Fast Approximate Energy Minimization via Graph Cuts

Yuri Boykov, Olga Veksler and Ramin Zabih

Abstract

Many tasks in computer vision involve assigning a label (such as disparity) to every pixel. A common constraint is that the labels should vary smoothly almost everywhere. In many applications a labeling should also preserve any sharp discontinuities that may exist, e.g., at object boundaries. These tasks can be naturally stated in terms of energy minimization. In this paper we consider a wide class of energies allowing various smoothness constraints. Our major restriction is that the energy function’s smoothness term must only involve pairs of pixels.

We show that global minimization of our energy function is NP-hard even in the simplest interesting case. Our focus is therefore on efficient approximation algorithms. We compute a labeling that is locally optimal even when very large moves are allowed. For example, our $\alpha$-expansion moves allow any subset of pixels to switch to any given label $\alpha$ simultaneously. In comparison, many standard algorithms (including simulated annealing) use moves where only one pixel changes its label at a time. We show that the labeling we compute, which is locally optimal with respect to $\alpha$-expansion moves, is within a known factor of the global minimum.

We present algorithms based on graph cuts that efficiently find a local minimum with respect to two types of large moves. Both our algorithms allow important cases of discontinuity preserving energy functions. We experimentally demonstrate the effectiveness of our approach for stereo and motion. On real data with ground truth from the University of Tsukuba, we achieve 98% accuracy.

Index Terms — Energy minimization, minimum cut, maximum flow, stereo, motion, Markov Random Fields, Potts model, multiway cut problem, graph algorithms.

*Yuri Boykov is with Siemens Research, Princeton, NJ, yuri@scr.siemens.com. Olga Veksler is with NEC Research, Princeton, NJ, olga@research.nj.nec.com. Ramin Zabih is with the Computer Science Department, Cornell University, Ithaca, NY 14853, rdz@cs.cornell.edu
1 Energy minimization in early vision

Many early vision problems require estimating some spatially varying quantity (such as intensity or disparity) from noisy measurements. Such quantities tend to be piecewise smooth; they vary smoothly on the surface of an object, but change dramatically at object boundaries. Every pixel \( p \in \mathcal{P} \) must be assigned a label in some finite set \( \mathcal{L} \). For motion or stereo, the labels are disparities, while for image restoration they represent intensities. The goal is to find a labeling \( f \) that assigns each pixel \( p \in \mathcal{P} \) a label \( f_p \in \mathcal{L} \), where \( f \) is both piecewise smooth and consistent with the observed data.

These vision problems can be naturally formulated in terms of energy minimization. In this framework, one seeks the labeling \( f \) that minimizes the energy

\[
E(f) = E_{\text{smooth}}(f) + E_{\text{data}}(f).
\]

Here \( E_{\text{smooth}} \) measures the extent to which \( f \) is not piecewise smooth, while \( E_{\text{data}} \) measures the disagreement between \( f \) and the observed data. Many different energy functions have been proposed in the literature. The form of \( E_{\text{data}} \) is typically

\[
E_{\text{data}}(f) = \sum_{p \in \mathcal{P}} D_p(f_p),
\]

where \( D_p \) measures how appropriate a label is for the pixel \( p \) given the observed data. In image restoration, for example, \( D_p(f_p) \) is typically \( (f_p - i_p)^2 \), where \( i_p \) is the observed intensity of the pixel \( p \).

The choice of \( E_{\text{smooth}} \) is a critical issue, and many different functions have been proposed. For example, in some regularization-based approaches \([18, 26]\), \( E_{\text{smooth}} \) makes \( f \) smooth everywhere. This leads to poor results at object boundaries. Energy functions that do not have this problem are called discontinuity preserving. A large number of discontinuity preserving energy functions have been proposed (see for example \([17, 23, 32]\)).

The major difficulty with energy minimization for early vision lies in the enormous computational costs. Typically these energy functions have many local minimums (i.e., they are non-convex). Worse still, the space of possible labelings has dimension \( |\mathcal{P}| \), which is many thousands.

The energy functions that we consider in this paper arise in a variety of different contexts, including the Bayesian labeling of first-order Markov Random Fields (see the appendix for details). We consider energies of the form

\[
E(f) = \sum_{\{p,q\} \in \mathcal{N}} V_{p,q}(f_p, f_q) + \sum_{p \in \mathcal{P}} D_p(f_p),
\]
where $\mathcal{N}$ is the set of interacting pairs of pixels. Typically $\mathcal{N}$ consists of adjacent pixels, but it can be arbitrary. We allow $D_p$ to be nonnegative but otherwise arbitrary. In our choice of $E_{\text{smooth}}$ only pairs of pixels interact directly (as opposed to triples and larger groups of pixels directly influencing each other), but this is still a dramatic step from independence. Note that each pair of pixels $\{p, q\}$ can have its own distinct interaction penalty $V_{p,q}$, independent of any other pair of interacting pixels. This turns out to be important in many applications, as shown in Section 8.2. However, to simplify the notation, we will frequently write $V_{p,q}$ in equation (1) as $V$.

