Sparse Higher Order Functions of Discrete Variables — Representation and Optimization

MSR-TR-2011-45 (under review with IJCV)

Carsten Rother
Microsoft Research Cambridge
carrot@microsoft.com

Pushmeet Kohli
Microsoft Research Cambridge
pkohli@microsoft.com

Abstract

Higher order energy functions have the ability to encode high level structural dependencies between pixels, which have been shown to be extremely powerful for image labeling problems. Their use, however, is severely hampered in practice by the intractable complexity of representing and minimizing such functions. We observed that higher order functions encountered in computer vision are very often "sparse", i.e. many labelings of a higher order clique are equally unlikely and hence have the same high cost. In this paper, we address the problem of minimizing such sparse higher order energy functions. Our method works by transforming the problem into an equivalent quadratic function minimization problem. The resulting quadratic function can be minimized using popular message passing or graph cut based algorithms for MAP inference. Although this is primarily a theoretical paper, we also show how labeling problems such as texture denoising and inpainting can be formulated using sparse higher order energy functions. We demonstrate experimentally that for some challenging tasks our formulation is able to outperform various state-of-the art techniques, especially the well-known patch-based approach of Freeman et al. [11]. Given the broad use of patch-based models in computer vision, we believe that our contributions will be applicable in many problem domains.

1 Introduction

Many computer vision problems such as object segmentation, disparity estimation, object recognition, and 3D reconstruction can be formulated as pixel or voxel labeling problems. The conventional methods for solving these problems use pairwise Conditional and Markov Random Field (CRF/MRF) formulations [38], which allow for the exact or approximate inference of Maximum a Posteriori (MAP) solutions using extremely efficient algorithms such as Belief Propagation (BP) [8, 31, 41], graph cut [3, 20, 23] and Tree-Reweighted (TRW) [18, 40] message passing. Although pairwise random field models permit efficient inference, they have restricted expressive power as they can only model interactions between pairs of random variables. They are unable to enforce the high level structural dependencies between pixels which have been shown to be extremely powerful for image labeling problems.

The last few years have seen the successful application of higher order CRFs and MRFs to some low level vision problems such as image restoration, disparity estimation and object segmentation [16, 27, 33, 42, 45, 9]. In spite of these encouraging results, the use of such models have not spread to other labeling problems. We believe that this is primarily due to the lack of efficient algorithms for performing inference in such models.

This paper proposes a new method for minimizing general higher order functions that can be used to

perform MAP inference in higher order random fields. Most methods for minimizing such energy functions can be placed in one of the following categories:

1. Message passing schemes [13, 24, 27, 39] which work by passing messages in the graphical model.
2. Transformations methods [2, 10, 14, 16, 17, 26, 32, 36] which convert the higher order functions into pairwise ones by adding auxiliary variables.
3. Local update methods based on gradient descent[33, 9].

We follow the classical transformation approach for minimizing higher order functions which can be broken down into two essential steps [2]: **(a)** *Transformation of the higher order energy into a quadratic function, and* **(b)** *Minimization of the resulting function using efficient inference algorithms.* The first step in this approach is also the most crucial one. Transformation of a general m-order function to an equivalent quadratic function involves the addition of exponential number of auxiliary variables [2, 14]. Alternatively, the addition of a single random variable with an exponential label space is needed. Both these approaches make the resulting quadratic function minimization problem intractable.

Recent work on solving higher order functions in vision have side-stepped the problem of minimizing general higher order functions. Instead they have focused on specific families of potential functions (such as the P^n Potts model [16], global co-occurrence potentials [26, 46]) which can be transformed to quadratic ones by the addition of a few auxiliary variables.

We present a method for minimizing general higher order functions. This is intrinsically a computationally expensive problem since even the parametrization of a general m order function of k-state variables requires k^m parameters. However, the higher order functions used in computer vision have certain properties which makes them easier to handle. For instance, certain higher order potentials assign a low cost to only a small number of labeling assignments (also called patterns), the rest of the k^m possible labelings are assigned a high constant cost. A typical example of such a potential would be a patch-based potential for image restoration. (A detailed discussed of patch-based models in computer vision is given in sec. 5). We propose a compact representation for such potentials which allows efficient inference. We then show that for some challenging tasks the compact representation is empirically superior to standard patch-based approaches, such as [11]. A related special-case of our compact representation was concurrently proposed by [24][1], which we discuss in more detail later.

Outline of the Paper We start by reviewing work on discrete energy minimization in section 2. In section 3, we show how higher order energy functions can be transformed to quadratic ones using a multi-state auxiliary variable. Section 4 explains how higher order pseudo-boolean functions can be transformed to quadratic pseudo-boolean functions by the addition of boolean variables, and specifically presents two types of such transformations. We discuss approaches of incorporating patch-based priors for image labeling problems in section 5. Section 6 describes the experimental evaluation of our transformation schemes on the binary texture restoration problem. We conclude by summarizing our work and providing directions for future work in section 7.

2 Notation and Preliminaries

Consider a random field defined over a set of latent variables $\mathbf{x} = \{x_i | i \in \mathcal{V}\}$ where $\mathcal{V} = \{1, 2, ..., n\}$. Each random variable x_i can take a label from the label set $\mathcal{L} = \{l_1, l_2, ..., l_k\}$. Let $\mathcal{C}$ represent a set of subsets of $\mathcal{V}$ (i.e., cliques), over which the higher order random field is defined. The MAP solution of a random field can be found by minimizing an energy function $E : \mathcal{L}^n \rightarrow \mathbb{R}$. Energy functions corresponding to higher order random fields can be written as a sum of higher order potential functions as: $E(\mathbf{x}) = \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x_c})$, where $\mathbf{x_c}$ represents the set of random variables included in any clique $c \in \mathcal{C}$. The higher order potential $\psi_c : \mathcal{L}^{|c|} \rightarrow \mathbb{R}$ is defined over

[1]The paper of Komodakis *et al.* [24] and the original version of our work [36] appeared both in CVPR 2009.

this clique assigns a cost to each possible configurations (or *labelings*) of $\mathbf{x_c}$. Here $|c|$ represents the number of variables included in the clique (also called the clique order) .

Minimizing Quadratic Functions Before proceeding further, we review the basics of discrete energy minimization algorithms in computer vision. As we mentioned earlier, the problem of MAP inference in a pairwise random field can be solved by minimizing a quadratic function of discrete variables. Algorithms for MAP inference can be classified into two broad categories: (a) message passing (BP/TRW) and (b) combinatorial algorithms such as graph cut. The readers should refer to [18, 40, 41] and [3, 23] for more information on message passing and graph cut based algorithms respectively. In their classical form, these algorithms allow for the exact or approximate minimization of quadratic energy functions with certain computation time and solution quality guarantees. We will next look at the minimization of functions of boolean variables.

Quadratic Pseudo-boolean Function Minimization An energy function is called a pseudo-boolean function if the label set $\mathcal{L}$ contains only two labels i.e. $\mathcal{L} = \{0, 1\}$. Formally, the energy is now defined as: $E : \{0, 1\}^n \rightarrow \mathbb{R}$ and can also be written as a set function. The minimization of pseudo-boolean functions is a well studied problem in combinatorial optimization [15] and operations research [2]. It is known that certain classes of pseudo-boolean functions such as *submodular* functions can be minimized exactly in polynomial time. Another important characteristic is that any *Quadratic* Pseudo-boolean Function (QPBF) can be minimized by solving an st minimum cut problem (st-mincut) [2, 20]. Further, if the QPF is *submodular*, all edges in the equivalent st-mincut problem have non-negative weights, which allows it to be solved exactly in polynomial time using maximum flow algorithms [10, 20].

In what follows, we will assume that we have algorithms for approximate minimization of arbitrary multi-label quadratic energy functions, and mainly focus our attention on converting a general higher order energy function to a quadratic one.

3 Transforming Multi-label Functions

We will now describe how to transform arbitrary higher order potential functions to equivalent quadratic ones. We start with a simple example to motivate our transformation. Consider a higher order potential function which assigns a cost θ_0 if the variables $\mathbf{x}_c$ take a particular labeling $\mathbf{X}_0 \in \mathcal{L}^{|c|}$, and θ_1 otherwise. More formally,

$$\psi_c(\mathbf{x}_c) = \begin{cases} \theta_0 & \text{if} \quad \mathbf{x}_c = \mathbf{X}_0 \\ \theta_1 & \text{otherwise.} \end{cases} \tag{1}$$

where $\theta_0 \leq \theta_1$, and $\mathbf{X}_0$ denotes a particular labeling of the variables $\mathbf{x}_c$. The potential is illustrated in figure 1(b). The minimization of this higher order function can be transformed to the minimization of a quadratic function using one additional *switching* variable z as:

$$\min_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) = \min_{\mathbf{x}_c, z \in \{0,1\}} f(z) + \sum_{i \in c} g_i(z, x_i) \tag{2}$$

where the *selection* function f is defined as: $f(0) = \theta_0$ and $f(1) = \theta_1$, while the *consistency* function g_i is defined as:

$$g_i(z, x_i) = \begin{cases} 0 & \text{if} \quad z = 1 \\ 0 & \text{if} \quad z = 0 \ \text{ and } \ x_i = \mathbf{X}_0(i) \\ \text{inf} & \text{otherwise.} \end{cases} \tag{3}$$

where $\mathbf{X}_0(i)$ denotes the label of variable x_i in labeling $\mathbf{X}_0$.

