A Fast Area-Based Stereo Matching Algorithm

L. Di Stefano[1], M. Marchionni[1], S. Mattoccia[2], G. Neri[1]
1. DEIS-ARCES, University of Bologna
Viale Risorgimento 2, 40136 Bologna, Italy
2. DEIS-ARCES, University of Bologna - Sede di Cesena
Viale Rasi e Spinelli 176, 47023 Cesena, Italy
ldistefano@deis.unibo.it, mmarchionni@litio.it, smattoccia@deis.unibo.it, gneri@deis.unibo.it

Abstract

This paper presents an area-based stereo algorithm suitable to real time applications. The core of the algorithm relies on the uniqueness constraint and on a matching process that allows for rejecting previous matches as soon as more reliable ones are found. The proposed approach is compared with the left-right consistency constraint, being the latter the basic method for detecting unreliable matches in many area-based stereo algorithms. The algorithm has been carefully optimised to obtain a very fast implementation on a Personal Computer. This paper describes the computational optimisation strategy, which is based on a very effective incremental calculation scheme. Finally, we provide experimental results obtained on stereo pairs with ground-truth as well as computation-time measurements; we compare these data with those obtained using a well-know, fast, area-based algorithm relying on the left-right consistency constraint.

1 Introduction

Dense depth measurements are required in applications such as teleconferencing, robot navigation and control, exploration and modelling of unstructured environments, virtual reality. According to a recent taxonomy [17], stereo algorithms that generate dense depth measurements can be roughly divided into two classes, namely *global* and *local* algorithms. *Global* algorithms, e.g. [14], rely on iterative schemes that carry out disparity assignments on the basis of the minimisation of a global cost function. These algorithms yield accurate and dense disparity measurements but exhibit a very high computational cost that renders them unsuited to real-time applications. *Local* algorithms, e.g. [6, 8, 13, 15], also referred to as area-based algorithms, calculate the disparity at each pixel on the basis of the photometric properties of the neighbouring pixels. Compared to *global* algorithms, *local* algorithms yield significantly less accurate disparity maps but, nowadays, thanks to both research and technology advances, can run fast enough to be deployed in many real-time applications. Numerous examples of dense stereo applications which require real-time performance can be found at the web sites [2, 3].

As far as local matching algorithms are concerned, and considering the more common case of a binocular stereo imaging system, a widely adopted method [8, 15, 6] aimed at detecting unreliable matches, such for example those due to occlusions or photometric distortions, is the so called *left-right consistency constraint* [9], also referred to as *bidirectional matching* or *left-right check*. The method can be described as follows. Initially, for each point of the left image find the best match into the right image. Then, reverse the role of the two images and for each point of the right image find the best match into the left image. Finally, keep only those matches that turn out to be coherent when matching left-to-right (direct matching phase) and right-to-left (reverse matching phase). It is worth observing that in both phases the match associated with each pixel is established independently of those found at neighbouring pixels, since the other matching phase will highlight ambiguous matches. The *left-right check* has proven to be particularly effective in detecting and discarding the erroneous matches necessarily yield by area-based algorithms in presence of occlusions [7, 15, 6]. However, this approach is characterised by a significant computational cost. In fact, it requires two matching phases (direct and reverse) and, although some authors have proposed calculation schemes aimed at reducing the impact of the left-right check on the overall stereo execution time [6], in most implementations this implies doubling the computational complexity of the matching process.

This paper presents a fast *local* algorithm which enables real-time dense stereo applications on a standard Personal Computer. The algorithm is based on a matching core that detects unreliable matches during the direct matching phase and therefore does not require a reverse matching phase.

2 The proposed matching approach

We assume a binocular stereo pair and images in standard form, i.e. with corresponding epipolar lines lying on corresponding image scanlines. Should the latter assumption not be verified, a suitable transformation, known as rectification (see for example [11]), must be applied to obtain a pair of images in standard form from the original ones.

Hence, in *local* algorithms, given a point in the reference image the homologous point is selected by searching along the corresponding scanline in the other image, and within a certain disparity range, for the point that minimize (maximize) an error (similarity) function, ε, representing the degree of dissimilarity (similarity) between two small regions of the stereo pair. Unlike algorithms based on the left-right check, which rely on a direct (i.e. left-to-right) and a reverse (i.e. right-to-left) matching phase, our algorithm uses only a direct matching phase. Our approach relies on the *uniqueness constraint*, which states that a 3D point can be projected at most in one point of each image of the stereo pair, as well as on the ability of modifying disparity measurements dynamically as long as the matching process proceeds.

