Efficient Sampling of Disparity Space for Fast And Accurate Matching

Jan Čech and Radim Šára
Center for Machine Perception
Czech Technical University, Prague, Czech Republic
{cechj,sara}@cmp.felk.cvut.cz

Abstract

A simple stereo matching algorithm is proposed that visits only a small fraction of disparity space in order to find a semi-dense disparity map. It works by growing from a small set of correspondence seeds. Unlike in known seed-growing algorithms, it guarantees matching accuracy and correctness, even in the presence of repetitive patterns. This success is based on the fact it solves a global optimization task. The algorithm can recover from wrong initial seeds to the extent they can even be random. The quality of correspondence seeds influences computing time, not the quality of the final disparity map. We show that the proposed algorithm achieves similar results as an exhaustive disparity space search but it is two orders of magnitude faster. This is very unlike the existing growing algorithms which are fast but erroneous. Accurate matching on 2-megapixel images of complex scenes is routinely obtained in a few seconds on a common PC from a small number of seeds, without limiting the disparity search range.

1. Introduction

Traditional area-based dense stereoscopic matching algorithms perform exhaustive search of the entire disparity space, i.e. they need to compute a correlation statistic for all putative correspondences [18]. Although efficient implementations for computing most of the commonly used statistics (SSD, SAD, NCC) are known [20], this is still one of the most expensive phases of stereo matching.

To avoid visiting the entire disparity space, algorithms were proposed that greedily grow corresponding patches from a given set of reliable seed correspondences. Such algorithms assume that neighboring pixels have similar disparity, not exceeding disparity gradient limit [15] or a similar constraint.

The principle of growing a solution from initial seeds had long been known in segmentation [6]. The first algorithms using this principle in stereo were proposed in photogrammetric community: by Otto and Chau [14], O'Neill and Denos [13], and by Kim and Muller [7].

Later, Lhuillier and Quan [9] employed the uniqueness constraint and proposed an algorithm both with and without the epipolar constraint in which images play a symmetric role. This work has later been used in a 3D surface reconstruction pipeline [10]. The basic idea is to grow contiguous *components*[1] in disparity space from initial correspondence seeds sorted in decreasing value of image similarity and to stop the growth process at image pixels where uniqueness constraint would be violated. The growth occurs in the neighborhood of previous matches in disparity space. This creates new seeds that are put to the priority queue. A decision on match acceptance is never revised. This inherent greediness of the algorithm may cause a complete failure in the presence of repetitive texture in the scene, as will be discussed later.

Independently, Chen and Medioni [2] proposed an alternative scheme in which the growth is not constrained by uniqueness. The best-first strategy is not used and the seeds are taken in arbitrary order. When a match of better image similarity is found at a given pixel, it overrides the previous match but it does not grow further, hence the correction is only local and there is no ability to follow a new disparity component of high image similarity. If the seeds in the queue are processed in a different order, (very) different results are obtained. Moreover, images in this algorithm do not have a symmetric role, which means the resulting matching violates uniqueness constraint.

The disadvantage of both approaches is that the decision on a match is local in the sense that other matches do not influence it (no global optimization is involved).

The following two works use variations of the above methods which means they suffer from the same drawbacks, especially in the presence of repeated structures. These methods are interesting not because of the baseline growth

[1] The term *disparity component* has been coined in [1].

mechanism but because of improvements in other respects.

The first, by Zeng et al. [22, 23] uses the best-first strategy in a multi-image algorithm that replaces the small pixelwise growth increments by an optimal choice of a whole surface patch extending the currently found 3D segment. Final selection is done by marching cube tracing in 3D which may not be able to recover from the earlier stage errors, especially in complex scenes.

The second, by Megyesi et al. [11] uses the Otto and Chau's algorithm with adaptive affine deformation of the domain of image similarity statistic, where the affine parameters are estimated from surface normals which are propagated within the growth process.

In stereo literature, also *progressive algorithms* have been proposed [24, 21, 3]. Earlier matches guide the subsequent matching by postponing ambiguous decision until enough confidence is accumulated to resolve the ambiguity. Although this may look like a growth from seeds, it does not explicitly use spatial coherence: the growth does not necessarily occur in the immediate neighborhood of previously accepted matches. These algorithms never revise the decision on match acceptance, as well. They need to visit a large fraction of disparity space (satisfying a set of constraints) before making the first decision. This is unlike the true growing algorithms (whose decision is *online*).

A problem common to all known growing algorithms is their reliance on high-quality seeds. In dense stereo, this means there must be at least one seed in each true disparity component, which is very hard to fulfill. To our surprise, it turned out that there is a suitable algorithm that has the ability to recover from errors, which means that good-quality initial seeds are not needed: In fact, even quite complex scenes can be matched from a few *random* seed correspondences, as will be discussed in Sec. 3.

To overcome drawbacks of the discussed methods, we temporarily forego uniqueness constraint and propose an algorithm which, under a theoretically well-grounded rule, keeps growing disparity components regardless of their overlap in disparity space. The rule facilitates the efficiency of the growth process by stopping it at loci where the final, optimal result cannot be improved. As a result, it is not necessary to visit a large fraction of disparity space to obtain optimal solution. We then solve a global optimality task by a fast robust matching algorithm that selects among the competing components in disparity space. This leads to a significant improvement in the quality of the result at only a small computational cost and it is equivalent to the ability to revise decisions made at the growth phase.