3.1 General Higher Order Potentials

The method used to transform the simple potential function (1) can also be used to transform any higher order function into a quadratic one. We observed that higher order potentials for many vision problems assign a low cost (or energy) to only a few label assignments. The rest of the labelings are given a high (almost constant) cost (see figure 1(a)). This motivated to develop a parameterization

of higher order potentials which exploits this sparsity. We parameterize higher order potentials by a list of possible labelings (also called patterns [24]) $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_t\}$ of the clique variables $\mathbf{x}_c$, and their corresponding costs $\mathbf{\Theta} = \{\theta_1, \theta_2, ..., \theta_t\}$. We also include a high constant cost $\theta_{\max}$ for all other labelings. Formally, the potential functions can be defined as:

$$\psi_c(\mathbf{x}_c) = \begin{cases} \theta_q & \text{if } \mathbf{x}_c = \mathbf{X}_q \in \mathcal{X} \\ \theta_{\max} & \text{otherwise.} \end{cases} \quad (4)$$

where $\theta_q \leq \theta_{\max}, \forall \theta_q \in \mathbf{\Theta}$. The higher order potential is illustrated in Figure 1(c). This representation was concurrently proposed by Komodakis *et al.* [24].

The minimization of the above defined higher order function can be transformed to a quadratic function using a $(t+1)$-state switching variable as:

$$\min_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) = \min_{\mathbf{x}_c, z \in \{1,2,...,t+1\}} f(z) + \sum_{i \in c} g(z, x_i) \quad (5)$$

$$\text{where} \quad f(z) = \begin{cases} \theta_q & \text{if} \quad z = q \in \{1, .., t\} \\ \theta_{\max} & \text{if} \quad z = t+1, \end{cases} \quad (6)$$

$$\text{and } g_i(z, x_i) = \begin{cases} 0 & \text{if} \quad z = q \in \{1, .., t\} \text{ and } x_i = \mathbf{X}_q(i) \\ 0 & \text{if} \quad z = t+1 \\ \text{inf} & \text{otherwise.} \end{cases} \quad (7)$$

where $\mathbf{X}_q(i)$ denotes the label of variable x_i in labeling $\mathbf{X}_q$. The reader should observe that the last i.e. $(t+1)^{th}$ state of the switching variable z does not penalize any labeling of the clique variables $\mathbf{x}_c$. It should also be noted that the transformation method described above can handle the P^n model potentials proposed in [16]. In fact it can be used to transform any general higher order potential. However, in the worst case, the addition of a switching variable with $|\mathcal{L}|^{|c|}$ states is required, which makes minimization of even moderate order functions infeasible.

3.2 Compact Parameterization

The above defined parametrization significantly reduces the complexity of performing inference in higher order cliques. However, the computation cost is still quite high for potentials which assign a low cost to many labelings. Notice, that the representation defined in equation (5) requires a $t+1$-state auxiliary variable for representing a higher order function where t labelings are assigned a low cost (less than the constant cost $\theta_{\max}$). This would make the use of this representation infeasible for higher order potentials where a large number of labelings of the clique variables are assigned low weights ($< \theta_{\max}$).

We observed that many low cost label assignments tend to be close to each other in term of the difference between labelings of pixels. For instance, consider the case of the two label foreground (f) / background (b) image segmentation problem. It is conceivable that the cost of a segmentation labeling ($fffb$) for 4 adjacent pixels on a line would be close to the cost

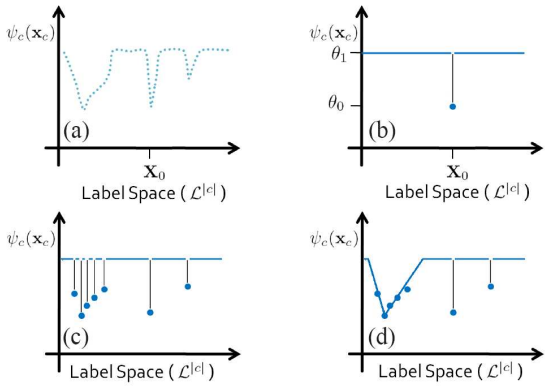

Figure 1: *Different parameterizations of higher order potentials. (a) The original higher order potential function. (b) The higher order basis function defined in equation (1). (c) Approximating function (a) using the functional form defined in equation (4). It can be seen that this representation requires the definition of 7 labelings (t=7), and thus would require the addition of a $t+1 = 8$-state auxiliary variable for its transformation to a quadratic function (as described in equation 5). (d) The compact representation of the higher function using the functional form defined in equation (8). This representation (8) requires only $t = 3$ deviation functions, and thus needs only a $t+1 = 4$-state label to yield a quadratic transformation.*

of the labeling ($ffbb$). We can encode the cost of such groups of *similar* labelings in the higher order potential in such a way that their transformation to quadratic functions does not require increasing the number of states of the switching variable z. The differences of the representations are illustrated in figure 1(c) and (d).

We parameterize the compact higher order potentials by a list of labeling deviation cost functions $\mathcal{D} = \{d_1, d_2, ..., d_t\}$, and a list of associated costs $\theta = \{\theta_1, \theta_2, ..., \theta_t\}$. We also maintain a parameter for the maximum cost $\theta_{\max}$ that the potential can assign to any labeling. The deviation cost functions encode how the cost changes as the labeling moves away from some desired labeling. Formally, the potential functions can be defined as:

$$\psi_c(\mathbf{x}_c) = \min\{\min_{q\in\{1,2,...,t\}} \theta_q + d_q(\mathbf{x}_c), \theta_{\max}\} \quad (8)$$

where deviation functions $d_q : \mathcal{L}^{|c|} \rightarrow \mathbb{R}$ are defined as: $d_q(\mathbf{x}_c) = \sum_{i\in c; l\in\mathcal{L}} w_{il}^q \delta(x_i = l)$, where w_{il}^q is the cost added to the deviation function if variable x_i of the clique c is assigned label l. The function $\delta(x_i = l)$ is the Kronecker delta function that returns value 1 if $x_i = l$ and returns 0 for all assignments of x_i. This higher order potential is illustrated in Figure 1(d). It should be noted that the higher order potential (8) is a generalization of the pattern-based potential defined in equation (4) and in [24]. Setting weights w_{il}^q as:

$$w_{il}^q = \begin{cases} 0 & \text{if } \mathbf{X}_q(i) = l \\ \theta_{\max} & \text{otherwise} \end{cases} \quad (9)$$

makes potential (8) equivalent to equation (4).

The minimization of the above defined higher order function can be transformed to that a quadratic function using a $(t+1)$-state switching variable as:

$$\min_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) = \min_{\mathbf{x}_c, z\in\{1,2,...,t+1\}} f(z) + \sum_{i\in c} g(z, x_i) \quad (10)$$

$$\text{where} \quad f(z) = \begin{cases} \theta_q & \text{if } z = q \in \{1, .., t\} \\ \theta_{\max} & \text{if } z = t+1, \end{cases} \quad (11)$$

$$g_i(z, x_i) = \begin{cases} w_{il}^q & \text{if } z = q \text{ and } x_i = l \in \mathcal{L} \\ 0 & \text{if } z = t+1. \end{cases} \quad (12)$$

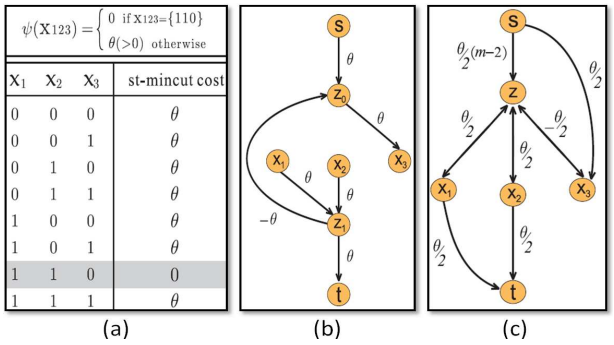

Figure 2: *Transformation of higher order pseudo-boolean functions to equivalent quadratic functions. A higher order pseudo-boolean function (a) represented by its truth table. Type-I graph construction (b) and Type-II graph construction (c) for minimizing its equivalent quadratic function. In graph (c), m denotes the number of variables included in the clique (in this case m = 3).*

The role of the switching variable in the above mentioned transformation can be seen as that of finding which *deviation* function will assign the lowest cost to any particular labeling. The final higher order function generated using the parametrization (8) is a lower envelop of the *linear* deviation cost functions $\theta_q + d_q(\mathbf{x}_c)$. For instance, the function shown in figure 4(c) is a lower envelop of the higher order functions shown in figure 4(b). This transformation method can be seen as a generalization of the method proposed in [17] for transforming the Robust P^n model potentials.

Related work on representing sparse higher order potentials Komodakis *et al.* [24] and we [36] concurrently introduced the pattern-based representation (defined in equation 4) for higher order potentials. However, deviation functions (equation 8) were not considered in [24]. Komodakis *et al.* [24] primarily focused on minimizing the pattern based potentials using dual decomposition. In their experiments, they used the pattern based representation to model potentials defined over up to 4 random vari-

ables[2]. Note, in this work we show experiments with up to 144 random variables.

