

Wavelet Belief Propagation for Large Scale Inference Problems

Ruxandra Lasowski^{1,2}
lasowski@mpi-inf.mpg.de

Art Tevs¹
tevs@mpi-inf.mpg.de

Michael Wand^{1,2}
wand@mpi-inf.mpg.de

Hans-Peter Seidel¹
hpseidel@mpi-inf.mpg.de

1. Max-Planck Institute Informatik, Saarbrücken, Germany
2. Saarland University, Saarbrücken, Germany

Abstract

Loopy belief propagation (LBP) is a powerful tool for approximate inference in Markov random fields (MRFs). However, for problems with large state spaces, the runtime costs are often prohibitively high. In this paper, we present a new LBP algorithm that represents all beliefs, marginals, and messages in a wavelet representation, which can encode the probabilistic information much more compactly. Unlike previous work, our algorithm operates solely in the wavelet domain. This yields an output-sensitive algorithm where the running time depends mostly on the information content rather than discretization resolution. We apply the new technique to typical problems with large state spaces such as image matching and wide-baseline optical flow where we observe a significantly improved scaling behavior with discretization resolution. For large problems, the new technique is significantly faster than even an optimized spatial domain implementation.

1. Introduction

Loopy belief propagation (LBP) [1] is a widely used technique for approximate inference in Markov random fields (MRFs). Although there are in the general case no strict guarantees of convergence (which can be fixed [1]) and optimality, belief propagation often yields very good approximations in practical computer vision problems such as stereo reconstruction [2, 3], optical flow [3], or correspondence computations [4]. The generality of the algorithm is particularly appealing: Unlike other approximate inference techniques, LBP is mostly independent of the structure of the Markov random field and the probabilistic potentials used in the model. The algorithm can be applied to graphs with general clique sizes, arbitrary potentials, and general label sets. For n random variables that each can be in k different states (or labels) with potentials of order (clique size) c , each iteration of the basic algorithm

needs $\Theta(nk^c)$ running time. For large state spaces, this is often prohibitively large, even in the case of pairwise MRFs ($c=2$). This is particularly unfavorable for MRFs with continuous states. Such MRFs are frequently used in application such as shape and image matching, where labels for example represent positions in a 2D image. Representing such distributions accurately with a grid of discrete bins often leads to high costs despite coarse resolutions.

However, the marginal distributions within the algorithm typically show a lot of coherence so that a full histogram representation is wasteful. For example, in image matching applications, we obtain sharply peaked initial beliefs at salient features and rather uniform, uninformative beliefs within uniformly colored regions. The LBP algorithm then distributes this knowledge under some regularizer. This successively creates more peaked beliefs until only a few final sharp peaks are left. Fig. 1 shows such typical marginal distribution before and after belief propagation is applied. In this paper, we are examining a more compact representation of marginals, which leads to a novel belief propagation algorithm that scales better to state spaces discretized at a high resolution. Instead of the original histograms, we operate in a space of linearly transformed histograms. A natural choice for the basis of such a message-passing space is a wavelet basis because wavelets are able to efficiently encode both low frequency distributions, referring to high uncertainty, as

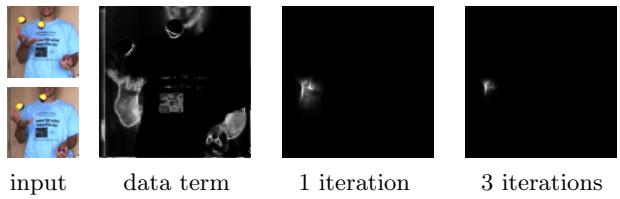


Figure 1: Marginal distributions in an optical flow application of loopy belief propagation (for a point on the right hand of the subject). Typical marginals are usually sparse and consist of some smooth regions with a few salient peaks.

well as sharply peaked distributions. Uncertain knowledge is easily captured by a few low frequency wavelet coefficients. On the contrary, a Dirac delta pulse represented at resolution $\mathcal{O}(\epsilon)$ can be encoded with $\mathcal{O}(\log \frac{1}{\epsilon})$ coefficients. Because of the linearity of the transform, mixtures of such cases can also be handled.

In order to work with this representation, there is a major technical obstacle: While the transformation and compression of individual messages and beliefs into the wavelet domain is straightforward, the message passing algorithm itself cannot be easily formulated in the wavelet domain. A straightforward solution would be to transform back to the spatial domain at each message passing step, which would void the computational advantages of the encoding. The other obvious alternative is using n -ary wavelet products for message evaluation, which in a naïve implementation leads to exponential costs with respect to n . Our algorithm employs recent results on efficient evaluation of n -ary wavelet product integrals [5, 6] in order to perform the direct algebraic product evaluation in linear rather than exponential time. Furthermore, each message passing step requires an integration over pairwise potential functions that weight the incoming messages. Here, we develop a wavelet-space encoding and a hierarchical scheduling scheme for the computations that can perform this step efficiently. Furthermore, we will show that we can formulate a resolution independent version of LBP that does not require any a priori discretization of the domain.