The paper is structured as follows: Sec. 2 formulates the dense stereoscopic matching task as a global discrete optimization problem and describes an efficient disparity component growth algorithm. Experimental validation comparing the proposed algorithm with a baseline algorithm is done in Sec. 3, where we discuss efficiency and the ability to deal with repetitive image structures correctly and the ability to find a large number of disparity components from a small set of seeds, even if they are random. Sec. 4 concludes the paper and hints some extensions of the work.

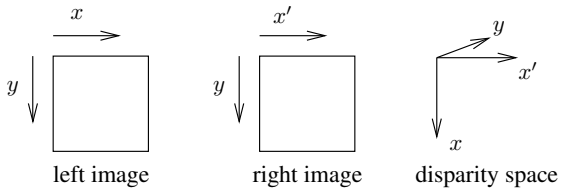


Figure 1. Coordinate system used in this paper. The y is a common row coordinate in rectified images. The x, x' are the column coordinates in the left and right image, respectively.

2. Matching Algorithm

To help understanding the proposed algorithm, we make a thought decomposition of the matching task to two phases: (1) an unconstrained growth of disparity components from an initial set of seeds and (2) an optimal matching working with the set of components found in the first step. We will then show that some of the work of the second phase can already be done during the component growth without loosing optimality of the algorithm, while significantly speeding up the first phase.

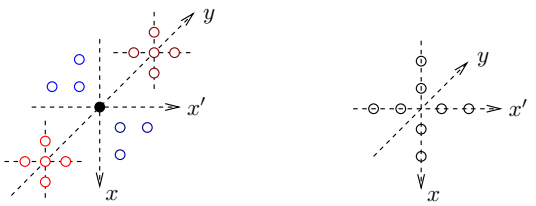
To simplify the description of the algorithm, we assume a pair of horizontally rectified stereo images is used. Generalization to unrectified images is possible but it will not be discussed in the present paper. We therefore assume we are working with matching table. It represents a 3D discretized disparity space in which each element (x, x', y) denotes a possible correspondence $(x, y) \leftrightarrow (x', y)$, see Fig. 1. Each matching table element (x, x', y) may be associated with some parameters θ modeling relative distortion of image neighborhoods. The parameters θ may be updated during the growth process to accommodate to the slant of the 3D surface, as in [11, 14, 13, 7]. This is important in wide-baseline stereo. For simplicity, we omit the distortion model here.

2.1. Disparity Component Growth

Suppose we are given an unsorted list of disparity seeds $\mathcal{S}$. Each seed is a point in disparity space, $\mathbf{s} = (x, x', y)$. Its neighborhood $\mathcal{N}(\mathbf{s})$ in disparity space consists of 16 points constructed from four sub-sets $\mathcal{N}_1(\mathbf{s}) \cup \mathcal{N}_2(\mathbf{s}) \cup \mathcal{N}_3(\mathbf{s}) \cup \mathcal{N}_4(\mathbf{s})$, see Fig. 2(a) where we use the same colors for:

$$\mathcal{N}_1(\mathbf{s}) = \big\{(x-1,\, x'-1,\, y),\, (x-2,\, x'-1,\, y),\, (x-1,\, x'-2,\, y)\big\},$$

$$\mathcal{N}_2(\mathbf{s}) = \big\{(x+1,\, x'+1,\, y),\, (x+2,\, x'+1,\, y),\, (x+1,\, x'+2,\, y)\big\},$$



(a) neighborhood $\mathcal{N}(\mathbf{s})$ (b) inhibition zone $\mathcal{X}(\mathbf{s})$

Figure 2. Disparity space neighborhood used in this paper (a). The X-inhibition zone for $\mathbf{s} = (x, x', y)$ (b).

$$\mathcal{N}_3(\mathbf{s}) = \big\{(x,\, x',\, y-1),\, (x \pm 1,\, x',\, y-1),\\ (x,\, x' \pm 1,\, y-1)\big\},$$

$$\mathcal{N}_4(\mathbf{s}) = \big\{(x,\, x',\, y+1),\, (x \pm 1,\, x',\, y+1),\\ (x,\, x' \pm 1,\, y+1)\big\}.$$

The neighborhood is selected so as to limit the magnitude of disparity gradient to unity and to improve the ability to follow a disparity component even if the image similarity peak falls in between pixels in the matching table. This improves performance in both the baseline and the proposed algorithms (see below).

Assuming similarity is computed from small image windows around pixels (u, v) and (u', v) by e.g. the normalized cross-correlation, we prepare an empty matching table $\mathcal{T}$ and start growing disparity components by drawing an arbitrary seed $\mathbf{s}$ from $\mathcal{S}$, adding it to $\mathcal{T}$, individually selecting the best-similarity neighbors $\mathbf{q}_i$ over its four subneighborhoods $\mathcal{N}_i(\mathbf{s})$:

$$\mathbf{q}_i = (u, u', v)_i = \underset{(x,x',y)\in\mathcal{N}_i(\mathbf{s})}{\operatorname{argmax}} \operatorname{simil}(x, x', y),$$

and putting these neighbors $\mathbf{q}_i$ to the seed list if their inter-image similarity exceeds a threshold τ. Hence, up to four new seeds are created. If we draw a seed from the list $\mathcal{S}$ that is already a member of the matching table, then we discard it. The growth must stop in a finite number of steps by exhausting the list $\mathcal{S}$. The output from the growth phase is a partially filled matching table whose connected regions in 3D represent disparity components grown around the initial seeds. Note that disparity components obtained this way are nothing more than contiguous segments in disparity space. In the extreme case when $\tau = -\infty$ the entire disparity space is filled by a single component grown from the first seed.

