

Multi-label Moves for MRFs with Truncated Convex Priors

Olga Veksler

Received: date / Accepted: date

Abstract Optimization with graph cuts became very popular in recent years. While exact optimization is possible in a few cases, many useful energy functions are NP hard to optimize. One approach to approximate optimization is the so-called *move making* algorithms. At each iteration, a move-making algorithm makes a proposal (move) for a pixel p to either keep its old label or switch to a new label. Two move-making algorithms based on graph cuts are in wide use, namely the swap and expansion. Both of these moves are binary in nature, that is they give each pixel a choice of only two labels. An evaluation of optimization techniques shows that the expansion and swap algorithms perform very well for energies where the underlying MRF has the Potts prior. However for more general priors, the swap and expansion algorithms do not perform as well. The main contribution of this paper is to develop *multi-label* moves. A multi-label move, unlike expansion and swap, gives each pixel has a choice of more than two labels to switch to. In particular, we develop several multi-label moves for truncated convex priors. We evaluate our moves on image restoration, inpainting, and stereo correspondence. We get better results than expansion and swap algorithms, both in terms of the energy value and accuracy.

Keywords discrete optimization · Markov random fields (MRF) · graph cuts · truncated convex priors

1 Introduction

Energy optimization with graph cuts [8, 15, 9, 22] is very popular in computer vision and graphics. It has been used for such diverse applications as image restoration [9], stereo and multi-view reconstruction [8, 20, 9, 21], motion [40], texture synthesis [25], segmentation [6, 5, 31, 17], digital photomontage [2], digital tapestry [30], image generation [28], computational photography [26], image completion [13], digital panoramas [1].

Optimization with graph cuts is successful, largely, because either an exact minimum or an approximate minimum with certain quality guarantees is found, unlike the older optimization techniques, such as Simulated Annealing [11] or ICM [4]. It is important, therefore, to continue seeking better optimization methods for computer vision problems. This paper is about improved optimization methods for a subclass of energy functions that is useful for computer vision.

Typically an energy minimization problem is formulated in a pixel labeling framework as follows. We have a set of sites \mathcal{P} , and a set of labels \mathcal{L} . \mathcal{P} is often the set of all image pixels, and \mathcal{L} is a finite set that represents the property that needs to be estimated at each site, such as intensity, color, etc. The task is to assign a label f_p to each site p so that some energy function $E(f)$ is minimized. Here f is the collection of all pixel-label assignments.

If \mathcal{L} has size two and $E(f)$ is submodular [22], then $E(f)$ can be optimized exactly by finding a minimum cut on a certain graph. All algorithms presented in this paper ultimately rely on binary (or pseudo-boolean) submodular energy optimization. While there are other methods for pseudo-boolean minimization, and we could certainly use them. However in [36] they show that

This research was supported in part by NSERC, CFI, ERA.

Computer Science Department
University of Western Ontario, London, Canada
Tel.: +1-519-6612111
Fax: +11-519-6613515
E-mail: olga@csd.uwo.ca

the graph-cut based ones have the best performance in terms of running time for computer vision problems.

Many vision problems require a multi-label \mathcal{L} . The condition of submodularity can be generalized to the multi-label case [35], and a submodular pairwise multi-label $E(f)$ can be, again, optimized exactly with a minimum graph cut. Unfortunately, the class of submodular multi-label energies is not that widely useful for vision problems, see the discussion in section 2.3.

Optimizing many important multi-label energy functions is NP-hard [9]. Interestingly, there are so called *move making* algorithms for approximation that are based on pseudo-boolean optimization. Two most popular such algorithms are expansion and swap [9]. A move making algorithm performs iterative optimization starting with an initial guess \hat{f} . At each move, each site p has a choice restricted to only two labels, one of them is usually the initial label \hat{f}_p . The best move (or switch) to a new labeling f' is found with restriction to only two labels per site. This is a pseudo-boolean optimization problem, solvable with graph cuts in the submodular case. The swap and expansion algorithms differ in the choice of two particular labels allowed for a pixel. These move making algorithms are powerful because they search efficiently a combinatorially large space.

Recently, Szeliski et.al. [36] performed an experimental evaluation of energy minimization methods common in vision: the expansion and swap algorithms [9], loopy belief propagation (LBP) [29], and sequential tree-reweighted message passing (TRW-S) [16]. Their results show that for energies based on the Potts model (see section 2.1) both expansion and swap algorithms perform very well, getting an answer within a small percentage of the global minimum. TRW-S performs as well as graph cuts, but takes significantly longer to converge.

For other useful energies, such as those arising from truncated convex priors (see section 2.1) the expansion and swap algorithms do not perform as well. In this paper, we develop several new move making algorithms for truncated convex priors that work better than expansion or swap moves. To get better optimization, we develop moves that are multi-label in nature, unlike swap and expansion. That is our moves give each pixels a choice of more than two labels to switch to. All of the new moves are based on the well known methods for exact optimization of multi-label submodular energies [14, 35]. However instead of exact optimization, these methods form a basis for our new multi-label moves for approximate optimization.

There are several move types that we develop. One type can be seen as multi-label generalization of the

expansion, and another as multi-label generalization of the swap algorithms, so we call them, respectively, multi-label expansion and multi-label swap. Note that the optimal multi-label expansion move can be found only approximately. Another interesting move that we explore and that is distinct from multi-label expansion and swap is a multi-label smooth swap.

We also develop a multi-label move for the Potts model that we call a *double* expansion. We show that it is more powerful (namely the set of expansion moves is strictly contained in the set of double expansion moves) than the expansion algorithm. We show an artificial example where it escapes a local minimum an expansion algorithm gets stuck in, and point out some practical applications where it may be useful.

The main idea of our approach (preliminary versions in [38,39]) is to design multi-label moves (as opposed to binary moves) in order to improve optimization. Since our original work, there were several related developments that further explore the idea of multi-label moves. The authors of [27] develop multi-label moves for a very specific energy function with strong ordering constraints on labels. In [23], simultaneously but independently [37], they develop a move similar to our multi-label expansion, and prove approximation bounds. Their graph construction is very similar, with some minor differences in the edge weights. An extended version of their work appears in [24]. The approach in [10] is similar in spirit to our multi-label swap moves, but they apply it for approximate optimization of a multi-label submodular $E(f)$ in order to reduce the time and space complexity of the exact optimization algorithm [35]. Interestingly, the set of labels can be different for each pixel. In [12] they develop approximation algorithms that are based on multi-label moves for very general energy functions. Their moves are also called expansion moves, with the range of labels to expand on chosen, possibly dynamically, for each pixel individually. They cannot guarantee that the optimal move can be found. However, in cases when it can be found, they prove certain optimality guarantees. They use an LP-based solver for approximating the best expansion move. In comparison, in our work, due to our restriction to the truncated-convex priors, we can find the optimal move in most cases.

We evaluate our method on problems of image restoration, inpainting, and stereo correspondence. Our results show that we are able to get more accurate answers, both in terms of energy value and accuracy of the labeling.

2 Energy Optimization with Graph Cuts

In this section, we briefly explain the relevant prior work on optimization with graph cuts.

2.1 Energy Function

We first formulate the energy function to be optimized. Recall that \mathcal{P} is the set of pixels, \mathcal{L} is the set of labels, $f_p \in \mathcal{L}$ is the label assigned to pixel p , and f is the collection of all pixel-label assignments. In the energy optimization framework, usually the following energy is minimized:

$$E(f) = \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{(p,q) \in \mathcal{N}} V_{pq}(f_p, f_q). \quad (1)$$

The first sum in Eq. (1) is called the data term, because it is usually modeled from the observed data. Individual pixel-label preferences are given by $D_p(f_p)$. The second sum in Eq. (1) is called the smoothness term, and it represents the prior knowledge about the likely labelings f . The name smoothness comes from the fact that the prior knowledge frequently encodes smoothness assumptions about f . The second term is the sum over ordered pixel pairs $(p, q) \in \mathcal{N}$. Usually \mathcal{N} is the 4 or 8 connected grid, however longer range interactions are also useful [20]. Without loss of generality, we assume that if $(p, q) \in \mathcal{N}$ then $p < q$. The energy function in Eq. (1) arises in Maximum A Posteriori (MAP) estimation in Markov Random Fields (MRF).

Usually the difficulty of optimizing the energy in Eq. (1) is determined by the form of V_{pq} 's, whereas the form of D_p 's is often inconsequential. Different choices of V_{pq} 's correspond to different smoothness assumptions. A common choice is the Potts model which is $V_{pq}(f_p, f_q) = w_{pq} \cdot \min\{1, |f_p - f_q|\}$. Intuitively, Potts model assumes that f is piecewise constant, that is f consists of several pieces where pixels inside the same piece share the same label.

