# GRSA: Generalized Range Swap Algorithm for the Efficient Optimization of MRFs

Kangwei Liu, Junge Zhang, Peipei Yang and Kaiqi Huang
Center for Research on Intelligent Perception and Computing (CRIPAC)
National Laboratory of Pattern Recognition (NLPR)
Institute of Automation, Chinese Academy of Sciences (CASIA)
{kwliu,jgzhang,ppyang,kqhuang}@nlpr.ia.ac.cn

## Abstract

*Markov Random Field (MRF) is an important tool and has been widely used in many vision tasks. Thus, the optimization of MRFs is a problem of fundamental importance. Recently, Veskler and Kumar et. al propose the range move algorithms, which are one of the most successful solvers to this problem. However, two problems have limited the applicability of previous range move algorithms: 1) They are limited in the types of energies they can handle (i.e. only truncated convex functions); 2) These algorithms tend to be very slow compared to other graph-cut based algorithms (e.g. $\alpha$-expansion and $\alpha\beta$-swap). In this paper, we propose a generalized range swap algorithm (GRSA) for efficient optimization of MRFs. To address the first problem, we extend the GRSA to arbitrary semimetric energies by restricting the chosen labels in each move so that the energy is submodular on the chosen subset. Furthermore, to feasibly choose the labels satisfying the submodular condition, we provide a sufficient condition of the submodularity. For the second problem, unlike previous range move algorithms which execute the set of all possible range moves, we dynamically obtain the iterative moves by solving a set cover problem, which greatly reduces the number of moves during the optimization. Experiments show that the GRSA offers a great speedup over previous range swap algorithms, while it obtains competitive solutions.*

## 1. Introduction

Markov Random Field (MRF) is an important tool and has been widely used in many vision problems such as stereo reconstruction [25], image restoration [5], segmentation [3], image matching [21] and medical image analysis [4]. Solving these problems refers to the maximum a posteriori (MAP) estimation, or obtaining the label assignment that minimizes the MRFs energy. Therefore, optimizing the MRFs efficiently while ensuring good quality of solutions is a problem of fundamental importance.

In the last decades, many kinds of optimization approaches have been developed, such as *iterated conditional modes* (ICM) [2], *sequential belief propagation* (BPS) [25, 26], and *sequential tree-reweighted message passing* (TRW-S) [14]. Recently, graph-cut based algorithms [5, 9–11,16,17,28] have attracted significant attention due to their good optimality properties. Boykov *et al.* [5] propose the popular $\alpha$-*expansion* and $\alpha\beta$-*swap*, both of which optimize the MRFs by a series of iterative moves. In these two algorithms, each move refers to solving the *st*-mincut problem of a corresponding graph. Although $\alpha$-expansion and $\alpha\beta$-swap have been successfully applied in many vision tasks, there is a main limitation which has greatly prevented the algorithms from achieving a better solution. That is only a choice of two labels is provided for the vertices in every move [27]. Veksler [27, 28] effectively solves this problem by developing the so-called *range move algorithms*, which also decrease the energy by making a sequence of moves. Different from $\alpha$-expansion and $\alpha\beta$-swap, they allow every vertex to have a choice of more than two labels, and this brings about better solutions in practice. In [17, 18], Kumar *et al*. propose an improved range move algorithm and point that the range move algorithm obtains the same guarantee as the linear programming (LP) relaxation [6].

However, although the range move algorithms outperform $\alpha$-expansion and $\alpha\beta$-swap in many cases, there are two main problems which have limited their applicability in practice: (i) they are limited in the types of energy functions they can handle; (ii) the speed of the optimization is too slow, and as a result, the range move algorithms are not so popular as $\alpha$-expansion and $\alpha\beta$-swap. For the former limitation, previous range move algorithms [17, 18, 27, 28] are restricted to only truncated convex functions. However, there are many more general energy functions, which have been successfully used in different vision problems,

such as piecewise linear function [13] and Geman-McClure function [20]. Veksler in [28] points out that previous range move algorithms can be extended to more general energies by restricting the set of labels so that the energy on the restricted subset is submodular. Unfortunately, it is still a challenging problem to judge which labels satisfy the submodular condition, while given an arbitrary energy function. For the latter problem, previous range move algorithms execute the set of all possible range moves, which contain many repeated labels and lead to computational inefficiency. As a result, the range move algorithms run much slower than $\alpha$-expansion and $\alpha\beta$-swap. In theory, the larger the set of allowed moves, the better is the performance. However, in practice, we find that almost the same performance can be obtained with a much reduced set of moves. Therefore, we raise the following questions: (i) how to feasibly choose the labels satisfying the submodular condition, while we are given an arbitrary energy function? (ii) Whether we should execute all the possible moves, and if not, which iterative moves are required? (iii) How to schedule the moves to reduce the number of unnecessary moves, while ensuring comparable solutions compared to previous range move algorithms?

To solve the problems above, we propose a *generalized range swap algorithm* (GRSA). Firstly, we extend the range swap method to arbitrary semimetric[1] functions by restricting the chosen labels in every move to be a submodular set. More importantly, we provide a sufficient condition of the submodularity to feasibly choose labels satisfying the submodular condition. Secondly, we give the requirement of iterative moves in the GRSA, that every vertex should have the opportunity to swap its current label with any other labels. This requirement guarantees the GRSA obtains at least as good solutions as $\alpha\beta$-swap. Thirdly, to reduce the running time of the GRSA, we dynamically obtain the series of moves meeting the requirement by novelly solving a *set cover problem*, which attempts to get the expected moves covering all the pairs of labels. Experimental results show that the GRSA greatly reduces the running time of range swap moves (Figure 1), while obtains competitive solutions.