Obviously, such growth is not a very efficient way of selecting high-similarity tentative matches. Instead, we see it only as an elementary mechanism for traveling in disparity space. This phase has been introduced just to show correctness of the entire proposed algorithm which will be described shortly. Note, finally, that the order of selecting the seeds from the list $\mathcal{S}$ is arbitrary, so far.

We refer to the above procedure as *unconstrained growth*. Its output is not a matching. This is the reason why various authors proposed a modification which stops the growth whenever a violation of uniqueness constraint is detected. This already requires drawing the seeds in the order of their image similarity. The result of this modification is shown in pseudo-code[2] as Alg. 1. We call it the *baseline growth algorithm* in this paper. It is very close to Lhuiller's algorithm [9]. Note that both images have a symmetric role. We will show in Sec. 3 that Alg. 1 is fast but its performance is not very good. The reason for its bad performance is that it does not solve any reasonable optimization task. In the next paragraph, we show how to incorporate a formal discrete optimization task into the algorithm. It will turn out that the necessary changes are small and still the performance improves dramatically (at the cost of a small increase in computational complexity).

Algorithm 1 The Baseline Growing Algorithm

Require: Rectified images $\mathbf{I}_l$, $\mathbf{I}_r$, initial correspondence seeds $\mathcal{S}$, image similarity threshold τ.

1.1: Compute similarity $\operatorname{simil}(\mathbf{s})$ for every seed $\mathbf{s} \in \mathcal{S}$.
1.2: Initialize empty matching table $\mathcal{T} := \emptyset$.
1.3: **repeat**
1.4: Draw the seed $\mathbf{s} \in \mathcal{S}$ of the best similarity $\operatorname{simil}(\mathbf{s})$.
1.5: **for** each of the four best neighbors

$$\mathbf{q}_i = (u, u', v) = \underset{\mathbf{t}\in\mathcal{N}_i(\mathbf{s})}{\operatorname{argmax}} \operatorname{simil}(\mathbf{t}),\ i \in \{1,2,3,4\}$$

do
1.6: $c := \operatorname{simil}(\mathbf{q}_i)$.
1.7: **if** $c \geq \tau$ **and** $\mathcal{X}(\mathbf{q}_i) \notin \mathcal{T}$ **then**
1.8: Update the matching table $\mathcal{T} := \mathcal{T} \cup \{\mathbf{q}_i\}$ and
1.9: the seed queue $\mathcal{S} := \mathcal{S} \cup \{\mathbf{q}_i\}$.
1.10: **end if**
1.11: **end for**
1.12: **until** $\mathcal{S}$ is empty.
1.13: **return** matching as a partially filled matching table $\mathcal{T}$.

2.2. Matching

Let $\mathbf{s} = (u, u', v)$ be an element of the matching table $\mathcal{T}$ obtained from the unconstrained growth procedure. Let $(:, u', v)$ represent all elements (w, u', v) such that $w \neq u$ and the colon in $(u, :, v)$ has a similar meaning. The set of all $(:, u', v)$ together with all $(u, :, v)$ will be called the X-inhibition zone of $\mathbf{s}$ and denoted as $\mathcal{X}(\mathbf{s})$, see Fig. 2(b). Note that $\mathbf{s} \notin \mathcal{X}(\mathbf{s})$ and that $\mathbf{q} \in \mathcal{X}(\mathbf{s}) \Leftrightarrow \mathbf{s} \in \mathcal{X}(\mathbf{q})$. The relation $\mathbf{q} \in \mathcal{X}(\mathbf{s})$ represents the relation of occlusion [17]. We say element $\mathbf{q} \in \mathcal{T}$ is a *competitor* to $\mathbf{s} \in \mathcal{T}$ if $\mathbf{q} \in \mathcal{X}(\mathbf{s})$ and it has better image similarity, i.e. $\operatorname{simil}(\mathbf{q}) \geq \operatorname{simil}(\mathbf{s})$. Element $\mathbf{q}$ is a *strict competitor* to element $\mathbf{s}$ if $\mathbf{q} \in \mathcal{X}(\mathbf{s})$ and $\operatorname{simil}(\mathbf{q}) > \operatorname{simil}(\mathbf{s}) + \mu$, where μ is called the *stability margin*.

The matching task is done by solving a graph-theoretic problem known as the *maximum strict sub-kernel* (SSK) [17, 16]. The basic SSK algorithm, which can be

[2] The X-inhibition zone $\mathcal{X}(\mathbf{q}_i)$ in Step 1.7 is defined early in Sec. 2.2.

used for our problem class and which we call the *dominant element reduction algorithm* works as follows. Given matching table $\mathcal{T}$, partially filled with disparity components from the unconstrained growth phase, one finds a *dominant element* $\mathbf{s}$ of the table whose image similarity satisfies

$$\text{simil}(\mathbf{s}) > \max_{\mathbf{t}\in\mathcal{X}(\mathbf{s})} \text{simil}(\mathbf{t}) + \mu, \tag{1}$$

