PMBP: PatchMatch Belief Propagation for Correspondence Field Estimation

Frederic Besse[1]
f.besse@cs.ucl.ac.uk
Carsten Rother[2]
carrot@microsoft.com
Andrew Fitzgibbon[2]
awf@microsoft.com
Jan Kautz[1]
j.kautz@cs.ucl.ac.uk

[1] University College London
London, UK
[2] Microsoft Research Cambridge
Cambridge, UK

Abstract

PatchMatch is a simple, yet very powerful and successful method for optimizing continuous labelling problems. The algorithm has two main ingredients: the update of the solution space by sampling and the use of the spatial neighbourhood to propagate samples. We show how these ingredients are related to steps in a specific form of belief propagation in the continuous space, called Particle Belief Propagation (PBP). However, PBP has thus far been too slow to allow complex state spaces. We show that unifying the two approaches yields a new algorithm, PMBP, which is more accurate than PatchMatch and orders of magnitude faster than PBP. To illustrate the benefits of our PMBP method we have built a new stereo matching algorithm with unary terms which are borrowed from the recent PatchMatch Stereo work and novel realistic pairwise terms that provide smoothness. We have experimentally verified that our method is an improvement over state-of-the-art techniques at sub-pixel accuracy level.

1 Introduction

This paper draws a new connection between two existing algorithms for estimation of correspondence fields between images: Belief Propagation [13, 19] and PatchMatch [1, 2]. Correspondence fields arise in problems such as dense stereo reconstruction, optical flow estimation, and a variety of computational photography applications such as recoloring, deblurring, high dynamic range imaging, and inpainting. By analysing the connection between the methods, we obtain a new algorithm which has performance superior to both its antecedents, and in the case of stereo matching, represents the current state of the art on the Middlebury benchmark at sub-pixel accuracy. The first contribution of our work is a detailed description of PatchMatch and belief propagation in terms that allow the connection between the two to be clearly described. This analysis is largely self-contained, and comprises the first major section of the paper. Our second contribution is in the use of this analysis to define a new algorithm: PatchMatch Belief Propagation (PMBP) which, despite its relative simplicity, is more accurate than PatchMatch and orders of magnitude faster than PBP.

Belief propagation (BP) is a venerable approach to the analysis of correspondence problems. The correspondence field is parametrized by a vector grid $\{\mathbf{u}_s\}_{s=1}^n$, where s indexes *nodes*, typically corresponding to image pixels, and $\mathbf{u}_s \in \mathbb{R}^d$ parametrizes the correspondence vector at node s. We shall consider a special case of BP, viewed as an energy minimization algorithm where the energy combines *unary* and *pairwise* terms

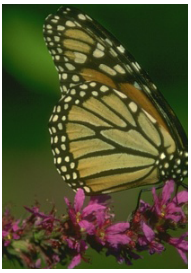
$$E(\mathbf{u}_1,\ldots,\mathbf{u}_n) = \sum_{s=1}^{n} \psi_s(\mathbf{u}_s) + \sum_{s=1}^{n} \left[\sum_{t\in N(s)} \psi_{st}(\mathbf{u}_s,\mathbf{u}_t) \right], \tag{1}$$

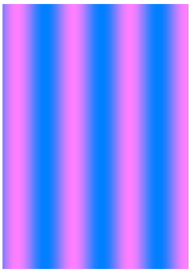
with $N(s)$ being the set of *pairwise neighbours* of node s. The unary energy $\psi_s(\mathbf{u}_s)$, also called the *data term*, computes the local evidence for the correspondence $\mathbf{u}_s$. For example, if $\mathbf{u}_s = (u_s, v_s)$ is a parametrization of a 2D flow field between images I_1 and I_2, then one might define a weighted patch data term (where (x_s, y_s) are the image coordinates of pixel s)

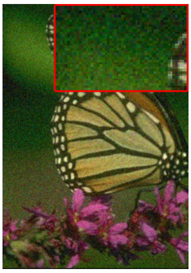
$$\psi_s^{\text{wpf}}\left(\begin{bmatrix} u_s \\ v_s \end{bmatrix}\right) = \sum_{i=-h}^{h} \sum_{j=-h}^{h} w_{sij} \left\| I_1\left(x_s+i,\ y_s+j\right) - I_2\left(x_s+i+u_s,\ y_s+j+v_s\right) \right\|. \tag{2}$$