In our opinion, there are two main contributions in this paper: (i) we break the limitation of previous range move algorithms that they can only handle truncated convex functions. What is more, we propose a method to choose the submodular labels feasibly. This is a further research on the submodular condition to make it practically applicable in general functions. (ii) We novelly formulate the iterative optimization as solving a set cover problem, and in return, this formulation reduces a large number of unnecessary moves and offers a great speedup over previous algorithms without losing much in accuracy. Besides the main contributions, the GRSA can be regarded as a generalization of several graph cut based algorithms including $\alpha\beta$-swap, range swap algorithm and the method of Ishikawa [11]. This provides a new view towards the relationship among the graph-cut based algorithms. Meanwhile, the idea of the GRSA can also be naturally extended to the range expansion algorithms [17, 28], which can provide a theoretical guarantee.

[1]Here, "semimetric" means that the pairwise function should satisfy $\theta(\alpha,\beta) = 0 \Leftrightarrow \alpha = \beta$ and $\theta(\alpha,\beta) = \theta(\beta,\alpha) \geq 0$. If a function also satisfy $\theta(\alpha,\beta) \leq \theta(\alpha,\gamma) + \theta(\gamma,\beta)$, it is metric.

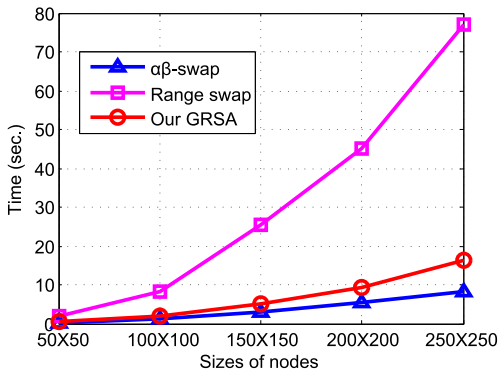



Figure 1. The running time of previous range swap algorithm and the GRSA on MRFs with different sizes. The results are average running time tested on 50 MRFs with truncated convex function.

## 2. Background and related work

### 2.1. The preliminaries of MRF

Many vision problems can be naturally formulated in terms of the *maximum a posteriori* (MAP) inference of an MRF. The MRF is defined as an undirected graph $\mathcal{G} = (\mathcal{P}, \mathcal{E})$, where $\mathcal{P}$ is the set of vertices, and $\mathcal{E}$ is the set of edges connecting neighboring vertices. Given an MRF, a *labeling* $f = \{f_p | p \in \mathcal{P}\}$ is the label assignment of all the vertices $p \in \mathcal{P}$. The probability of the labeling is given by the Gibbs distribution: $p(f|\mathbf{D}) = \exp(-E(f))/Z$, where $\mathbf{D}$ is the observed data and $Z$ is the normalization constant. The MAP estimation of the labeling can be solved by minimizing the Gibbs energy, which is typically given as follows:

$$E(f) = \sum_{p\in\mathcal{P}} \theta_p(f_p) + \sum_{(p,q)\in\mathcal{E}} \theta_{pq}(f_p, f_q) \qquad (1)$$

where $f_p, f_q \in \mathcal{L}$, and $\theta_p, \theta_{pq}$ denote the unary and pairwise potential respectively. The edge $(p,q) \in \mathcal{E}$ if and only if $p$, $q$ are neighboring vertices.

### 2.2. Graph cut based optimization

In recent years, graph cut has been a standard technique for the optimization of MRFs. The GRSA is also based on graph cut, and thus we give a brief review of the graph-cut based algorithms in this section.

The primary idea of the graph-cut based algorithms is to construct a special graph $\mathcal{G}_C$, where there is a one-to-one

correspondence between the cut of $\mathcal{G}_C$ and the labeling $f$. Meanwhile, the value of energy $E(f)$ is exactly equal to the cost of the cut of $\mathcal{G}_C$. Thus, the minimization of $E(f)$ can be obtained by solving the $st$-mincut problem. However, the weight of edges in the $st$-mincut graph is required to be non-negative, but not all the energy functions can be ensured that such a corresponding graph can be exactly constructed. As a result, most algorithms optimize $E(f)$ by a series of moves, each of which only considers a subset of the labels. According to the number of considered labels in each move, the graph-cut based algorithms can be divided into three categories:

**The $\alpha$-expansion and $\alpha\beta$-swap algorithms** Among the graph-cut based algorithms, $\alpha$-expansion and $\alpha\beta$-swap [5] may be the most popular methods. Due to their good optimality properties, both algorithms have been successfully applied in many vision tasks [12, 25], and there are a large number of improved move making algorithms based on them. For example, one class of the works [19, 20] attempt to extend their application to arbitrary energies using QPBO [15,22] to construct the graph, and other works [1, 8] attempt to improve the efficiency of the algorithms by reducing the label space which will be searched over in each move. However, all of these algorithms only provide every vertex a choice of two labels in each move.

**The range move algorithms** To obtain better solutions with the graph-cut techniques, Veksler [27, 28] and Kumar *et al*. [17, 18] develop the range move algorithms for truncated convex functions (*e.g.*, $\theta(f_p, f_q) = \min\{|f_p - f_q|, T\}$). The range move algorithms break the limitation of previous move making algorithms in which only two labels are considered in every move. In the range swap algorithms, every move considers a consecutive label subset $\mathcal{L}_{\alpha\beta} = \{\alpha, \alpha+1, \cdots, \beta\}$ by restricting $|\alpha - \beta| = T$. However, there is a problem in the iterative moves of previous range swap algorithms: they execute the set of all possible range moves, and this leads to computational inefficiency. In contrast, the GRSA does not suffer from the problem, since we give the requirement to guarantee the solution quality, and obtain the expected moves by solving a set cover problem, which greatly reduces the number of moves in the iterations.

**The case of global optimization** Although the optimization of MRFs is usually NP-hard, there are a few energy functions whose exact solution can be obtained by considering all the labels in one $st$-mincut. Ishikawa [11] develops an exact method for the optimization of multi-label MRFs. However, the pairwise potential $\theta_{pq}$ is restricted to be convex[2]. More generally, Schlesinger [23] points out that all the energies with *submodular* pairwise functions can be exactly minimized by the graph cut techniques. However, neither the convex nor submodular function is so popularly used in practice since they cannot preserve discontinuity on boundaries [5]. Our GRSA is based on the graph construction of [23]. However, rather than requiring the energy function to be convex or submodular on the whole label set, the GRSA only needs the chosen subsets in every move to satisfy the submodular condition.