Let's assume that the left image is the reference, that disparity, d, belongs to the interval $[0 \dots d_{max}]$ and that the left image is scanned from top to bottom and from left to right during the matching process. The algorithm, starting from one point of left image, say $L(x - d_{max}, y)$, searches for the best candidate by evaluating function ε, within the interval $[R(x - d_{max}, y) \dots R(x, y)]$. Then, for the successive point of reference image $L(x + 1 - d_{max}, y)$ the procedure is repeated searching for the best match within $[R(x + 1 - d_{max}, y) \dots R(x + 1, y)]$. The process is then iterated for the successive points along the scanline. This procedure is outlined in Figure 1, which shows for each point of the left image belonging to the interval $[L(x - d_{max}, y) \dots L(x, y)]$ the potential matching points in the right image within the disparity range $[0 \dots d_{max}]$. Note also that in the Figure the arcs are marked with the disparity value that brings one point of the left image into the same point $R(x, y)$ of the right image.

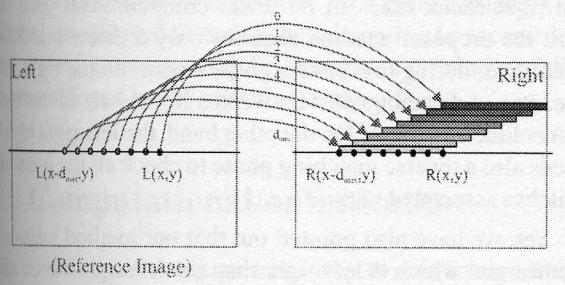

Figure 1: Matching from left to right.

Suppose now that the best match found for $L(x + \beta - d_{max}, y)$ is $R(x, y)$, with similarity score $\varepsilon(x + \beta - d_{max}, x, y)$. We adopt the notation $L(x + \beta - d_{max}, y) \rightharpoonup R(x, y)$ to indicate that this match from left to right has been established.

As it is common in area-based algorithms we use photometric properties, encoded by the error (similarity) function, as the main cue driving the matching process, even though this cue may be ambiguous, due to many causes such as for example photometric distortions, occlusions and signal noise. However, wrong matches expose inconsistencies within the set of matches already established that can be deployed to detect and discard them.

Thus, let's suppose that another point of the left image, say $L(x + \alpha - d_{max}, y)$, with $\alpha \leq \beta$, has previously matched with $R(x, y)$ with score $\varepsilon(x + \alpha - d_{max}, x, y)$. This situation, that violates the uniqueness constraint, is used in our approach to detect wrong matches. In fact, based on the uniqueness constraint we assume that at least one of the two matches, i.e. $L(x + \beta - d_{max}, y) \rightharpoonup R(x, y)$ or $L(x + \alpha - d_{max}, y) \rightharpoonup R(x, y)$, is wrong and retain the match having the better score. Thus, if the point currently analyzed $L(x + \beta - d_{max}, y)$ has a better score than $L(x + \alpha - d_{max}, y)$ (i.e. $\varepsilon(x + \beta - d_{max}, x, y) \leq \varepsilon(x + \alpha - d_{max}, x, y)$) our algorithm will reject the previous match and accept the new one. This implies that, although the proposed approach relies on a direct matching phase only, it allows for recovering from possible previous matching errors.

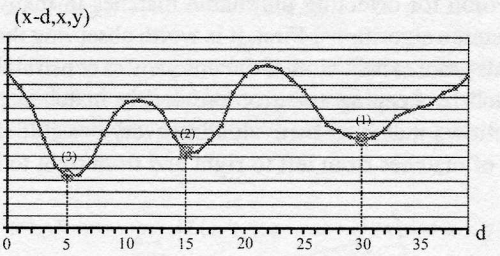

Figure 2: Scores associated with point $R(x, y)$.

The capability of our algorithm to recover from previous errors as long as better matches are found during the search is also shown in Figure 2, which plots as a function of $d \in [0 \dots d_{max}]$ all the scores between the point $R(x, y)$ of right image and the points in the reference image $[L(x - d_{max}, y) \dots L(x, y)]$ that are allowed to establish a correspondence with $R(x, y)$ (see Figure 3).