For evaluation, we apply the algorithm to image matching and optical flow tasks that require large label spaces. We compare our implementation to a simple histogram-based implementation of LBP as well as an optimized version with pruning. The observed scaling behavior of our new technique is significantly better than that of both spatial domain algorithms: Computational costs grow significantly weaker with discretization resolution, and for large label spaces, the new algorithm outperforms the spatial domain implementations.

2. Related Work

Compression techniques have already been applied to LBP to improve space and/or time complexity. Yu et al. [7] investigate PCA and an envelope transform for the compression in a 1D domain. The technique achieves good results but cannot be applied directly in the compression domain. Zhao et al. [8] compress the messages in the wavelet domain however each time the message is uncompressed when arriving at the node and compressed when going out. Minka [9] describe expectation propagation, a variant of LBP for potential

functions belonging to exponential families allowing for continuous state spaces in this case. Felzenszwalb et al. [3] describe linear and near-linear time message passing algorithms for restricted types of pairwise potentials. Additionally the authors show a speed up of message update computation by FFT-based fast convolution for shift invariant potentials. Our method does not have restrictions on potentials. Furthermore, our method is completely independent of discretization resolution, which is impossible with the FFT-based algorithm which relies on regularly sampled domains. Kondakis et al. [10] propose dynamic pruning of states of low probability. However, this approach cannot handle smeared out uncertainty, but would distort such distributions by removing the uncertainty information altogether. Sudderth et al. [11] introduce *nonparametric belief propagation* (NBP) that represents messages by samples. Fitting of Gaussian mixture models and Gibbs sampling is used for message propagation. This reconstruction from sampling leads to numerical smoothing, adding artificial uncertainty in each propagation step. For large graphs, this impacts precision (the paper only discusses results in the range of 5×5 nodes). Our wavelet representation is more restrictive with respect to the domain of the *state space* (restriction to hypercubes in low dimensions). However, we obtain good results even for large graphical models with many random variables and very high PSNR. A restricted NBP for Gaussian potential functions is presented in [12]. Han et al. [13] employ mean-shift to estimate the mode and weight for the outgoing message, but this neglects uncertainty information. Following this line, Parks et. al [14] discretize the continuous state space into a grid and use only local samples to perform mean-shift on the marginals. In contrast to these approaches, we are aiming at representing the full distribution, not just the main modes.

3. Wavelet Belief Propagation

3.1. Standard Belief Propagation

We assume that we are given a MRF consisting of n random variables \mathbf{x}_i , with $i = 1..n$ and mutual conditional independence encoded in a graphical model $G = (V, E)$, $V = 1..n$, $E \subseteq V \times V$. Each variable is assumed to have a continuous, d -dimensional state space $\Omega = [0, 1]^d \subseteq \mathbb{R}^d$. The layout of G can be arbitrary. Furthermore, we assume that the probability distribution factors into singleton node potentials ϕ_i (“data terms”) and pairwise edge potentials ψ_{ij} (“compatibility terms”):

$$p(\mathbf{x}_1, \dots, \mathbf{x}_n) = \frac{1}{Z} \prod_{i=1..n} \phi_i(\mathbf{x}_i) \prod_{i,j \in E} \psi_{ij}(\mathbf{x}_i, \mathbf{x}_j), \quad (1)$$

where Z is a normalization constant. Our goal is to approximate the marginal distributions

$$p(\mathbf{x}_i) = \int_{\substack{\mathbf{x}_1 \dots \mathbf{x}_n \in \Omega \\ j \neq i}} \dots \int p(\mathbf{x}_1, \dots, \mathbf{x}_n) d\mathbf{x}_1 \dots d\mathbf{x}_n. \quad (2)$$

Loopy belief propagation approximates the marginal distributions $p(\mathbf{x}_i)$ for each random variable by assigning a message m_{ij} to each edge in the graph and updates these iteratively by the following rule [1]:

$$m_{ij}(\mathbf{x}_j) = \int_{\mathbf{x}_i \in \Omega} \left(\Psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) \Phi_i(\mathbf{x}_i) \prod_{k \in N_i \setminus j} m_{ki}(\mathbf{x}_i) \right) d\mathbf{x}_i \quad (3)$$

where N_i denotes the set of direct neighbors of node i in G . The *belief* function f_i of node i is given by:

$$f_i(\mathbf{x}_i) = \frac{1}{Z} \Phi_i(\mathbf{x}_i) \prod_{k \in N_i \setminus j} m_{ki}(\mathbf{x}_i) \quad (4)$$

Belief propagation initializes all beliefs by the singleton potentials Φ_i and then iteratively passes messages in the graph until convergence. The final approximations to the marginals $p(\mathbf{x}_i)$ are the beliefs at each node.