[2] A function $g(\cdot)$ is convex if it satisfies $g(x+1) - 2g(x) + g(x-1) \geq 0$ for any integer $x$. Note that convex is a special case of submodular.

## 3. The algorithm

The GRSA starts from an initial labeling, and optimizes $E(f)$ by making a series of moves, each of which refers to a $st$-mincut problem. The GRSA converges when there is no move can be found to decrease $E(f)$. Firstly, we explain the notions of the generalized range swap move in Section 3.1. Every range swap move is executed on a subset of labels satisfying the submodular condition, but it is a hard problem to judge the labels satisfying the submodular condition. In Section 3.2, we novelly propose a sufficient condition of the submodularity, and show how to choose the labels flexibly with this condition. In Section 3.3, we focus on solving the problem of the iterative process in the GRSA. We first give the requirement of the iterative moves, and then obtain the series of moves by novelly solving a set cover problem.

### 3.1. The generalized range swap move

Let $\mathcal{L} = \{0, \cdots, n\}$ be the label set, and $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ $(l_i < l_{i+1})$ be a subset chosen from $\mathcal{L}$. Note that $\mathcal{L}_s$ is an arbitrary subset of $\mathcal{L}$, and is not necessary to be a consecutive sequence as previous algorithms [11, 17, 27]. Let $\mathcal{P}_l = \{p \in \mathcal{P} | f_p = l\}$ be the set of vertices assigned label $l$, and $\mathcal{P}_\mathcal{S} = \{p \in \mathcal{P} | f_p \in \mathcal{L}_s\}$ denote the set of vertices whose labels belong to $\mathcal{L}_s$. Then, a move from $f$ to $f'$ is called a *range swap move* (RSM) on $\mathcal{L}_s$, if $\mathcal{P}'_\mathcal{S} = \mathcal{P}_\mathcal{S}$, and $\mathcal{P}'_l = \mathcal{P}_l$ for any label $l \notin \mathcal{L}_s$. In other words, a RSM only allows the vertices belonging to $\mathcal{P}_\mathcal{S}$ to swap their labels in $\mathcal{L}_s$. Each range swap move refers to minimizing the following energy:

$$E_s(f) = \sum_{p \in \mathcal{P}_\mathcal{S}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_\mathcal{S} \neq \varnothing} \theta_{pq}(f_p, f_q). \quad (2)$$

Naturally, we have $E(f) = E_s(f) + E_{\hat{s}}(f)$, where

$$E_{\hat{s}}(f) = \sum_{p \notin \mathcal{P}_\mathcal{S}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_\mathcal{S} = \varnothing} \theta_{pq}(f_p, f_q).$$

With the minimization of $E_s(f)$, the move effectively decreases $E(f)$, since the RSM on $\mathcal{L}_s$ will not change the value of $E_{\hat{s}}(f)$. The RSM on $\mathcal{L}_s$ will lead to a better solution, if there are more labels considered in $\mathcal{L}_s$ (meanwhile, more vertices will be in $\mathcal{P}_\mathcal{S}$). However, $\mathcal{L}_s$ cannot be chosen arbitrarily. It should satisfy the following submodular condition [23] to guarantee the optimal RSM can be obtained:

**Definition 1.** *Given a pairwise potential $\theta(\alpha,\beta)$, we call $\mathcal{L}_s$ a **submodular set**, if it satisfies*

$$\theta(l_{i+1},l_j)-\theta(l_{i+1},l_{j+1})-\theta(l_i,l_j)+\theta(l_i,l_{j+1}) \geq 0 \qquad (3)$$

*for any pair of labels $l_i$, $l_j \in \mathcal{L}_s (1 \leq i, j < m)$.*

The optimal RSM on a submodular set $\mathcal{L}_s$ can be achieved with the graph construction of [11, 23].

### 3.2. Candidate submodular sets

Unfortunately, given an arbitrary energy, it is still a challenging problem to obtain the submodular sets with these inequalities (3). Therefore, we propose a sufficient condition of the submodularity, which allows the labels satisfying (3) to be chosen feasibly in practice. The sufficient condition is given by the following theorem:

**Theorem 1.** *Given a pairwise function $\theta(\alpha,\beta) = g(x)$ $(x = |\alpha - \beta|)$ on domain $X = [0, c]$, assume there is an interval[3] $X_s = [a, b]$ $(0 \leq a < b \leq c)$ satisfying: (i) $g(x)$ is locally convex on $[a, b]$, and (ii) $a \cdot (g(a+1) - g(a)) \geq g(a) - g(0)$. Then $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ is a submodular subset, if $|l_i - l_j| \in [a, b]$ for any pair of labels $l_i, l_j \in \mathcal{L}_s$.*

*Proof.* The proofs of the theorem and corollary presented in this paper are available in the supplementary material. □

In the follows, we focus on explaining the above theorem. For brevity, we call the interval $X_s$ satisfying the conditions in Theorem 1 a *candidate interval*. It is obviously that any convex interval $[0, b]$ is a candidate interval, since $a \cdot (g(a+1) - g(a)) = g(a) - g(0)$ when $a = 0$.

**General energy functions** For most functions successfully applied in vision problems, they are usually neither convex functions nor concave functions, such as truncated convex functions and piecewise functions. Although these functions are neither convex nor submodular on the whole domain, there are usually some convex candidate intervals. Theorem 1 implies that the submodular label subsets $\mathcal{L}_s$ can be obtained by restricting the difference between each pair of labels belonging to the same candidate interval. To explain this clearly, we use the example of piecewise linear function as shown in Figure 2. There are two candidate intervals: $[0, 3]$ and $[5, 12]$ in this pairwise function. As previous range move algorithms [27] [28], we can obtain the series of submodular sets $\{\alpha, \alpha+1, \alpha+2, \alpha+3\}$ where $0 \leq \alpha \leq n-3$ with the first candidate interval $[0, 3]$. Meanwhile, it can be seen that the subsets $\{\alpha, \alpha+2, \alpha+3\}$ or $\{\alpha, \alpha+1, \alpha+3\}$ are also submodular sets, since $|l_i - l_j| \in [0, 3]$ for any pair of labels. Furthermore, we can also obtain the submodular sets: $\{\alpha, \alpha+5, \alpha+10\}$, $\{\alpha, \alpha+6, \alpha+11\}$,$\cdots$ with the second candidate interval $[5, 12]$. More generally,

[3]Here, the interval $[a, b]$ denotes the set of integers $\{x | a \leq x \leq b\}$.