They investigated two different decompositions schemes. In the first one, each pattern-based higher order clique is a so-called "slave", i.e. optimized individually. In the second (denoted as Pat-B) all higher order cliques in an image row are optimized jointly. Pat-B exploited the structure of the pattern based potential to perform efficient inference. However, it is not obvious how it could handle energy functions containing both pattern based potentials and other pairwise potentials. In this work we show experimentally that such combinations of pairwise and higher order potentials results in good labeling solutions.

An alterative way of using message passing for sparse higher order potentials has been discussed in the recent, follow-up work by Tarlow et al. [39]. They showed that our, and [24], sparse higher order potentials can be optimized via belief propagation in the given factor-graph.

4 Transforming Pseudo-Boolean Functions

In previous section, we discussed how to transform multi-label higher order functions to quadratic ones by the addition of a multi-state auxiliary variable. The same method can also be applied to transforming higher order pseudo-boolean functions. However, the resulting quadratic function is not pseudo-boolean as it contains multi-state switching variables. In this section, we discuss two alternative transformation approaches for transforming higher order pseudo-boolean functions which works by adding boolean auxiliary variables. These methods will produce a quadratic pseudo-boolean function (QPBF) which can be minimized using algorithms for quadratic pseudo-boolean optimization (QPBO) [2].

In what follows, we assume binary labels, i.e., $\mathcal{L} = \{0, 1\}$. Consider a higher order potential which assigns a cost $\theta \geq 0$ if the variables $\mathbf{x}_c$ take the labeling $\mathbf{X}_0 \in \{0,1\}^{|c|}$, and $\theta_{\max} \geq \theta$ otherwise. More formally,

$$\psi_c(\mathbf{x}_c) = \begin{cases} \theta_0 & \text{if} \quad \mathbf{x}_c = \mathbf{X}_0 \\ \theta_{\max} & \text{otherwise,} \end{cases} \tag{13}$$

where $\mathbf{X}_0$ denotes the preferred labeling of the variables $\mathbf{x}_c$. We will call this higher order potential a δ *basis function* since it assigns a low cost to only one single labelling. A constant θ_0 can be subtracted from this potential to yield:

$$\psi_c(\mathbf{x}_c) = \begin{cases} 0 & \text{if} \quad \mathbf{x}_c = \mathbf{X}_0 \\ \theta & \text{otherwise,} \end{cases} \tag{14}$$

where $\theta = \theta_{\max} - \theta_0 > 0$.

Type-I Transformation The minimization of higher order potential function (14) can be transformed to the minimization of a quadratic function using two additional *switching* variables $z_0, z_1 \in \{0, 1\}$ as: $\min_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) =$

$$\begin{aligned} \min_{\mathbf{x}_c; z_0, z_1 \in \{0,1\}} \quad & \theta z_0 + \theta(1 - z_1) - \theta z_0(1 - z_1) \\ & + \theta \textstyle\sum_{i \in S_0(\mathbf{X}_0)} (1 - z_0) x_i \\ & + \theta \textstyle\sum_{i \in S_1(\mathbf{X}_0)} z_1 (1 - x_i). \end{aligned} \tag{15}$$

Here, $S_0(\mathbf{X}_0)$ is the set of indices of random variables which were assigned the labels 0 in the assignment $\mathbf{X}_0$. Similarly, $S_1(\mathbf{X}_0)$ represents the set of variables which were assigned the label 1 in $\mathbf{X}_0$. The minimization problem (15) involves a quadratic function with at most one non-submodular term (i.e., $-\theta z_0(1 - z_1)$).[3] It can be easily verified that the transformed QPBF in (15) is equivalent to (14). The transformation is illustrated for a particular higher order function in Figure 2(b).

Type-II Transformation We now describe an alternative method to transform higher order pseudo-boolean functions which requires the addition of only *one* auxiliary variable, but results in a slightly complex transformation.

[2]Note, [24] showed also experiments using the robust P^n model with 9 random variables in each clique. This is an instance of pattern-based potentials that can be optimized globally in polynomial time [17].

[3]This is independent of the clique size $|c|$ or the enforced labeling $\mathbf{X}_0$.

Theorem 1 *The minimization of the higher order pseudo-boolean function (14) is equivalent to the following* QPBF *minimization problem:*

$$\psi_c(\mathbf{x}_c) = \theta + \frac{\theta}{2} \min_{z \in \{0,1\}} f(z, \mathbf{x}_c), \tag{16}$$

where the QPBF *function f is defined as:*

$$\begin{aligned} f(z, \mathbf{x}_c) &= z(m-2) \\ &+ z\Big(\sum_{i \in S_1(\mathbf{X}_0)} (1 - x_i) - \sum_{i \in S_0(\mathbf{X}_0)} (1 - x_i)\Big) \\ &+ (1-z)\Big(\sum_{i \in S_1(\mathbf{X}_0)} x_i - \sum_{i \in S_0(\mathbf{X}_0)} x_i\Big) \\ &+ \sum_{i \in S_0(\mathbf{X}_0)} x_i - \sum_{i \in S_1(\mathbf{X}_0)} x_i \end{aligned} \tag{17}$$

where $m = |c|$ is the order of the clique[4]*. The transformation is illustrated for a particular higher order function in Figure 2(c).*

Proof By expanding equation (17) and substituting $|S_1(\mathbf{X}_0)| + |S_0(\mathbf{X}_0)| = |c| = m$, we obtain:

$$\begin{aligned} f(z, \mathbf{x}_c) &= 2z(|S_1(\mathbf{X}_0)| - 1) \\ &+ 2z\Big(\sum_{i \in S_0(\mathbf{X}_0)} x_i - \sum_{i \in S_1(\mathbf{X}_0)} x_i\Big) \end{aligned} \tag{18}$$

$$f(z, \mathbf{x}_c) = 2z\Big(\sum_{i \in S_0(\mathbf{X}_0)} x_i + \sum_{i \in S_1(\mathbf{X}_0)} (1 - x_i) - 1\Big) \tag{19}$$

Combining (19) with (16), we can easily find that

$$z^* = \arg\min_z f(z, \mathbf{x}_c) = 1 \tag{20}$$

if and only if $\mathbf{x}_c = \mathbf{X}_0$. Under this condition, $f(z^*, \mathbf{x}_c) = -2$, and from (16) we get $\psi_c = 0$. For all other $\mathbf{x}_c \in \{0,1\}^{|c|}$, $z^* = \arg\min_z f(z, \mathbf{x}_c) = 0$, and hence $\psi_c(\mathbf{x}_c) = \theta$.

[4]Note that the coefficient of each monomial in (17) exactly corresponds to an edge capacity in Type-II graph construction.

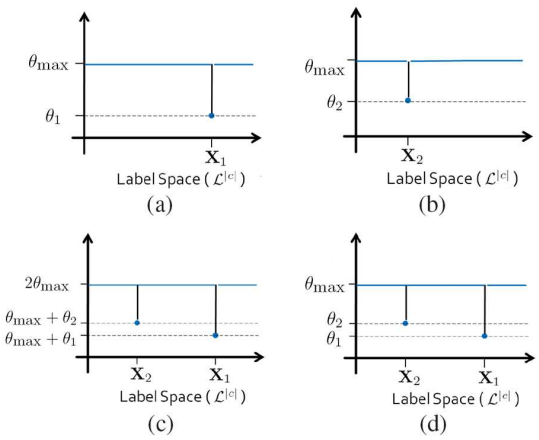

Figure 3: *Composing higher order pseudo-boolean functions by adding basis functions of the form 14.(a) and (b) are two δ basis functions. (c) The potential obtained by summing the basis functions shown in (a) and (b). (d) The potential function obtained after subtracting a constant $\theta_{\max}$ from (c) (this doesn't change the labeling with the minimal energy).*

4.1 Composing General Pseudo-boolean Functions

Multiple instances of the *δ basis* higher order pseudo-boolean potentials (14) can be used to compose general higher order energy functions. The composition method works by summing these pseudo-boolean potential. The equivalent QPBF of the target higher order pseudo-boolean function is obtained by summing the QPBFs corresponding to the individual basis pseudo-boolean potentials of the form (13) (see Figure 3 for illustration). The composition scheme requires the selection of $\theta_{\max}$ which is the highest possible energy that can be assigned to any labeling.

The reader should observe that this way of obtaining equivalent quadratic functions for general higher order functions is fundamentally different from the strategy employed in Section 3.1. There, we use a multi-state switching variable to select among different constituent higher order functions; in contrast, here we sum the constituent higher order functions.

4.2 Compact Representation for Higher Order Pseudo-boolean Potentials

The composition method described above would in the worst case require the addition of $2^{|c|+1}$ auxiliary variables for Type-I transformation and $2^{|c|}$ auxiliary variables for Type-II transformation [5]. To reduce the number of auxiliary variables, we use a scheme similar to the one discussed in Section 3.2 to model the cost of multiple labelings using only two auxiliary variables for Type-I transformation and one auxiliary variable for Type-II transformation.

We define the *deviation basis* higher order potential ψ_c^f which assigns the minimum of a deviation cost function $f(\mathbf{x}_c)$ and a constant threshold cost θ. Formally,

$$\psi_c^f(\mathbf{x}_c) = \min\{f(\mathbf{x}_c), \theta\}. \tag{21}$$

where the deviation function $f : \{0,1\}^{|c|} \rightarrow \mathbb{R}$ specifies the penalty for deviating from the favored labeling $\mathbf{X}_0$, and is written as:

$$f(\mathbf{x}_c) = \theta \sum_{i \in c} abs(w_i)(x_i \neq \mathbf{X}_0(i)) \tag{22}$$

where the absolute value of the weights w_i control the cost of different labelings deviating from $\mathbf{X}_0$. The function f can also be seen as assigning a cost equal to a weighted hamming distance of a labelling from $\mathbf{X}_0$.