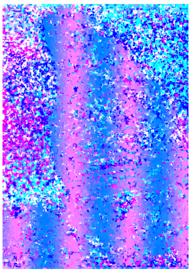
Here, the weights w_{sij} are precomputed based on the intensity values surrounding pixel s, and the norm $\|\cdot\|$ represents magnitude of difference in an appropriate colour space. For stereo correspondence, with $\mathbf{u}_s = [\Delta_s]$ being the single scalar disparity, the equivalent data term is $\psi_s^{\text{wps}}([\Delta_s]) = \psi_s^{\text{wpf}}([\Delta_s, 0]^\top)$. The problem with such a data term is that it implicitly assumes a constant correspondence field in the $(2h+1)\times(2h+1)$ patch surrounding every pixel. For large h, this oversmooths the solution, even with clever choices of w_{sij}. The oversmoothing can be addressed by using more complex parametrizations of the field within the patch (see ψ_s^{pms} below), but within traditional BP frameworks, this comes at intractable computational cost. Alternatively, h may be reduced, but as h decreases, the data term becomes increasingly ambiguous. This ambiguity is addressed by the introduction of pairwise terms, typically encouraging piecewise smoothness of the correspondence field, by assigning low energy to neighbouring nodes with similar parameter vectors, for example $\psi_{st}(\mathbf{u}_s,\mathbf{u}_t) := \min(\tau_{st}, \omega_{st}\|\mathbf{u}_s - \mathbf{u}_t\|^2)$ for image-derived constants τ_{st}, ω_{st}. It is generally understood that the presence of such pairwise term makes energy minimization difficult. For discrete problems, where the $\mathbf{u}$ live in a finite set of size D, this is clearly true in principle: without pairwise terms, minimization can be computed in $O(nD)$ time, while with pairwise terms, the worst-case complexity becomes $O(D^n)$. In practice, although BP offers no strong guarantees, it often finds good minimizers in time far below this worst case prediction. For correspondence problems, however, the $\mathbf{u}$ live in an effectively continuous space, so D must be very large (say 10^2–10^5), meaning that even the $O(nD)$ complexity of tabulating the unary costs is extremely high. Some algorithms have been proposed to address this complexity [7, 13, 14, 17], and it is on this class of methods that we improve in this paper. First, however, let us consider another school of related work.

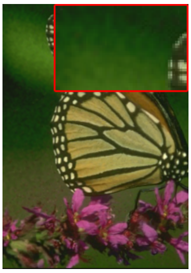
The **PatchMatch** algorithm [1] was initially introduced as a computationally efficient way to compute a *nearest neighbour field* (NNF) between two images. The NNF is then used for image editing operations such as denoising, inpainting, deblurring, as illustrated in figure 1. In terms of energy minimization, the NNF is the global minimizer of an energy comprising unary terms only ($\psi_{st} = 0$). The PatchMatch algorithm computes good minima while evaluating the unary term many fewer than D times per node. With such a powerful optimizer, more complex unary terms can be defined, yielding another class of state-of-the-art correspondence finders, exemplified by the recent introduction of PatchMatch Stereo [3].

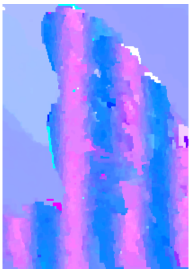

(a) Source

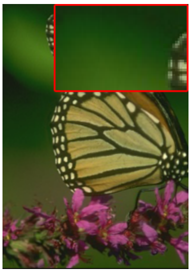

(b) Displacem. field

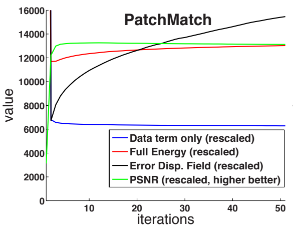

(c) Warped target


(d) EPE = 4.63×10^6

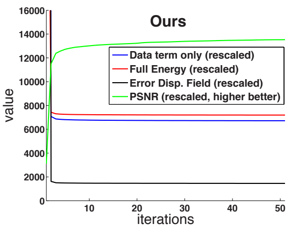

(e) PSNR = 37.39


(f) EPE = 0.44×10^6


(g) PSNR = 38.36



(h) PM: min of data term only



(i) PMBP: min of full energy

Figure 1: **Example: denoising with a reference image**. (a) Source image. (b) Synthetic displacement field $\mathbf{u}_s^{\text{gt}} := [\sin x_s, 0]^\top$. (c) Warped target image with 10% Gaussian noise added. (Note, red rectangle is a zoom of the top left corner. All images can be found in supplementary material). Estimated displacement field using PatchMatch (d) and our method (f), with total end-point error EPE = $\sum_s \|\mathbf{u}_s - \mathbf{u}_s^{\text{gt}}\|^2$. Reconstructed target image using PatchMatch (e) and our method (g), with peak signal-to-noise ratio (PSNR). Our method is considerably better for both error measures. The difference between (e) and (g) is especially noticeable in the smooth, green background where PatchMatch suffers from the ambiguous data term. (h,i) Plots error and energy for patchmatch and our method. It is noticeable that the full energy with pairwise terms is a much better fit for the task, since in (i) both error measures are well correlated with the regularized energy, in contrast to (h), where the error curves increase as the PatchMatch iterations decrease the unary-only energy.