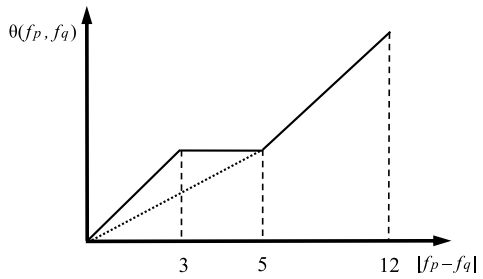



Figure 2. An example of piecewise linear function $g(x)$. The function is locally convex on the intervals $[0, 3]$ and $[5, 12]$. Here, we assume that the function satisfies that $\frac{g(6)-g(5)}{6-5} \geq \frac{g(5)-g(0)}{5-0}$. Thus, both $[0, 3]$ and $[5, 12]$ are candidate intervals, and the submodular sets can be obtained on these two candidate intervals.

we give the following corollary which is equivalent to Theorem 1:

**Corollary 1** (Thereom 1)**.** *Assuming the interval $[a, b]$ is a candidate interval, then $\{\alpha, \alpha + x_1, \alpha + x_1 + x_2, \cdots, \alpha + x_1 + \cdots + x_m\} \subseteq \mathcal{L}$ is a submodular set for any $\alpha \geq 0$, if $x_1, \cdots, x_m \in [a, b]$ and $x_1 + \cdots + x_m \leq b$.*

**Concave functions** If the pairwise function is a concave function (*e.g.*, $g(x) = \sqrt{x}$), there is no convex interval can be found. Meanwhile, it can be easily proved that there is no submodular set that contains more than two labels. In this case, there are only two labels that can be considered in every move. Therefore, the GRSA is equivalent to $\alpha\beta$ swap algorithm, when the energy function is concave.

**Convex functions** If the pairwise function is a convex function (*e.g.*, $g(x) = x$), the domain $[0, n]$, where $n$ is the number of labels, is a candidate interval. Therefore, the whole label set $\mathcal{L}$ is a submodular set. The optimal solution can be achieved in one move, and thus the GRSA is equivalent to the global method of Ishikawa [11] in this case.

Thus, we can obtain a series of candidate submodular sets with the above method while given an arbitrary semi-metric functions. The range swap move executed on any of these submodular sets can be exactly solved by computing the $st$-mincut problem.

### 3.3. The iterative optimization

Before proposing the process of iterative optimization in the GRSA, we firstly review the iterative process in $\alpha\beta$-swap and previous range swap algorithms, and then give the requirement of the moves to ensure the solution quality of the GRSA.

**$\alpha\beta$-swap and previous range swap** In $\alpha\beta$-swap, the requirement of the swap moves is that any pair of labels should be visited in each *cycle*[4] of iterations. This is

[4]In $\alpha\beta$-swap, we call these iteration moves considering all the pairs of labels once as a "cycle", and $\alpha\beta$-swap usually takes several cycles to converge [5].

necessary since it guarantees that every vertex has a chance to swap its current label $f_p$ with any other labels in $\mathcal{L}$.

In previous range swap algorithms, the moves are executed on all the subsets $\mathcal{L}_{\alpha\beta} = \{\alpha, \alpha+1, \cdots, \beta\}$, where $|\alpha - \beta| = T$, and $T$ is the truncated factor in a truncate convex function (*e.g.* $\theta = \min\{|f_p - f_q|, T\}$). However, there are many repeated labels in these moves. For example, in the two moves $\{\alpha, \alpha+1, \cdots, \beta\}$ and $\{\alpha+1, \alpha+2, \cdots, \beta+1\}$, most of the labels (*i.e.* $\{\alpha+1, \cdots, \beta\}$) are repeated. Thus, these moves cost much time, and cannot efficiently decrease $E(f)$. This is why previous range swap algorithms run much slower than $\alpha\beta$-swap.

**The GRSA** As the method described in Section 3.2, given an arbitrary energy function, we usually can obtain a large number of submodular sets, each of which corresponds to one possible range swap move. However, it is time-consuming and unnecessary to perform the exhaustive set of range moves. In practice, we find the requirement is sufficient to ensure the quality of solutions that *any pair of labels should be simultaneously considered once in one cycle of iterative moves*, *i.e.*, every vertex should have chance to swap its current label with any other labels. It is the same as the requirement of $\alpha\beta$-swap, and in theory, the range swap moves meeting this requirement can be guaranteed to obtain at least as good solutions as $\alpha\beta$-swap.

The following problem is how to choose a series of moves (*i.e.*, submodular sets) in one cycle, such that (i) these submodular sets cover all pairs of labels; (ii) the submodular set containing more labels should be chosen preferentially, and (iii) there should be as less repeated labels as possible in these submodular sets to reduce the running time. Naturally, this problem can be formulated as a *set cover problem* (SCP) [7].

In a SCP, we are usually given an *universe* $U$ of $m$ elements, a collection of set $\mathcal{S} = \{S_1, ..., S_k\}$ where $S_i \subseteq U$, and a cost function $c : \mathcal{S} \to \mathbf{R}$. A set cover is a subcollection of the sets in $\mathcal{S}$, that covers all the element in $U$. Then, the objective of the set cover problem is to find a cover $\mathcal{S}' \subseteq \mathcal{S}$ which minimizes the costs.