The function f can alternatively be defined as:

$$f(\mathbf{x}_c) = \theta \sum_{i \in c} w_i x_i + \theta K \tag{23}$$

where the constant $K = \sum_{i|w_i<0} w_i$, and the weight w_i specifies what is the cost of assigning the label 1 to variable x_i. Naturally, the weights would be negative for pixels which have been assigned the label 1 in the favored labeling $\mathbf{X}_0$. Similarly, variables labeled 0 will be assigned a positive weight. On substituting the value of K, f becomes:

$$f(\mathbf{x}_c) = \theta \sum_{i \in c; w_i > 0} w_i x_i + \theta \sum_{i \in c; w_i < 0} (-w_i)(1 - x_i) \tag{24}$$

[5] There are $2^{|c|}$ possible labelings of the variables involved in the clique c.

We now show how the higher order pseudo-boolean function defined in equation (21) can be transformed to a QPBF.

Transformation using Type-I Construction

Theorem 2 *Using Type-I transformation, the minimization of higher order pseudo-boolean potential function (21) can be transformed to the following* QPBF *minimization problem:* $\min_{\mathbf{x}_c} \psi_f(\mathbf{x}_c) =$

$$\begin{aligned} \min_{\mathbf{x}_c; z_0, z_1 \in \{0,1\}} \quad & \theta z_0 + \theta(1 - z_1) - \theta z_0(1 - z_1) \\ & + \theta \textstyle\sum_{i|w_i \geq 0} w_i (1 - z_0) x_i \\ & + \theta \textstyle\sum_{i|w_i < 0} (-w_i) z_1 (1 - x_i) \end{aligned}$$

Transformation using Type-II Construction

Theorem 3 *Using Type-II transformation, the minimization of higher order function (21) can be written as the result of the following* QPBF *minimization problem:*

$$\psi_c(\mathbf{x}_c) = \theta + \frac{\theta}{2} \min_{z \in \{0,1\}} F(z, \mathbf{x}_c)), \tag{25}$$

where the QPBF *function F is defined as:*

$$\begin{aligned} F(z, \mathbf{x}_c) \quad = \quad & z(\sum_{i \in c} abs(w_i) - 2) \\ + \quad & z(\sum_{i|w_i<0} (-w_i)(1 - x_i) - \sum_{i|w_i \geq 0} w_i (1 - x_i)) \\ + \quad & (1 - z)(\sum_{i|w_i<0} (-w_i) x_i - \sum_{i|w_i \geq 0} w_i x_i) \\ + \quad & \sum_{i|w_i \geq 0} w_i x_i - \sum_{i|w_i<0} (-w_i) x_i \end{aligned} \tag{26}$$

where $abs(w_i)$ is the absolute value of the weight w_i.

Proof On expanding equation and simplifying equation (26), we get:

$$\begin{aligned} f(z, \mathbf{x}_c) \quad = \quad & 2z(-\sum_{i|w_i<0} w_i - 1) \\ + \quad & 2z(\sum_{i|w_i \geq 0} w_i x_i + \sum_{i|w_i<0} w_i x_i) \end{aligned} \tag{27}$$

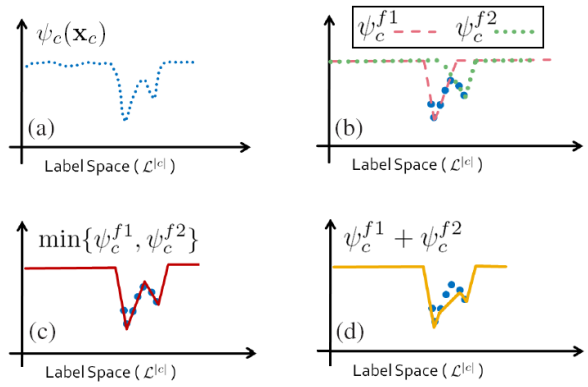

Figure 4: *Difference between the min and sum composition schemes.(a) Original function to be represented. (b) Two deviation basis functions (21). (c) The composition of the functions shown in (b) by taking their lower envelope (by minimizing over a multi-state variable as explained in section 3.2). (d) The composition of the functions shown in (b) by summing them (causes misrepresentations in regions where the basis functions overlap).*

$$f(z, \mathbf{x}_c) = 2z(\sum_{i|w_i \geq 0} w_i x_i - \sum_{i|w_i < 0} w_i(1-x_i) - 1) \quad (28)$$

Substituting this in equation (25) we see that ψ_c is a lower envelop of two functions, i.e.

$$\psi_c(\mathbf{x}_c) = \min\{\psi_c^1(\mathbf{x}_c), \psi_c^2(\mathbf{x}_c)\} \quad (29)$$

where $\psi_c^1(\mathbf{x}_c) = \theta$ and

$$\psi_c^2(\mathbf{x}_c) = \theta(\sum_{i|w_i \geq 0} w_i x_i - \sum_{i|w_i < 0} w_i(1-x_i)) \quad (30)$$

which makes it equivalent to the higher order deviation function defined in equation (21).

Behavior of the Summation Composition Scheme The method of composing higher order potentials using the summation scheme described in the previous sub-sections suffers from a problem when using compact representations of higher order potentials. This occurs when there is significant overlap in the subset of labelings which are assigned a non-threshold cost θ by multiple higher order potentials. This problem is illustrated in figure 4.

5 Sparse Potentials versus Patch-based Methods for Image Labeling

Suppose we had a dictionary containing all possible 10×10 patches that are present in natural real-world images. One could use this dictionary to define a higher order prior for the image restoration problem which can be incorporated in the standard MRF formulation. This higher order potential is defined over sets of variables, where each set corresponds to a 10×10 image patch. It enforces that patches in the restored image come from the set of natural image patches. In other words, the potential function assigns a low cost (or energy) to the labelings that appear in the dictionary of natural patches. The rest of the labelings are given a high (almost constant) cost.

It is well known that only a relatively small fraction of all possible labeling of a 10×10 patch actually appear in natural images. We can use this sparsity property to compactly represent the higher order prior potential by storing only the labelings that need to be assigned a low cost, and assigning a (constant) high cost to all other labelings. Unfortunately, for most applications, the number of patches might still be too large to be represented explicitly. There are several ideas in the literature to overcome this problem. For some application there exist an underlying (input) image which can guide the selection process such that at each position in the image a different set of candidate patches is chosen (i.e. a classical conditional random field). It has been shown that with a small set of candidate patches (e.g. k between 4 and 25) convincing results can be achieved for super-resolution [11], or new-view synthesis [9, 44][6]. An-

[6]These works use implicitly a small dictionary, since the problem of new-view synthesis is cast as a multi-labeling problem, where a label corresponds to one out of k possible colors

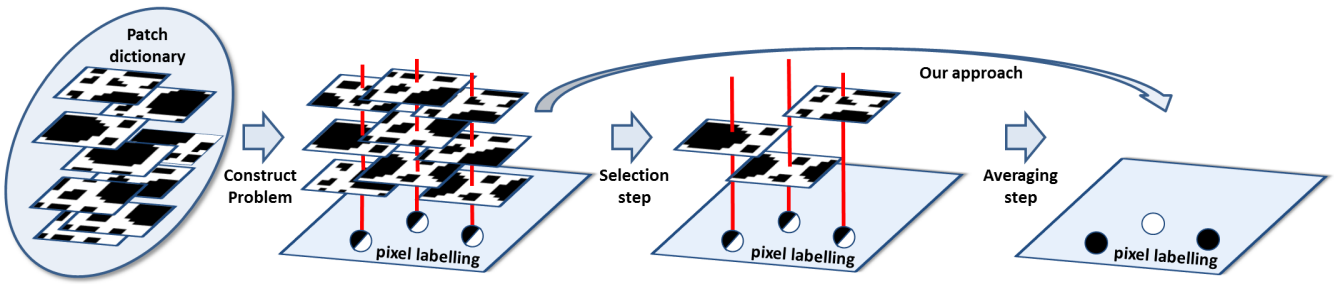

Figure 5: Illustration of the main steps involved in the "classical" framework for patched-based methods such as [11]. Here for the task of binary texture restoration. A dictionary of e.g. all 10×10 patches of natural (binary) textures is given. The first step is to select for each 10×10 pixel region a set of k candidate patches. This defines implicitly a higher order random field (see details in text). Instead of optimizing this random field, the classical approach is to construct an auxiliary labeling problem where each pixel has k possible labels, which corresponds to the k possible candidate patches. The solution to the auxiliary problem gives for each pixel one candidate patch ("selection step"). Finally, the "averaging step" constructs the output labeling by averaging all selected (overlapping) patches. The contribution of this work is to replace this two step procedure: "selection" and "averaging" with a one step procedure which directly optimizes the underlying higher order random field.

other idea is to iterate the selection process: An initial solution with a small set of candidate patches can be used to select a better set of candidate patches in the next iteration[7]. In principle, the sparse energy can have the same global optimum as the original energy defined over all natural patches. Such an energy can be constructed by using even one candidate patch with a low cost (patch from the global optimum labelling of the pixel) and assigning a very high cost to all other patches.