Recalling the arcs drawn in Figure 1, we can notice that smaller d values correspond to scores computed more recently while greater d values to scores computed earlier. Considering again the two matches $(1) : L(x + \alpha - d_{max}, y) \rightharpoonup R(x, y)$ and $(2) : L(x + \beta - d_{max}, y) \rightharpoonup R(x, y)$, the algorithm will discard the old one, (1), since the new one, (2), has a better score with $R(x, y)$. Moreover, if we find a new "col-

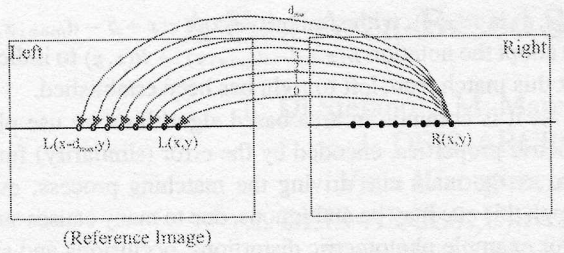

Figure 3: Potential matches in the left image for $R(x,y)$.

lision" when analysing the successive points of the left image: (3) : $L(x+\gamma-d_{max},y) \rightharpoonup R(x,y)$, the score of this new match will be compared with that associated with the current best match for $R(x,y)$, so as to retain only one single match. That is, referring to the example of Figure 2, since $\varepsilon(x+\gamma-d_{max},x,y) \leq \varepsilon(x+\beta-d_{max},x,y)$, $L(x+\beta-d_{max},y) \rightharpoonup R(x,y)$ will be discarded and the current match for $R(x,y)$ set to $L(x+\gamma-d_{max},y) \rightharpoonup R(x,y)$.

3 A comparison with bidirectional matching

In this section, we compare the proposed matching approach with bidirectional matching, being the latter the basic method for detecting unreliable matches in many area-based stereo algorithms. First, it is worth observing that both methods grant satisfaction of the uniqueness constraints: our approach by keeping trace of previously matched points, bidirectional matching by evaluating every possible combination of matches from left to right and from right to left.

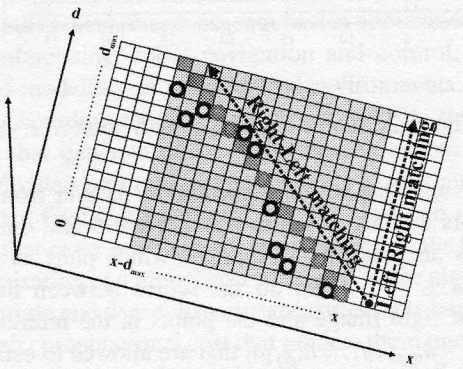

Figure 4: The search paths of bidirectional matching from the computational point of view.

From the computational point of view the behaviour of the left-right check can be described observing Figure 4, which, for a given scanline, shows on the x-axis the x-coordinates of the points of the left image that are allowed to match with point $R(x,y)$ (see Figure 3) and on the d-axis the disparity range; the score of the error (similarity) function, not shown in the Figure, would be represented on the ε-axis. The light-gray area defines all the potential matches for the points in the left image within the interval $[L(x-d_{max},y)\ldots L(x,y)]$ with points of the right image. For each point in the left image, bidirectional matching chooses in the direct phase the best score along a column in the light-gray area (the matches found when matching left-to-right have been marked with a circle in the Figure). Then, in the reverse phase, when matching $R(x,y)$ chooses the best score along the diagonal path marked with the darker-gray level: a match is accepted only if the match found along this path turns out to be one of those found when matching left-to-right. It is worth noticing that, although during the reverse phase bidirectional matching checks all of the potential matches along the path in darker-gray, the allowed ones for $R(x,y)$ turn out to be only those that in the direct phase fall in the darker-gray path (i.e. the circles lying in the darker gray path).

Conversely, our matching approach when matching left-to-right chooses the best match along a column and at the same time checks if this is also the best match among those already found falling on the darker-gray path (i.e. left-to-right matches that collide on $R(x,y)$). Looking at Figure 4, the matches cross-checked by our algorithm are only those represented by the circles lying on the darker-gray path. In our matching approach collisions are the cues to reject potential wrong matches. The matching constraint embodied into our algorithm turns out to be less tight than the left-right consistency constraint. In fact, with our method a point of the right image, $R(x,y)$, will be certainly matched if there is at least one match in the darker-gray path of Figure 4 associated with $R(x,y)$. This is not true for bidirectional matching: even though the direct phase finds a match for $R(x,y)$ lying in the darker-gray path, this may not correspond to the best match found along the darker-gray path in the reverse matching phase; in such a case $R(x,y)$ will not be matched.

The major advantage of our method with respect to the left-right check relies in its lower computational cost. In fact, the proposed method requires only a direct matching phase and, during this phase, given a point on the right image, $R(x,y)$ disambiguates between a limited set of matches that collide on $R(x,y)$. On the other hand, the left-right check needs also a reverse matching phase to check all the possible matches associated with $R(x,y)$.