In the GRSA, $\mathcal{L} = \{0, \cdots, n\}$ is the set of labels, and let $\{\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_k\}$ be the series of submodular sets. We define $\mathcal{C}(\mathcal{L}) = \{(0,1), (0,2), \cdots, (n-1, n)\}$ to be the set containing all the pairs of labels in $\mathcal{L}$. In the set cover formulation, the universe $U = \mathcal{C}(\mathcal{L})$, and the collection of set $S_i = \mathcal{C}(\mathcal{L}_i)$. Therefore, the moves can be obtained by solving the following set cover problem:

$$\min \sum_{S_i \in \mathcal{S}'} c(S_i) \qquad \text{s.t.} \bigcup_{S_i \in \mathcal{S}'} S_i = U \ . \tag{4}$$

Although the SCP is an NP hard problem, fortunately, the greedy algorithm [24] can successfully achieve an approximate solution in polynomial time. Algorithm 1 describes the iterative process of the GRSA, where the moves are chosen by dynamically solving the SCP with the greedy algorithm. In the algorithm, step 6-11 is a cycle of iterative moves, and each move in step 7 chooses the set:

$$S_i = \underset{S_i \in \mathcal{S}}{\operatorname{argmin}} \frac{c(S_i)}{|S_c \cup S_i| - |S_c|} \tag{5}$$

where $S_c$ denotes the set of elements which have been chosen in the greedy algorithm. $|S_c \cup S_i| - |S_c|$ denotes the number of increased elements when set $S_i$ is added into $S_c$. This means the greedy algorithm always chooses the set which minimizes the cost of per increased element.

Using the SCP, we can design different iterative processes by assigning different cost to the submodular sets. In this paper, we set[5] : $c(S_i) = 1$, if $S_i \cap S_c = \varnothing$; $\infty$, otherwise.

This means that every iterative move in step 7 chooses the set: (i) there is no repeated element with the collections which have been chosen in previous moves; (ii) it contains maximal number of labels among the sets satisfying condition (i). For an example of the truncated convex function whose truncated factor is $T$, the GRSA is possible to execute the moves on the following series of submodular sets: $\{0, \cdots, T\}$, $\{T, \cdots, 2T\}$,$\cdots$,$\{mT, \cdots, n\}$ and all the pair of labels which are missing.

**Algorithm 1** The Generalized Range Swap Algorithm

**Input:**

1: The label set $\mathcal{L} = \{0, \cdots, n\}$, and the pairwise function $\theta(\alpha, \beta) = g(x)$ $(x = |\alpha - \beta|)$.

**Initialization:**

2: Find the series of submodular sets $\mathcal{L}_i$ with the total enumeration according to Corollary 1.

3: Get the collection of sets $\mathcal{S} = \{S_1, ..., S_k\}$ where $S_i = \mathcal{C}(\mathcal{L}_i)$, and initialize $U = \mathcal{C}(\mathcal{L})$, $S_c \leftarrow \varnothing$.

4: Initialize the labeling $f$.

**Iteration:**

5: **repeat**

6: **while** $S_c \neq U$ **do**

7: Choose $S_i \in \mathcal{S}$, which minimizes the cost per element $\frac{c(S_i)}{|S_c \cup S_i| - |S_c|}$.

8: Set $S_c := S_i \cup S_c$ and $\mathcal{L}_s := \mathcal{L}_i$ where $S_i = \mathcal{C}(\mathcal{L}_i)$.

9: Get the new labeling $f' = \operatorname{argmin} E(f)$ within the range swap move on $\mathcal{L}_s$.

10: If $E(f') < E(f)$, set $f := f'$.

11: **end while**

12: **until** No moves can be found to decrease $E(f)$.

**Output:**

13: Return the labeling $f$.

[5] We just set the cost function described in the paper for simplicity, but a better iterative process may be developed with other design of cost (*e.g.* set $c(S_i) = 1 + |S_i|$), since a small number of repeated labels may lead to a better solution without increase of running time. However, the experiments show that the GRSA obtain promising results without a fine design of the SCP.

## 4. Experiments

In this section, we evaluate our GRSA on both synthetic data and real vision applications of image restoration and stereo matching. The performance is compared with some state-of-the-art methods, including $\alpha$-expansion, $\alpha\beta$-swap, BPS [26], TRWS [14], as well as previous range swap and range swap + t algorithm[6]. To ensure the fairness of comparisons, the GRSA and all the other algorithms are initialized with the same labeling (every vertex is initialized label 0). The iterative moves of the GRSA are performed as Algorithm 3.3. In the experiment, we use the codes provided by Kumar *et.al* [18] for previous range swap and range swap + t algorithms. On the experiments of image restoration, we perform "*all swap moves*" plus "*all range swap moves*" in the iterative moves of previous range swap and range swap + t algorithms. This is because that some important moves are missing in previous range swap moves, *i.e.*, the pairs of labels $\{\alpha, \beta\}$ are not considered for any pair of labels $\alpha$, $\beta$ satisfying $|\alpha-\beta| > T$[7]. For a vertex $p$, whose current label is $\alpha$ and real label is $\beta$ ($|\alpha-\beta| > T$), unfortunately there is no move from $\alpha$ to $\beta$. This may lead to a bad solution especially when the label space is large (*e.g.* $|\mathcal{L}| = 256$ in image restoration).

### 4.1. Data and experimental setting

**Synthetic data** The computation time of previous range swap and the GRSA is affected by multiple factors, such as the parameters of energy function or the size of MRF. To give a comparison of our GRSA and previous range swap algorithm under various cases, we evaluate them on the MRFs whose parameters are generated randomly. Following [17,18], the data term $\theta_p(f_p)$ are sampled uniformly from the interval $[0, 10]$. For the pairwise term, we use the truncated convex function $\theta_{pq} = 3\min\{(f_p - f_q)^2, T^2\}$. In the experiments, we firstly fix $T = 5$, and test the running time of the algorithms with the increase of MRFs' size. Then, we fix the size of the MRFs as $100\times100$, and evaluate the influence of the truncated factor $T$. In each group, we use 50 random fields to avoid errors, and compare the average running time and average energy.