There, disparity is *overparametrized* by a 3-dimensional vector at each node $\mathbf{u}_s = [a_s, b_s, c_s]^\top$, parametrizing a planar disparity surface $\Delta_s(x,y) = a_s(x - x_s) + b_s(y - y_s) + c_s$, giving a unary cost whose essential form is:

$$\psi_s^{\text{pms}}([a_s, b_s, c_s]^\top) = \sum_{i=-h}^{h} \sum_{j=-h}^{h} w_{sij} \left\| I_1\left(x_s + i,\ y_s + j\right) - I_2\left(x_s + i + (a_s i + b_s j + c_s),\ y_s + j\right) \right\|. \quad (3)$$

Without PatchMatch, optimization of an energy containing such a data term, even without pairwise terms, would be computationally demanding, requiring millions of operations per pixel. Intriguingly, the key operations to which PatchMatch owes its efficiency are very close to those used in continuous BP, and in particular to the message-passing that is central to optimization in the presence of pairwise terms. Conversely, a key deficiency of PatchMatch is that it lacks an explicit smoothness control on the output field. Indeed, recent developments of PatchMatch have noted that PatchMatch "has difficulty finding reliable correspondences in very large smooth regions" [5]. He *et al.* [6] require a smooth field when applying PatchMatch to an alpha matting problem, but impose smoothness as a postprocess, by solving the matting Laplacian. Boltz et al. [4] achieve smoothness by dividing the images into superpixels and running PatchMatch on these, meaning that a failure of superpixelization cannot be recovered from. A related deficiency is the tendency of PatchMatch to require a form of "early stopping": the global optimum of the unary energy is not necessarily the best solution in terms of image error, as we show in figure 1(h), and as can be seen in figure 9 of [12]. These difficulties are exacerbated by more powerful PatchMatch algorithms [2, 10] which, although getting closer to the globally optimal NNF, lose the implicit smoothness that early stopping provides. We characterize this tradeoff by looking at *error* versus *energy*: the correlation between ground-truth errors (e.g. peak signal-to-noise ratio (PSNR) for denoising problems, end-point error (EPE) for 2D correspondence fields, or disparity error for stereo) and the values of the energy functions the algorithms implicitly or explicitly minimize.

The contribution of this paper is to define a new family of algorithms, called *PatchMatch Belief Propagation* (PMBP), which combine the best features of both existing approaches, and which includes the existing methods as special cases. We first describe both existing algorithms using a unified notation, showing the close relations between the two (also illustrated as an "algorithm by numbers" in table 1). We then investigate the combination in various experimental settings, in order to explore the key terms which contribute to the combined algorithm's performance. The paper closes with a discussion of future directions.

Notation To simplify the descriptions below, the following notation will be helpful. Define the application of a function f to a set S by $f(S) := \{f(s) | s \in S\}$. Define the function $\text{fargmin}_K(S, f)$ as the function that returns the K elements of S which minimize f:

$$\text{fargmin}_K(S, f) := S_K \subset S \quad \text{s.t. } |S_K| = \min(K, |S|) \text{ and } \max f(S_K) \leq \min f(S \backslash S_K) \tag{4}$$

1.1 PatchMatch with Particles

In this section we describe Generalized PatchMatch [2] in terms that will allow easy unification with standard descriptions of continuous-domain BP. With each node s, we associate a set of K *particles* $P_s \subset \mathbb{R}^d$, where each particle $p \in P_s$ is a candidate solution for the minimizing correspondence parameters $\mathbf{u}_s^*$. Initializing these sets uniformly at random gives good performance, which may be improved slightly by using some more data-driven strategy, as discussed in §3.1.

One PatchMatch iteration then comprises a linear sweep through all nodes. The order in which nodes are visited is defined by a *schedule function* $\phi(s)$, so that s is visited before s' if $\phi(s) < \phi(s')$. We also define the *predecessor set* $\Phi s = \{s' | \phi(s') < \phi(s)\}$. On odd-numbered iterations, the typical choice of scheduling function $\phi(\cdot)$ defines a top-left to bottom-right ordering, while even-numbered iterations reverse the ordering, from bottom-right to top-left. If *iter* is an iteration counter, we write $\phi_{iter}(\cdot)$ to select the appropriate schedule. At node s, two update steps are performed: *propagation* and *resampling*:

- In the **propagation** step, the particle set is updated to contain the best K particles from the union of the current set and the set C_s of already-visited neighbour candidates

$$C_s = \bigcup \left\{ P_t \mid t \in N(s) \cap \Phi s \right\}, \tag{5}$$

  where "best" is defined as minimizing the unary cost $\psi_s(\cdot)$:

$$P_s \leftarrow \text{fargmin}_K(P_s \cup C_s, \psi_s). \tag{6}$$

- The local **resampling** step (called "random search" in [2]) perturbs the particles locally according to a proposal distribution which we model as a Gaussian $\mathcal{N}(0, \sigma)$. The second step of the PatchMatch iteration updates P_S with any improved estimates from the local resampling set, for m resampling steps:

$$R_s = \{p + \mathcal{N}(0,\sigma) \mid p \in P_s\} \tag{7}$$

$$P_s \leftarrow \text{fargmin}_K(P_s \cup R_s, \psi_s). \tag{8}$$

After several alternating sweeps, the best particle in each set typically represents a good optimum of the unary-only energy. At first sight, it may appear surprising that such a simple algorithm can effectively minimize complex energies such as $\sum_s \psi_s^{\text{pms}}$, but as the analysis in [1] shows, the piecewise smoothness in typical image flow fields[1] effectively shares the optimization burden among neighbouring pixels in the same smooth segment, without any need to identify those segments in advance.

1.2 Particle Belief Propagation (PBP)

As mentioned above, our view of belief propagation is as a minimizer of the energy (1). Thus we present a rather spartan description of max-product BP, sufficient to derive our new algorithm. BP is a *message-passing* algorithm, where messages are defined as functions from nodes to their neighbours, so that the message $M_{t\to s}(\mathbf{u}_s)$ represents, in words, "node t's opinion of the [negative log of the] likelihood that node s has value $\mathbf{u}_s$". Before defining the messages, which are themselves defined recursively, it is useful to define the *log disbelief*[2] at node s as

$$B_s(\mathbf{u}_s) := \psi_s(\mathbf{u}_s) + \sum_{t \in N(s)} M_{t\to s}(\mathbf{u}_s), \tag{9}$$

in terms of which the messages are defined as

$$M_{t\to s}(\mathbf{u}_s) := \min_{\mathbf{u}_t} \; \psi_{st}(\mathbf{u}_s, \mathbf{u}_t) + B_t(\mathbf{u}_t) - M_{s\to t}(\mathbf{u}_t) \tag{10}$$

or, in words: "the belief at t, modified by the pairwise term, and neglecting s's contribution to t's belief". When implemented as an iterative algorithm, messages are updated according to a schedule, like PatchMatch, and messages on the right-hand side of (10) are those of the previous iteration, or those computed earlier in the current iteration. Messages are typically initialized to all-zero. At convergence, $\hat{\mathbf{u}}_s := \text{argmin}_{\mathbf{u}} B_s(\mathbf{u})$ is the estimate of the minimizer.

The key to implementing BP for continuous state variables $\mathbf{u}$ is in the representation chosen for the messages and beliefs. Isard *et al.* [8] propose a solution by discretizing the space in a way that minimises a Kullback-Leibler (KL) divergence measure. Noorshams *et al.* [13] work on large discrete spaces, and use a randomisation step to incrementally and stochastically update partial messages, reducing the complexity from quadratic to linear. Pal *et al.* [14] also operate on large discrete spaces, and maintain sparse local marginals by using Kronecker delta functions, keeping only labels carrying the highest probability mass. Sudderth *et al.* [17] extend particle filters to Loopy BP, and use a regularisation kernel to ensure that message products are well defined. Particle Convex BP [16] uses a local resampling step like PBP, but instead of keeping the K best particles per node, or drawing from a distribution, it

[1]Note that "flow field" is intentionally left imprecise here. The key is that the globally optimum NNF is *not* smooth, but the approximate NNF found by PatchMatch tends to be, due to the smoothness of the underlying real-world physical process which generates the image correspondences.

[2]This energy-based formulation can be converted to a probabilistic form using the conversions: belief $b_s(\mathbf{u}_s) := \exp(-B_s(\mathbf{u}_s))$ and message $m_{t\to s}(\mathbf{u}_s) = \exp(-M_{t\to s}(\mathbf{u}_s))$.

Let P_s be the set of particles at node s, and K the desired number of particles.
Let N be the number of iterations, and m the number of randomization iterations.
Let $\mathcal{I}$ be the initialization distribution: uniform or local potentials ψ.