where μ is the stability margin. If there was no dominant element, the algorithm stops. Next, given the dominant element $\mathbf{s}$, one removes all elements $\mathcal{X}(\mathbf{s})$ from $\mathcal{T}$. In the reduced table, new dominant element is found and the reduction process is repeated. The procedure finishes in a finite number of steps. The fully reduced table $\mathcal{T}$ contains a set that is the largest (i.e. maximum) one-to-one matching that is a SSK. It can be proved that our problem always has at most one maximum SSK and that the above algorithm is able to find it or confirm there is none [17]. The defining quality of a SSK is its *strict stability*, which can be considered a global 'optimality' property in the following sense: Let $\mathcal{T}$ be the set of all elements in the initial matching table. Let $\mathcal{K}$ be a subset of $\mathcal{T}$. We say $\mathcal{K}$ is strictly stable if for every $\mathbf{p}, \mathbf{q} \in \mathcal{K}$ it holds that $\mathbf{p} \notin \mathcal{X}(\mathbf{q})$ (i.e. $\mathcal{K}$ is a one-to-one matching) and every element $\mathbf{s} \in \mathcal{K}$ is strictly stable with respect to $\mathcal{K}$. An element $\mathbf{s} \in \mathcal{K}$ is strictly stable wrt $\mathcal{K}$ if every competitor $\mathbf{q} \in \mathcal{T}$ to $\mathbf{s}$ has a strict competitor $\mathbf{t}$ in $\mathcal{K}$. Hence, SSK is the only set that is strictly stable. Note that a SSK can be incomplete. In the extreme case it can be empty if there was no dominant element. Incompleteness is a necessary prerequisite for *robustness*. See [17] for a discussion of properties of SSK related to robustness.

The paper [17] formulates the SSK problem in rigorous graph-theoretical language and should be referred to for necessary conditions under which the algorithm is valid (for our problem it is valid).[3] The above algorithm directly performs *guiding* or the *least commitment strategy*, as discussed in [24]: most reliable decisions are made prior to unreliable ones that wait until the set of putative solutions becomes more constrained.[4] The accuracy of stereoscopic matching based on the SSK is studied in [8].

Besides the dominant element reduction algorithm, there is another algorithm for finding an SSK, which is more suitable for our matching problem and which produces an equivalent result [16, 17]. It has two phases: The first phase runs as in the dominant element reduction algorithm but with $\mu = 0$ and with the dominance test inequalities not sharp (in such case we are obtaining *weakly dominant elements*). We call its result a *weak kernel* (WK). It can be shown that maximum SSK is always a subset of a WK, irrespective of the order of processing the weakly dominant elements [17]. In the second phase, the WK is converted to a maximum SSK as follows:

Require: The output $\mathcal{T}$ from unconstrained growth, the WK $\mathcal{K} \subseteq \mathcal{T}$.
1: **loop**
2: Find an element $\mathbf{q} \in \mathcal{T}$, $\mathbf{q} \notin \mathcal{K}$ such that
$$\mathcal{X}(\mathbf{q}) \cap \mathcal{K} \text{ is empty } \textbf{or}$$
$$\text{simil}(\mathbf{q}) + \mu \geq \max_{\mathbf{t}\in\mathcal{X}(\mathbf{q})\cap\mathcal{K}} \text{simil}(\mathbf{t}).$$
3: **if** no such $\mathbf{q}$ was found **then**
4: **return** reduced WK $\mathcal{K}$
5: **else**
6: Remove $\mathbf{q}$ and $\mathcal{X}(\mathbf{q})$ from $\mathcal{T}$ and $\mathcal{K}$.
7: **end if**
8: **end loop**

The $\mathbf{q}$ in Step 2 are called the *converting elements*. Obviously, their neighbors in $\mathcal{X}(\mathbf{q}) \cap \mathcal{K}$ violate the strong stability condition. Correctness of this algorithm has been proved [17]. A fast algorithm for simultaneous finding WK and its progressive conversion to SSK, is described in [16].

The formulation given here clearly shows what to do in the disparity component growth procedure: Instead of stopping whenever the uniqueness constraint would be violated as in Step 1.7 of Alg. 1, we have to continue the growth unless image similarities of the overlapping disparity components differ more than μ. Hereby, the components that cannot survive the final competition are not grown further. The resulting modification of Alg. 1 is shown in pseudocode as Alg. 2. The difference is in Step 2.7 and in some inexpensive bookkeeping in Step 2.10. The difference is seemingly subtle but it has a strong impact on the ability to reject some of the bad matches, including some of the seeds, as will be shown in Sec. 3.

Of course, the partially filled matching table output from Alg. 2 is not yet a matching because of the overlap of disparity components in matching table $\mathcal{T}$ which violates the uniqueness constraint. The overgrown components undergo a competition in a subsequent selection process. To this end the output from Alg. 2 is processed by the dominant element reduction algorithm discussed above or by the equivalent WK conversion algorithm. We use an implementation that has been proposed for semi-dense stereo matching [16] and that does not require a fully populated matching table. This final matching is computationally very efficient because of the sparsity of $\mathcal{T}$. Our experiments show it takes less than 15% of the overall computing time.

[3] Strict sub-kernel is a general notion valid for oriented graphs, in which the graph structure represents the structure of the underlying problem (the structure of constraints) and the orientation represents evidence (data) [17]. The notion of SSK is related to the well-known Stable Marriage and Stable Roommates Problems [5].

[4] Note that the fact the above algorithm proceeds in this way is only due to the special structure of our matching problem. In more general graph orientations, the SSK algorithm is more complex and the general problem of finding a SSK is NP-complete [17]. It can be shown that when $\mu = 0$ the SSK approximates the max-sum independent vertex set problem within a factor of two (in our problem) [17].

Algorithm 2 The Proposed Growing Algorithm

Require: Rectified images $\mathbf{I}_l$, $\mathbf{I}_r$, initial correspondence seeds $\mathcal{S}$, image similarity threshold τ, and margin μ.