**Image restoration** In image restoration, the given input images are corrupted with noise and the objective is to reconstruct the original images by removing the noise. We use two popular images from the Corel database: *penguin* and *house*. In the experiments, we set $\mathcal{L} = \{0, 1, \cdots, 255\}$, and test the GRSA on two pairwise functions: 1) the truncated convex function $\theta_{pq}(f_p, f_q) = 25\min\{(f_p - f_q)^2, 200\}$, whose parameters are set as [12, 28]; and 2) the piecewise linear function:

$$\theta_{pq}(f_p, f_q) = \begin{cases} 25|f_p - f_q|, & \text{if } |f_p - f_q| \le 15; \\ 25 \times 15, & \text{if } 15 < |f_p - f_q| < 45; \\ 25(|f_p - f_q| - 35), & \text{if } 45 \le |f_p - f_q|. \end{cases}$$

**Stereo matching** In stereo correspondence, the goal is to get the correspondence of pixels in the left and right images. In the experiment, we use two popular image pairs *tsukuba* and *venus* from the Middlebury Stereo Database. The size of label space is equal to the number of disparity on the image pairs: *tsukuba* (16), *venus* (20). We use two kinds of energy functions: the truncated function $\theta_{pq}(f_p, f_q) = 30\min\{(f_p - f_q)^2, T^2\}$ and the pairwise linear function

$$\theta_{pq}(f_p, f_q) = \begin{cases} 30|f_p - f_q|, & \text{if } |f_p - f_q| < T; \\ |f_p - f_q| + 30T, & \text{otherwise.} \end{cases}$$

We set $T = 8$ for *tsukuba*, and $T = 10$ for *venus*. We also evaluate the accuracy of results on Middlebury On-line Evaluation[8] with two different *error thresholds* (ET).

### 4.2. Performance comparison and analysis

**Comparison with traditional range swap algorithms**

*Efficiency*: To quantify the efficiency of the GRSA, we compare the running time of the GRSA and previous range swap algorithm on truncated convex functions. To avoid the influence of implementation details, we use the code[9] provided by Kumar *et al.* [17] for previous range swap algorithm. Figure 3 shows the running time of the algorithms on synthetic data with different influential factors. We can observe that the running time of previous range swap greatly increases with the MRFs' size or truncated factor. In contrast, the running time of the GRSA increases much slower than previous range swap, and it reduces the running time of previous range swap by more than $80\%$ in many cases (*e.g.* $T = 8$, size $100 \times 100$). In real problems, the GRSA also runs much faster. As shown in Figure 4 (b) and (e) and Table 1, we can see that the GRSA runs at least 3-6 times faster than previous range swap, and 5-14 times faster than range swap + t algorithm in image restoration. The GRSA shows high efficiency because the set cover formulation greatly reduces the number of unnecessary moves, and thus the GRSA takes much less time in each cycle of iterations. As shown in Figure 3 (d), Figure 4 (d) and (g), we can see that the GRSA takes similar number of cycles to converge, but runs several times faster in each cycle compared to previous range swap algorithms.

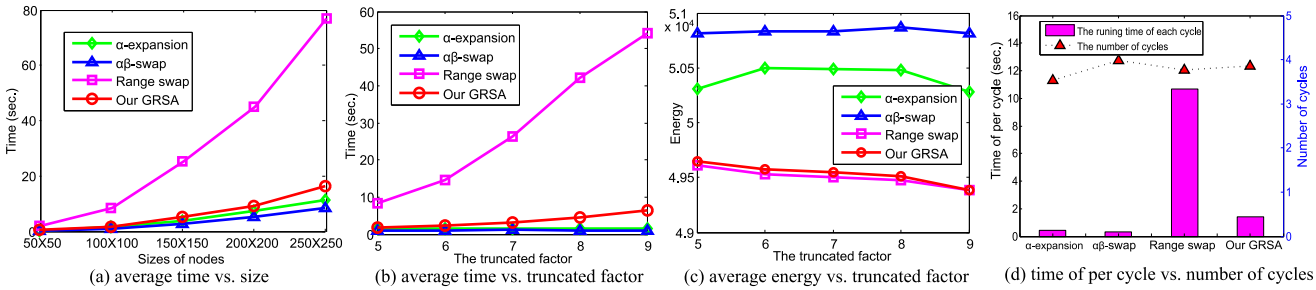
*Performance*: We quantify the performance of the GRSA and previous range swap algorithms on image restoration and stereo matching. In stereo matching, the range swap + t algorithm obtains the best results among the compared

[6] In every move, the range swap + t algorithm [27] considers the label set $\{\alpha - t, \alpha - t + 1, \cdots, \beta + t\}$, where $|\beta - \alpha| = T$ to obtain a better solution. We set $t = 2$ in all the experiments.

[7] $T$ is the truncated factor in a truncate convex function

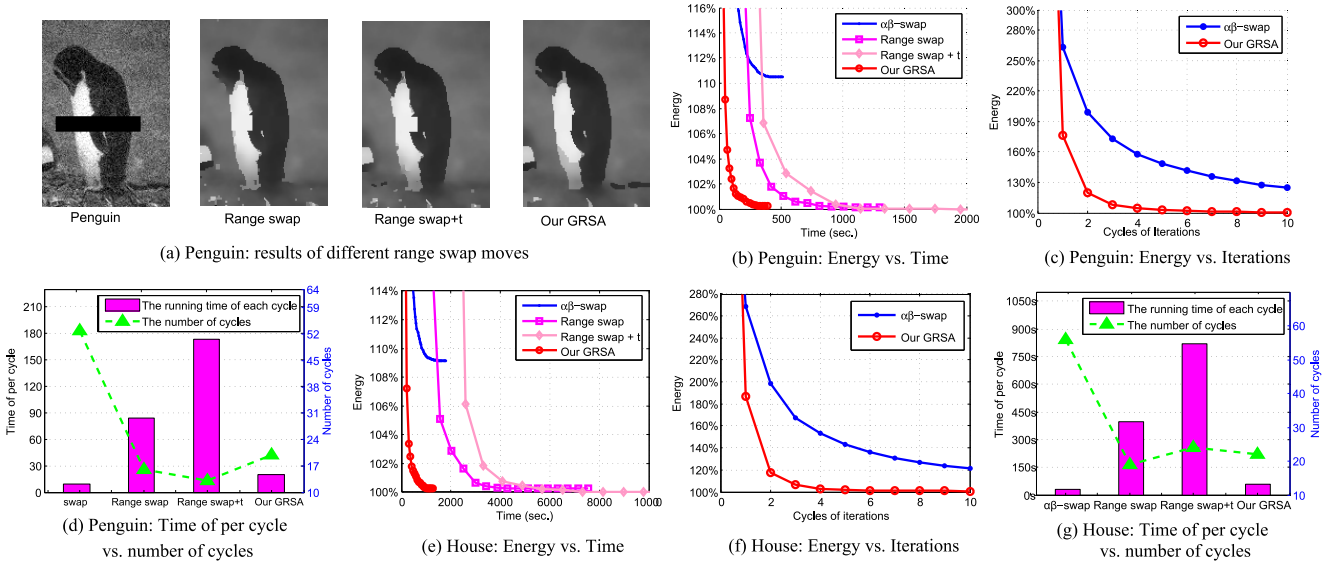
[8] http://vision.middlebury.edu/stereo/eval/