Other common choices for V_{pq} are

$$V_{pq}(f_p, f_q) = w_{pq} \cdot \min\{T, |f_p - f_q|\}$$

and

$$V_{pq}(f_p, f_q) = w_{pq} \cdot \min\{T^2, (f_p - f_q)^2\},$$

the truncated linear and truncated quadratic, respectively. These V_{pq} 's correspond to the piecewise smooth assumption on f , that is f is expected to consist of several pieces, where the labels inside each piece vary "smoothly"¹. It is important to limit the penalty from

above by a *truncation* constant T . Otherwise $|l_1 - l_2|$ or $(l_1 - l_2)^2$ might be prohibitively large, and assigning labels l_1 and l_2 to neighboring pixels will be too costly, resulting in an oversmoothed labeling f . Without the truncation, that is if V_{pq} is the absolute linear or quadratic difference, the energy in Eq. (1) can be optimized exactly with a graph cut [14]. Energy in Eq. (1) is NP-hard to optimize for truncated linear or quadratic V_{pq} 's, as well as for the Potts V_{pq} [9].

2.2 Assumptions on the Label Set

For the rest of the paper we assume that the labels can be represented as integers in the range $\{0, 1, \dots, k\}$, which is necessary since we base our method on the construction in [14]. Assuming integer labels rules out directly using our methods for motion estimation, since in motion, labels are two dimensional. However, there are indirect ways to apply our methods to motion, by fixing one component of a motion vector and letting the other one vary [32].

2.3 Convex Priors

Ishikawa [14] develops a method to find the exact minimum of the energy function in Eq. (1) in the case when the terms V_{pq} are convex functions of the label differences. We say that $V_{pq}(l_1, l_2) = w_{pq} \cdot g(l_1 - l_2)$ is convex if for any integer x , $g(x+1) - 2g(x) + g(x-1) \geq 0$. It is assumed that $g(x)$ is symmetric and non-negative.

Convex V_{pq} include the absolute and squared difference functions as a special case. While the energy arising from convex priors may oversmooth the answer, Ishikawa's construction gives us an important tool for energy optimization with truncated convex priors, which correspond to energies that are less likely to oversmooth. To explain Ishikawa's construction, we could define a pseudo boolean submodular energy that encodes the original multi-label problem. However, the original presentation [14] in terms of the graph construction is easier to understand, so we choose to follow it. Note that the construction we give in this section slightly differs in the edge weights from that in [14].

Ishikawa's method is based on computing a minimum cut on a particular graph. The reader unfamiliar with minimum cuts can review this topic in [7]. There are two special nodes in the graph, the source s and the sink t . For each $p \in \mathcal{P}$, we create a set of nodes p_0, p_1, \dots, p_{k+1} . We identify p_0 with the source s , and we identify p_{k+1} with the sink t . We connect node p_i to node p_{i+1} with a directed edge e_i^p for $i = 0, 1, \dots, k$. In addition, for $i = 0, 1, \dots, k$, node p_{i+1} is connected to node p_i with a directed edge of infinite weight. This

¹ The term "smoothly" is used informally here.

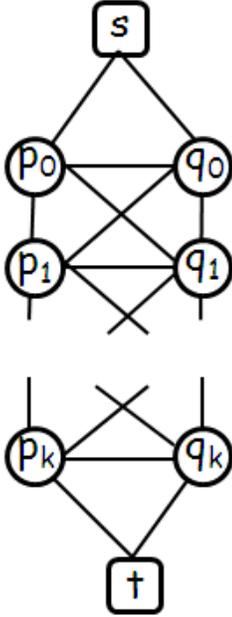


Fig. 1 Sketch of the graph construction from [14].

ensures that for each p , only one of the edges of type e_i^p will be in the minimum cut, see [14]. If an edge e_i^p is cut, then pixel p is assigned label i . Thus a cut C of finite cost corresponds to a labeling f^C in a unique way.

Furthermore, for any $(p, q) \in \mathcal{N}$, an edge e_{ij}^{pq} which connects node p_i to node q_j is created for $i = 0, \dots, k+1$ and $j = 0, \dots, k+1$. The weight of this edge is

$$w(e_{ij}^{pq}) = \frac{w_{pq}}{2} [g(i-j+1) - 2g(i-j) + g(i-j-1)]. \quad (2)$$

The edge weight defined by Eq. (2) is non-negative, since $g(x)$ is convex. This is important, since min-cut algorithms require non-negative edge weights. The graph construction is illustrated in fig. 1.

Let C be a cut of finite cost. Let $(p, q) \in \mathcal{N}$. If edges e_i^p and e_j^q are in the cut C , then all the edges in the set S_{pq}^{ij} , defined below, also have to be in the cut C .

$$S_{pq}^{ij} = \{e_{lm}^{pq} | 0 \leq l \leq i, j+1 \leq m \leq k+1\} \cup \\ \{e_{lm}^{pq} | i+1 \leq l \leq k+1, 0 \leq m \leq j\}.$$

When summing up over S_{pq}^{ij} , most of edge weight cancel out, and we are left with

$$\sum_{e \in S_{pq}^{ij}} w(e) = w_{pq} [g(i-j) + g(k+2) + h(i) + h(j)],$$

where $h(i) = -\frac{1}{2}[g(k+1-i) + g(i+1)]$. Recall that the cut C corresponds to a labeling f^C . Except some extra terms, the sum above is almost exactly $V_{pq}(i, j) =$

$V_{pq}(f_p^C, f_q^C) = w_{pq} \cdot g(i-j)$. The term $g(k+2)$ can be ignored since it is a constant and does not change the minimum cut, just its cost. Terms $h(i)$ and $h(j)$ can be subtracted from the costs of edges e_i^p and e_j^q . Therefore we define the weights e_i^p as follows:

$$w(e_i^p) = D_p(i) - \sum_{q \in \mathcal{N}_p} w_{pq} \cdot h(i),$$

where \mathcal{N}_p is the set of neighbors of pixel p . Under this edge weights assignment, the cost of any finite cut C is exactly $E(f^C)$ plus a constant. Therefore the minimum cut gives the optimal labeling.

For the absolute linear V_{pq} this construction leads to a graph with $O(|\mathcal{P}| \cdot |\mathcal{L}|)$ (where $|S|$ denotes the size of set S) vertices and edges, assuming 4-connected grid. This is because the edges of type e_{ij}^{pq} have zero weight unless $i = j$. For more general V_{pq} , for example the squared difference V_{pq} , the number of vertices is still $O(|\mathcal{P}| \cdot |\mathcal{L}|)$, but the number of edges is $O(|\mathcal{P}| \cdot |\mathcal{L}|^2)$.

Note that [19] develops an algorithm for minimizing energy with convex V_{pq} which is more memory and time efficient. However it can be used only when the D_p 's are convex. Also note that [35] generalizes the results in [14] and develops a method to exactly optimize submodular multi-label energies.

2.4 Expansion and Swap Algorithms

Boykov et.al. [9] develop expansion and swap algorithms based on graph cuts for minimizing the energy in Eq. (1). The swap algorithm can be applied when V_{pq} is Potts, truncated linear or quadratic, and the expansion algorithm can be applied to the Potts and truncated linear V_{pq} . The answer is only approximate, which is not surprising, since the energy is NP-hard to optimize [9]. However, in case of the Potts V_{pq} , the expansion algorithm gives an answer within a factor of 2 from the optimal [9], although in practice, the answer is much closer to the optimal [36].

Both the expansion and swap algorithms find a local minimum of the energy function in the following sense. For each f , we define a set of "moves" $M(f)$, which is just a set of labelings that we are allowed to move to from f . We say that f is a local minimum with respect to a set of moves $M(f)$, if $E(f') \geq E(f)$ for any $f' \in M(f)$. A move from f to f' is *standard* if there is at most one pixel p s.t. $f_p \neq f'_p$. Such moves are used, for example, in the ICM algorithm [4]. The number of standard moves is $O(|\mathcal{P}| |\mathcal{L}|)$, therefore an optimal standard move is easy to compute.

Swap moves are defined as follows. Given a labeling f and a label pair (α, β) , a move from f to f' is called

an α - β swap if $f_p \neq f'_p \Rightarrow f_p, f'_p \in \{\alpha, \beta\}$. That is an α - β swap reassign labels α, β among pixels that are labeled either α or β in f . $M(f)$ is then defined as the collection of α - β swaps for all pairs of labels $\alpha, \beta \in \mathcal{L}$.

The expansion moves are defined as follows. Given a labeling f and a label α , a move f' is called an α -expansion if $f_p \neq f'_p \Rightarrow f'_p = \alpha$. That is the set of pixels assigned α can only expand from f to f' . $M(f)$ is then defined as the collection of α -expansions for all labels $\alpha \in \mathcal{L}$.

The expansion and swap algorithm finds a local minimum with respect to expansion or swap moves, correspondingly. In either case, the number of moves is exponential in the number of pixels, and so the exhaustive search is ruled out. In [9] they describe how to compute, for a given a labeling f , the optimal α -expansion and the optimal α - β swap by finding a minimum cut on a certain graph. The conditions on V_{pq} for the expansion or swap algorithm to work were generalized from those in [9] by [22].