3.2. Wavelet representation

Our goal is now to perform this algorithm completely in the wavelet domain. We therefore use a wavelet basis $B = \{b_1, \dots, b_N\}$ with $N = (2^k)^d, k \in \mathbb{N}$, i.e., the resolution in each dimension is a power of two. In this paper, we will use a Haar wavelet basis in “non-standard” decomposition (for more details on wavelets, see for example [15]). This restriction is necessary because for this basis, efficient algorithms for computing products of wavelet encoded signals are known [5, 6]. The non-standard Haar basis consists of one constant scaling function b_1 , covering the whole domain, as well as wavelets that describe local differences at different scales. The basis functions together form a d -dimensional quadtree. The basis is orthonormal and all functions except from b_1 integrate to zero over Ω . We now express all marginal distributions that show up in the LBP algorithm as linear combinations of wavelet functions:

$$m_{ij}(\mathbf{x}) = \sum_{p=1}^N \lambda_p^{m_{ij}} b_p(\mathbf{x}) \quad (5)$$

$$\Phi_i(\mathbf{x}) = \sum_{p=1}^N \lambda_p^{\Phi_i} b_p(\mathbf{x}) \quad (6)$$

$$\Psi_{ij}(\mathbf{x}, \mathbf{y}) = \sum_{p=1}^N \sum_{q=1}^N \lambda_{pq}^{\Psi_{ij}} b_{pq}(\mathbf{x}, \mathbf{y}). \quad (7)$$

In the following, we will use just $\lambda^{m_{ij}}, \lambda^{\Phi_i}$, and $\lambda^{\Psi_{ij}}$, without the indices p, q , to denote the complete vector of wavelet coefficients. Before we go on, Eq. 7 needs some more attention: since Ψ is a $2 \cdot d$ -dimensional function we therefor cannot use the same basis as for the other, d -dimensional functions. At this point, we opt for a tensor product construction $b_{pq}(\mathbf{x}_i, \mathbf{x}_j) := b_p(\mathbf{x}_i) \cdot b_q(\mathbf{x}_j)$. For the 1D-case, this yields a standard decomposition Haar wavelet basis, and for higher dimensions, we obtain functions that are non-standard shaped restricted to the first and last d dimensions, and standard shaped for combinations of these dimensions. Choosing the tensor product basis at this point facilitates the computation of the message passing integral, as we will see in the next step. We now plug this representation into the message passing equation 3. After simple algebraic reordering, we obtain:

$$m_{ij}(\mathbf{y}) = \sum_q b_q(\mathbf{y}) \left[\sum_{p=1}^N \lambda_{pq}^{\Psi_{ij}} \int_{\mathbf{x} \in \Omega} b_p(\mathbf{x}) R_{ij}(\mathbf{x}) \right] \quad (8)$$

with

$$R_{ij}(\mathbf{x}) := \left(\sum_{r=1}^N \lambda_r^{\Phi_i} b_r(\mathbf{x}) \right) \prod_{k \in N_i \setminus j} \left(\sum_{l=1}^N \lambda_l^{m_{ki}} b_l(\mathbf{x}) \right) d\mathbf{x}. \quad (9)$$

As we see, due to our tensor product basis representation of the Ψ_{ij} , we can split the basis functions b_{pq} into two parts and directly obtain a wavelet representation for the new message m_{ij} . The coefficients of the basis are given by the expression in square brackets. We now deal with this expression in two steps: First, we evaluate the products in the term $R_{ij}(\mathbf{x})$ (Eq. 9) using the n -ary wavelet product integral algorithm of Sun et al. [6]. In a second step, we perform the outer summation. We show that this step corresponds to a matrix vector product and provide an adaptive approximation algorithm for this step.

3.3. Wavelet Product Integrals

The term $R_{ij}(\mathbf{x})$ is a product of functions represented in the same wavelet basis B . Algebraically, this is a product of $\#N_i$ sums of up to N terms each. Assuming that r of them are non-zero, a naïve expansion of this product would result in $\mathcal{O}(r^{\#N_i})$ terms, exponential in the degree of G . Clearly, this is not acceptable.

Recently, there has been considerable interest in computing such products of wavelet-represented functions in the field of three-dimensional rendering, where integrals of products of transformed functions occur for example in lighting computations. Ng et al. [5] provide a fast solution for triple product integrals, which has been extended to general products by Sun et al. [6].

In both cases, a non-standard Haar wavelet basis is used. The key observation is that the functions are orthogonal and have only limited support. Due to the quadtree-structure of the basis functions, only direct and indirect descendant nodes can cause non-zero contributions in a pointwise multiplication, and the integral over most combinations is still zero. Based on these observations, Sun et al. [6] construct an algorithm that can evaluate the product of m wavelet-represented functions with at most r non-zero coefficients each in time $\mathcal{O}(r \cdot m)$.