[9] http://cvn.ecp.fr/personnel/pawan/research/truncated-moves.html




(a) average time vs. size (b) average time vs. truncated factor (c) average energy vs. truncated factor (d) time of per cycle vs. number of cycles

Figure 3. The results obtained on synthetic data. Each group of result is evaluated on 50 MRFs with truncated convex functions. (a) and (b) show the algorithms' running time with different sizes of MRFs or different truncated factors $T$. (c) shows the energy with different truncated factors $T$ (size $100\times100$). (d) shows the number of cycles and the running time of each cycle when $T=8$, size is $100\times100$.




(a) Penguin: results of different range swap moves (b) Penguin: Energy vs. Time (c) Penguin: Energy vs. Iterations

(d) Penguin: Time of per cycle vs. number of cycles (e) House: Energy vs. Time (f) House: Energy vs. Iterations (g) House: Time of per cycle vs. number of cycles

Figure 4. The results obtained on image restoration. (a) shows the results obtained by range swap [17], range swap + t and the GRSA on *penguin*. (b) and (e) show the energy obtained by different algorithms with running time on *penguin* and *house* respectively. (c) and (f) show the obtained energy with the number of cycles on *penguin* and *house* respectively. The value of energy is plotted in percentage points, where 100% is set to be the lowest value achieved by any algorithm in (b), (c), (e) and (f). (d) and (g) show the running time taken by every cycle of iterations in different algorithms, and the number of cycles that each algorithm takes to converge.

algorithms (Table 2). Meanwhile, previous range swap and the GRSA get promising and similar results. In image restoration, the range swap + t algorithm ("*all swap moves*" plus "*all range swap + t moves*") obtains the best results. However, we can see that the GRSA obtains very similar solutions compared to both the range swap + t and range swap algorithms as shown in Figure 4.

Therefore, the GRSA offers a great speedup over previous range swap algorithms, while achieves competitive solutions without losing much in accuracy on both synthetic data and real problems.

**Comparison with $\alpha\beta$-swap algorithms** We compare the GRSA with $\alpha\beta$-swap on both synthetic data and real problems. As expected, the GRSA outperforms $\alpha\beta$-swap on both synthetic data and the real problems of image restoration and stereo matching, because the GRSA considers more labels in every move. The GRSA achieves not only

| Algorithm | Energy (penguin) | Time (penguin) | Energy (house) | Time (house) |
|---|---|---|---|---|
| $\alpha\beta$-swap | 17367822 | 512.4 | 45114530 | 1789.8 |
| Range swap | 15740426 | 1448.7 | 41438765 | 7536.5 |
| Range swap+t | **15716390** | 2258.0 | **41332881** | 15753.3 |
| Our GRSA | 15758765 | **392.0** | 41452670 | **1278.7** |
| $\alpha\beta$-swap | 9395637 | 221.0 | 26648428 | 453.5 |
| Our GRSA | **8675869** | **149.3** | **23827954** | **371.1** |

Table 1. The results on image restoration. The optimization algorithms are evaluated on two images. The 2-3 columns show the energy and running time on image *penguin*, respectively. The 4-5 columns show the results on image *house*. The 2-7 rows show the results on the truncated convex function, and 8-9 rows show the results on the piecewise linear function.

(a) Tsukuba
error1=13.9 # error2 =6.22 (b) α-expansion
error1=16.1 # error2=10.3 (c) αβ-swap
error1=13.4 # error2=4.50 (d) TRWS
error1=12.9 # error2=4.48 (e) GRSA

Figure 5. The stereo matching results obtained by $\alpha$-expansion, $\alpha\beta$-swap, TRW-S and the GRSA with the truncated convex function. The errors (%) are tested on Middlebury Stereo Evaluation. Error1 is tested with the error threshold $ET = 1$, and error2 is tested with $ET = 2$.

a lower energy but also a better accuracy compared to $\alpha\beta$-swap. For example, as shown in Figure 5, the error of the results obtained by $\alpha\beta$-swap on tsukuba is 10.3 (ET $= 2$), while the error of the GRSA is 4.5.

In the experiments, we find that the GRSA takes similar time to converge compared to $\alpha\beta$-swap in most cases, *e.g.* in synthetic data and stereo matching. However, in image restoration, the GRSA surprisingly converges faster than $\alpha\beta$-swap with both the truncated convex and piecewise linear functions. This is because that although the GRSA takes more time in each cycles of iterations, it needs much fewer cycles to converge as shown in Figure 4) (d) and (g). For example, the GRSA takes 21 cycles while $\alpha\beta$-swap takes 55 cycles to converge in image *penguin*. In Figure 4 (c) and (f), we can also see that each cycle of the GRSA decreases $E(f)$ much more than $\alpha\beta$-swap does.