According to [22], the swap algorithm may be used whenever $V_{pq}(\alpha, \alpha) + V_{pq}(\beta, \beta) \leq V_{pq}(\alpha, \beta) + V_{pq}(\beta, \alpha)$ for all $\alpha, \beta \in \mathcal{L}$. The expansion algorithm may be used whenever $V_{pq}(\alpha, \alpha) + V_{pq}(\beta, \gamma) \leq V_{pq}(\beta, \alpha) + V_{pq}(\alpha, \gamma)$ for all $\alpha, \beta, \gamma \in \mathcal{L}$. Therefore the requirements for expansion algorithm are more strict than those for the swap algorithm. For example, the energy with truncated linear V_{pq} can be optimized by both expansion and swap algorithms, whereas for truncated quadratic V_{pq} , only the swap algorithm applies. In practice, however, it is possible to apply the expansion algorithm with a ‘‘truncation trick’’ [30] to energies which do not satisfy the necessary inequality above. The truncation trick lowers the value of V_{pq} for the non-submodular terms in order to make them submodular. The resulting labeling is no longer guaranteed to be a local minimum with the respect to expansion moves, but the energy is guaranteed to go down.

Unlike the method in [14], the expansion and swap algorithms are iterative. We start with an initial labeling f . Then we iterate until convergence over labels $\alpha \in \mathcal{L}$ for the expansion and over pairs of $\alpha, \beta \in \mathcal{L}$ for the swap algorithm. At each iteration, we find the optimal α -expansion (or α - β -swap) from the current labeling, and then replace the current labeling with it.

3 Multi-label Swap

We are now ready to develop multi-label moves for optimizing energies with truncated convex V_{pq} . In this section we develop the multi-label swap and in sections 4 and 5 we develop the multi-label expansion and multi-label smooth swap, respectively.

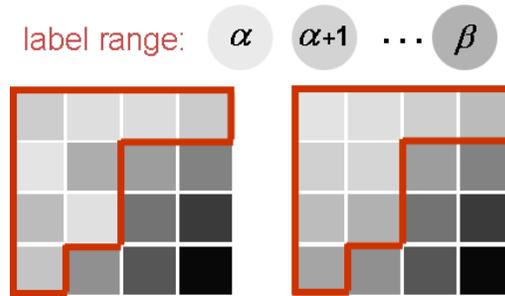


Fig. 2 Illustration for the multi-label swap. Left: initial labeling. Right: multi-label swap from labeling on the left. Allowed range of labels $\{\alpha, \alpha + 1, \dots, \beta\}$ is on top. Sites participating in the move are outlined with a thick line.

First we define the truncated convex V_{pq} . V_{pq} is said to be truncated convex if there exists a constant T and a symmetric function $g(x)$ such that $g(x + 1) - 2g(x) + g(x - 1) \geq 0$ and

$$V_{pq}(l_1, l_2) = w_{pq} \cdot \min\{g(l_1 - l_2), g(T)\}. \quad (3)$$

We now develop the multi-label swap move. Recall that the label set is $\mathcal{L} = \{0, 1, \dots, k\}$. Let $\mathcal{L}_{\alpha\beta} = \{\alpha, \alpha + 1, \dots, \beta\}$, where $\alpha < \beta \in \mathcal{L}$. That is $\mathcal{L}_{\alpha\beta}$ is the subset of labels containing consecutive integers. Given a labeling f , we say that f' is an α - β multi-label swap from f , if $f_p \neq f'_p \Rightarrow \{f_p, f'_p\} \subset \mathcal{L}_{\alpha\beta}$. The α - β multi-label swap is a generalization of α - β swap moves. An α - β swap move reassigns labels α, β among pixels that are currently labeled α and β . An α - β multi-label swap reassigns the labels in the range $\{\alpha, \alpha + 1, \dots, \beta\}$ among the pixels that currently have labels in the range $\{\alpha, \alpha + 1, \dots, \beta\}$. An illustration of a multi-label swap is in Fig. 2

Of course, if we knew how to find the best α - β range move for $\alpha = 0$ and $\beta = k$, we would find the global optima of the energy function, which is not feasible since the problem is NP-hard, in general. However, we can find the optimal α - β multi-label swap if $|\alpha - \beta| \leq T$.

3.1 α - β Multi-label Swap for $|\alpha - \beta| \leq T$

To simplify the description, we will assume that $|\alpha - \beta| = T$, but it is trivial to extend the construction in this section to handle the case when $|\alpha - \beta| < T$. Suppose that we are given a labeling f and we wish to find the optimal α - β multi-label swap, where $|\alpha - \beta| = T$. The graph construction is similar to that in section 2.3. Let $\mathcal{T} = \{p | \alpha \leq f_p \leq \beta\}$. Notice that the truncated convex terms V_{pq} become convex when $p, q \in \mathcal{T}$, since for any $p, q \in \mathcal{T}$, $V_{pq}(f_p, f_q) = w_{pq}g(f_p - f_q)$.

We identify label set $\mathcal{L}_{\alpha\beta}$ with set $\{0, 1, \dots, T\}$ and employ the construction in section 2.3 but only on the pixels in the subset \mathcal{T} and with a slight modification.

A modification is needed because the construction in section 2.3 does not consider the effect of terms V_{pq} on the boundary of \mathcal{T} , that is those V_{pq} for which we have $|\{p, q\} \cap \mathcal{T}| = 1$. An adjustment to the weights of edges e_i^p , described below, solves this boundary problem.

First we need more notation. Given a $\mathcal{T} \subset \mathcal{P}$, let

$$E_{\mathcal{T}}(f) = \sum_{p \in \mathcal{T}} D_p(f_p) + \sum_{(p,q) \in \mathcal{N}, \{p,q\} \cap \mathcal{T} \neq \emptyset} V_{pq}(f_p, f_q).$$

In words, $E_{\mathcal{T}}(f)$ is the sum of all the terms of the energy function which depend on pixels in \mathcal{T} . Also let

$$E_{\mathcal{T}}^{open}(f) = \sum_{p \in \mathcal{T}} D_p(f_p) + \sum_{(p,q) \in \mathcal{N}, \{p,q\} \subset \mathcal{T}} V_{pq}(f_p, f_q).$$

In words, $E_{\mathcal{T}}^{open}(f)$ is the sum of all the terms of the energy function which depend *only* on pixels in \mathcal{T} . Note that $E_{\mathcal{T}}^{open}(f) \neq E_{\mathcal{T}}(f)$ in most cases.

Let $M^{\alpha\beta}(f) = \{f' | f'_p \neq f_p \Rightarrow f'_p, f_p \in \mathcal{L}_{\alpha\beta}\}$. That is $M^{\alpha\beta}(f)$ is exactly the set of all α - β multi-label swap moves from labeling f . If we directly use the construction in section 2.3 on pixels in \mathcal{T} and labels in $\mathcal{L}_{\alpha\beta}$ then we will find the $f' \in M^{\alpha\beta}(f)$ s.t. $E_{\mathcal{T}}^{open}(f')$ is as small as possible. However, we actually want to find $f' \in M^{\alpha\beta}(f)$ that makes $E_{\mathcal{T}}(f')$ is as small as possible. This is since for $f' \in M^{\alpha\beta}(f)$,

$$E(f') = E_{\mathcal{T}}(f') + E_{\mathcal{P}-\mathcal{T}}^{open}(f') = E_{\mathcal{T}}(f') + E_{\mathcal{P}-\mathcal{T}}^{open}(f), \quad (4)$$

where $\mathcal{P} - \mathcal{T}$ denotes set difference. Thus the labeling $f' \in M^{\alpha\beta}(f)$ which minimizes $E_{\mathcal{T}}(f')$ gives the biggest decrease in energy from f to f' among all $f' \in M^{\alpha\beta}(f)$. Notice that since $f \in M^{\alpha\beta}(f)$, we are guaranteed to have $E(f') \leq E(f)$, for any $f' \in M^{\alpha\beta}(f)$ that makes $E_{\mathcal{T}}$ as small as possible.

This boundary problem is easy to fix. For each pixel $p \in \mathcal{T}$, if there is a neighboring pixel $q \notin \mathcal{T}$, we add to the weight of edge e_i^p an additional cost which equals to $V_{pq}(i, f_q)$, for all $i = 0, 1, \dots, k$. Recall that we identified the label set $\{\alpha, \alpha + 1, \dots, \beta\}$ with the label set $\{0, 1, \dots, T\}$. Therefore $V_{pq}(i, f_q) = V_{pq}(i + \alpha, f_q)$. This additional weight to edges e_i^p makes sure that the terms V_{pq} on the boundary of \mathcal{T} are accounted for. Now this fixed construction will find the $f' \in M^{\alpha\beta}(f)$ which optimizes $E_{\mathcal{T}}(f')$.

Just as with α - β swaps, the algorithm starts at some labeling f . Then it iterates over a set of label ranges $\{\alpha, \dots, \beta\}$ with $|\alpha - \beta| = T$, finding the best α - β multi-label swap move f' and switching the current labeling to f' .

The memory requirement grows linearly with T for the truncated linear V_{pq} , and quadratically with T for the truncated quadratic V_{pq} . Thus the larger is T , the more powerful the move is, but the more time and memory will be required.