To summarize, sparse functions have played an important role in designing computer vision models and in optimizing them. Sparsity in patch appearance has also been exploited by other methods on sparse coding and dictionary learning for image analysis and restoration [30, 29].

As motivated above, sparse patch-based higher order random fields are good models for many vision applications, however, they are very hard to optimize, as we have seen in the previous sections. To the best of our knowledge only very few papers have attempted to optimize them directly, such as [9], which used the very local Iterated Conditional Modes (ICM) approach, and the two recent papers [24, 39], which used message passing-based techniques. The majority of papers use an alternative way to deal with patch-based observations as outlined below. The key contribution of this paper is to show that sparse patch-based higher order random fields can be optimized efficiently (using the transformation approach) and that the results are (empirically) superior to the classical, alternative way to deal with patch-based observations.

Classical Patch-based Methods Freeman *et al.* [11] was probably one of the first to describe a way to deal with patch-based observation while circumventing the need of a higher order random field. This work inspired many other patch-based algorithms for various problems such as image inpainting [7, 22], object recognition [28], and new view synthesis [9, 44].

Let us review the "classical" patch-based approach for the problem of binary texture restoration, as depicted in fig. 5 (a detailed description is given in sec. 6). Note, we only explain the main ideas. (Details differ considerably between various realizations

for each pixel.

[7]Such ideas have been addressed in [43, 12, 4].

of these ideas). Given a (very) large dictionary of all 10×10 patches of natural (binary) textures, the first step is to construct an auxiliary, pairwise MRF energy function. The solution to this auxiliary problem will help in finding the solution to the true underlying higher order random field. The auxiliary problem is also defined over pixels but has a different label set, in contrast to the true higher order function. Each position (or pixel) in the image can take one out of k labels, where each label corresponds to a (10×10) patch. The pairwise terms between two neighboring pixels (e.g. 4-connected) encode how well their chosen (overlapping) patches agree in the labeling of the pixels[8]. The auxiliary energy is minimized to get a patch labeling for the image, which we denote in fig. 5 as the "selection step" (i.e. one patch is assigned to each pixel). To get a unique binary (or RGB) value for each image pixel, a second "averaging step" is needed which extracts an binary (or RGB) pixel labeling from the patch labeling. For instance, in the case of binary texture denoising, one can average (and round) the values assigned to a pixel by various patches. Note, the non-local mean procedure [4] can also be seen as variant of this two step procedure. The idea is to select and average all possible patches, and iterate this procedure many times.

The contribution of our work is to replace this two step procedure: "selection" and "averaging" with a one step procedure which directly optimize the underlying higher order random field. This direct approach has three key advantages over the classical one:

- **"Patch robustness", i.e. inconsistency with candidate patch labelings.** If none of the k candidate patches fits at a certain position, we allow for choosing any arbitrary patch. All these arbitrary patches have a constant cost which has to be higher than all of the costs of the candidate patches.

- **Deviation from patch labelings.** In the auxiliary problem, one would ideally like to represent as many patch labelings (patterns) as possible. We propose a way to represent a set of similar patch labeling (patterns) by using a simple patch labeling (pattern) and a deviation function (sec. 3.2). This representation allows us to consider many more patch labelings for each pixel with no extra overhead on the memory or computational cost compared to using a single patch labeling.

- **Incorporation of generic pairwise (and higher order) costs on the pixel labeling.** We allow for any generic unary, pairwise (and higher order - after decomposing into pairwise terms) cost function on the pixel level to be added to the energy function. In our experiments, we demonstrate that the addition of pairwise functions is crucial to obtain good results.

Let us motivate the importance of the last point with an example from the domain of object-class specific segmentation. A standard MRF/CRF model for segmentation has pairwise potentials which incorporate a boundary length prior (shorter boundaries are encouraged). While segmenting a specific object-class such as cars, one might want to incorporate a higher order patch-based prior which encourages parts of the segmentation to take the shape of sharp corners (edge of car) and round corners (wheel of car). We believe that using both, generic length-based priors and object-specific patch-based priors may lead to best results. We illustrate this point in our experiments on texture inpainting in sec. 6.

6 Experiments

We evaluated our theoretical contributions on the problems of texture restoration and inpainting, which are popular test-beds for energy minimization methods [5, 19, 35]. The main focus of this section is to illustrate the advantages of sparse higher order models over the "classical" patch-based approaches [11], and other state-of-the art techniques such as [43]. In the following we concentrate on exploring the different aspects of our model, rather than promoting one universal model.

Note, an experimental comparison to other related work, such as the submodular triple-clique model for

[8] Various connectivity, and patch-overlap, structures have been considered in the past, see e.g. [11, 44].

texture restoration of [5] and the FRAME model [47], is beyond the scope of this work.

We first introduce our experimental set-up in sec. 6.1. We then proceed, in sec. 6.2, to compare the performance of the three transformation schemes proposed in sec. 3 and 4. After that we consider the application of texture restoration (sec. 6.3) and after that texture inpainting (sec. 6.4).

6.1 Problem set-up

Given an image of a texture as in fig. 6a (crop of Brodatz texture D101) our goal is to build a sparse higher order function which models the texture well. The model is then used to solve inference tasks such as: restoring (denoising) a noisy version of a test image (fig. 6c) or reconstructing (inpainting) an unobserved area of the texture (fig. 13a). Our model consists of pairwise and sparse higher order terms. For all applications we use the same pairwise terms, as explained next. The higher order terms are constructed in different ways (application dependent), as explained in the individual subsections. Note, for most parts of the experiments the in- and output will be binary images (0/1 labeling). (We will show one experiment with multiple labels, i.e. gray-scale images.)

To extract pairwise terms we followed the procedure outlined in [19][9], which builds on [5]. In brief, given a training image (e.g. fig. 6a), we first compute the joint histogram of all pixel-pairs with the same shift (s_x, s_y), where we constrained the maximum shift length, i.e. $\max\{|s_x|, |s_y|\} \leq 30$. The pairwise potentials are then defined as $\theta_{i,j}(x_i, x_j) = -\log Pr(x_i, x_j)$, with x_i, x_j being the output labeling at the two pixels i, j. For restoration, the unary potential is given as $\sum_{i \in \mathcal{V}} -\lambda/(1 + |T_i - x_i|)$, where T_i is the value of the pixel i in the noisy test image (e.g. fig. 6c). To obtain good result, it is important to select those pairwise potentials which best describe the texture, and to learn the optimal value for λ, i.e. the trade-off between unary and pairwise terms. As in [19], we used the discriminative learning procedure on a validation dataset, which resulted in 9 pairwise terms (7 sub- and 2 non-submodular) for the texture in fig. 6. For a test image (fig. 6b) with 60% noise (fig. 6c) such a pairwise model, optimized with QPBO [19], gives a reasonable result shown in fig. 6f with 16.4% error rate (number of misclassified pixels).

[9] We also used their data, which is available online.

6.2 Comparing Transformation Schemes

We compared the performance of our three transformations: Multi-label (sec. 3), binary Type-I and Type-II (sec. 4) on various textures (real and toy data). For optimization, we used the top performing publicly available methods for minimizing non-submodular binary and multi-label functions: (Sequential) Tree-ReWeighted message passing (TRW-S) and Belief Propagation (BP) from [38], and the roof-dual relaxation based approaches (QPBO, QPBOP) from [2, 35]. Note, in order to enable other researchers to run alternative inference techniques, we will make some problem instances publicly available

Table 1 shows results for two examples models, which capture the main conclusions we have drawn from our experiments. Some of the results of table 1 are shown in fig. 7. The toy texture (top row in table 1 and fig. 7) is a perfectly repeated texture (30×30pixels) which has only four distinctively different patterns of size 3×3. We encoded these patterns as higher order cliques, and did not use any pairwise terms. The second texture (bottom row in table 1 and fig. 7) is a 30×30 crop of the real texture in fig. 10 where we used 25 5×5 patterns without deviation cost function and additionally pairwise terms (see details in sec. 6.3).

The first conclusion is that the two binary constructions (Type-I and -II) perform very similar. For both constructions the graph-cut based techniques QPBO and QPBOP performed badly, i.e. left all nodes unlabeled for these instances. (Note it is not surprising that TRW-S also has a weak lower bound since it is solving the same LP relaxation as QPBO). Secondly, the binary constructions are considerably inferior to the multi-label construction, for all instances. For example, the multi-label construction finds the global minimum with TRW-S for the toy texture. (Unfortunately, real textures are more chal-