Yet, we have also pointed out that our method relies on a constraint which is less tight than the left-right check and therefore tends to accept more matches. This implies that the method is intrinsically more prone to mismatches. The reliability of the disparity measurements provided by our algorithm can be improved by incorporating additional con-

straints into the basic matching core. Since we are interested in a fast algorithm, suited to real-time, PC-based stereo applications, we should introduce new constraints that could be implemented very efficiently. To this end, rather than carry out new, expensive calculations we try to exploit the information related to match reliability which is already embodied into the data computed by the matching core. Following these guidelines, we have devised a two-test strategy, *distinctiveness test* and *sharpness test*, aimed at improving match reliability based on the evaluation of the behaviour of the error function scores. This strategy, described in [5], checks if the minimum is well localized and if its magnitude is significantly smaller than the error score produced by any possible point candidate to the match.

4 The overall stereo algorithm

The overall stereo algorithm consists of four main steps. (A) The input images are normalized by subtraction of the mean values of the intensities computed in a small window centered at each pixel [8]. This allows for compensating for different settings of the cameras and-or different photometric conditions. Moreover, since matching turns out to be highly unreliable when dealing with poorly textured areas, the variance of the intensities is calculated at each pixel considering a window of the same size as that used to obtain the means. This information is used to detect regions with lack of texture [8]. (B) The normalized images are matched according to the matching approach described in Section 2, which is independent of the error (similarity) function. Currently, we use the SAD error function but other similarity measures (e.g. Sum of Squared Differences (SSD) or Normalized Cross Correlation (NCC)) could be used. (C) The reliability of the matches provided by the basic matching core is improved by means of the "distinctiveness" and "sharpness" tests. In addition, this step uses the variance map computed in the pre-processing step to reject the matches found in poorly textured areas. (D) The final step performs sub-pixel refinement of disparities. Sub-pixel accuracy ($\frac{1}{16}$ of pixel) is achieved detecting the minimum of a second degree curve interpolating the SAD scores in proximity of the minimum found by the matching core.

5 Computational optimisation

The most expensive task performed by the stereo algorithm is the computation of SAD scores, which are needed to carry out the direct matching phase. In this section we outline the optimisation techniques adopted to avoid redundant calculations. We show first the basic calculation scheme, already decribed in [6], and then propose an additional level of incremental calculation aimed at achieving further speed-up.

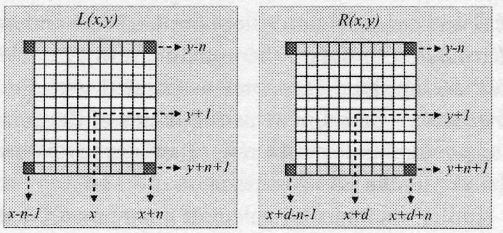

Figure 5: Recursive calculation of SAD scores.

Suppose that $SAD(x, y, d)$ is the SAD score between a window of size $(2n+1)\cdot(2n+1)$ centered at coordinates (x, y) in the left image and the corresponding window centered at $(x + d, y)$ in the right image:

$$SAD(x, y, d) = \sum_{i,j=-n}^{n} |L(x + j, y + i) - R(x + d + j, y + i)| \tag{1}$$

Observing Figure 5, it is easy to notice that $SAD(x, y + 1, d)$ can be attained from $SAD(x, y, d)$:

$$SAD(x, y + 1, d) = SAD(x, y, d) + U(x, y + 1, d) \tag{2}$$

with $U(x, y+1, d)$ representing the difference between the SADs associated with the lowermost and uppermost rows of the matching window (shown in light-gray in Figure 5):

$$U(x, y + 1, d) = -\sum_{j=-n}^{n} |L(x + j, y - n) - R(x + d + j, y - n)| + \sum_{j=-n}^{n} |L(x + j, y + n + 1) - R(x + d + j, y + n + 1)| \tag{3}$$

Furthermore, $U(x, y + 1, d)$ can be computed from $U(x - 1, y + 1, d)$ by simply considering the contributes associated with the four points shown in dark-gray in Figure 5:

$$\begin{aligned} &U(x, y + 1, d) = U(x - 1, y + 1, d) + \\ &(|L(x + n, y + n + 1) - R(x + d + n, y + n + 1)| - \\ &|L(x + n, y - n) - R(x + d + n, y - n)|) - \\ &(|L(x - n - 1, y + n + 1) - R(x + d - n - 1, y + n + 1)| - \\ &|L(x - n - 1, y - n) - R(x + d - n - 1, y - n)|) \end{aligned} \tag{4}$$

This allows for keeping complexity small and independent of the size of the matching window, since only four

elementary operation are needed to obtain the SAD score at each new point.