3.2 Generalized Multi-label Swap

We can generalize the construction in the previous section. As before, let $|\alpha - \beta| = T$ (the case of $|\alpha - \beta| < T$ is basically identical) and let $\mathcal{T} = \{p | \alpha \leq f_p \leq \beta\}$. Let

$$\mathcal{L}_{\alpha\beta t} = \{\alpha - t, \alpha - t + 1, \dots, \beta + t - 1, \beta + t\} \cap \mathcal{L},$$

that is $\mathcal{L}_{\alpha\beta t}$ extends the range of $\mathcal{L}_{\alpha\beta}$ by t in each direction, making sure that the resulting range is still a valid range of labels in \mathcal{L} . Let

$$M^{\alpha\beta t}(f) = \{f' | f'_p \neq f_p \Rightarrow f_p \in \mathcal{L}_{\alpha\beta}, f'_p \in \mathcal{L}_{\alpha\beta t}\}.$$

That is $M^{\alpha\beta t}(f)$ is a set of moves that change pixels labels from $\mathcal{L}_{\alpha\beta}$ to labels in $\mathcal{L}_{\alpha\beta t}$. Notice that $M^{\alpha\beta}(f) \subset M^{\alpha\beta t}(f)$. We actually cannot find the optimal move in $M^{\alpha\beta t}(f)$, but we can find $\hat{f} \in M^{\alpha\beta t}(f)$ s.t. $E(\hat{f}) \leq E(f^*)$, where f^* is the optimal move in $M^{\alpha\beta}(f)$. Thus labeling \hat{f} is not worse than the optimal move in $M^{\alpha\beta}(f)$, and if we are lucky, $E(\hat{f})$ could be significantly better than the optimal move in $M^{\alpha\beta}(f)$.

We use basically the same construction as in section 3.1. We construct a graph for pixels in $\mathcal{T} = \{p | \alpha \leq f_p \leq \beta\}$. However, the label range is now $\mathcal{L}_{\alpha\beta t}$, and as before, we identify it with label set $\{0, 1, \dots, |\mathcal{L}_{\alpha\beta t}| - 1\}$. The rest of the graph construction is identical to that in section 3.1.

This construction finds a labeling under the energy $E^{-\mathcal{T}}(f)$, where $E^{-\mathcal{T}}(f)$ is the same as $E(f)$, except there is no truncation of V_{pq} terms for $p, q \in \mathcal{T}$. That is for $p, q \in \mathcal{T}$, $V_{pq}(f_p, f_q) = w_{pq} \cdot g(f_p - f_q)$ under the energy $E^{-\mathcal{T}}(f)$. There is a correct truncation if either p or q are not in \mathcal{T} .

Clearly for any labeling f , $E^{-\mathcal{T}}(f) \geq E(f)$. Let \hat{f} be the labeling returned by our construction, i.e. it is the labeling that minimizes $E^{-\mathcal{T}}(f')$ over $f' \in M^{\alpha\beta t}(f)$. Let f^* be the optimal move in $M^{\alpha\beta}(f)$. Since $f^* \in M^{\alpha\beta t}(f)$, we have that $E^{-\mathcal{T}}(\hat{f}) \leq E^{-\mathcal{T}}(f^*)$, and therefore $E(f) \leq E^{-\mathcal{T}}(f^*)$. However, $E^{-\mathcal{T}}(f^*) = E(f^*)$, since V_{pq} terms do not need to be truncated for $p, q \in \mathcal{T}$ for any $f' \in M^{\alpha\beta}(f)$. Therefore we have the desired result, namely $E(\hat{f}) \leq E(f^*)$.

If time and memory resources were not an issue, the best option would be to set t to a large value. However, with larger t the running time increases dramatically, especially for the truncated quadratic V_{pq} , since the size of the graph is quadratic in the number of labels in this case. Also, the larger is the value of t , the more the graph construction overestimates the term $V_{pq}(f_p, f_q)$, and it is too costly to assign labels f_p and f_q under such overestimated cost. Experimentally, the best trade off that we found between improvement in the energy and increase in the running time is when t is set to a small constant, such as 2 or 3 for the examples in the experimental section.

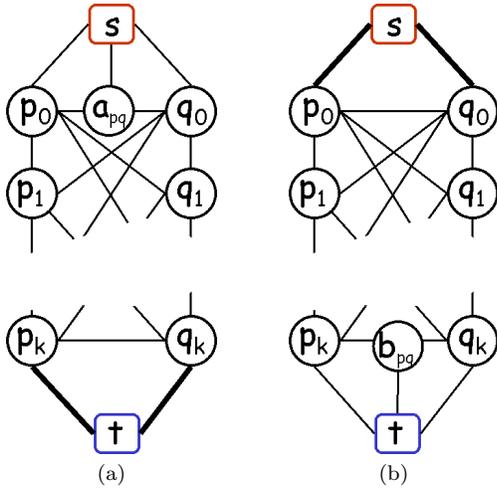


Fig. 3 Graph construction for the multi-label expansion.

4 Multi-label Expansion

The expansion algorithm usually performs better than the swap algorithm [36], and has optimality guarantees in some cases [9]. Therefore it is promising to extend a binary expansion to a multi-label expansion, in a hope of better performance. Multi-label expansion is the topic of this section.

Let α and β be two labels s.t. $\alpha < \beta$. A multi-label expansion generalizes the binary expansion move. Each pixel can either stay with its old label, or switch to a label in the set $\{\alpha, \alpha + 1, \dots, \beta\}$. The name “expansion”, as before, reflects the fact that labels in the set $\{\alpha, \alpha + 1, \dots, \beta\}$ expand their territory.

Let $M^{\alpha\beta}(f) = \{f' | f'_p \neq f_p \Rightarrow f'_p \in \mathcal{L}_{\alpha\beta}\}$. That is $M^{\alpha\beta}(f)$ is exactly the set of all α - β multi-label expansions from labeling f . Unfortunately, the optimal expansion move cannot be computed exactly, so we are forced to approximate it.

Suppose that we are given a labeling f and we wish to approximate the optimal α - β multi-label expansion, where $|\alpha - \beta| = T$. The construction is similar to that in section 3. We identify label set $\{\alpha, \alpha + 1, \dots, \beta\}$ with set $\{0, 1, \dots, T\}$. One of the differences is that now all pixels participate in a move. First we build a graph exactly like in section 3, except the links between the source and p_0 are not set to infinite, for all pixels p . That is, unlike in section 3, p_0 is not identified with the source. We create an auxiliary pixel a_{pq} between each pair of neighboring pixels (p, q) . We connect p_0 to a_{pq} , q_0 to a_{pq} , and s to a_{pq} , as illustrated in Fig. 3 (a). The costs of the new links that we create for the expansion algorithm are as in Fig. 4. If the minimum cut severs the edge between s and p_0 , then p is assigned its old

link	weight
p_0 to a_{pq}	$V_{pq}(f_p, \alpha) + C/2$
q_0 to a_{pq}	$V_{pq}(\beta, f_q) + C/2$
s to a_{pq}	$V_{pq}(f_p, f_q) + C$
s to p_0	$D_p(f_p)$
s to q_0	$D_p(f_q)$

Fig. 4 Weights of the new links.

label in the move. Otherwise, the label assignment is exactly like in section 3.

For the construction in section 3, if we sever links e_i^p and e_j^q , then the cost of all the links e_{ij}^{pq} that have to be severed adds up to $C + V_{pq}(i, j)$.

This construction insures that if links between s and p_0 and between q_0 and q_1 are broken, then the cost of all edges severed corresponds exactly to $V_{pq}(f_p, \alpha)$ plus a constant, which is exactly what is needed. Similarly the correct thing happens if links between s and q_0 and between p_0 and p_1 are broken, and if the link between s and a_{pq} is broken. Unfortunately in other cases, as long as the new links in Fig. 4 are involved, the V_{pq} value can be either underestimated or overestimated. The minimum graph cut is not even guaranteed to reduce the energy from that of the current labeling f . Still in practice we found that many minimum cuts correspond to an assignment with a lower energy, and therefore many of such moves are useful in decreasing the energy. Notice that there are other methods, such as in [23, 12], where multi-label moves developed also do not guarantee that the energy is decreased. To insure that the energy never goes up, if f' is the assignment returned by our approximate multi-label expansion, we first test if $E(f') < E(f)$, where f is the current labeling. If yes, we accept f' as the new current labeling. If no, we reject it.

As with the multi-label swap, the range of labels involved in multi-label expansion can be extended by some t . The construction changes appropriately, similar to what is done when extending the range of multi-label swap moves, see section 3.

In practice, we found the following version of the multi-label expansion to work better. Let $\mathcal{U} = \{p \in \mathcal{P} | f_p \leq \beta\}$ and let $\mathcal{B} = \{p \in \mathcal{P} | f_p \geq \alpha\}$. We perform a multi-label expansion on pixels in set \mathcal{U} using the graph like in Fig. 3(a), and another multi-label expansion on pixels in set \mathcal{B} using the graph like in Fig. 3(b), with symmetrically modified weights in Fig. 4 for the second case. The weights also have to be corrected because there are pixels not participating in the move, so the “border” conditions resulting from such pixels have to be incorporated into edge weights e_i^p , just like in section 3. The improvement is probably due to the fact that more V_{pq} ’s are correctly represented by this split

graph construction. Another improvement is probably due to the fact that pixels on the border not participating in the move pull the energy in the right direction by having their V_{pq} terms correctly modeled through the edge weights e_i^p .