PM	PBP	PMBP		Steps
●	●	●	1	**for all** nodes $s \in \{1..n\}$, **repeat** K times: // initialization
●	●	●	2	Draw $p \sim \mathcal{I}$, add to P_s
●	●	●	3	**for** $i = 1$ to N: // main loop
●	●	●	4	**for all** nodes $s \in \{1..n\}$ **orderby** ϕ_i: // PatchMatch schedule
●	●	●	5	**for all** proposal sets R_s in:
●	-	●	6	$R_s = \bigcup\{P_t \mid t \in N(s) \cap \Phi_i(s)\}$ // resampling using neighbours
●	●	●	7	$R_s = P_s$ // local resampling
●	●	●	8	**do**
●	●	●	9	**for all** particles $p \in R_s$:
●	●	●	10	**repeat** m times // Possibly different m for each R_s
●	●	●	11	$p' = p + \mathcal{N}(0, \sigma)$
●	-	-	12	Compute $B_s(p') = \psi_s(p')$
-	●	●	13	Compute $B_s(p') = \psi_s(p') + \sum_{t \in N(s)} M_{t \to s}(p')$
●	-	●	14	$P_s = \text{fargmin}_K(P_s \cup \{p'\}, B_s)$ // Update best K in P_s.
-	●	-	15	**if** $B_s(p') < B_s(p) - \log(\text{rand})$: $p \leftarrow p'$ // MCMC sampling
-	●	-	16	Replace p with p' in P_s // Only after MCMC
●	●	●	17	**for all** nodes $s \in 1..n$: // read out the final solution
●	-	-	18	return fargmin(P_s, B_s) where $B_s(p) = \psi_s(p)$
-	●	●	19	return fargmin(P_s, B_s) where $B_s(p) = \psi_s(p) + \sum_{t \in N(s)} M_{t \to s}(p)$

Table 1: **Pseudo-code for different algorithms**. PM is PatchMatch; PBP is Max Product Particle BP; PMBP is PatchMatch BP. Note that whenever B_s is computed, for PBP and PMBP, we have to also recompute the minimizations in the messages $M_{t \to s}$.

keeps the one particle which optimizes a discrete MRF with K candidate particles per node. Very recently, Yamaguchi *et al.* [18] apply it to dense stereo estimation, combining the plane parameterization from (3) with a discrete line process. However, to allow tractable inference, they use a superpixelization into 1200 regions, meaning the results are strongly dependent on an accurate segmentation.

In our case, a natural representation already presents itself, closely related to the Max Product Particle BP of Kothapa *et al.* [11], based in turn on [7]. For brevity, we refer to [11] as "PBP". As above, we associate with each node s a particle set P_s. Then all messages and beliefs evaluated at any node η are in terms of the particles P_η, so the message definition becomes

$$M_{t \to s}(\mathbf{u}_s) := \min_{\mathbf{u}_t \in P_t} \psi_{st}(\mathbf{u}_s, \mathbf{u}_t) + B_t(\mathbf{u}_t) - M_{s \to t}(\mathbf{u}_t). \tag{11}$$

We note that this definition is still in terms of a continuous $\mathbf{u}_s$, not restricted to the current particle set P_s, but the continuous minimization over $\mathbf{u}_t$ in (10) is replaced by a discrete minimization over the particles P_t.

The final step of each iteration at node s is to choose a new set of particles P_s to represent the belief at s. The ideal set of particles would be a draw (including the mode, as our goal is to minimize the energy) from the true belief $b_s^*(\cdot)$, which is of course unavailable. As an alternative, Kothapa *et al.* [11] propose MCMC sampling from the current belief estimate with a Gaussian proposal distribution. We show that other alternatives can be valuable.

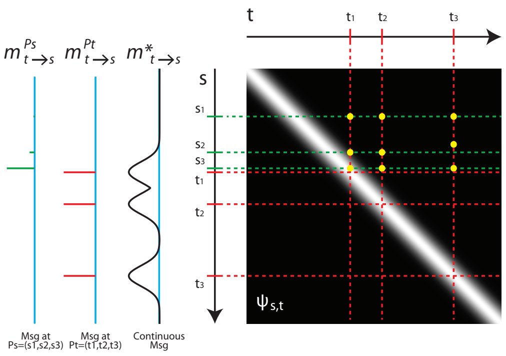


Figure 2: **Message calculation**. Green bars represent the set of particles at s, $P_s = (s_1, s_2, s_3)$ and the red bars represent $P_t = (t_1, t_2, t_3)$. In PBP [7, 11], the continuous message $m^*_{t\to s}\mathbf{u}_s$ is evaluated only at particles in P_s, and minimized only over P_t, evaluated at the yellow dots. When P_s and P_t differ, much of the message may be uninformative (represented by the green particles $m^{Ps}_{t\to s}$). If the pairwise potential favours smoothness, including particles from P_t increases the likelihood that high probability parts of the message are included.

2 PatchMatch Belief Propagation

We are now in a position to make the second of our contributions, combining the PatchMatch and PBP algorithms. We shall consider PBP our base, as the goal is to minimize a more realistic energy than PatchMatch, that is to say, an energy with pairwise terms encouraging piecewise smoothness. Referring to table 1, two key differences between PM and PBP are evident.