We develop algorithms that approximately minimize the energy $E(f)$ for an arbitrary finite set of labels $\mathcal{L}$ under two fairly general classes of interaction penalty $V$: metric and semi-metric. $V$ is called a metric on the space of labels $\mathcal{L}$ if it satisfies

$$V(\alpha, \beta) = 0 \iff \alpha = \beta,$$

$$V(\alpha, \beta) = V(\beta, \alpha) \geq 0,$$

$$V(\alpha, \beta) \leq V(\alpha, \gamma) + V(\gamma, \beta),$$

for any labels $\alpha, \beta, \gamma \in \mathcal{L}$. If $V$ does not satisfy the triangle inequality of equation (4), it is called a semi-metric.

Note that both semi-metrics and metrics include important cases of discontinuity preserving interaction penalties.\(^1\) Examples of such interaction penalties for a one-dimensional label set $\mathcal{L}$ include the truncated quadratic $V(\alpha, \beta) = \min(K, |\alpha - \beta|^2)$ (a semi-metric) and the truncated absolute distance $V(\alpha, \beta) = \min(K, |\alpha - \beta|)$ (a metric), where $K$ is some constant. If $\mathcal{L}$ is multidimensional, we can replace $| \cdot |$ by any norm, e.g. $\| \cdot \|_{L^2}$. Another important discontinuity preserving interaction penalty is given by the Potts model $V(\alpha, \beta) = K \cdot T(\alpha \neq \beta)$ (a metric), where $T(\cdot)$ is 1 if its argument is true, and otherwise 0.

We begin with a review of previous work on energy minimization in early vision. In Section 3 we give an overview of our energy minimization algorithms. Our first algorithm, described in Section 4, is based on $\alpha$-$\beta$-swap moves and works for any semi-metric $V$. Our second algorithm, described in Section 5, is based on the more interesting $\alpha$-expansion moves but works only for metric $V$’s. Optimality properties of our algorithms are discussed in Section 6. For example, we show that our $\alpha$-expansion algorithm produces a solution within a known factor of the global minimum of $E$. In Section 7 we describe an important special case of our energy which arises from the Potts interaction penalty. This is a very simple type of discontinuity preserving smoothness penalty, yet we prove that computing the global minimum is NP-hard. Experimental data is presented in Section 8.

\(^1\)Informally, a penalty $V(x, y)$ is discontinuity preserving if it is bounded, i.e. $\sup_{(x, y) \in \mathbb{R}^2} V(x, y) < \infty$. 

3
2 Related work

The energy functions that we are interested in, shown in equation (1), arise quite naturally in early vision. Energy based methods attempt to model some global image properties that can not be captured, for example, by local correlation techniques. The main problem, however, is that interesting energies are often difficult to minimize. We show in section 7.2 that one of the simplest discontinuity preserving cases of our energy function minimization is NP-hard; it is therefore impossible to rapidly compute the global minimum unless P=NP.

Due to the inefficiency of computing the global minimum, many authors have opted for a local minimum. However, in general a local minimum can be arbitrarily far from the optimum. It thus may not convey any of the global image properties that were encoded in the energy function. In such cases it is difficult to determine the cause of an algorithm’s failures. When an algorithm gives unsatisfactory results, it may be due either to a poor choice of the energy function, or to the fact that the answer is far from the global minimum. There is no obvious way to tell which of these is the problem.\(^2\) Another common issue is that local minimization techniques are naturally sensitive to the initial estimate.

In general, a labeling \(f\) is a local minimum of the energy \(E\) if

\[
E(f) \leq E(f') \quad \text{for any } f' \text{ “near to” } f. \tag{5}
\]

In case of discrete labeling, the labelings near to \(f\) are those that lie within a single move of \(f\). Many local optimization techniques use what we will call standard moves, where only one pixel can change its label at a time (see Figure 2(b)). For standard moves, equation (5) can be read as follows: if you are at a local minimum with respect to standard moves then you cannot decrease the energy by changing a single pixel’s label. In fact, this is a very weak condition. As a result, optimization schemes using standard moves frequently generate low quality solutions. For instance, consider the local minimum with respect to standard moves shown in Figure 1(c).

An example of a local method using standard moves is Iterated Conditional Modes (ICM), which is a greedy technique introduced in [4]. For each site (pixel or voxel), the label which gives the largest decrease of the energy function is chosen, until the iteration converges to a local minimum.