**Comparison with other state-of-the-art algorithms** In stereo matching, we also compare the performance of the GRSA with $\alpha$-expansion, BPS and TRW-S, all of which have been successfully applied in many applications. Table 2 shows the performance obtained with both truncated convex functions and piecewise linear functions. We can see that the range swap + t algorithm obtains the best results on truncated convex functions. The GRSA, range swap algorithm and TRW-S also get competitive results. With the piecewise linear functions, the GRSA obtains the best results compared to $\alpha$-expansion, $\alpha\beta$-swap, BPS and TRW-S. Besides the energy, the GRSA also gets competitive results in terms of accuracy. Figure 5 shows the solutions obtain by $\alpha$-expansion, $\alpha\beta$-swap, TRW-S and the GRSA. It can be seen that the GRSA gives the best results among these algorithms.

| Algorithm | Energy 1 (Tsukuba) | Energy 2 (Tsukuba) | Energy 1 (Venus) | Energy 2 (Venus) |
|---|---|---|---|---|
| $\alpha$-expansion | 1482258 | 1369101 | 2712361 | 2611608 |
| $\alpha\beta$-swap | 2189043 | 1371721 | 2922213 | 2952128 |
| Range swap | 1449334 | - | 2629676 | - |
| Range swap+t | **1431401** | - | **2629664** | - |
| TRW-S | 1449548 | 1371572 | 2630498 | 2603053 |
| BPS | 1686577 | 1460496 | 3299917 | 2633931 |
| Our GRSA | 1445585 | **1367681** | 2630819 | **2600259** |

Table 2. The results on stereo matching. "Energy 1" denotes the energies obtained on the truncated convex function, while "Energy 2" denotes the energies obtained on the piecewise linear function.

## 5. Conclusions and discussions

In this paper, we have presented a generalized range swap algorithm for the approximate optimization of MRFs with any semimetric energy functions. The algorithm considers a submodular set in every move, and to choose the submodular labels feasibly, we have proposed a sufficient condition of the submodularity. In the iterative optimization, we dynamically obtain the range swap moves by solving a set cover problem (SCP). The SCP gets the moves covering all the pairs of labels in $\mathcal{L}$, while reduces many unnecessary moves. The experiments show that the GRSA runs several times faster than previous range swap algorithms, while it achieves competitive solutions. Moreover, the GRSA can be regarded as a generalization of $\alpha\beta$-swap, previous range swap algorithms and the global optimization of Ishikawa.

In future work, the GRSA still can be improved in some aspects. First, we designed a simple cost function in the set cover problem, and we believe that more powerful iterative methods can be developed with the SCP by designing a better cost function. Second, although Theorem 1 provides a practical method to obtain the submodular sets, however it is not a necessary condition. It is an interesting problem whether we can find a better way to allow more labels to be feasibly taken in submodular sets. Third, we observe that there are still many moves that do not efficiently lower $E(f)$ in the iterations of the GRSA. Therefore, it will be an interesting direction to find the move which leads to the biggest decrease of $E(f)$ in each iteration as [1], and this may greatly reduce the running time of the GRSA.

**Acknowledgement** This work is funded by the National Basic Research Program of China (Grant No. 2012CB316302), National Natural Science Foundation of China (Grant No. 61322209, Grant No. 61175007 and Grant No. 61403387). We thank Olga Veksler for her great help to this work, and we thank Pushmeet Kohli for his valuable comments.

## References


[1] D. Batra and P. Kohli. Making the right moves: Guiding alpha-expansion using local primal-dual gaps. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[2] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, 48(3), 1986.

[3] Y. Boykov and Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in nd images. In *IEEE International Conference on Computer Vision*, 2001.

[4] Y. Boykov and M.-P. Jolly. Interactive organ segmentation using graph cuts. In *Medical Image Computing and Computer-Assisted Intervention*, pages 276–286, 2000.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11), 2001.

[6] C. Chekuri, S. Khanna, J. Naor, and L. Zosin. A linear programming formulation and approximation algorithms for the metric labeling problem. *SIAM Journal on Discrete Mathematics*, 18, 2004.

[7] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45:634–652, 1998.

[8] S. Gould, F. Amat, and D. Koller. Alphabet soup: A framework for approximate energy minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[9] D. Greig, B. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, pages 271–279, 1989.

[10] I. Gridchyn and V. Kolmogorov. Potts model, parametric maxflow and k-submodular functions. In *IEEE International Conference on Computer Vision*, 2013.

[11] H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10), 2003.

[12] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, J. Lellmann, N. Komodakis, and C. Rother. A comparative study of modern inference techniques for discrete energy minimization problem. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[13] P. Kohli, A. Osokin, and S. Jegelka. A principled deep random field model for image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[14] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10), 2006.

[15] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts-a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7), 2007.

[16] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2), 2004.

[17] M. P. Kumar and P. H. Torr. Improved moves for truncated convex models. In *Advances in Neural Information Processing Systems*, 2008.

[18] M. P. Kumar, O. Veksler, and P. H. Torr. Improved moves for truncated convex models. *The Journal of Machine Learning Research*, 12, 2011.

[19] V. Lempitsky, C. Rother, and A. Blake. Logcut-efficient graph cut optimization for markov random fields. In *IEEE International Conference on Computer Vision*, 2007.

[20] V. Lempitsky, C. Rother, S. Roth, and A. Blake. Fusion moves for markov random field optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8), 2010.

[21] K. Liu, J. Zhang, K. Huang, and T. Tan. Deformable object matching via deformation decomposition based 2d label mrf. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[22] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary mrfs via extended roof duality. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[23] D. Schlesinger and B. Flach. *Transforming an arbitrary minsum problem into a binary one*. TU, Fak. Informatik, 2006.

[24] P. Slavłk. A tight analysis of the greedy algorithm for set cover. *Journal of Algorithms*, 25, 1997.

[25] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for mrfs with smoothness-based priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6), 2008.

[26] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo using identical mrf parameters. In *IEEE International Conference on Computer Vision*, 2003.

[27] O. Veksler. Graph cut based optimization for mrfs with truncated convex priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[28] O. Veksler. Multi-label moves for mrfs with truncated convex priors. *International Journal of Computer Vision*, 98(1), 2012.