First, PM resamples P_s from the neighbours of node s, while PBP's resampling is only via MCMC from the elements of P_s. As illustrated in figure 2, this may be viewed as sampling from the continuous incoming messages at s, with the property that important modes of the belief may be uncovered, even when P_s lacks particles at those modes. It should be clarified that the samples are evaluated using B_s, so this is a resampling of the particle set under the current belief, as proposed in PBP, but with a quite different source of particle proposals. Thus PMBP augments PBP with samples from the neighbours (or, as argued in Figure 2, samples from the incoming messages). This can also viewed as a return to the sampling strategies of Nonparametric BP [17], but with a much simpler message representation. One way to look at this contribution is simply to say we are running some form of NBP but with algorithm settings (number of particles, number of samples) that would never make sense for NBP, and that this in itself is a useful contribution. Note that taking directly particles from the neighbouring node only works because our pairwise term is a smoothing term, i.e. has the lowest value when both entries are the same. Hence for arbitrary pairwise terms this strategy has to be modified.

Second, PBP uses an MCMC framework where particles are replaced in P_s with probability given by the Metropolis acceptance ratio, while PatchMatch accepts only particles with higher belief than those already in P_s. We have found that this non-Metropolis replacement strategy further accelerates convergence, so it is included in PMBP.

Making these two modifications yields “PatchMatch BP”, a powerful new optimization algorithm for energies with pairwise smoothness terms. In the case of a zero pairwise term $\psi_{st} = 0$, PMBP exactly yields Generalized PatchMatch. Conversely, running PMBP with a nonzero pairwise term is a strict generalization of GPM, allowing the incorporation of an explicit smoothness control which directly addresses the deficiencies of PatchMatch while retaining its speed.

Note that we can also use any external information to get reasonable candidate particles, such as matching nodes between image pairs in the stereo matching case, similarly to [3].

3 Experiments

Experiments were performed to quantify the effects of the various algorithm components, as well as real-world performance on a stereo benchmark.

3.1 Initialization

As mentioned above, there are two ways of initializing the particles: using a uniform distribution, or using the local potentials, as suggested in [7]. However, sampling from the local potentials is not an easy task, as they are defined on a continuous, high dimensional space. The original PatchMatch algorithm, optimizing only the unary energy, can be used to find an approximation of these local potentials. A benchmark can be seen in figure 3, which shows that PMBP outperforms PBP, with both types of initializations, and that convergence is orders of magnitude faster. Furthermore, we show that resampling using the neighbours is the key step of our algorithm. To do so, we run PMBP with MCMC instead of using the PM randomisation mechanism, which in effect replicates PBP, the only difference now being the use of the neighbours for resampling, and we see that although much slower than PMBP, it converges to the same energy.

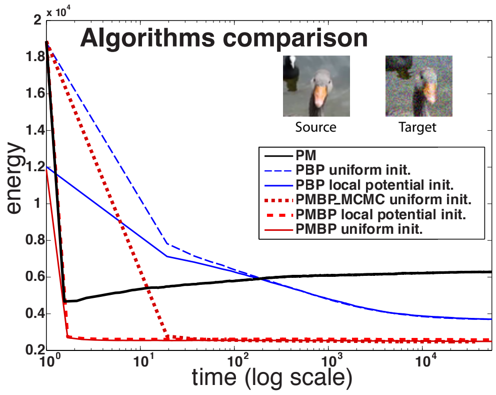


Figure 3: Comparison of the energies produced by the different algorithms on a denoising experiment. Notice that PBP cannot reach the energy of PMBP even if allowed four orders of magnitude longer, supporting our claim that previous BP implementations were intractable.

3.2 Stereo

In the following we demonstrate the benefits of introducing smoothness for the stereo matching case, and by doing so we are able to achieve state-of-the art results.

For the data term we use the same energy as in PatchMatch Stereo [3]. The weight w_{sij} is defined as

$$w_{sij} = \exp(- \parallel I(x_s,y_s) - I(x_s+i, y_s+j) \parallel / \omega). \tag{12}$$

In this equation ω is a user-defined parameter and $\parallel I_s - I_t \parallel$ is the L_1 distance between s and t in RGB space. The image difference is adapted to include an image gradient term, so that $\|I_1(x,y) - I_2(x',y')\|$ in (3) is replaced by

$$(1-\alpha)\min(\|I_1(x,y)-I_2(x',y')\|, \tau_{\text{col}}) + \alpha \min(\|\nabla I_1(x,y) - \nabla I_2(x',y')\|, \tau_{\text{grad}}) \tag{13}$$

where $\|\nabla I - \nabla I'\|$ is the L_1-distance between the grey-level gradient, and α is a parameter controlling the influence of the colour and the gradient terms. τ_{col} and τ_{grad} are the truncated costs used to add robustness.