We now use their wavelet product algorithm to compute the coefficients of the function $R_{ij}(\mathbf{x})$. Because we will always be looking at a fixed message passing step from node i to j , we will in the following omit the indices i, j to simplify the exposition. The wavelet product algorithm builds explicit hierarchies (d -dimensional quadtrees) to represent each term in the product and then incrementally merges these trees by a simultaneous hierarchical traversal. The output of the algorithm is a quadtree hierarchy that stores all non-zero wavelet basis coefficients for the product. We denote these by $\lambda^R = (\lambda_1^R, \dots, \lambda_N^R)$. For further details of the wavelet product algorithm, see [6].

Next, we deal with the integral in Eq. 8: The integration over $b_p(\mathbf{x})$ multiplied with $R(\mathbf{x})$ is actually easy to evaluate [6]. As B is an orthonormal basis, this integral just outputs the coefficient with index p from the wavelet tree of R . In the following, we will write the wavelet coefficients of Ψ as an $N \times N$ matrix $\Lambda^\Psi := [\lambda_{pq}^\Psi]_{p,q}$. In this notation, the summation in square brackets in Eq. 8 reduces to a dot product of the wavelet coefficients λ^R and the q -th column of Λ^Ψ . Including the outer sum yields the matrix-vector product:

$$\lambda^m = \Lambda^\Psi \lambda^R, \quad (10)$$

Our task is now to evaluate this product efficiently, which we will address in the next subsection.

3.4. Adaptive Approximate Summation

A naïve multiplication leads to $\mathcal{O}(N^2)$ running time, identical to the standard algorithm. To speed up, we make use of the special structure of this data. Both the input vector λ^R and the output vector λ^m consist of wavelet coefficients structured as d -dimensional quadtrees. The support of the associated basis functions is strictly hierarchically nested. We use this property to compute hierarchical bounds on contribution of subtrees within \mathbf{r} and Λ^Ψ . For \mathbf{r} , this is easy: In linear time, we can compute the sum of squared coefficients within the subtree of each node.

Next, we determine hierarchical bounds on Λ^Ψ . Because of the tensor product construction, the structure

is a bit more complex: The compatibility potentials Ψ are $2 \cdot d$ -dimensional objects. For example, in the case of a 2D state space, the Ψ are 4D functions. Each coefficient in the wavelet representation describes the coupling of a 2D square in the input and another 2D square in the output domain. A node n_a is contained in another node n_b if and only if both input squares of n_a are contained in those of n_b . The same holds for the corresponding hyper-squares in arbitrary dimension. This defines a hierarchy, which we use for pruning. The actual bounds are easy to obtain if we know the difference between maximum and minimum absolute values of Ψ within this domain. No wavelet coefficient can be larger in absolute values than this bound scaled with area and the normalization constant of the wavelet basis.

Using the hierarchical bounds on Λ^Ψ and λ^R , we can now evaluate the vector-matrix product efficiently. Fig. 2 shows the key steps. The blue tree represents the output tree for \mathbf{m} , which is empty in the beginning. The orange colored tree encodes the input \mathbf{r} . An edge connecting nodes of both trees represents a product operation $\lambda^m \rightarrow \lambda^m + \lambda_{pq}^\Psi \cdot \lambda_p^R$. We now try to restrict ourselves to the most important interactions by using a priority queue to schedule the most interesting regions of interaction first. For each link, we set the priority to the upper bound of the square-integral error that we would incur if we leave out this whole subtree of computation. This error is estimated by multiplying the upper bound on corresponding region in the domain of Ψ with the upper bound of the corresponding region in the domain of R .

In the first step (Fig. 2(a)) of our tree scheduling algorithm, the roots of both trees are connected with a link of infinite cost. This represents a product between the root of λ^R and the complete output tree λ^m . We now iteratively pick the most important pair from the queue, which is the pair with the largest upper bound on the potential error. We perform the multiplication for this pair and store the result. Next, all link combinations between the children of both nodes are put on the queue (Fig. 2(b)). Additionally (Fig. 2(c)) links between every child node and parent nodes are built. For these links, we mark the parent nodes as fixed, which enforces that the iteration will never subdivide these nodes but keep the connection to the fixed parent node. These links correspond to asymmetric combinations of hyper-squares of different size, which form nested side hierarchies, not contained in the symmetric nodes resulting from the symmetric subdivision (see Fig. 2(d)).