2.1: Compute image similarity for all seeds $\mathbf{s} \in \mathcal{S}$.
2.2: Initialize matching table $\mathcal{T} := \emptyset$, and auxiliary arrays $\mathbf{C}_{\text{best}}(:,:) := -\infty$, $\mathbf{C}'_{\text{best}}(:,:) := -\infty$.
2.3: **repeat**
2.4: Draw the seed $\mathbf{s} = (x, x', y) \in \mathcal{S}$ of the best image similarity $\text{simil}(\mathbf{s})$.
2.5: **for** each of the four best neighbors

$$\mathbf{q}_i = (u, u', v) = \underset{\mathbf{t} \in \mathcal{N}_i(\mathbf{s})}{\text{argmax}}\, \text{simil}(\mathbf{t}),\ i \in \{1, 2, 3, 4\}$$

do
2.6: $c := \text{simil}(\mathbf{q}_i)$.
2.7: **if** $c \geq \tau$ **and** $\mathbf{q}_i \notin \mathcal{T}$ **and**

$$c + \mu \geq \min\{\mathbf{C}_{\text{best}}(u, v),\ \mathbf{C}'_{\text{best}}(u', v)\}$$

then
2.8: Update the matching table $\mathcal{T} := \mathcal{T} \cup \{\mathbf{q}_i\}$,
2.9: the seed queue $\mathcal{S} := \mathcal{S} \cup \{\mathbf{q}_i\}$,
2.10: and the best image similarities

$$\mathbf{C}_{\text{best}}(u, v) := \max\{c,\ \mathbf{C}_{\text{best}}(u, v)\},$$
$$\mathbf{C}'_{\text{best}}(u', v) := \max\{c,\ \mathbf{C}'_{\text{best}}(u', v)\}.$$

2.11: **end if**
2.12: **end for**
2.13: **until** $\mathcal{S}$ is empty.
2.14: **return** table $\mathcal{T}$ with traced-out disparity components.

2.3. Implementation Notes

The matching table $\mathcal{T}$ of Alg. 1 is implemented as two 2D arrays of the input image sizes, each containing a match index to the other image. The second 2D array is needed for a fast check of the symmetric uniqueness constraint. The correlation map is stored separately. This is possible because there is at most one match per pixel.

In the proposed algorithm Alg. 2, the data structure for $\mathcal{T}$ is more complicated, since the uniqueness does not hold during the growth process. The $\mathcal{T}$ is very sparse, therefore, it is implemented as a 2D array of binary search trees with disparities as the keys. Inserting an element and the test on element's presence both take logarithmic time.

3. Experiments

We first demonstrate the principal differences between the baseline and the proposed algorithms on synthetic data and then compare their performance on some real data and on a ground-truth dataset.

We use Moravec's NCC [12] on 5×5 window as image similarity statistic in all experiments. The default parameters of the algorithms are $\tau = 0.6$, $\mu = 0.1$. Unless stated otherwise, we use a simple pre-matcher to obtain initial seeds which we call *Harris seeds*: we take all correspondences of Harris interest points whose image similarity over 5×5 window exceeds a threshold of 0.9. Note that the Harris seeds are not necessarily a one-to-one matching.

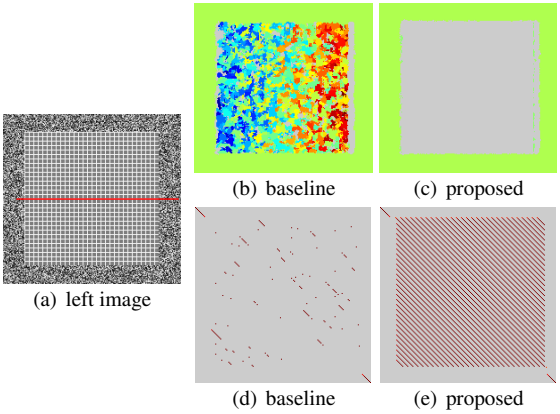


Figure 3. The ability of the baseline and proposed algorithms to handle repetitive pattern. Disparity maps (b,c) and cross-sections of disparity space (d,e) for the red line in the image.

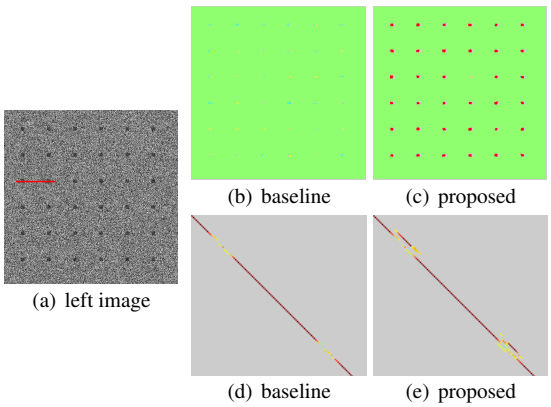
Disparity maps are shown in color: colder colors code smaller disparities, warmer colors larger disparities, gray areas have unassigned disparity. CPU times are measured on a PC C2 2.4 GHz. Our code combines Matlab and C++.

3.1. Basic Behavior of The Algorithms

We show that the proposed algorithm can handle repetitive patterns unlike in the baseline algorithm and that it has a greater ability to find all disparity components, even from a small set of random seeds. Synthetic scenes in this experiment were piecewise planar random dot 500×500 pixel stereograms.