The computational scheme of equations (2), (4) makes use of a *vertical* recursion to obtain the SAD and an *horizontal* recursion to obtain the updating term, U. Hence, it requires storing the SAD scores associated with the previous row ($W \cdot d_r$ values, if W is the width of the image and $d_r = d_{max} + 1$ the disparity range) and the d_r values of U associated with the previous point.

A similar incremental-calculation approach has been adopted in the stereo algorithms developed at INRIA [8] and CMU [13]. However, the scheme described in [8, 13], make use of a *vertical* recursion to obtain the updating term and of an *horizontal* recursion to obtain the similarity (INRIA) or error (CMU) scores. Hence, in order to sustain the recursion the INRIA and CMU scheme requires storing the $W \cdot d_r$ values of the updating term associated with the previous row and the d_r values of the similarity-error scores associated with the previous point. To implement efficiently our matching algorithm, which is based on disambiguating between the collisions occurring while matching left-to-right along a row, when matching a point of the left image it is necessary to be able to obtain quickly the SAD scores associated with the previous points along the row. Hence, the scheme of equations (2), (4) is particularly suited to our matching algorithm: since, as the computation proceeds along a row, the scheme requires storing SAD scores to sustain the recursion, when matching a point of the left image the SAD scores of the previous points of the row are already available and can be accessed to disambiguate in case of a collision. Note that this would not be the case of the INRIA and CMU scheme, for which the values stored as the computation proceeds along a row to sustain the recursion are those of the updating term.

As shown in Section 4, the pre-processing step requires computation of the mean and variance of the two images. Considering for example the left image, and posing $N^2 = (2n+1) \cdot (2n+1)$, the mean is given by

$$\mu(x,y) = \frac{1}{N^2} \sum_{i,j=-n}^{n} L(x+j, y+i) = \frac{1}{N^2} S_1(x,y) \qquad (5)$$

while the variance can be expressed [4] as

$$\begin{aligned} \sigma^2(x,y) &= \frac{1}{N^2} \sum_{i,j=-n}^{n} L^2(x+j, y+i) - \mu^2(x,y) \\ &= \frac{1}{N^2} S_2(x,y) - \mu^2(x,y) \end{aligned} \qquad (6)$$

Since equations (5) and (6) rely on the same basic operation, namely scanning the image and summing-up intensities - or squared intensities, it can be easily verified that the computation of mean and variance can be carried out using the following schemes:

$$S_1(x, y+1) = S_1(x,y) + U_{S_1}(x, y+1) \qquad (7)$$

$$U_{S_1}(x, y+1) = \sum_{j=-n}^{n} (L(x+j, y+n+1) - L(x+j, y-n)) \qquad (8)$$

$$\begin{aligned} &U_{S_1}(x, y+1) = U_{S_1}(x-1, y+1) + \\ &(L(x+n, y+n+1) - L(x+n, y-n)) - \\ &(L(x-n-1, y+n+1) - L(x-n-1, y-n)) \end{aligned} \qquad (9)$$

$$S_2(x, y+1) = S_2(x,y) + U_{S_2}(x, y+1) \qquad (10)$$

$$U_{S_2}(x, y+1) = \sum_{j=-n}^{n} (L^2(x+j, y+n+1) - L^2(x+j, y-n)) \qquad (11)$$

$$\begin{aligned} &U_{S_2}(x, y+1) = U_{S_2}(x-1, y+1) + \\ &(L^2(x+n, y+n+1) - L^2(x+n, y-n)) - \\ &(L^2(x-n-1, y+n+1) - L^2(x-n-1, y-n)) \end{aligned} \qquad (12)$$

In both the matching and pre-processing steps it is possible to introduce a third level of incremental computation aimed at achieving additional speed-up. Both steps use the four pixels at the corners of the correlation window. Formulas 4, 9 and 12 show that these pixels contribute to two terms, say A and B, where A includes the two pixels on the left side of the correlation window and B those on the right side. Observing that term B plays the role of term A when the correlation window is shifted horizontally by $2n+1$ units, we can store, at a very small memory cost, the most recent $2n+1$ B terms so that they can be re-used $2n+1$ units later in place of the A terms. Naming T the array of the B terms, each element can be referenced with the index $\tilde{x} = x \ mod \ (2n+1)$ and thus all elements are visited each time the correlation window is shifted horizontally by $2n+1$ units. When shifting the window by one unit, a new B term is calculated while the needed A term is fetched from $T(\tilde{x})$. After both terms have been used, $T(\tilde{x})$ is updated to the newly calculated B term.