5 Multi-label Smooth Swap

We now present our last move for a truncated convex prior, the multi-label smooth swap. This move is closely related to the multi-label swap described in section 3. The idea of this new move is to involve a potentially larger group of pixels than that of a multi-label swap. In a multi-label swap, the pixels participating in a move have labels in a range limited by truncation, i.e. all the labels are between some α and β with $|\alpha - \beta| < T$. In a multi-label smooth swap, the domain of pixels participating in a move can be larger than that compared to the multi-label swap. That is the pixels participating in the move can have labels between some α and β with $|\alpha - \beta| > T$. The restriction is that the pixels participating in a multi-label smooth swap must form a “smooth” component in the current labeling f , that is the labels of any two neighbors cannot differ by more than T . First, we need a definition.

Definition Given a labeling f and a subset $\mathcal{T} \subset \mathcal{P}$, f is called *totally smooth with respect to (w.r.t) \mathcal{T}* , if for any $(p, q) \in \mathcal{N}$, whenever $\{p, q\} \subset \mathcal{T}$, then $|f_p - f_q| \leq T$, where T is the truncation constant in Eq. (3). \square

In words, if f is totally smooth with respect to \mathcal{T} , then the label difference for any two neighboring pixels contained in \mathcal{T} is not larger than the truncation constant.

Let f be the current labeling and $\mathcal{T} \subset \mathcal{P}$ be s.t. f is totally smooth w.r.t. \mathcal{T} . Let $\mathcal{L}(\mathcal{T}, f) = \{f_p | p \in \mathcal{T}\}$, that is $\mathcal{L}(\mathcal{T}, f)$ is the collection of labels of pixels in \mathcal{T} under labeling f .

Given a subset $\mathcal{T} \subset \mathcal{P}$ s.t. f is totally smooth w.r.t. \mathcal{T} , let $M_{smooth}(f, \mathcal{T}) = \{f' | f'_p \neq f_p \Rightarrow p \in \mathcal{T} \text{ and } f'_p \in \mathcal{L}(\mathcal{T}, f)\}$. $M_{smooth}(f, \mathcal{T})$ describes exactly the set of all multi-label smooth swaps. In words, a multi-label smooth swap takes a subset of pixels \mathcal{T} s.t. f is totally smooth with respect to \mathcal{T} , collects the labels of pixels in \mathcal{T} , and reassigns these labels among pixels in \mathcal{T} .

Just as it was possible to generalize the multi-label swap and expansion by extending the range of labels, it is possible to generalize the multi-label smooth swap. Let t be a constant for extending the range of labels $\mathcal{L}(\mathcal{T}, f)$. Let us define the extended range of labels as

$$\mathcal{L}'(\mathcal{T}, f, t) = \{l \in \mathcal{L} | \exists l' \in \mathcal{L}(\mathcal{T}, f) \text{ s.t. } |l - l'| \leq t\}$$

In words, to get $\mathcal{L}'(\mathcal{T}, f, t)$ we add to $\mathcal{L}(\mathcal{T}, f)$ those labels that are at distance no more than t from some

label already in $\mathcal{L}'(\mathcal{T}, f)$. Let the set of smooth swap moves augmented by t be denoted by $M_{smooth}(f, \mathcal{T}, t)$.

There are two questions that remain to be answered: how to choose the smooth subsets \mathcal{P}' and how to optimize with smooth swap moves. Let us first consider the question of optimization.

In general, it is not possible to find the optimal smooth swap move. However, we are able to find a good smooth swap, the one that improves the current labeling f . Let f be the current labeling, and let \mathcal{T} be s.t. f is totally smooth w.r.t. \mathcal{T} .

We use construction that is very similar to that in section 3. We construct a graph for pixels in \mathcal{T} . However, the label range is $\mathcal{L}'(\mathcal{T}, f, t)$, and we identify it with label set $\{0, 1, \dots, |\mathcal{L}'(\mathcal{T}, f, t)| - 1\}$. Otherwise, the graph construction is identical to that in section 3.

Just as in section 3.2, this construction finds a labeling under the energy $E^{-\mathcal{T}}(f)$, where $E^{-\mathcal{T}}(f)$ is the same as $E(f)$, except there is no truncation of V_{pq} terms for $p, q \in \mathcal{T}$. Let \hat{f} be a smooth swap found by this construction. Using a reasoning very similar to that in section 3.2, it is easy to show that $E(\hat{f}) \leq E(f)$. So even though we cannot find the optimal multi-label smooth swap, we can at least find a smooth swap does not increase the energy.

The question remains of how to find subsets \mathcal{T} s.t. current labeling f is totally smooth under \mathcal{T} . In general, there are many possibilities. We take the following approach. Given a current labeling f , we can partition it into a set of $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_d$, s.t. $\bigcap_i \mathcal{P}_i = \mathcal{P}$ and f is totally smooth w.r.t. each \mathcal{P}_i . This partition can be performed by computing connected components. This partition is not unique, however. To remove any bias due to visitation order, we compute connected components in random order. Then we compute multi-label smooth swaps for each \mathcal{P}_i . This is not the only way to proceed, but we found it to be effective. Computing all smooth swap moves for a partition $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_d$ constitutes one iteration of the algorithm. We perform iterations until convergence.

The advantage of the multi-label smooth swap move over the multi-label swap is that it converges faster. If we start from a good solution (typically we start from the results of the binary expansion algorithm), the number of smooth subsets in a partition of \mathcal{P} is small, so the number of moves is smaller compared to the multi-label swap. The disadvantage is that it gives energies that are slightly higher in practice, see section 7.

6 Double Expansion

In this section we give an example of a multi-label move that is useful for an energy with V_{pq} other than

truncated convex. We develop a *double* expansion move for the Potts model, see section 2.1. Recall that Potts $V_{pq}(f_p, f_q) = w_{pq} \cdot \min\{1, |f_p - f_q|\}$.

A double-expansion move is an analogue of the expansion move for two, not necessarily consecutive, labels. That is two labels are allowed to increase their territory at the same time. Formally, the double expansion moves are defined as follows. Given a labeling f and labels α, β , a move f' is called an $\alpha - \beta$ expansion if $f_p \neq f'_p \Rightarrow f'_p \in \{\alpha, \beta\}$. That is the set of pixels assigned to α and β can only expand from f to f' .

Just as with multi-label expansion, we cannot find the optimal $\alpha - \beta$ expansion, since optimization with Potts V_{pq} is NP-hard for three labels [9]. However, we can find an approximation that is at least as good as the optimal α -expansion *and* the optimal β -expansion.

Let $\alpha, \beta \in \mathcal{L}$ and a current labeling f^c be given. In this section, we choose to formulate the problem of finding a (possibly sub-optimal) $\alpha - \beta$ expansion as three-label (ternary) energy optimization.

Let h be a ternary labeling of pixels in \mathcal{P} , i.e. $h_p \in \{0, 1, 2\}$. We are going to use h to encode a double-expansion move. The *transformation* function $T_{\alpha\beta}(f^c, h)$ of a $\alpha - \beta$ expansion takes a labeling f^c and a ternary labeling h and returns the new labeling f^n which is induced by h and is an $\alpha - \beta$ expansion from f^c .

The transformation function $T_{\alpha\beta}()$ for an $\alpha - \beta$ expansion transforms the current label f_p^c of pixel p into a new label defined by:

$$f_p^n = T_{\alpha\beta}(f_p^c, h_p) = \begin{cases} \alpha & \text{if } h_p = 0 \\ f_p^c & \text{if } h_p = 1 \\ \beta & \text{if } h_p = 2 \end{cases} \quad (5)$$

To simplify the notation, we are going to use $T(h)$ instead of $T_{\alpha\beta}(f^c, h)$ and $T(h_p)$ instead of $T_{\alpha\beta}(f_p^c, h_p)$. There is no risk of confusion since α, β and f^c are fixed for a double-expansion move computation. We want the energy of h to be equal to the energy of the labeling f^n it induces i.e. $E'(h) = E(T(h))$. However, this results in non-submodular ternary energy in most cases. Instead we define:

$$E'(h) = \tilde{E}(T(f^c, h)) = \sum_{p \in \mathcal{P}} D_p(T(h_p)) + \sum_{(p,q) \in \mathcal{N}} \tilde{V}_{pq}(T(h_p), T(h_q)), \quad (6)$$

where \tilde{V}_{pq} is defined as:

$$\tilde{V}_{pq}(l, l') = \begin{cases} 2w_{pq} & \text{if } f_p^c = f_q^c \notin \{\alpha, \beta\} \\ & \text{and } l \neq l' \in \{\alpha, \beta\} \\ V_{pq}(l, l') & \text{otherwise} \end{cases} \quad (7)$$

It is trivial but tedious to check that the energy in Eq. (6) is submodular and therefore can be optimized exactly with the method in [35].

The term \tilde{V}_{pq} is almost always equal to V_{pq} , except in one case. If the current labels of p and q are the same and are not equal to either α or β , then the cost of assigning α to pixel p and β to pixel q (or vice versa) is overestimated by a factor of two. Therefore if an $\alpha - \beta$ expansion, is, in fact, a pure α -expansion, its cost is modeled correctly. Similarly for a pure β -expansion. This implies that the best $\alpha - \beta$ expansion found by optimizing the energy in Eq. (6) is not worse than the optimal α -expansion *and* the optimal β -expansion. In fact, if the α and β regions do not have a common boundary in the optimal $\alpha - \beta$ expansion, this optimal $\alpha - \beta$ expansion will be found by optimizing the energy in Eq. (6).