The pairwise term captures the deviation between the two local planes in $(x, y, \mathit{disparity})$ space. Let the plane normal at node s be $\mathbf{n}_s = \text{orth}([a_s, b_s, -1]^\top)$, where $\text{orth}(v) := v/\|v\|$, and let $\mathbf{x}_s = [x_s, y_s, c_s]^\top$ be a point on the plane. Then the pairwise energy is given by

$$\psi_{st}(\mathbf{u}_s, \mathbf{u}_t) = \beta w_{st} \left(|\mathbf{n}_s \cdot (\mathbf{x}_t - \mathbf{x}_s)| + |\mathbf{n}_t \cdot (\mathbf{x}_s - \mathbf{x}_t)| \right). \tag{14}$$

The data-dependent term w_{st} is defined as in eqn. (12) with $i = x_t - x_s$ and $j = y_t - y_s$. The weight β is a constant weighting of the pairwise term with respect to the unary term. Note,

	Tsukuba			Venus			Teddy			Cones		
	nocc	all	disc	nocc	all	disc	nocc	all	disc	nocc	all	disc
PM Stereo	15.0 $_{45}$	15.4 $_{44}$	20.3 $_{56}$	1.00 $_{6}$	1.34 $_{6}$	7.75 $_{9}$	5.66 $_{2}$	11.8 $_{2}$	16.5 $_{2}$	3.80 $_{2}$	10.2 $_{2}$	10.2 $_{2}$
Ours	11.9 $_{27}$	12.3 $_{24}$	17.8 $_{29}$	0.85 $_{5}$	1.10 $_{3}$	6.45 $_{6}$	**5.60** $_{1}$	12.0 $_{3}$	**15.5** $_{1}$	**3.48** $_{1}$	**8.88** $_{1}$	**9.41** $_{1}$

Table 2: Results on the Middlebury dataset with subpixel threshold (t=0.5). Bold entries indicates where our algorithm is ranked first. Our method has the first rank, with an average rank of 8.5, in contrast to 14.9 for PatchMatch Stereo.

for $\beta = 0$ we obtain PatchMatch Stereo.

The energy ψ^{pms} is augmented to symmetrize left and right views, and we label the left and right images in two consecutive steps. To be precise, the main loop at line 3 in table 1 is first executed for the left view and then for the right view. Furthermore, as in the PatchMatch Stereo algorithm, we have implemented the concept of "view propagation". The idea is that a good particle for a pixel s in the left view, may be in the particle set P_t of the corresponding (warped) pixel t in the right view, and vice versa. In terms of code, lines $5-8$ in table 1 are duplicated, with the change that in line 5, the neighbourhood $N(s)$ is $t = (x_s + c_s, y_s)$. Finally, after optimizing the energy, there is a post-processing which is the same left-right consistency check as in [3] in order to fill-in occluded pixels.

We use the same parameters as [3], which are $\{\omega, \alpha, \tau_{col}, \tau_{grad}\} = \{10, 0.9, 10, 2\}$, with a larger patch size of 40x40 pixels. The weighting of the pairwise terms is set to $\beta = 7.5$.

We tested our algorithm on stereo pairs of the Middlebury dataset. We run our PMBP on the full energy and compare it to PatchMatch Stereo with no smoothness cost, i.e. $\beta = 0$. In both cases we use the same number of particles $K = 5$. The results are summarized in table 2 and figure 4. We observe that we are superior to PM Stereo in all cases. For the sub-pixel accuracy level, we are overall Rank 1 of all methods. Note that we perform particularly well on the challenging datasets "Teddy" and "Cones".

Figure 5 illustrates again the importance of the smoothness term. As expected, PatchMatch stereo struggles in areas of low textures (e.g. middle of the bowling ball (top row), and white pages of the book (bottom row)). By increasing the weight β of the pairwise term, the output becomes increasingly smoother. Naturally, overshooting occurs after a certain a point, which can be seen in figure 5 for large values of β. Please refer to the supplementary material for further results. There we also demonstrate the positive effect of using a large number of particles.

A comment on the performance, PMBP has a 20% overhead compared to PM, due to the message computations being more expensive.

4 Conclusion

In this work we have made the link between the popular PatchMatch method and the very well-known Belief propagation algorithm. By doing so, we were able to extend the PatchMatch algorithm by introducing additional pairwise terms. We validated experimentally that we achieve state-of-the art results for stereo matching at sub-pixel accuracy level.

There are many exciting avenues for future work, both in terms of applications, such as optical flow, as well as algorithms, such as different forms of message passing e.g. Tree-reweighted message passing [9].

Acknowledgements We thank Christoph Rhemann and Michael Bleyer for their help with the PatchMatch Stereo code and also for fruitful discussions.