To introduce this third level of incremental computation into the pre-processing step, formula 9 for the mean calculation is rewritten as follows:

$$U_{S_1}(x, y+1) = U_{S_1}(x-1, y+1) + (L(x+n, y+n+1) - L(x+n, y-n)) - T_1(\tilde{x}) \quad (13)$$

where

$$T_1(\tilde{x}) = L(x-n-1, y+n+1) - L(x-n-1, y-n) \quad (\text{with} \quad \tilde{x} = x \; mod \; (2n+1)) \quad (14)$$

Similarly formula 12 for the variance calculation becomes:

$$U_{S_2}(x, y+1) = U_{S_2}(x-1, y+1) + (L^2(x+n, y+n+1) - L^2(x+n, y-n)) - T_2(\tilde{x}) \quad (15)$$

where

$$T_2(\tilde{x}) = (L^2(x-n-1, y+n+1) - L^2(x-n-1, y-n)) \quad (\text{with} \quad \tilde{x} = x \; mod \; (2n+1)) \quad (16)$$

In the matching step the third level of incremental computation is applied for each disparity value $d \in [0, d_{max}]$; thus, the array T grows by one dimension and formula 4 is rewritten as follows:

$$U(x, y+1, d) = U(x-1, y+1, d) + (|L(x+n, y+n+1) - R(x+d+n, y+n+1)| - |L(x+n, y-n) - R(x+d+n, y-n)|) - T(\tilde{x}, d) \quad (17)$$

$$T(\tilde{x}, d) = |L(x-n-1, y+n+1) - R(x+d-n-1, y+n+1)| - |L(x-n-1, y-n) - R(x+d-n-1, y-n)| \quad (\text{with} \quad \tilde{x} = x \; mod \; (2n+1), \quad d \in [o, d_{max}]) \quad (18)$$

The described computational optimisations can be extended easily to other error (similarity) functions such as SSD and NCC.

The most expensive portions of the optimised algorithm have been mapped on a general purpose processor with parallel, SIMD-style (Single Instruction Multiple Data) instructions (e.g. Pentium III processor with SSE technology) A detailed description of the parallel-mapping process, not reported in this paper for the sake of brevity, can be found in [5].

6 Experimental results

In this section we show the experimental results obtained using the "Tsukuba" stereo pair, from University of Tsukuba (the left image of the stereo pair is shown in Figure 6). We also compare our results with those obtained with SVS 2.0 [3, 15], which is a well-known area-based algorithm based on bidirectional matching. We discuss experimental results using the ground truth provided with the stereo pair, shown at the bottom of Figure 6.

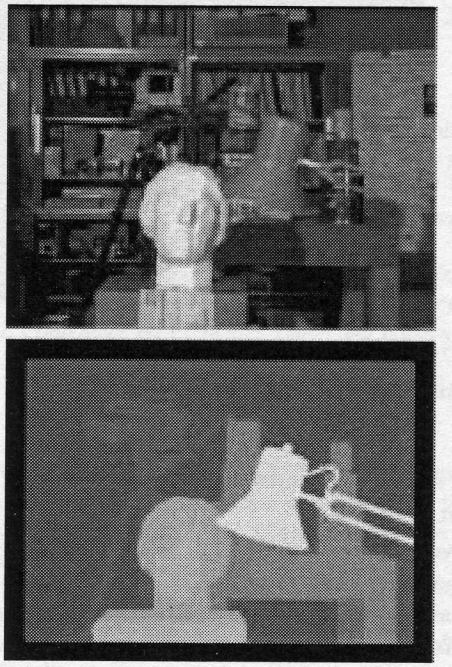

Figure 6: (Top) Left image of the "Tsukuba" stereo pair and (Bottom) ground-truth.