These new double expansion moves can avoid some local minima that expansion algorithm gets stuck in, as illustrated in an example in fig. 5.

If we need to perform expansion as quickly as possible, that is expanding on each label exactly once, then only half as many double expansions are needed, compared to the regular expansions. The double expansions are done on graphs that are 1.5 times larger, compared to the expansion graphs. Therefore one could hope that performing double expansions is faster. However in practice, with the min-cut/max-flow algorithm of [7], we found that this was not the case. That is performing $k/2$ double expansions took approximately the same time as performing k expansion. Potentially, however, a future min-cut/max-flow algorithm could work faster with double expansions.

7 Experimental Results

In this section we evaluate the performance of our multi-label moves on the problems of image restoration, inpainting, and stereo correspondence. For the min-cut computation, we use the max-flow algorithm in [7].

7.1 Image Restoration

In image restoration, we want to reconstruct the original image from the given noisy one. In this case, \mathcal{P} is the set of all image pixels, \mathcal{L} is the set of all gray levels, that is $\mathcal{L} = \{0, 1, \dots, 255\}$. We set $D_p(f_p) = (I_p - f_p)^2$, where I_p is the intensity of pixel p in the given noisy image. We used truncated quadratic $V_{pq}(f_p, f_q) = 8 \cdot \min\{(f_p - f_q)^2, 50\}$ and multi-label swap with $t = 3$.

Fig. 6(a) shows an artificial image we constructed, which consists of a circle and a square in front of the background, and the intensities inside the circle, square and background vary smoothly. Fig. 6(b) shows image

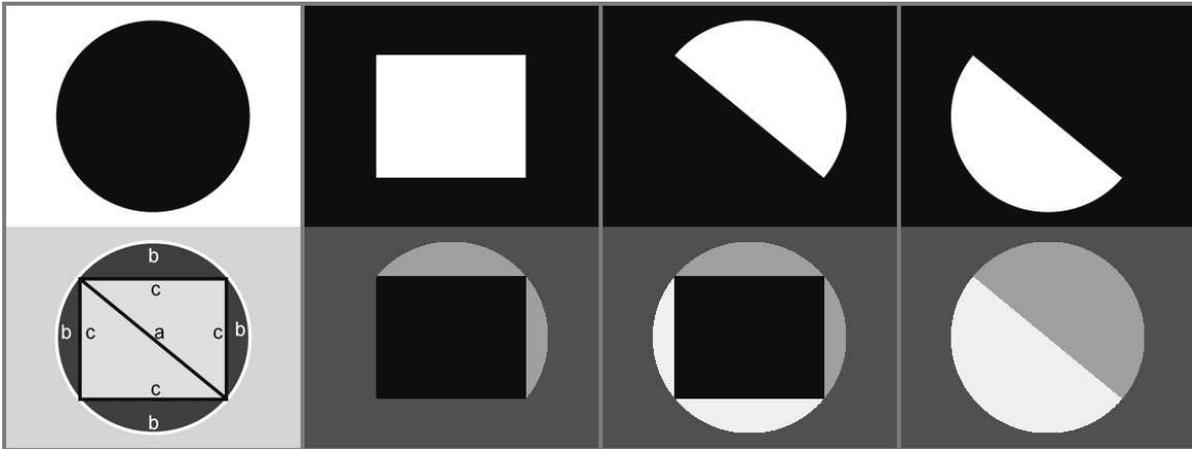


Fig. 5 Example where expansion algorithm gets stuck in a local minimum, and double-expansion finds the global optimum. Here $\mathcal{L} = \{0, 1, 2, 3\}$. The top row shows the data terms for these labels in the consecutive order. White means zero cost and black means an infinitely high data cost. Bottom row, left image, illustrates w_{pq} terms. All of them are infinite except those along the arcs and segments outlined in black and white. The accumulated cost of cutting along these arcs and segments is shown. For example, cutting along the top part of the rectangle costs c in total, and cutting along the diagonal of the rectangle costs a in total. Here $a > 2b$, and $2c = a - b$. The expansion algorithm is initialized with all pixels labeled as 0. Expansion proceeds on labels 1 and 2, the results of which are shown in the second picture, bottom row. Expansion on label 3 results in the solution shown in the third picture, bottom row, at which point the algorithm converges to a local minimum with cost $C_{sub} = 2a + 2a$. The optimum is shown in the last picture bottom row, and its cost is $C_{opt} = 4b + a < C_{sub}$. Double expansion on labels 2 and 3 finds the optimal labeling, starting from where expansion gets stuck, i.e. from the third picture, bottom row.

in (a) corrupted by zero mean Gaussian noise with variance 16. Figs. (c) and (d) show the result of the expansion and multi-label swap, respectively. We omit the results of the swap algorithm because they are visually similar to the results of the expansion algorithm. The energies of the ground truth, expansion algorithm and our multi-label swap are listed in the figure. Notice that our algorithm not only produces an answer with a significantly lower energy, but also gives the answer which looks smoother². The expansion algorithm tends to assign the same label to a subset of pixels that is too large, and the resulting answer looks piecewise-constant as opposed to piecewise smooth. This is because expansion moves seeks to change a large subset of pixels to the *same* label, as opposed to our algorithm which can change a subset of pixels to a smooth range of labels. In addition to producing a labeling which is more piecewise smooth, our answer is much closer to the ground truth. This is due not only to the fact that our energy is lower, but also to the fact that truncated quadratic energy is more appropriate for piecewise smooth restoration. The absolute average error (compared to the ground truth in (a)) for our answer

is 0.82, for the swap algorithm the error is 1.35, and for the expansion algorithm the error is 1.38. Our algorithm does take twice longer to run than the expansion algorithm on this example. Expansion algorithm takes about 40 seconds, and our algorithm takes about 80 seconds. The running time for the multi-label swap is only twice longer because for many moves the multi-label swap is run on graphs with small residual flow.

7.2 Image Inpainting

Image inpainting problem is similar to image restoration, except that some pixels have been “occluded” and therefore they have no preference for any label, that is for an occluded pixel p , $D_p(l) = 0$ for all $l \in \mathcal{L}$. For non-occluded pixels, we set $D_p(l) = (I_p - l)^2$, where I_p is the intensity of pixel p . We took a “Penguin” example from [36], available from D. Scharstein’s web site³. We used the same energy as in [36], namely $V_{pq}(f_p, f_q) = 25 \cdot \min\{(f_p - f_q)^2, 200\}$. The final energies are summarized in Figure 7, and the results are in Figure 8. Again, we set $t = 3$ for this experiment.

Multi-label swap obtains the best energy on this inpainting example. However the smooth range move achieved the energy very close to that of the multi-label swap, with a running time that is several times

² Depending on the printer resolution, the ground truth and our answer in the hard copy version of the paper may actually appear not piecewise smooth but piecewise constant. Zooming in on the electronic version, one will see images that do look piecewise smooth for our answer and the ground truth, and piecewise constant for the expansion algorithm.

³ <http://vision.middlebury.edu/MRF/>

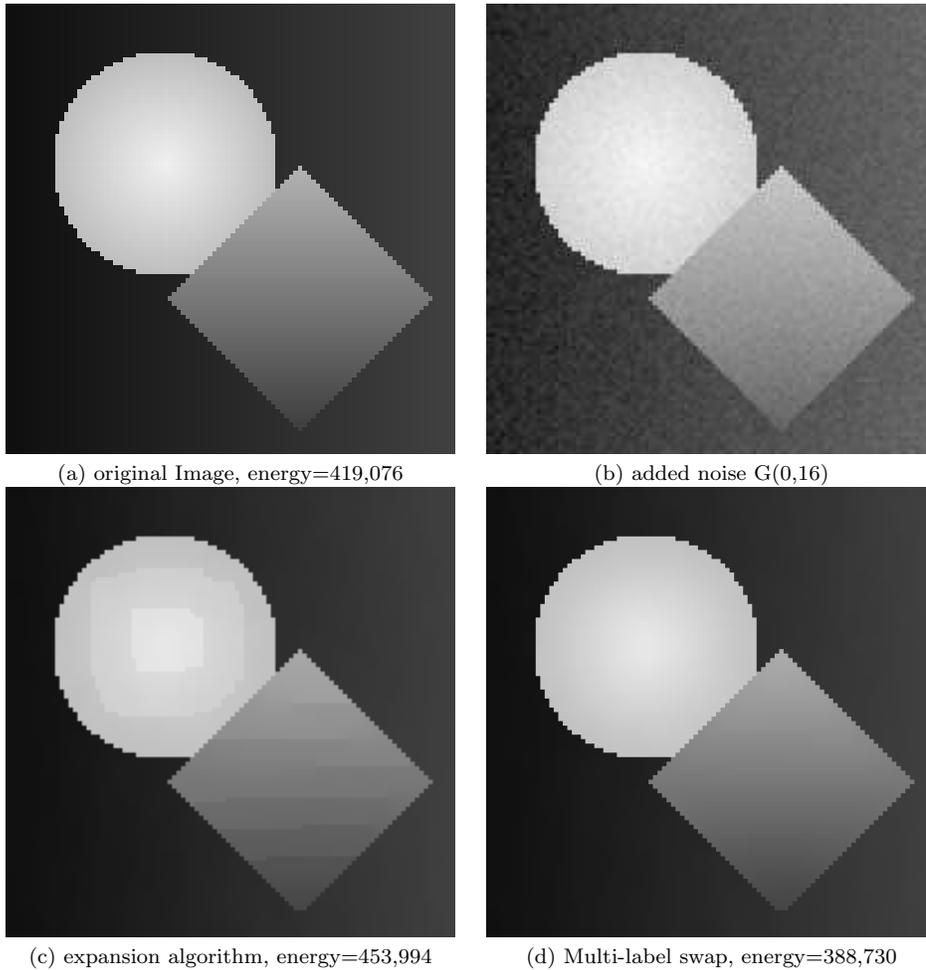


Fig. 6 Image Restoration Results.