The stopping criterion is the relative l_2 error of the current estimate: While building the message, we can keep track of its current square integral norm. At the same time, we can track the amount of potential error

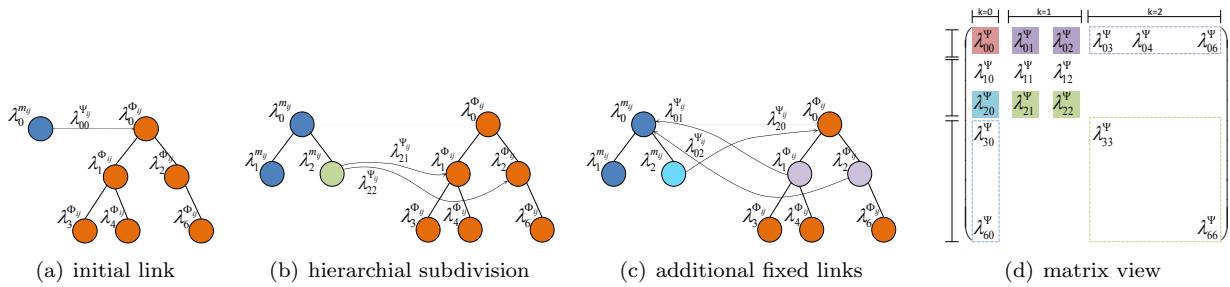


Figure 2: Illustration of the adaptive matrix vector product. Trees on the left (blue) are outputs, trees on the right (orange) are input. (a) Initial link on the priority queue. (b) After processing a link, new connections are added. (c) Additionally, “fixed” links between nodes and their counterpart’s children added. (d) Matrix interpretation: Colors indicate the corresponding cases of (a) - (c). Due to illustration purpose we omitted the scaling coefficient λ_0^Ψ .

left in the priority queue by keeping track of the sum of the bounds. If the ratio is smaller than a user specified constant (and not less than a constant minimum number of allowable operations has been performed yet), the iteration is ended¹.

The tree based message update algorithm presented here is independent on the domain resolution. Solution gathering is stopped if either there is no more input summands given or the desired signal approximation is reached. Therefore, we can work with an unbounded potential number N of wavelet coefficients and let the algorithm automatically choose a representation that in each step is accurate up to a fixed l_2 error bound. In other words, the algorithm adaptively determines a hierarchical discretization on the fly rather than fixing it upfront. This automatic adaptation cannot be obtained with traditional algorithms that work on fixed grids. Non-parametric sampling methods [11] are resolution independent by design but available methods cannot guarantee strict error bounds.

For moderately sized state spaces, this property might not be necessary so that we can simplify our algorithm by performing the priority queue-driven schedule for each row of the matrix-vector multiplication instead of globally. We have implemented this as an alternative. As we will see in Section 4, the scaling behavior is not as good but the constants in the runtime are quite a bit lower because the access to a sparse data structure storing the result coefficients is avoided (the fully adaptive version uses a hash table to store coefficients while the row-wise version can use a simple array).

3.5. Conversion to the Wavelet Domain

The algorithm described above relies on the availability of certain information about the Φ_i and the Ψ_{ij} . So far, we have formulated this as an abstract oracle; now we propose different implementations. The first

¹Our current practical implementation actually uses a simpler absolute error threshold.

type of information is the wavelet transform of these functions. Because the Haar-basis $\{b_p\}_p$ (as well as the tensor-product basis $\{b_{pq}\}_{p,q}$) is orthogonal, the transformation is given by just an inner product between the input function and a basis function. Because of the simple, piecewise constant form of the basis, this can be reduced to a sum of a small number of integrals over the input function. The second type of information we need are lower and upper bounds for Ψ_{ij} : For two quadtree-aligned squares \mathcal{X}, \mathcal{Y} , we need to be able to bound the minimum and maximum value of $\Psi_{ij}(\mathbf{x}, \mathbf{y})$ for $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$.

We have evaluated two variants how to do this in practice: First, we have discretized all (different) Φ_i and Ψ_{ij} on a regular $2^K \times 2^K$ grid for fixed K and precomputed both the bounds as well as the wavelet transforms using a simple fast-wavelet transform as described in [15]. For large K , this is not feasible anymore. In particular, the size of the Ψ -table becomes too large because of the quadratic growth. For pairwise compatibility functions Ψ that are symmetric and stationary with respect to \mathbf{x} or \mathbf{y} , the space can be reduced from $\Theta(N^2)$ to $\Theta(N)$ for each table by exploiting this redundancy. However, a resolution free operation is not possible in this way.

In this case, we have to use analytic computations: We omit all precomputed tables and perform an analytic integration and bounding at runtime, on demand. This is possible for many classes of functions Ψ (directly or through a piecewise Taylor approximation) but requires additional manual effort. In addition, runtime costs are higher than for precomputation.