The "Tsukuba" stereo pair contains objects at different depths generating, several occlusions, as well as poorly-textured regions in the background, such as the wall at the top-right corner. Moreover, this stereo pair also contains some specular regions (i.e. the face of the statue and some regions of the lamp) that could render quite difficult the stereo matching process. Comparing the output of our algorithm (top of Figure 7) with the ground truth image (bottom of Figure 6) we can observe first that the rough 3D structure has been clearly recovered: the camera an its trestle on the background have been recovered almost correctly as well as the objects closer to the stereo acquisition system, such as the statue and the lamp's head. Moreover, it is worth observing that the major occlusions have been discarded (since in Figure 7 the points left unmatched are represented in red, this can be seen clearly if the paper is printed with colours or looking at the results available at our web site [1]), confirm-

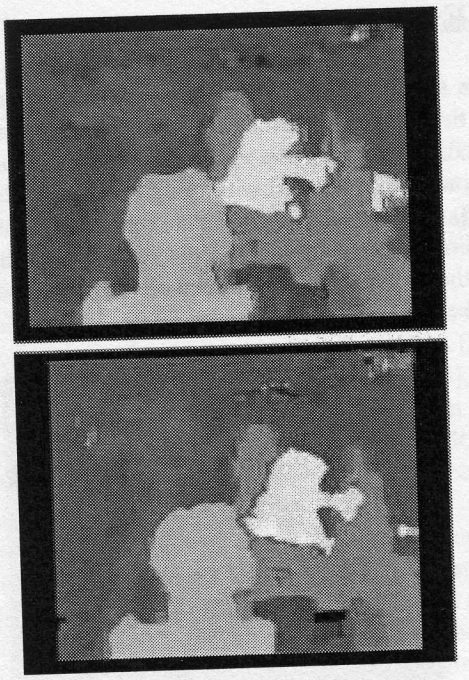

Figure 7: (Top) Disparity map computed with the proposed approach and (Bottom) with SVS 2.0.

ing the ability of our matching approach to deal with this problem.

However, some details such as for example the lamp's wire, the lamp's switch and the two roads that sustain the lamp, have vanished. Moreover the disparity map shows the *border-localisation* problem, i.e. the objects' borders are not perfectly localised with respect to their original position, as it can be seen comparing the ground truth with the disparity map computed by the algorithm. This causes clearly an unaccurate fitting of the object's silhoutte in the original image into the disparity map.

These problems are inherent to local algorithms since they depend on the method adopted to establish correspondences, which relies on the use of a local support area centered at the point under examination (i.e. the correlation window). Local algorithms behave correctly when the correlation window covers a region at constant depth but are likely to produce artifacts when the correlation window covers regions at different depths.

The vanishing of details in the disparity map arises when the details are small compared to the size of the correlation window. In this case the signal strenght embodied in the texture of the detail can be overcame by the contributions of the other points within the correlation window, resulting in a low-pass filtering effect. The *border-localisation* effect, analysed in detail in [16, 12], occurs when the correlation window is placed across regions at different depths. In such a situation, the correlation windows that are compared when trying to match a given point do not cover exactly the same points of the 3D scene; more precisely, taking one image as reference, the correlation window centered at the point under examination and placed across a depth discontinuity consists of foreground and background portions that due to the different viewpoints and the possible presence of an occlusion do not mantain the same spatial relationships within the correlation windows belonging to the other image. Thus, a foreground or a background portion of the correlation window can drive the matching process, resulting in an uncertainity in the localisation of the border. These problems can be mitigated reducing the size of the correlation window; however this solution has the side-effect of reducing the signal to noise ratio leading to more matching errors within low textured areas.

Some authors [10, 16, 12] propose local algorithms aimed at reducing the *border-localisation* problem. However, it is worth pointing out that the results provided by these algorithms are still less accurate that those generated with global, slow algorithms such as [14].

The results obtained by SVS on the "Tsukuba" image pair (bottom of Figure 7) are very similar: the rough 3D structure is recovered and the major occlusions are correctly detected. This result confirm the expected similar behavior, discussed in section 3, of the left-right constraint and of the proposed matching approach. Additional experimental results can be found at the web site [1].

Finally, we report in Table 1 some measurements aimed at assessing the speed of the two algorithms with different image sizes and disparity ranges. These measurements have been obtained on an Intel Pentium *III* processor running at 800 MHz. From Table 1 we can see that for a small disparity range (i.e. 16) SVS is always faster, much faster for small images (i.e. (320×240)) and slightly faster for bigger images. Yet, as the disparity range is increased, our algorithm gets faster than SVS, significantly faster for big images and large disparity range. For example, with 800×600 stereo pairs and a disparity range of 16 our algorithm runs at 5.56 fps while SVS at 6.96. But with the same image size and a disparity range of 80 our algorithm is nearly twice faster that SVS (i.e. 2.89 fps for our algorithm and 1.51 for SVS).