Repetitive Pattern. The scene in Fig. 3(a) consists of a foreground plane with a repetitive texture in front of a randomly textured background plane. Harris seeds are used. Disparity map from the baseline algorithm in Fig. 3(b) is a set of patchy mismatches. All the patches have high correlation and are too large to be filtered out by any kind of post-processing. The proposed algorithm grows all seeds into mutually competing components shown in Fig. 3(e) in a cross-section of matching table $\mathcal{T}$ marked red in Fig. 3(a), the cross-section shows similarity values, grey are unoccupied elements in $\mathcal{T}$. The final robust matching algorithm then correctly labels the repetitive area as ambiguous, assigning no disparity there, see Fig. 3(c). The bad behavior in the baseline algorithm is due to a forced stop by the 'accept the first match for a pixel' in Step 1.7. As a result, the components found in disparity space are of high image similarity but they are only partial and the information on ambiguity is lost, see Fig. 3(d).

The Capture of All Disparity Components. The scene in Fig. 4(a) consists of 36 planar patches of 10×10 pixels in front of a planar 500×500 pixel background. The



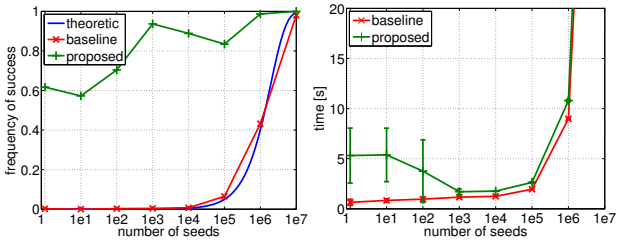
(a) left image (b) baseline (c) proposed (d) baseline (e) proposed

Figure 4. The ability of the baseline and proposed algorithms to find all disparity components (small patches). The disparity space cross-sections (d,e) are shown for the red line in the image.

foreground-to-background disparity difference is five pixels. About 1700 Harris seeds are used. We used $\tau = -\infty$ to promote growth of disparity components in both algorithms. The baseline algorithm assigned correct disparity to no foreground component in Fig. 4(b), unlike the proposed algorithm which missed only 1 of 36 in Fig. 4(c). We conclude the proposed algorithm has the ability to locate a high-correlated disparity component even if there is no seed in it. This is again due to the ability to temporarily forego uniqueness constraint, the benefit of which illustrates Fig. 4(e), as opposed to the baseline algorithm in Fig. 4(d): The behavior helps bridge the gap between the components over a set of elements with low image similarity. Clearly, even a seed out of any true component may give rise to a path that ends up on a high-similarity component. Greater μ helps encourage this behavior. When $\mu = \infty$, the entire matching table is visited and all disparity components are found. Nevertheless, our experiments confirm that $\mu = 0.1$ suffices in practice.

For quantitative evaluation, we performed a randomized experiment. A 500×500 pixel image with a *single* 10×10 pixel patch in a 5-pixel distance from a background is used in a repeated experiment with a set of n uniformly distributed *random* seeds. A thousand trials were performed for each n. The relative frequency of finding the small disparity component is our measure of success as a function of n. The result is shown in Fig. 5(a) for the baseline (red) and the proposed (green) algorithms. We can see that the ability of the proposed algorithm to find the small component is indeed much larger for even a small number of seeds. The baseline algorithm has a negligible ability to find a disparity component unless the seed is located directly in it. The non-monotonic behavior of the proposed algorithm is due to a high number of wrong correspondences clogging the matching table and preventing growth over bridges of low image similarity when μ is small.

The probability of hitting the small component with a single random seed is[5] $p = 64/500^3 \approx 5 \cdot 10^{-7}$. The probability the component is hit by at least one of n random seeds is $P = 1 - (1 - p)^n$, which is shown as the blue curve in Fig. 5(a). Hence, the baseline algorithm experiment correspond with the theory quite well which shows the experimental method is correct.



(a) (b)

Figure 5. Random sampling of disparity space. The relative frequency of finding a small disparity component in the foreground (a) and the total computational time as a function of the number of random initial seeds n (b).

We also measured the CPU time of the algorithm as a function of the number of random initial correspondences, as shown in Fig. 5(b), where errorbars show standard deviation. The proposed algorithm is less efficient for a small number of seeds because it has to travel over large subset of the disparity space. Both algorithms are less efficient when the number of seeds is too large. By comparing the plots in Fig. 5(a) and 5(b) we can see the proposed algorithm finds a solution of similar density in shorter time.

3.2. Results on Real Scenes

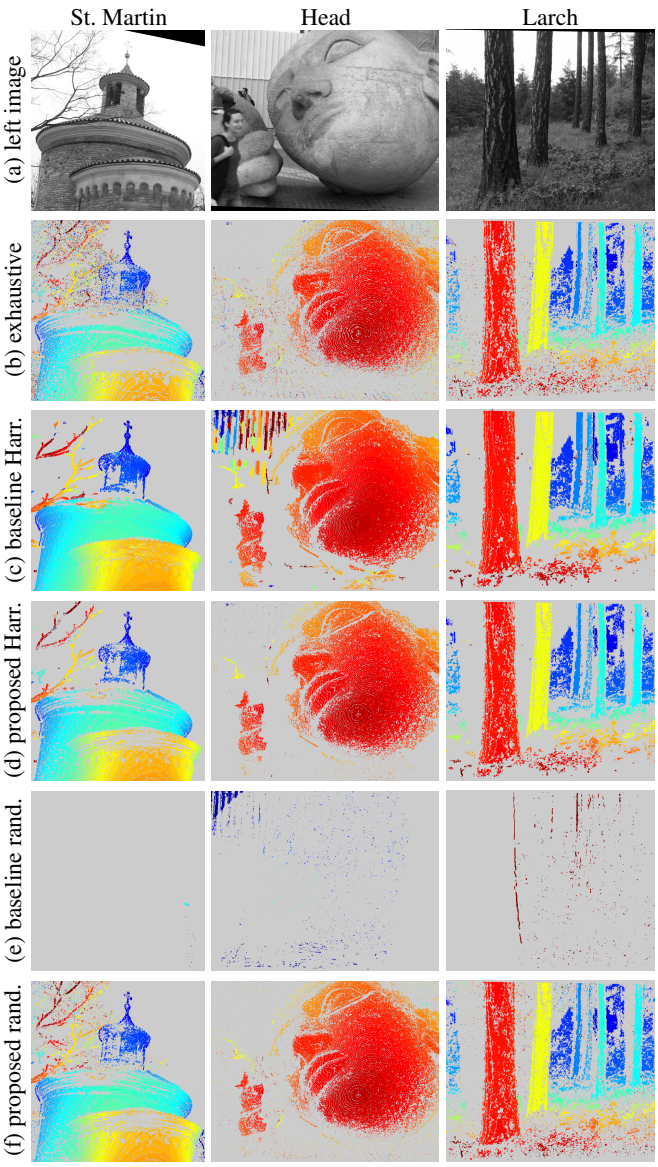
We tested the algorithms on several complex scenes to confirm the predictions on their behavior from the synthetic-data experiment and to show the proposed algorithm is indeed fast in practice. We did not limit disparity search range for any of the three algorithms in this experiment.