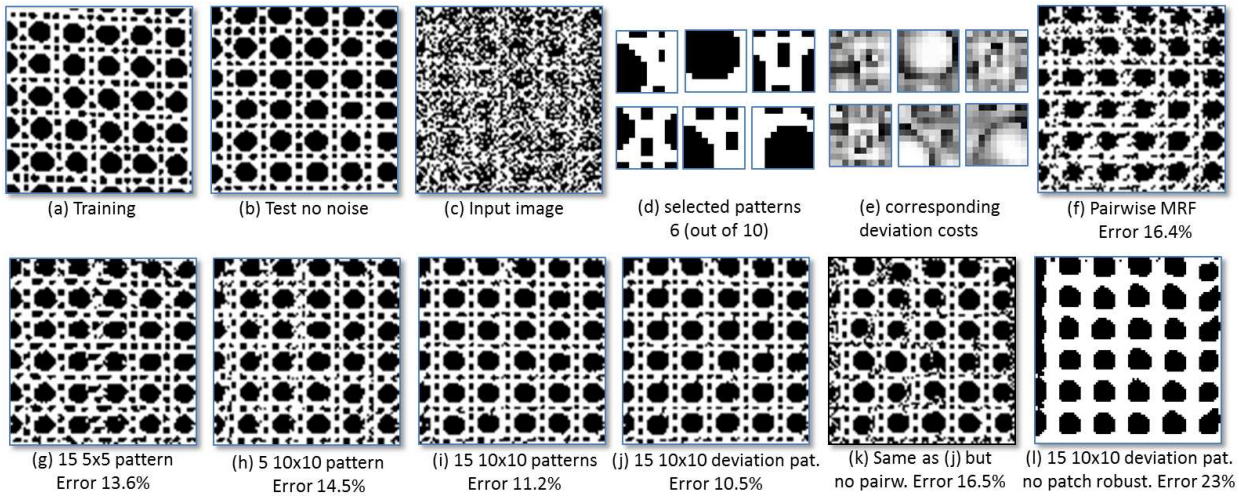

Figure 6: *Binary texture restoration for Brodatz texture D101. (a) Training image (*86×86 *pixels). (b) Test image. (c) Test image with 60% noise used as input. (d) 6 (out of 10) selected patterns of size* 10×10*pixels. (e) Their corresponding deviation cost function. (f-l) Results of various different models (see text for details).*

	Binary Type-I		Binary Type-II		Multi-label	
	BP	TRW-S	BP	TRW-S	BP	TRW-S
fig. 7 top row	86.1; 38.5%	100(-179); 60.8%	86.0; 38.5%	100(-179); 60.8%	**0; 0%**	**0(0), 0%**
fig. 7 bot. row	91.8; 19.9%	100(-624); 31.7%	91.8; 19.9%	100(-624); 31.7%	**91.2; 19.6%**	99.9(0); 30.9%

Table 1: *Comparing different transformation schemes for two different textures and various optimization methods. We report for BP: energy; error, for TRW-S energy(lower bound); error. Note, the energy values are shifted and scaled so that highest energy is* 100 *and the best lower bound is* 0*. Best performing methods are shown in bold.*

lenging and the lower bound of TRW-S can not give insights on the optimality of the solution for the real texture in table 1). The third conclusion is that BP performs consistently better than TRW-S, both in terms of lower energy and speed. Fig. 8 illustrates a typical run of BP for the model in fig. 6j. Within a few (here 3) iterations the energy reaches a low value. In the remaining iterations the energy oscillates around this energy value (here the lowest energy is reached after 489 (out of 1000) iterations, which is nearly the same energy as reached after 20 iterations). Note, for this case the convergence of TRW-S is much slower (about 1000 iterations), and the result is inferior (best energy $3.74 * 10^5$ and lower bound $-4.97 * 10^4$).

Given the above results, one might question the usefulness of the binary constructions. Two arguments in support of the binary constructions are **(a)** *we found a few instances where the binary construction slightly outperformed the multi-label construction using BP*, and **(b)** *a lot of active research in the optimization community is on pseudo-boolean optimization which hopefully will yield in the near future methods, like [21], which are superior to QPBO and QPBOP.*

Finally, we see that low energies correlate with low reconstruction error which confirms the quality of our model. In terms of runtime of BP, we see that

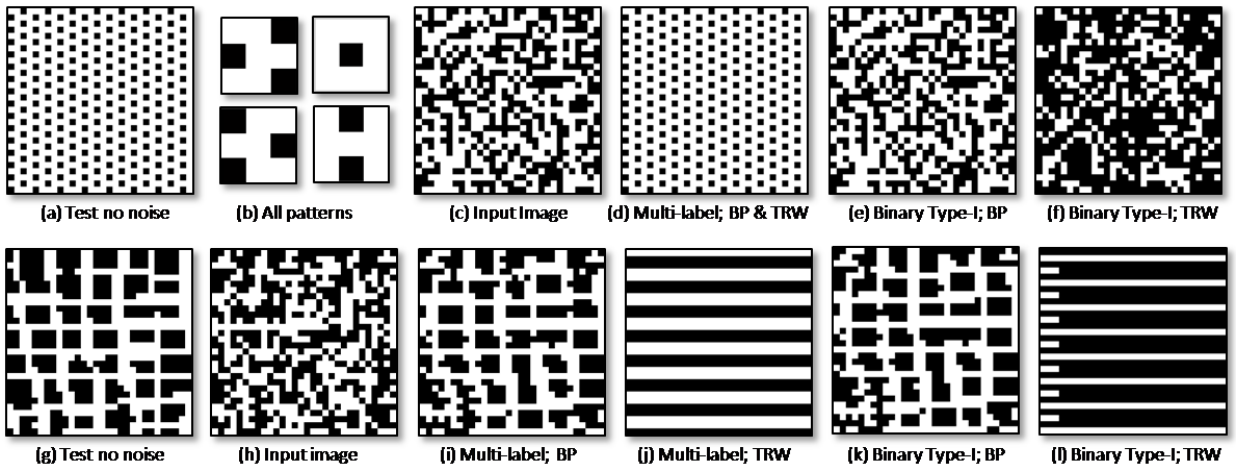

Figure 7: *Comparing different transformation schemes for a toy texture (top row) and a real texture (bottom row). The results (d-f, i-l) correspond to some experiments shown in table 1. (b) Shows the four patterns of size* 3×3*pixels which are sufficient to fully represent the toy texture in (a). We see that, especially for the toy texture (top row), the performance of the multi-label construction is considerably superior to the binary construction Type-I. Furthermore, we see that TRW-S is (apart from (d)) inferior to BP.*

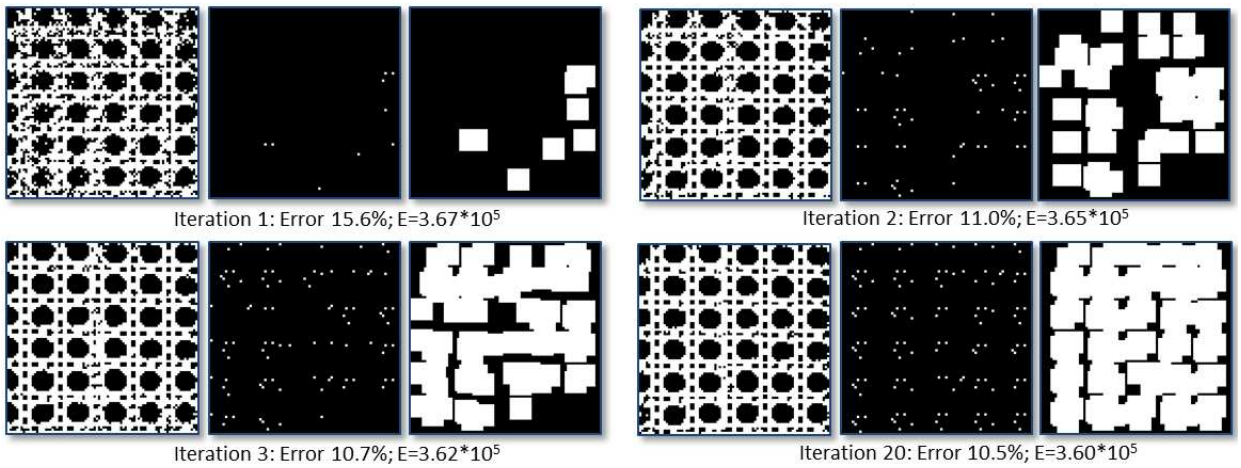

Figure 8: *A typical run of BP to achieve the result in fig. 6j. Intermediate results for* 4 *different iterations are shown. The left images show the current labeling. The middle images illustrate in black those pixels where the maximum (robustness) patch cost* $\theta_{\max}$ *is paid. It can be observed that only a few pixels do not utilize the maximum cost. The right images illustrate all* 10×10 *patches which are utilized, i.e. each white dot in the middle images relates to a patch. Note that after* 20 *iterations there is no area in the right image where a patch could be used which does not overlap with any other patch. Also, note that many patches do overlap.*

the multi-label and binary type-II constructions have about the same runtime, e.g. 2.1sec for the real texture in table 1, while binary type-I has slightly higher runtime (2.5sec) since more auxiliary nodes are used.

6.3 Texture Restoration

For the task of texture restoration we investigated two different ways to construct the sparse higher order functions. In the first one, the sparse higher order functions are constructed independently of the input image, i.e. a pure prior. In the second one, the higher order functions depend on the input image, which means that for every clique of pixels the higher order function may be different. This gives a so-called Conditional Random Field (CRF). In the following we will call the first option *"static"* higher order functions and the second on *"data-driven"* higher order functions.

Static higher order functions Given a training image as in fig. 6a we aim at select a few patterns which occur frequently and are as different as possible in terms of their hamming distance. We achieve this by k-means clustering over all training patches. Fig. 6d depicts 6 (out of k=10) such patterns. To compute the deviation function for each particular pattern $\mathbf{X}_p$ we consider all patterns which belong to the same cluster as $\mathbf{X}_p$. For each position within the patch we record the frequency of having the same value as $\mathbf{X}_p$. Fig. 6e shows the associate deviation costs, where a bright value means low frequency (i.e. high cost). As expected, lower costs are at the edge of the pattern. Note, the scale and truncation of the deviation functions, as well as the weight of the higher order function with respect to unary and pairwise terms, are set by hand in order to achieve best performance.