Swap	17,076,141
Expansion	15,918,631
Multi-label Swap	15,382,317
Multi-label Smooth Swap	15,448,641
Multi-label Expansion	15,512,729

Fig. 7 Energies for “Penguin” inpainting. The minimum in each column is highlighted.

faster. All multi-label versions performed better than the regular swap and expansion algorithms.

TRW-S algorithm does give a slightly better energy than we get, namely the energy of 15,349,028, see [36] Graph cuts, however, perform better than TRW-S when longer range interactions are present. Szeliski et.al. [36] studied only the case when \mathcal{N} is the 4-connected grid. Kolmogorov and Rother [18] performed a comparison between graph cuts and TRW-S when longer range interactions are present in \mathcal{N} , and they concluded that graph cuts perform significantly better in terms of energy than TRW-S in this case.

7.3 Stereo Correspondence

In this section, we evaluate our multi-label swap, multi-label expansion, and multi-label smooth swap on the problem of stereo correspondence. We use the Middlebury database stereo images⁴. This database was constructed by D. Scharstein and R. Szeliski, and these images are the top benchmark in evaluating the performance of stereo algorithms [33, 34].

For stereo correspondence, \mathcal{P} is the set of all pixels in the left image, \mathcal{L} is the set of all possible stereo disparities. We take the disparity labels at sub-pixel precision, in quarter of a pixel steps. That is if $|f_p - f_q| = 1$, then the disparities of pixels p and q differ by 0.25 pixels. Let d_l stand for the actual disparity corresponding to the integer label l , for example label 2 corresponds to disparity 0.75. The data costs are $D_p(l) = |I_L(p) - [I_R(p - \bar{d}_l) \cdot (d_l - \underline{d}_l) + I_R(p - \underline{d}_l)(\bar{d}_l - d_l)]|$,

⁴ The images were obtained from www.middlebury.edu/stereo

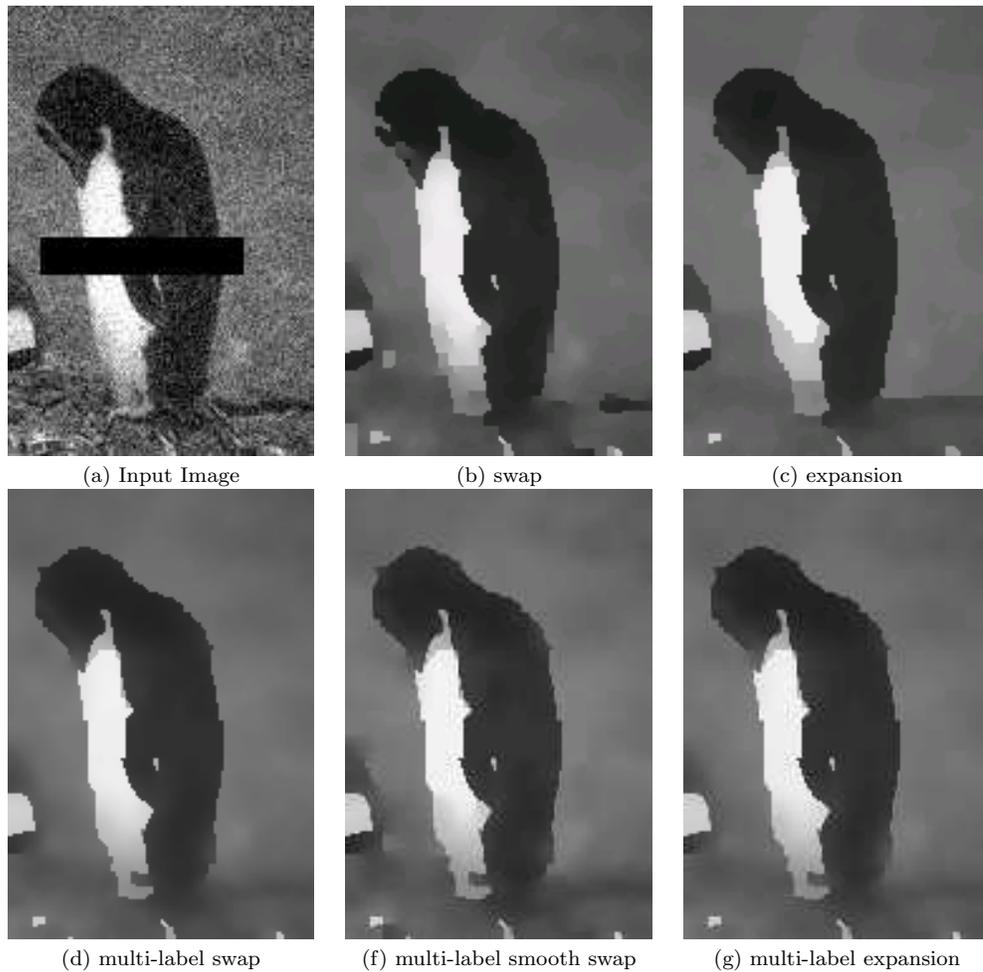


Fig. 8 Image Inpainting Results.

	Venus	Sawtooth	Teddy	Cones
Swap	7,871,677	9,742,107	16,376,181	21,330,284
Expansion	8,131,203	9,418,529	15,829,221	21,020,174
Multi-label Swap	7,188,393	9,371,745	15,421,437	20,490,753
Multi-label Smooth Swap	7,193,823	9,373,126	15,616,999	20,515,493
Multi-label Expansion	7,188,404	9,377,494	15,408,234	20,626,809

Fig. 9 Energies on Middlebury database. The minimum in each column is highlighted.

where \underline{x} stands for rounding down, \bar{x} stands for rounding up, and $p - x$ stands for the pixel that has the coordinates of pixel p shifted to the left by x .

Parameter t was set to 2 for all multi-label moves. We use the truncated quadratic $V_{pq}(f_p, f_q) = 100 \cdot \min\{(f_p - f_q)^2, 25\}$. Using spatially varying weights w_{pq} improves results of stereo correspondence, since it helps to align disparity discontinuities with the intensity discontinuities. We set all $w_{pq} = 10$, since the main purpose of our paper is to evaluate our multi-label moves, and not to come up with the best stereo algorithm.

Fig. 9 summarizes the energies we obtain with different algorithms. First let us consider the “binary” swap and expansion moves vs. the multi-label moves. The swap and expansion algorithms are clearly inferior when it comes to truncated convex priors. Even though the swap algorithm is guaranteed to find a best swap move and the expansion algorithm is not guaranteed to find the best move under the truncated quadratic model, expansion algorithm does perform better for all scenes except “Venus”. This is probably explained by the fact that expansion moves are more powerful than the swap moves. Even if we do not find the optimal

	Tsukuba	Venus	Teddy	Cones
Multi-label swap	6.7	3.25	15.1	6.79
Swap	7.47	4.04	15.8	7.64
Expansion	7.14	4.19	16.0	7.81

Fig. 10 Accuracy on the Middlebury database.

expansion, a good expansion may be better than the optimal swap.

Now let us discuss the multi-label moves. First of all, the running times for the multi-label swap move was on the order of minutes (from 5 to 10 minutes). The smooth range move achieved the energy very close to that of the multi-label swap, but its running time is about 2 or 3 times faster. The multi-label expansion move is almost always slightly worse than the multi-label swap, it is better only on the “Teddy sequence”. One would expect a better performance from the expansion move, but since we cannot find the optimal one, only an approximate one, these results are not entirely surprising. The running time for the expansion move is much worse than for other multi-label moves, since the graphs are much bigger. Multi-label expansion takes about 9-10 times longer than multi-label swap.

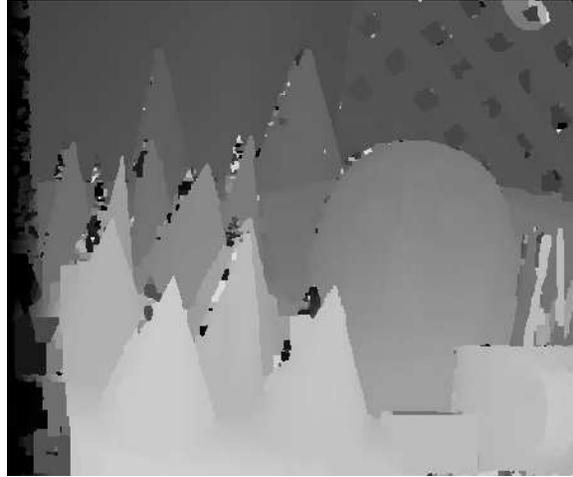
The accuracy of the labelings is summarized in Fig. 10. Each number in Fig. 10 gives the percentage of pixels away from ground truth by more than 0.5 pixels. *Tsukuba*, *Venus*, *Teddy*, *Cones* are the name of the four scenes in the Middlebury stereo database. Notice that our algorithm performs better not only in terms of energy, but also in terms of ground truth. The accuracy improvement is slight, but consistent across all the images in the database.

