2003

[17] B.D. Lucas, T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision", Proceedings of the International Joint Conference on Artificial Intelligence, pp. 674-679, 1981

[18] V.S. Kluth, G.W. Kunkel, U.A. Rauhala, "Global Least Squares Matching" Proceedings of the International Geoscience and Remote Sensing Symposium, vol. 2, pp. 1615-1618, 1992

[19] X. Zhou, P. Boulanger, "Radiometric Invariant Stereo Matching Based on Relative Gradients", ICIP, pp. 2989-2992, 2012

[20] U.R. Dhond, J.K. Aggarwal, "Structure from Stereo – A Review", IEEE Transactions on Systems, Man and Cybernetics, vol. 19, pp. 1489-1510, 1989

[21] V. Venkateswar, R. Chellappa, "Hierarchical Stereo and Motion Correspondence Using Feature Groupings. International Journal of Computer Vision, vol. 15, pp. 245-269, 1995

[22] S. Birchfield, C. Tomasi, "Depth Discontinuities by Pixel-to-Pixel Stereo", Proceedings of the International Conference on Computer Vision, vol. 1, pp. 489-495, 1999

[23] M.J. McDonnell, "Box filtering techniques", Computer Graphics and Image Processing, vol. 17, pp. 65–70, 1981

[24] C. Tomasi, R. Manduchi, "Stereo Matching as a Nearest-Neighbor Problem", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 3, March 1998

[25] O. Veksler, "Stereo Correspondence by Dynamic Programming on a Tree", Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 384-390, 2005

[26] Y. Boykov, O. Veksler, R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, pp. 1222-1239, November 2001

[27] L. Hong, G. Chen, "Segment-based Stereo Matching Using Graph Cuts", Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 74-81, 2004

[28] J. Sun, N.N. Zheng, H.Y. Shum, "Stereo Matching using Belief Propagation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 7, July 2003

[29] D. Scharstein, R. Szeliski, "Stereo Matching with Nonlinear Diffusion", International Journal of Computer Vision, vol. 28, no.2, pp. 155-174, 1998

[30] N. Lazaros, G.C. Sirakoulis, A. Gasteratos, "Review of Stereo Vision Algorithms: From Software to Hardware", International Journal of Optomechatronics, vol. 2, pp. 435-462, 2008

[31] H. Hirschmuller, "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 807-814, 2005

[32] M. Michael, J. Salmen, J. Stallkamp, M. Schlipsing, "Real-time Stereo Vision: Optimizing Semi-Global Matching", IEEE Intelligent Vehicles Symposium (IV), pp. 1197-1202, June 2013

[33] D.G. Lowe, "Object recognition from local scale-invariant features", ICCV, pp.1150-1157, September 1999

[34] Stereo Matching, Available from http://campar.in.tum.de/twiki/pub/Chair/TeachingWs10Cv2/3D_CV2_WS_2010_StereoMatching.pdf, Accessed 03/05/2017

[35] M.J. Hannah, "Computer Matching of Areas in Stereo Images", PhD. Thesis, Stanford University, 1974

[36] Kanade, T., Kano, H., Kimura, S., Kawamura, E., Yoshida, A. and Oda, K., "Development of a video-rate stereo machine", Journal of the Robotics Society of Japan, vol. 15, no 2, pp.261-267, 1997

[37] Stereo Vision, Available from http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/HENKEL/research/stereo/Coop/, Accessed 04/05/2017

[38] D. Scharstein, View synthesis using stereo vision, Springer, 1999

[39] S. Patil, J.S. Nadar, J. Gada, S. Mothghare, S.S. Nair, "Comparison of Various Stereo Vision Cost Aggregation Methods", International Journal of Engineering and Innovative Technology, vol. 2, pp.222-226, February 2013

[40] Correspondence, Available from http://www.csd.uwo.ca/courses/CS4487a/Lectures/lec07_correspondence.pdf, Accessed 06/07/2017

[41] Y. Boykov, V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, No. 9, pp. 1124-1137, September 2004

[42] Y. Boykov, O. Veksler, R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, pp. 1222-1239, November 2001