The results for various models are shown in fig. 6(f-l). Further results for Brodatz texture D103,D20, and D22 are shown in fig. 10, 11 and [34]. Let us summarize the main findings. The higher order functions always manage to capture visually the main (large-scale) characterizes of a texture, e.g. fig. 6j compare to fig. 6f. For most textures this is reflected in a considerably reduced error rate. Furthermore, as expected modelling more patterns helps (fig. 6h compare to fig. 6i). Also, large cliques give typically better results (fig. 6g compare to fig. 6i). Although, for texture D103 (fig. 10) the improvement of using 10×10 instead of 5×5 patterns was not noticeable. This is probably due to the fact that the repeated texture elements are of a smaller size than 10×10. Finally, the deviation functions help to improve results further (see fig. 6(i,j) and [34]). Fig. 6k shows the result with the same model as in (i) but where pairwise terms are switched off. The result is less good since those pixels which are not "covered" by a patch are unconstrained[10] and hence take the optimal noisy labeling. One way to eliminate such unconstrained pixels is to remove the option of patch robustness, i.e. for each higher order clique one candidate pattern has to be utilized. This is achieved by setting $\theta_{\max}$ to infinity and to assign to each pattern a deviation cost which is not infinite. Fig 6l shows the result. Although there are no "isolated" pixel errors, the result is not visually pleasing. This shows the importance of having patch robustness, which is missing in the classical patch-based approach (sec. 5).

In terms of runtime, using more and large higher order cliques is obviously more computationally expensive. The example in fig. 6j, which is an image of size 86×86, with 88935 patterns, each of size 100, takes 32sec (14 rounds of BP) on a 1.8Ghz CPU. Note that deviation functions do not increase the runtime.

Data-driven higher order functions The above approach is obvious suboptimal since the known input image is not exploited in the process of selecting candidate patches. We will explain such a data-driven approach now. The basic idea has been introduced in sec. 5. Given a large database of patches, e.g. all patches of the training images in fig. 9a, a set of k candidate patches is retrieved from the database, individually for each image position. As distance measure we simply use the hamming distance between the noisy test patch (fig. 6c) and all patches in the database. Since the selected patches are typically quite self-similar we did not use devi-

[10] Note, the black pixels in fig. 8(bottom, right) would be the unconstrained pixels for that example.

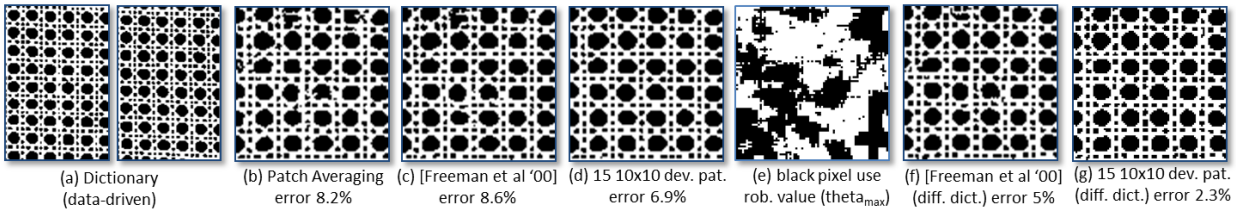

Figure 9: *The same restoration problem as in fig. 6 but now the used techniques select patches in a data-driven way by exploiting the given input image. All techniques use the same set of patches, i.e. 15* 10×10 *patches at each position. Our method (d,g) gives visually best results, since it solves the underlying higher order random field directly. Other techniques (b,c,f) are visually inferior due to the inherent averaging step of all selected patches. Note, results in (f,g) use a different dictionary which consists of the training images in (a) augmented by the ground truth input image in fig. 6b.*

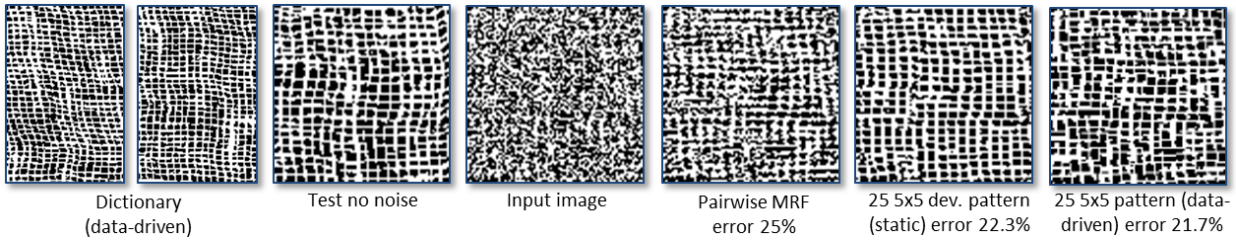

Figure 10: *Results for Brodtaz texture D103 with pairwise MRF, static higher order functions, and data-driven higher order functions, where the latter gives clearly the best result.*

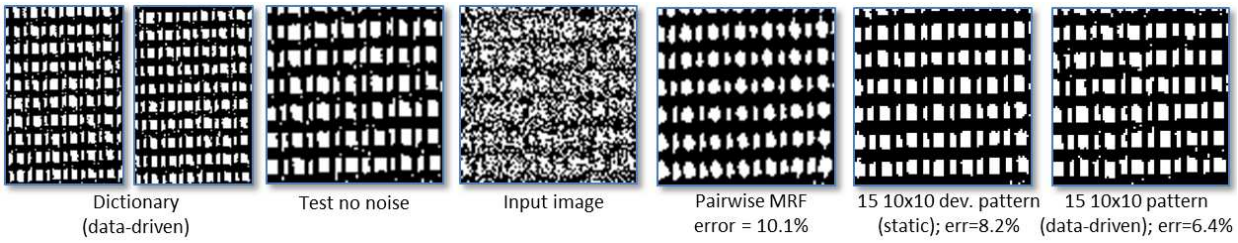

Figure 11: *Results for Brodtaz texture D20 with pairwise MRF, static higher order functions, and data-driven higher order functions, where the latter gives clearly the best result.*

ation functions. As above, pairwise functions were added to the model.

It can clearly be seen that the result with data-driven higher order functions, fig. 9d, is considerably better than with static higher order functions, fig. 6j. The same is true for other textures, see fig. 10, 11.

Figure 9e and fig. 8(bottom row, second image from right) explain why the result is so different. Both images show in white those image positions where a candidate patch has been utilized. It is apparent that in fig. 9e there are many more white pixels.

In order to emphasis the importance of solving the higher order random field directly, in contrast to the classical patch-based approach outlined in sec. 5, we run two alternative methods. Figure 9b shows the result of averaging and rounding all the available patches. While the result is not bad in terms of error rate, artifacts due the "averaging step" are visible (see e.g. top row of the texture). The same reason holds for the results with the classical patch-based approach of [11], outlined in sec. 5 — see fig. 9c. Here we used an auxiliary energy with a 4-connected pairwise MRF which models the patch overlap cost. In a final experiment we changed the dictionary to additionally include the ground truth input image (fig. 6b). Our method benefited most from the improved dictionary, with a low error rate of 2.3% (fig. 9g), followed by [11] with an error rate of 5.0% (fig. 9f) and the approach of averaging all patches (error rate 7.5%; not shown).

Multi-label output The goal is to denoise the gray-scale Brodatz texture D101 in fig. 12(b). For this we use the same "static" training procedure as above, but now with 15, instead of 2, labels for the output. The 15 output labels are equally distributed over the range of 0 to 255 gray-scale values. The input image has 256 labels. We have to use our multi-label construction (sec. 3), which also means that the candidate patterns have 15 labels.

The result with a pairwise MRF and 15 labels is shown in fig. 12d. For this we adjusted the 9 pairwise terms from sec. 6.1 which were defined for binary labels. Figure 12e depicts our result with 10 10×10 "static" patterns and pairwise terms. It is interesting to note that a noticeable improvement can be achieved by adding deviation functions, fig. 12f.

We expect that further improvements can be achieved by using data-driven functions, and by intelligently selecting the 15 candidate gray-scale colors (as done for new-view synthesis [9, 44]).

6.4 Texture Inpainting

The task of image inpainting is to invent new pixels in a region of the input image which is un-observed. In contrast to image restoration there is no information (like noisy input data) available inside the unknown region. This makes the problem quite challenging and a good prior knowledge about the image is crucial in order to achieve good results.

Figure 13a shows an inpainting task for the Brodatz texture D101 used before. The solutions of three well-known exemplar-based inpainting techniques [7, 6, 43] are shown in (b-d). These techniques only use patches of the observed part of the input image to construct a solution. They differ in the way *how* they use the patches. The approach [43], shown in (d), is probably state-of-the art and uses the patches in the most excessive way. It optimizes an objective function which forces every patch in the unknown area to be as similar as possible to an observed patch. However, even this technique does not achieve a pleasing result in this case[11].

Figure 13e shows our result with only pairwise terms used (here the same 9 pairwise terms as for texture restoration (sec. 6.1), which were trained using a different training image (fig. 6a)). We see that the result preserves the rough structure of the texture. Finally, fig. 13f shows our result using jointly pairwise terms and 15 12×12 patches with deviation function. Here we used the static higher order functions which were constructed in the same way as above (sec. 6.3), however only the observed part of the input image served as training data. In contrast

[11] We used our own implementation of [43] using the speed-up patch-match technique of [1]. The result of photoshop's inpainting method, which is based on [43], is of similiar visual quality as (d). Note, if the unknown area is smaller, then [43], as well as our approach, achieve visually satisfying results.