Results are shown in Fig. 6, where the row (b) shows disparity maps from the WK conversion algorithm which works with fully populated matching table, as described in [16] (the variant with no ordering constraint was used). We refer to this as the *exhaustive search algorithm.*

CPU timings are shown in the table in Fig. 6, together with the fraction of matching table that was visited by the respective algorithm. We initialized the growing algorithms by Harris seeds (as described above) and by random seeds.

Harris seeds. On the *St. Martin* scene, the proposed algorithm (d) achieves lower density but the holes occur in locations where the baseline algorithm (c) is erroneous (several-pixel errors in disparity in the cornice regions). Note the proposed algorithm grew the tree branches correctly, unlike the baseline algorithm.

[5]The high-correlation component is just about 8×8 pixels for a 5×5 correlation window.



	St. Martin		Head		Larch	
	t [sec]	f [%]	t [sec]	f [%]	t [sec]	f [%]
exhaustive	1023.0	100	889.0	100	182.0	100
baseline Harris	3.8	0.030	2.7	0.032	1.1	0.057
proposed Harris	7.9	0.048	9.1	0.100	1.6	0.076
baseline random	3.9	0.046	2.8	0.045	1.1	0.088
proposed random	31.2	0.320	30.0	0.510	11.9	0.680

Figure 6. Real complex scenes. The proposed algorithm is more accurate but slower than the baseline algorithm. The table shows CPU times t and the fraction f of matching table that was visited. The image sizes are 1.8 Mpx, 1 Mpx, and 0.5 Mpx; Approximately 2000 Harris (c,d) and 10 random (e,f) seeds were used.

There is a strong repetitive pattern in the *Head* scene due to the corrugated iron fence behind the sculpture. The baseline algorithm (c) suffers from illusions there as well as on the regular cobblestone paving below the sculpture.

The *Larch* is a scene of great depth and large occlusions. Results look visually similar, but the baseline algortihm (c) makes a few small mismatches at occlusion boundaries.

Random seeds. For all scenes, we used 10 random, i.e. wrong, seeds. We set $\tau = -\infty$, and deleted all disparities whose image similarity dropped below 0.6 in the final map.

The experiment shows that the proposed algorithm (f) can indeed obtain a dense map from a very small number of random initial seeds, showing their quality need not be high to succeed. The bad quality of the seeds just increases the runtime, as clearly visible in the table in Fig. 6. In a repeated experiment, the proposed algorithm (f) always succeeded unlike the baseline algorithm (e) which always failed finding any correct disparity component.

The results from the exhaustive search algorithm are dense, large ambiguous regions are correctly identified, but there are more mismatches unlike in the proposed algorithm. This is caused by a higher number of competing putative correspondences whose image similarity is high due to statistical fluctuations of the NCC statistic.

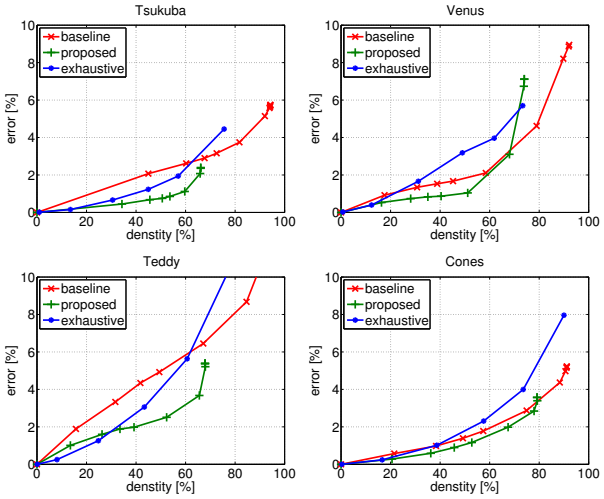


Figure 7. Middlebury new test results: ROC curves, error rate in non-occluded regions versus matching density.