7 Conclusion

We have presented an area-based stereo matching algorithm which relies only on a left-to-right matching phase and that detects unreliable matches via "colliding matches" i.e. matches that violate the uniqueness constraint. We have compared our approach with bidirectional matching i.e. left-to-right matching followed by right-to-left matching, since this is the method adopted to detect unreliab

Algorithm (size)	$d = 16$	$d = 32$	$d = 48$	$d = 64$	$d = 80$
P.A. (320 × 240)	39.59 fps	31.25 fps	27.44 fps	25.94 fps	25.96 fps
SVS (320 × 240)	57.99 fps	33.68 fps	20.49 fps	15.31 fps	12.71 fps
P.A. (640 × 480)	8.94 fps	6.92 fps	5.77 fps	5.17 fps	4.78 fps
SVS (640 × 480)	11.99 fps	5.93 fps	4.07 fps	3.18 fps	2.54 fps
P.A. (800 × 600)	5.56 fps	4.28 fps	3.60 fps	3.18 fps	2.89 fps
SVS (800 × 600)	6.96 fps	3.65 fps	2.53 fps	1.94 fps	1.51 fps
P.A. (1024 × 768)	3.32 fps	2.56 fps	2.09 fps	1.86 fps	1.67 fps
SVS (1024 × 768)	3.79 fps	2.07 fps	1.45 fps	1.06 fps	0.78 fps

Table 1: Speed measurements (in terms of frame per second, fps) for the proposed algorithm (P.A.) and SVS 2.0.

matches in many area-based stereo algorithms conceived for real time applications. Our analysis suggests that in most practical cases the two methods should behave similarly and that our approach is potentially faster. The proposed stereo algorithm exploits massively incremental computation schemes aimed at eliminating redundant calculations.

In addition, we propose a further level of incremental calculation which avoids redundant computations that take place within the correlation window. We have shown the experimental results obtained with the proposed algorithm on the "Tsukuba" stereo pair and compared these results with the available ground-truth and with those obtained by SVS 2.0, a well-known algorithm based on bidirectional matching. Since the two algorithms yield very similar disparity maps these results confirm the effectiveness of the proposed matching approach based on the handling of colliding matches. Finally, we have reported several measurements showing that, with the exception of small disparity ranges, our algorithm is typically faster than SVS 2.0. In particular, with big images and a large disparity range our algorithm turns out to be significantly faster.

References

[1] *Experimental Results*. World Wide Web, `http://labvisione.deis.unibo.it/~smattoccia/stereo.htm`.

[2] *Point Grey Research*. World Wide Web, `http://www.ptgrey.com`, 2001.

[3] *Videre Design*. World Wide Web, `http://www.videredesign.com`, 2001.

[4] S. Changming. A fast stereo matching method. In *Digital Image Computing: Techniques and Applications*, pages 95–100, Auckland, Nov. 1997.

[5] L. Di Stefano, M. Marchionni, S. Mattoccia, and G. Neri. Fast Dense Stereo Based on the Uniqueness Constraint: Theory, Implementation and Results. Technical Report CSITE-10-01, Oct. 2001.

[6] L. Di Stefano and S. Mattoccia. Fast stereo matching for the videt system using a general purpose processor with multimedia extensions. In *Fifth IEEE International Workshop on Computer Architecture for Machine Perception*, Padova, Italy, Sept. 2000.

[7] G. Egnal and R. Wildes. Detecting binocular half-occlusions: empirical comparisons of four approaches. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 466 –473, jun 2000.

[8] O. Faugeras et al. Real-time correlation-based stereo: algorithm, implementation and applications. INRIA Technical Report n. 2013, 1993.

[9] P. Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *12th. International Joint Conference on Artificial Intelligence*, pages 1292–1298, Sydney, Aug. 1991.

[10] A. Fusiello, V. Roberto, and E. Trucco. Symmetric stereo with multiple windowing. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 14:1053–1066, 2000.

[11] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.

[12] H. Hirschmuller, P. Innocent, and J. Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *Int. Journal of Computer Vision*, 47(1-3):229–246, 2002.

[13] T. Kanade, H. Kato, S. Kimura, A. Yoshida, and K. Oda. Development of a video-rate stereo machine. In *Proc. of International Robotics and Systems Conference (IROS '95)*, volume 3, pages 95 – 100, August 1995.

[14] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *Proceedings of International Conference on Computer Vision*, 2001.

[15] K. Konolige. Small vision systems: Hardware and implementation. In *8th Int. Symposium on Robotics Research*, pages 111–116, Hayama, Japan, 1997.

[16] M. Okutomi, Y. Katayama, and S. Oka. A simple stereo algorithm to recover precise object boundaries and smooth surfaces. *Int. Journal of Computer Vision*, 47(1-3):261–273, 2002.

[17] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Microsoft Research Technical Report MSR-TR-2001-81, 2001.