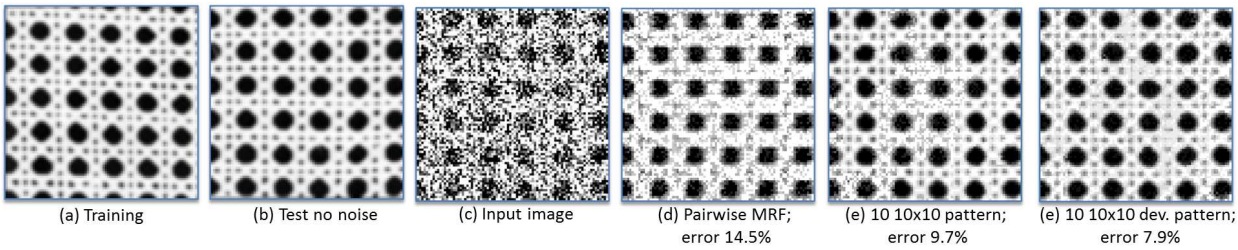

Figure 12: *Gray-scale texture restoration. (a) Training image, and (c) test image with* 60% *noise. The data (a-c) has* 256 *labels, while the results (d-f) have* 15 *labels. (d) Result with pairwise MRF. (e) Our result without deviation functions, and (f) with deviation functions.*

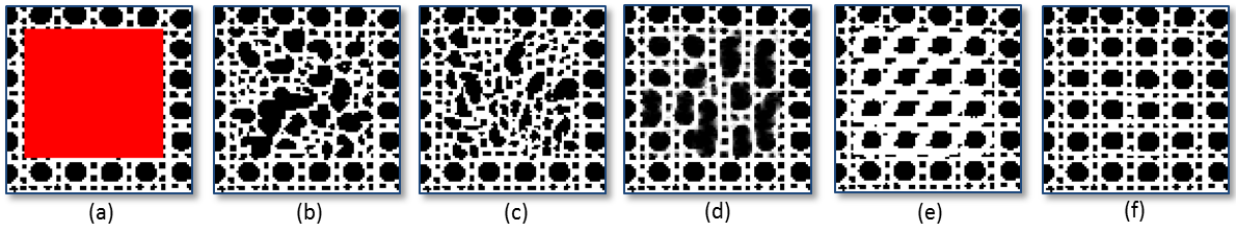

Figure 13: *The task is to inpainting the red area of the input image (a). State-of the art methods produce results: (b) Efros et al. [7], (c) Criminisi et al. [6], and (d) Wexler et al. [43]. Using our trained pairwise model gives result (e). Our result (f) is visually best, and uses static, sparse higher order functions (15* 12×12 *patterns with deviation function) and pairwise terms.*

to competitors is our solution visually appealing.

Note, in this case all ingredients of our model, i.e. patch robustness, deviation functions, and additional pairwise terms, were crucial to achieve the good result in fig. 13f. As mentioned already in sec. 5, we believe that the idea of combining generic priors (here pairwise terms) with patch-based priors may also be important for other tasks such as object-class specific segmentation.

7 Conclusion and Future Work

This paper provides a method for minimizing sparse higher order energy functions. We studied the behavior of our methods in dealing with different energy functions. We applied our theoretical contributions to the problems of binary texture restoration and inpainting, where we achieve results which improve on existing techniques, especially the classical patch-based approach of [11]. Since the classical approach is very wildly used in computer vision, such as object recognition and segmentation, we believe that our sparse higher order models will have high impact in the future.

Our different transformation methods result in difficult optimization problems which are NP-hard to solve. The use of sophisticated optimization algorithms such the ones proposed recently in [21, 25, 37, 42] in solving these problems is a interesting direction for future work.

Another future direction is to compare and analyze optimization schemes based on message passing with higher order functions. As discussed above, a sparse structure in the higher order potentials can be exploited within a dual-decomposition paradigm [24], or within a factor-graph representation [39].

Towards this end, we will make some instances of our problems publicly available in order to encourage other researchers to work on these exciting and challenging problems.

8 Acknowledgment

We would like to thank Alex Mansfield for creating results on texture inpainting with the method of [43]. We also gratefully thank Wei Feng and Jiaya Jia for their work on the conference article [36].

References

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. volume 28, 2009.

[2] E. Boros and P.L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 2002.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 2001.

[4] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *CVPR*, pages 60–65, 2005.

[5] D. Cremers and L. Grady. Learning statistical priors for efficient combinatorial optimization via graph cuts. In *ECCV*, 2006.

[6] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *CVPR*, 2003.

[7] A. A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *ICCV*, 1999.

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Belief Propagation for Early Vision. In *Proc. CVPR*, volume 1, pages 261–268, 2004.

[9] A. W. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. In *ICCV*, pages 1176–1183, 2003.

[10] D. Freedman and P. Drineas. Energy minimization via graph cuts: Settling what is possible. In *CVPR*, 2005.

[11] William T. Freeman, Egon C. Pasztor, and Owen T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[12] Stephen Gould, Fernando Amat, and Daphne Koller. Alphabet soup: A framework for approximate energy minimization. In *CVPR*, pages 903–910, 2009.

[13] Rahul Gupta, Ajit A. Diwan, and Sunita Sarawagi. Efficient inference with cardinality-based clique potentials. In *ICML*, pages 329–336, 2007.

[14] Hiroshi Ishikawa. Higher-order clique reduction in binary graph cut. In *CVPR*, 2009.

[15] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, 2001.

[16] P. Kohli, M. P. Kumar, and P. H. S. Torr. P^3 and beyond: Solving energies with higher order cliques. In *CVPR*, 2007.

[17] P. Kohli, L. Ladicky, and P. H. S. Torr. Robust higher order potentials for enforcing label consistency. In *CVPR*, 2008.

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

[19] V. Kolmogorov and C. Rother. Minimizing non-submodular functions with graph cuts – a review. *PAMI*, 2007.

[20] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts?. *PAMI*, 2004.

[21] N. Komodakis and N. Paragios. Beyond loose lp-relaxations: Optimizing mrfs by repairing cycles. In *ECCV*, 2008.

[22] N. Komodakis and G. Tziritas. Image completion using global optimization. In *CVPR*, 2006.

[23] N. Komodakis, G. Tziritas, and N. Paragios. Fast, approximately optimal solutions for single and dynamic MRFs. In *CVPR*, 2007.

[24] Nikos Komodakis and Nikos Paragios. Beyond pairwise energies: Efficient optimization for higher-order mrfs. In *CVPR*, pages 2985–2992, 2009.

[25] M. P. Kumar and P. Torr. Efficiently solving convex relaxations for map estimation. In *ICML*, 2008.

[26] Lubor Ladicky, Christopher Russell, Pushmeet Kohli, and Philip H. S. Torr. Graph cut based inference with co-occurrence statistics. In *ECCV*, pages 239–253, 2010.

[27] X. Lan, S. Roth, D. P. Huttenlocher, and M. J. Black. Efficient belief propagation with learned higher-order markov random fields. In *ECCV (2)*, pages 269–282, 2006.

[28] Anat Levin and Yair Weiss. Learning to combine bottom-up and top-down segmentation. In *ECCV*, pages 581–594, 2006.

[29] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *ICML*, 2009.

[30] Julien Mairal, Michael Elad, and Guillermo Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, 17(1):53–69, 2008.

[31] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artif. Intell.*, 29(3):241–288, 1986.

[32] Srikumar Ramalingam, Pushmeet Kohli, Karteek Alahari, and Philip H. S. Torr. Exact inference in multi-label crfs with higher order cliques. In *CVPR*, 2008.

[33] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *CVPR*, pages 860–867, 2005.

[34] C. Rother and P. Kohli. Transforming sparse higher order functions. Technical report, Microsoft Research Technical Report, 2008.

[35] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. In *CVPR*, 2007.

[36] Carsten Rother, Pushmeet Kohli, Wei Feng, and Jiaya Jia. Minimizing sparse higher order energy functions of discrete variables. In *CVPR*, 2009.

[37] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y . Weiss. Tightening lp relaxations for map using message passing. In *UAI*, 2008.

[38] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. F. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. In *ECCV (2)*, pages 16–29, 2006.

[39] Daniel Tarlow, Inmar Givoni, and Richard Zemel. Hop-map: Efficient message passing with high order potentials. In *AISTATS*, 2010.

[40] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005.

[41] Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Transactions on Information Theory*, 2001.

[42] T. Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (map-mrf). In *CVPR*, 2009.

[43] Y. Wexler, E. Schechtman, and M. Irani. Space-time completion of video. volume 29, 2007.

[44] O. J. Woodford, I. D. Reid, and A. W. Fitzgibbon. Efficient new-view synthesis using pairwise dictionary priors. In *CVPR*, 2007.

[45] O. J. Woodford, P. H. S. Torr, I. D. Reid, and A. W. Fitzgibbon. Global stereo reconstruction under second order smoothness priors. In *CVPR*, 2008.

[46] Oliver Woodford, Carsten Rother, and Vladimir Kolmogorov. A global perspective on map inference for low-level vision. In *ICCV*, 2009.

[47] S.-C. Zhu, Y. Wu, and D. Mumford. FRAME: Filters, Random fields And Maximum Entropy— towards a unified theory for texture modeling. *IJCV*, 27, 1998.