Fig. 11 shows the full results and Fig. 12 shows a zoom in on the detail in the *Cones* sequence. Notice that our algorithm gives results that look smoother over the surface of the cone than the expansion algorithm.⁵

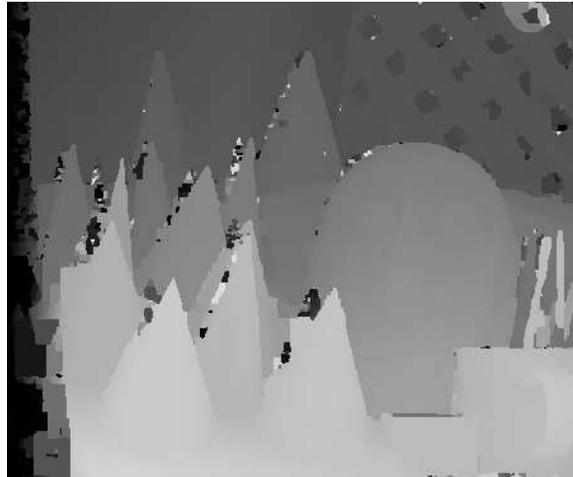
We should mention that the running times of our algorithms can be significantly improved using the ideas in [3]. They employ techniques such as good initialization, reducing the number of unknown variables by computing partially optimal solutions, and recycling flow. All of these are directly transferable to the implementation of our multi-label moves. Their speed ups are around a factor of 10 or 15.

8 Conclusions

The main contribution of this paper is the idea of the multi-label moves, as opposed to the commonly used

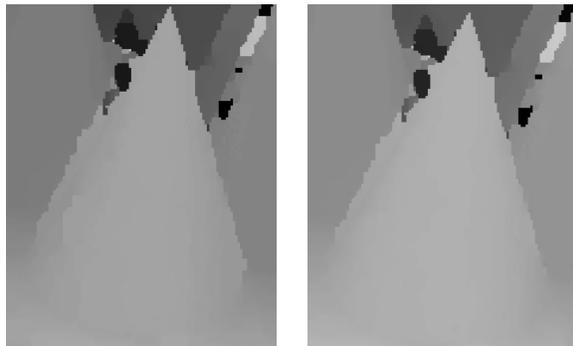


(a) expansion algorithm



(b) multi-label swap

Fig. 11 Results on “Cones”.



(a) expansion algorithm

(b) multi-label swap

Fig. 12 Zoom in on the detail.

⁵ To see the difference, it may be necessary to zoom in on these images in the electronic version of the paper.

swap and expansion moves that are binary in nature. We develop and compare several new multi-label moves for energies with a truncated convex prior, and a new multi-label move for the Potts model. We discuss the relative merits of the moves in terms of energy optimization and running times. Clearly, there are many more useful multi-label moves that can be developed for multi-label energies. An interesting question is whether it is possible to discover automatically new multi-label moves with good properties for a given energy, rather than develop them by hand.

The multi-label moves we develop in this paper for the truncated convex priors can be extended to the other non-submodular multi-label energies. The main idea is to restrict the set of labels for each pixel so that the restricted energy is submodular. The optimal move can be then found with the construction of [35] for multi-label submodular energies.

References

1. Agarwala, A., Agrawala, M., Cohen, M., Salesin, D., Szeliski, R.: Photographing long scenes with multi-viewpoint panoramas. *ACM SIGGRAPH* **25**(3), 853–861 (2006)
2. Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., Cohen, M.: Interactive digital photomontage. *ACM SIGGRAPH* pp. 294 – 302 (2004)
3. Alahari, K., Kohli, P., Torr, P.: Reduce, reuse, recycle: Efficiently solving multi-label mrfs. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8 (2008)
4. Besag, J.: On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B* **48**, 259–302 (1986)
5. Blake, A., Rother, C., Brown, M., Perez, P., Torr, P.: Interactive image segmentation using an adaptive gmmrf model. In: *ECCV*, pp. Vol I: 428–441 (2004)
6. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In: *ICCV*, pp. I: 105–112 (2001)
7. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI* **26**(9), 1124–1137 (2004)
8. Boykov, Y., Veksler, O., Zabih, R.: Markov random fields with efficient approximations. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 648–655 (1998)
9. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. *PAMI* **23**(11), 1222–1239 (2001)
10. Carr, P., Hartley, R.: Solving multilabel graph cut problems with multilabel swap. In: *Digital Image Computing: Techniques and Applications* (2009)
11. Geman, S., Geman, D.: Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *PAMI* **6**, 721–741 (1984)
12. Gould, S., Amat, F., Koller, D.: Alphabet soup: A framework for approximate energy minimization. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 903–910 (2009)
13. Hays, J., Efros, A.A.: Scene completion using millions of photographs. *ACM SIGGRAPH* **26**(3) (2007)
14. Ishikawa, H.: Exact optimization for markov random fields with convex priors. *PAMI* **25**(10), 1333–1336 (2003)
15. Ishikawa, H., Geiger, D.: Occlusions, discontinuities, and epipolar lines in stereo. In: *ECCV*, p. I:232 (1998)
16. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. *PAMI* **28**(10), 1568–1583 (2006)
17. Kolmogorov, V., Criminisi, A., Blake, A., Cross, G., Rother, C.: Probabilistic fusion of stereo with color and contrast for bilayer segmentation. *PAMI* **28**(9), 1480–1492 (2006)
18. Kolmogorov, V., Rother, C.: Comparison of energy minimization algorithms for highly connected graphs. In: *ECCV*, pp. II: 1–15 (2006)
19. Kolmogorov, V., Shioura, A.: New algorithms for convex cost tension problem with application to computer vision. *Discrete Optimization* **6**(4), 378–393 (2009)
20. Kolmogorov, V., Zabih, R.: Computing visual correspondence with occlusions via graph cuts. In: *ICCV*, pp. II: 508–515 (2001)
21. Kolmogorov, V., Zabih, R.: Multi-camera scene reconstruction via graph cuts. In: *ECCV*, p. III: 82 ff. (2002)
22. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? In: *ECCV* (2002)
23. Kumar, M.P., Torr, P.H.S.: Improved moves for truncated convex models. In: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (eds.) *Advances in Neural Information Processing Systems 21*, pp. 889–896 (2009)
24. Kumar, M.P., Veksler, O., Torr, P.: Improved moves for truncated convex models. *Journal of Machine Learning Research* **12**, 31–67 (2011)
25. Kwatra, V., Schdl, A., Essa, I., Turk, G., Bobick, A.: Graph-cut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics, SIGGRAPH 2003* **22**(3), 277–286 (2003)
26. Levin, A., Fergus, R., Durand, F., Freeman, W.T.: Image and depth from a conventional camera with a coded aperture. In: *ACM SIGGRAPH*, p. 70 (2007)
27. Liu, X., Veksler, O., Samarabandu, J.: Graph cut with ordering constraints on labels and its applications. In: *CVPR*, pp. 1–8 (2008)
28. Nguyen, M.H., Lalonde, J.F., Efros, A.A., de la Torre, F.: Image-based shaving. *Computer Graphics Forum Journal (Eurographics 2008)* **27**(2), 627–635 (2008)
29. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann (1988)
30. Rother, C., Kumar, S., Kolmogorov, V., Blake, A.: Digital tapestry. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. I: 589–596 (2005)
31. Rother, C., Minka, T., Blake, A., Kolmogorov, V.: Cosegmentation of image pairs by histogram matching: Incorporating a global constraint into mrfs. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. I: 993–1000 (2006)
32. Roy, S., Govindu, V.: Mrf solutions for probabilistic optical flow formulations. In: *ICPR00*, pp. Vol III: 1041–1047 (2000)
33. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV* **47**(1-3), 7–42 (2002)
34. Scharstein, D., Szeliski, R.: High-accuracy stereo depth maps using structured light. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. I: 195–202 (2003)
35. Schlesinger, D., Flach, B.: Transforming an arbitrary minsum problem into a binary one. Technical Report TUD-FI06-01, Dresden University of Technology (2006)
36. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A comparative study of energy minimization methods for markov

-
- random fields with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **30**(6), 1068–1080 (2008)
37. Torr, P.H.S.: In: personal communication (2008)
 38. Veksler, O.: Graph cut based optimization for mrfs with truncated convex priors. In: *CVPR*, pp. 1–8 (2007)
 39. Veksler, O.: Multi-label moves for mrfs with truncated convex priors. In: *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 1–13 (2009)
 40. Wills, J., Agarwal, S., Belongie, S.: What went where. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. I: 37–44 (2003)