3.3. Middlebury Dataset

Results on the standard new Middlebury dataset [18] are shown in Fig. 7. For both baseline and proposed algorithms, the ROC curves were obtained using the same method as in [4], spanning the $\tau \in [-1, 1]$. For the proposed algorithm, we set $\mu = 0.05$. The exhaustive search algorithm has parameters of a similar meaning. We can see the proposed algorithm is consistently better than baseline algorithm producing less errors at the same density. The difference in matching quality is large on the Tsukuba and Teddy scenes, due to ambiguous repetitive structures which are correctly handled by the proposed algorithm. On the Venus, the error of the proposed algorithm for high densities is

higher than in the baseline algorithm, since the simple planar scene is well suited for the greedy baseline algorithm. We observed that most of the errors in the above algorithms occur at occluding boundaries. Image pre-segmentation as e.g. in [21] would help reduce this error. Naturally, these results are inferior to semi-dense methods using a global optimality in the MAP sense, e.g. [3, 4, 19]. But the results are comparable to the exhaustive algorithm [16], confirming the efficiency of the disparity space sampling. The average time on Middlebury images is about 1 sec. The [3, 4] report faster times, but it is not clear if they count similarity computation and if they limit the disparity search range or not. Our runtime on small images is dominated by Matlab overhead. The time efficiency of the algorithm (due to visiting less than 1% of disparity space) becomes apparent in images above 1 Mpx, whereas Middlebury images are just about 0.15 Mpx.

4. Conclusions

We have proposed a novel disparity component growing algorithm that can cope with much more difficult cases (repetitive patterns, complex scene) than similar existing algorithms, that can recover from errors in initial seeds, and that does not require a seed on every component in disparity space. Hence, the seeds need not be salient image features, which opens a way to random sampling of disparity space.

The changes against the standard seed growing algorithms are small, but their consequences are deep and allow the resulting algorithm to be well grounded in the theory of robust matching.

Although this has not been demonstrated in the present paper, the algorithm is in no way restricted to narrow baseline stereo images, since whenever a new seed is created, it can inherit an updated set of parameters that describe relative image distortion, as briefly discussed in Sec. 2. The algorithm can be easily adapted for multi-image matching.

The proposed algorithm implementation is available at `http://cmp.felk.cvut.cz/~stereo`.

Acknowledgements. This work was supported by the EC projects FP6-IST-027113 and MRTN-CT-2004-005439, by the Czech Academy of Sciences under project 1ET101210406, and by the STINT Foundation under project Dur IG2003-2 062.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Disparity component matching for visual correspondence. In *Proc CVPR*, pp. 470–475, 1997.

[2] Q. Chen and G. Medioni. A volumetric stereo matching method: Application to image-based modeling. In *Proc CVPR*, pp. 1029–1034, 1999.

[3] M. Gong and Y.-H. Yang. Fast stereo matching using reliability-based dynamic programming and consistency constraints. In *Proc ICCV*, pp. 610–617, 2003.

[4] M. Gong and Y.-H. Yang. Fast unambiguous stereo matching using reliability-based dynamic programming. *IEEE Trans PAMI*, 27(6):998–1003, June 2005.

[5] D. Gusfield and R. W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. The MIT Press, 1989.

[6] R. M. Haralick and L. G. Shapiro. Image segmentation techniques. *CVGIP*, 29:100–132, 1985.

[7] T. Kim and J. Muller. Automated urban area building extraction from high resolution stereo imagery. *IVC*, 14:115–130, 1996.

[8] J. Kostková, J. Čech, and R. Šára. Dense stereomatching algorithm performance for view prediction and structure reconstruction. In *Proc SCIA*, pp. 101–107, 2003.

[9] M. Lhuillier and L. Quan. Match propagation for image-based modeling and rendering. *IEEE Trans PAMI*, 24(8):1140–1146, 2002.

[10] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Trans PAMI*, 27(3):418–433, 2005.

[11] Z. Megyesi, G. Kós, and D. Chetverikov. Dense 3D reconstruction from images by normal aided matching. *Machine Graphics and Vision*, 15:3–28, 2006.

[12] H. P. Moravec. Towards automatic visual obstacle avoidance. In *Proc IJCAI*, p. 584, 1977.

[13] M. A. O'Neill and M. I. Denos. Practical approach to the stereo matching of urban imagery. *IVC*, 10(2):89–98, 1992.

[14] G. P. Otto and T. K. W. Chau. 'Region-growing' algorithm for matching of terrain images. *IVC*, 7(2):83–94, 1989.

[15] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. PMF: A stereo correspondence algorithm using a disparity gradient constraint. *Perception*, 14:449–470, 1985.

[16] R. Šára. Finding the largest unambiguous component of stereo matching. In *Proc ECCV*, pp. 900–914, 2002.

[17] R. Šára. Robust correspondence recognition for computer vision. In *Proc COMPSTAT*, pp. 119–131. Physica-Verlag, 2006.

[18] D. Sharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1):7–42, 2002. `http://cat.middlebury.edu/stereo/`.

[19] O. Veksler. Extracting dense features for visual correspondence with graph cuts. In *Proc CVPR*, pp. 689–694, 2003.

[20] O. Veksler. Fast variable window for stereo correspondence using integral images. In *Proc CVPR*, pp. 556–561, 2003.

[21] Y. Wei and L. Quan. Region-based progressive stereo matching. In *Proc CVPR*, pp. 106–113, 2004.

[22] G. Zeng, S. Paris, L. Quan, and M. Lhuillier. Surface reconstruction by propagating 3D stereo data in multiple 2D images. In *Proc ECCV*, pp. 163–174, 2004.

[23] G. Zeng, S. Paris, L. Quan, and F. Sillion. Accurate and scalable surface representation and reconstruction from images. *IEEE Trans PAMI*, 29(1):141–158, 2007.

[24] Z. Zhang and Y. Shan. A progressive scheme for stereo matching. In *Proc European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, pp. 68–85, 2000.