4. Evaluation and Applications

4.1. Applications

We evaluate our method with two standard applications of LBP that typically require large state spaces: deformable image matching and wide-baseline optical flow [16, 17, 18]. In both cases, we are operating on

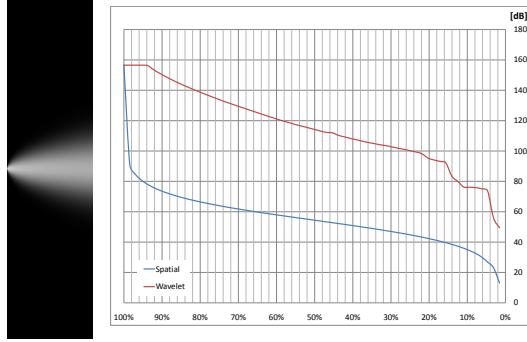


Figure 3: Left: Reference solution. Right: PSNR for the same percentage of coefficients, comparing LBP with simple binning (“spatial”) and a wavelet LBP algorithm.

a two-dimensional domain Ω . As data term Φ , we use Fourier descriptors [19] (compressed by PCA) that characterize local image content in a rotationally invariant way. The compatibility function Ψ differs for the two applications:

Deformable image matching: Here, we aim at preserving distances along the edges of the graph G , with increasing penalty for larger deviations. In other words, we are trying to keep the matching “as-rigid-as-possible”:

$$\Psi_{ij}^{def}(x_i, x_j) = \max \left(e^{-\beta(D_s(i,j) - D_t(x_i, x_j))^2}, \rho \right) \quad (11)$$

$D_s(i, j)$ and $D_t(x_i, x_j)$ are Euclidean distance between source and target point pairs, β determines how easily deformable the model is, and ρ is a truncation parameter that allows for discontinuities in the solution to make the matching robust.

Optical flow: Here, we use a regularizer that imposes a non-rotational invariant smoothness constraint [17]:

$$\Psi_{ij}^{F_d}(a, b) = \max \left(e^{-\beta(|a_x - b_x + d_x| + |a_y - b_y + d_y|)}, \rho \right) \quad (12)$$

(a_x, a_y) , (b_x, b_y) are the displacement vectors and (d_x, d_y) is one of the four stationary vectors $(1, 0)^T, (-1, 0)^T, (0, 1)^T, (0, -1)^T$, depending on current direction of the message propagation in the MRF graph, respectively. In order to favour parallel displacement vectors we move the highest response of the smoothness potential by the vectors equal to the displacement of the MRF nodes, which is encoded by the stationary vector (d_x, d_y) .

4.2. Baseline methods

We compare our approach with standard loopy belief propagation (LBP) as well with an optimized version, which we call *pruned* belief propagation (PBP). Pruned belief propagation employs a simple but very

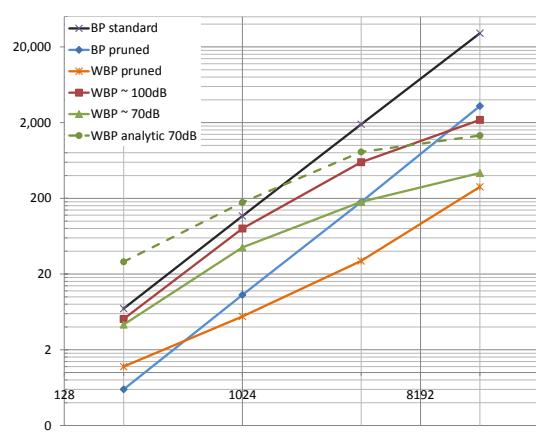


Figure 4: Execution time of one belief propagation iteration for different number of labels, $16^2 = 256$ to $128^2 = 16384$ labels. The number of nodes in the MRF is constant. Both axes of the diagram are logarithmic. Wavelet belief propagation scales differently and thus outperforms the spatial domain implementations for large state spaces.

effective optimization: Many pairwise compatibility potentials Ψ are rather sparse. For a fixed \mathbf{y} , $\Psi(\mathbf{x}, \mathbf{y})$ is typically non-constant only for a small area $\mathcal{A} \subseteq \Omega$. In this case, we can restrict the evaluation of the integral in Eq. 3 to \mathcal{A} . To account for the integral over constant Ψ , we compute the unweighted integral of Φ over \mathcal{A} and compute the integral over $\bar{\mathcal{A}}$ as complement to the integral over Ω , at no extra costs. Then, we weight this value with the constant value Ψ attains in $\bar{\mathcal{A}}$. In practice, this easy to implement optimization typically leads to large speedups. Therefore, it should be included for a fair comparison.

The comparison with WBP is based on PSNR (peak signal to noise ratio) where the corresponding mean-squared-error (MSE) is a sum of quadratic differences between belief vectors in wavelet domain and spatial domain.

4.3. Results

Image matching: Our first test is performing as-rigid-as-possible image matching on a strongly deformed example image. We use the first and fourth frame of the teddy image sequence [21] and additionally deformed the fourth frame by applying a spatial “wave” filter. In this example, we use precomputed wavelet decompositions and row-based approximative summation (Sec. 3.4). Fig. 5 shows the visual results. We are able to obtain a more than threefold speedup with WBP (1.25 hours), over the optimized PBP (4.8 hours), implementation at a PSNR of 95dB. The baseline standard LBP implementation would have used (estimated) 1200 hours for the same data set.