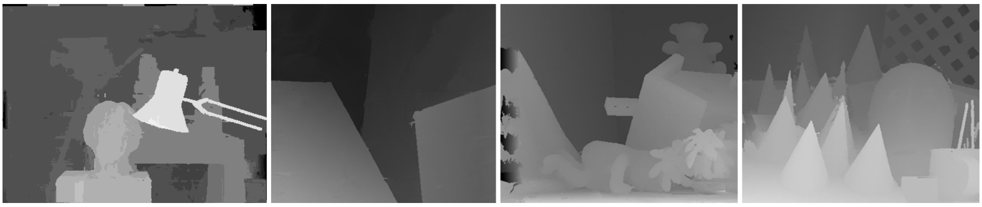


Figure 4: Qualitative results of PMBP on the Middlebury dataset.

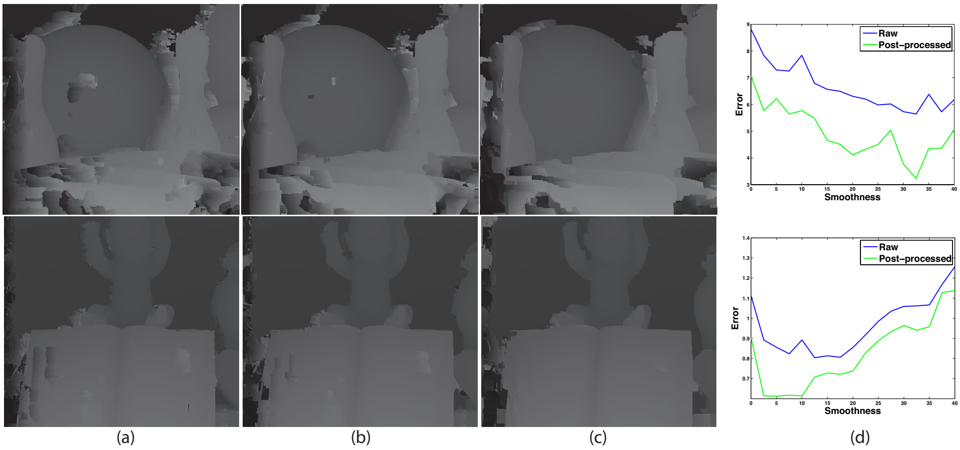


Figure 5: Evolution of the disparity map (before post-processing) with different weightings of the smoothness: (a) $\beta = 0$ (PatchMatch stereo). (b) $\beta = 5$. (c) $\beta = 17.5$. (d) Corresponding disparity error, for both raw and post-processed outputs. See supplementary material for the input images.

References

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), August 2009.

[2] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized PatchMatch correspondence algorithm. In *Proc. ECCV*, 2010.

[3] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch Stereo—Stereo matching with slanted support windows. In *Proc. BMVC*, 2011.

[4] S. Boltz and F. Nielsen. Randomized motion estimation. In *Proc. ICIP*, pages 781–784, 2010.

[5] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 30(4):70:1–70:9, 2011.

[6] K. He, C. Rhemann, C. Rother, X. Tang, and J. Sun. A global sampling method for alpha matting. In *Proc. CVPR*, pages 2049–2056, 2011.

[7] A. Ihler and D. McAllester. Particle Belief Propagation. In *Proc. AISTATS*, volume 5, pages 256–263, 2009.

[8] M. Isard, J. MacCormick, and K. Achan. Continuously-adaptive discretization for message-passing algorithms. In *Proc. NIPS*, pages 737–744, 2008.

[9] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, October 2006.

[10] S. Korman and S. Avidan. Coherency sensitive hashing. *Proc. ICCV*, pages 1607–1614, 2011.

[11] R. Kothapa, J. Pachecho, and E. B. Sudderth. Max-product particle belief propagation. Master's thesis, Brown University, 2011.

[12] A. Mansfield, M. Prasad, C. Rother, T. Sharp, P. Kohli, and L. Van Gool. Transforming image completion. In *Proc. BMVC*, 2011.

[13] N. Noorshams and M. J. Wainwright. Stochastic belief propagation: Low-complexity message-passing with guarantees. *Computing Research Repository*, arXiv:1111.1020v1 [cs.IT], 2011.

[14] C. Pal, C. Sutton, and A. McCallum. Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *Proc. ICASSP*, volume 5, 2006.

[15] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[16] J. Peng, T. Hazan, D. A. McAllester, and R. Urtasun. Convex max-product algorithms for continuous MRFs with applications to protein folding. In *Proc. ICML*, 2011.

[17] E. B. Sudderth, A. T. Ihler, M. Isard, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. *Proc. IEEE Intl. Conf. Acoustics, Speech, Signal Proc.*, 53(10): 95–103, October 2010.

[18] K. Yamaguchi, T. Hazan, D. A. McAllester, and R. Urtasun. Continuous Markov random fields for robust stereo estimation. *Computing Research Repository*, arXiv:1204.1393v1 [cs.CV], 2012.

[19] J. Yedidia, W. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Info. Theory*, 51(7):2282 – 2312, July 2005.