Optical flow: Our second test is optical flow, ap-

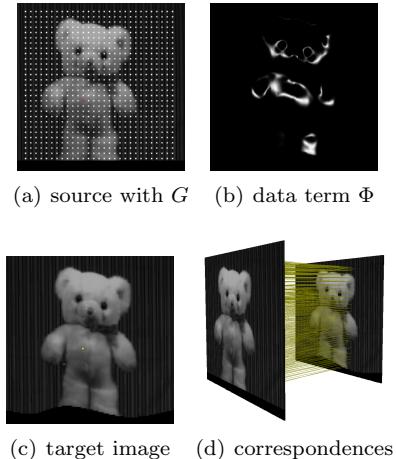


Figure 5: Teddy data set for the image matching experiment. (a) Source image; the $29^2 = 841$ nodes of G are shown as white dots. State space: $256^2 = 65536$ labels, i.e. 128×128 resolution with half-pixel accuracy. (b) Fourier data term for the point on the teddy’s belly marked in red. (c) Target image, created by applying a “wave” filter image distortion filter. Marked points indicate the correspondences. (d) Visualization of the correspondences.

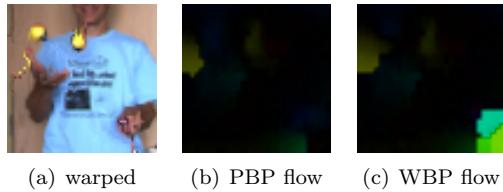


Figure 6: BeanBag data set [20]. (a) Warped image of flow computed with WBP. The MRF graph consists of 1024 nodes. On the target side we incorporate subpixel accuracy, by defining a label set of 16384 labels. (b) Flow computed with pruned BP in around 12 hours. (c) Flow computed with wavelet BP in 1.5 hours, the PSNR is 70dB.

plied to the BeanBag data set from the Middlebury database [20]. The displacement in this experiment are in the range of 10-20% of the image size, which is a good test case for large displacement optical flow. Fig. 6 shows the warped source to target image and corresponding optical flow. The color encoded displacement vectors computed with standard belief propagation are shown in Fig. 6(b) and those computed with wavelet belief propagation in Fig. 6(c). Please note that our PSNR error is computed over all belief vectors \vec{m} rather than the maximum label likelihood as shown in the resulting figures.

Comparison against downsampling: We have also performed an experiment for a 1D case (a 1D Markov chain with a 1D interval as state space) where the exact solution can be computed exactly. We run wavelet and standard BP at different resolutions to study how the error scales with the retained information. Our analytical example uses a Φ that is a Dirac

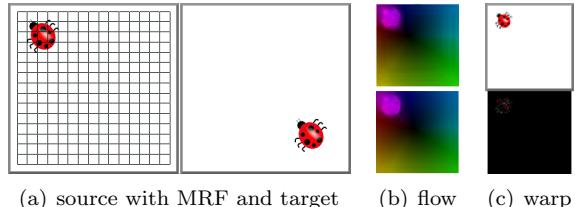


Figure 7: Synthetic example for time complexity evaluation. (a) Source and target images. The MRF consists of 256 nodes. (b) Flow fields computed with WBP at 70dB (top) and with PBP (bottom) at 4096 nodes and 4096 labels. The computation is 7min and 56min for WBP and PBP respectively. (c) Warped image computed with WBP at 70dB (top) and difference image to the source (bottom).

impulse for the first node and a constant function for all nodes $i \neq 0$. The regularizer consists of a Gaussian blur with a slight upward shift. The reference solution is computed with 1024 states. Fig. 3 shows the error for the same number of coefficients used in the state space. The accuracy of the wavelet approximation drops much more gracefully; the PSNR is between two to four orders of magnitude better than that for simple downsampling.

Scaling behavior: We study one more wide-baseline optical flow example on a synthetic “lady bug” data set (see Fig. 7). We vary the resolution of the discretization domain and compare the running times of standard LBP, pruned PBP, and wavelet BP. We use wavelet BP at 70dB and 100dB PSNR, compared against the *final results* of the reference LBP solution. For 70dB, we use both the precomputed and the analytical computation of the wavelet transform of Ψ , all other cases are precomputed. We also test the simplified row-based summation (Sec. 3.4), where very small entries in the precomputed table for Ψ are statically pruned by 1e-15, i.e. floating point accuracy; the method is therefore referred to as “WBP pruned”) and achieves a PSNR of more than 110dB.

Fig. 4 shows the running times for one iteration of belief propagation. As expected, the spatial domain implementations all scale quadratically (confirmed by a slope of 2 in the log-log plot). The row-based summation is very fast in absolute numbers, but scales slightly super-linearly (empirical slope 1.3). The full WPB implementations have the largest constant overhead factor (due to the more involved data structures), but show the best scaling behavior.

5. Conclusions

In this paper, we have introduced a new algorithm that performs sum-product belief propagation directly in a wavelet transformed representation. It represents all messages as sparse linear combination of wavelet

basis functions, which typically yields a much more concise representation than traditional histograms of labels. Our technique is general and applicable to pairwise MRFs of arbitrary topology (general graphical models) with general potential functions. Unlike previous work our algorithm performs all operations in the wavelet domain only, thereby only requiring computational costs dependent on the information content in this decorrelated representation rather than on the spatial resolution. In practice, we obtain a significantly better scaling behavior, however, at the price of a larger overhead (larger constant factor in the runtime). In absolute time, wavelet belief propagation outperforms even our optimized spatial domain base-line implementation for medium to large state spaces (depending on the chosen summation algorithm).

Limitations and future work: Our technique has a number of limitations. Probably the most important one is the larger constant in the runtime due to the more involved data structures. For small state spaces, the method does not offer advantages. So far, we have to leave low-level optimizations of the algorithm for future work. The usage of wavelets also limits the domain of the state space; we need to be able to define an orthogonal Haar-basis on this domain. So far, we have only looked at square domains, but generalizations may be possible. Another limitation is the use of sum-product belief propagation; the max-product algorithm cannot be supported. Finally, for very high and unlimited (purely adaptive) resolutions, some integral properties and bounds of Ψ must be derived analytically. A hybrid, partially precomputed technique with caching might reduce these issues. In terms of applications, we would also like to look at application domains where LBP has not been used, or only in restricted form due to the high computational costs, such as texture synthesis and shape matching. Finally, it would be interesting to investigate the usage for higher order potentials, where the wavelet representation might offer a different way to reduce computational costs.

References

- [1] Yedidia, J.S., Freeman, W.T., Weiss, Y.: Understanding belief propagation and its generalizations. (2003) 239–269 1, 3
- [2] Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vision **47** (2002) 7–42 1
- [3] Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient belief propagation for early vision. Int. J. Comput. Vision **70** (2006) 41–54 1, 2
- [4] Starck, J., Hilton, A.: Correspondence labelling for wide-timeframe free-form surface matching. ICCV (2007) 1
- [5] Ng, R., Ramamoorthi, R., Hanrahan, P.: Triple product wavelet integrals for all-frequency relighting. ACM Trans. Graph. **23** (2004) 477–487 2, 3
- [6] Sun, W., Mukherjee, A.: Generalized wavelet product integral for rendering dynamic glossy objects. ACM Trans. Graph. **25** (2006) 955–966 2, 3, 4
- [7] Yu, T., Lin, R.S., Super, B., Tang, B.: Efficient message representations for belief propagation. CVPR (2007) 2
- [8] Zhao, W., Liang, Y.: W-lbp: Wavelet-based loopy belief propagation for wireless sensor networks. International Conference on Sensor Technologies and Applications (2009) 617–622 2
- [9] Minka, T.P.: Expectation propagation for approximate bayesian inference. In: Conf. on Uncertainty in Artificial Intelligence. (2001) 362–369 2
- [10] Komodakis, N., Tziritas, G.: Image completion using efficient belief propagation via priority scheduling and dynamic pruning. IEEE Transactions on Image Processing **16** (2007) 2649–2661 2
- [11] Suderth, E.B., Ihler, A.T., Freeman, W.T., Willsky, A.S.: Nonparametric belief propagation. In: CVPR (1). (2003) 605–612 2, 5
- [12] Isard, M.: Pampas: real-valued graphical models for computer vision. In: CVPR. Volume 1. (2003) I–613–I–620 2
- [13] Han, T.X., Ning, H., Huang, T.S.: Efficient nonparametric belief propagation with application to articulated body tracking. CVPR **1** (2006) 214–221 2
- [14] Park, M., Brocklehurst, K., Collins, R.T., Liu, Y.: Deformed lattice detection in real-world images using mean-shift belief propagation. IEEE Trans. PAMI **31** (2009) 1804–1816 2
- [15] Stollnitz, E.J., Salesin, H., DeRose, D.T.D.: Wavelets for Computer Graphics: Theory and Applications. Morgan Kaufmann Publishers Inc. (1996) 3, 5
- [16] Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In Pajdla, T., Matas, J., eds.: ECCV. Volume 3024 of LNCS., Prague, Czech Republic, Springer (2004) 25–36 5
- [17] Steinbrucker, F., Pock, T., Cremers, D.: Large displacement optical flow computation without warping. In: CVPR. (2009) 1609 –1614 5, 6
- [18] Flow, O., Strecha, C., Fransens, R., Gool, L.V.: A probabilistic approach to large displacement. In: In proc of ECCV Workshop SMVP. (2004) 71–82 5
- [19] Kazhdan, M., Funkhouser, T., Rusinkiewicz, S.: Rotation invariant spherical harmonic representation of 3d shape descriptors. In: SGP, Aire-la-Ville, Switzerland, Switzerland, Eurographics Association (2003) 156–164 6
- [20] Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: (A database and evaluation methodology for optical flow) 7
- [21] Vision, Database, A.S.C.I.: Teddy dataset. <http://vassc.ri.cmu.edu//idb/html/motion/teddy4/index.html> (2003) 6