Another example of an algorithm using standard moves is simulated annealing, which was

\(^2\)In special cases where the global minimum can be rapidly computed, it is possible to separate these issues. For example, [16] points out that the global minimum of an Ising energy function is not necessarily the desired solution for image restoration. [8, 16] analyze the performance of simulated annealing in cases with a known global minimum.
popularized in computer vision by [15]. Annealing is popular because it is easy to implement, and it can optimize an arbitrary energy function. Unfortunately, minimizing an arbitrary energy function requires exponential time, and as a consequence simulated annealing is very slow. Theoretically, simulated annealing should eventually find the global minimum if run for long enough. As a practical matter, it is necessary to decrease the algorithm’s temperature parameter faster than required by the theoretically optimal schedule. Once annealing’s temperature parameter is sufficiently low, the algorithm will converge to a local minimum with respect to standard moves.

If the energy minimization problem is phrased in continuous terms, variational methods can be applied. These methods were popularized by [18]. Variational techniques use the Euler equations, which are guaranteed to hold at a local minimum.\(^3\) To apply these algorithms to actual imagery, of course, requires discretization.

Another alternative is to use discrete relaxation labeling methods; this has been done by many authors, including [10, 28, 31]. In relaxation labeling, combinatorial optimization is converted into continuous optimization with linear constraints. Then some form of gradient descent which gives the solution satisfying the constraints is used.

There are also methods that have optimality guarantees in certain cases. Continuation methods, such as graduated non-convexity [7], are an example. These methods involve approximating an intractable (non-convex) energy function by a sequence of energy functions, beginning with a tractable (convex) approximation. There are circumstances where these methods are known to compute the optimal solution (see [7] for details). Continuation methods can be applied to a large number of energy functions, but except for these special cases nothing is known about the quality of their output.

Mean field annealing is another popular minimization approach. It is based on estimating the partition function from which the minimum of the energy can be deduced. However computing the partition function is computationally intractable, and saddle point approximations [24] are used. [14] provides an interesting connection between mean field approximation and other minimization methods like graduated non-convexity.

There are a few interesting energy functions where the global minimum can be rapidly computed via dynamic programming. However, dynamic programming [2] is restricted essentially to energy functions in one-dimensional settings. This includes some important cases, such as snakes [21]. In general, the two-dimensional energy functions that arise in early vision cannot be solved efficiently via dynamic programming.

\(^3\)Note that in continuous cases the labels near to \(f\) in equation (5) are normally defined as \(||f - f'|| \leq \epsilon\) where \(\epsilon\) is a positive constant and \(|| \cdot ||\) is a norm, e.g. \(L_2\), over some appropriate functional space.
Figure 1: Comparison of local minimums with respect to standard and large moves in case of image restoration. We use energy (1) with quadratic data terms penalizing deviations from the observed intensities (b). The smoothness term is truncated $L_2$ metric. Both local minimums in (c) and (d) were obtained using labeling (b) as an initial solution.

Graph cuts can be used to find the global minimum for some multidimensional energy functions. When there are only 2 labels, equation (1) is called the Ising model. Greig et al. [16] showed how to find the global minimum in this case by a single graph cut. Note that the Potts model we discuss in Section 7 is the natural generalization of the Ising model. [12] develop a method optimal to within a factor of two for the Potts model energy function; however the data energy they use is very restrictive. Recently [29], [19], and [9] used graph cuts to find the exact global minimum of a certain type of energy functions. However, these energy functions apply only if the labels are one-dimensional which rules out motion estimation, for example. Most importantly these functions require $V$ to be convex, and hence are not discontinuity preserving.

3 Overview of our algorithms

The NP-hardness result given in Section 7.2 effectively forces us to compute a local minimum. However, our methods generate a local minimum with respect to very large moves. We show that such approach overcomes many of the problems associated with local minimums.

The algorithms introduced in this section generate a labeling that is a local minimum of the energy in (1) for two types of large moves: $\alpha$-expansion and $\alpha$-$\beta$-swap. In contrast to the standard moves described in Section 2 these moves allow large number of pixels to change their labels simultaneously. This makes the set of labelings within a single move of a locally optimal $f$ exponentially large, and the condition in (5) very demanding. For example,
\(\alpha\)-expansion moves are so strong that we are able to prove that any labeling locally optimal with respect to these moves is within a known factor of the global minimum (see Section 6). Figure 1 compares local minimums for standard moves (c) and for \(\alpha\)-expansion moves (d) obtained from the same initial solution (b). This and other experiments also show that in practice our solutions do not change significantly by varying the initial labelings. In most cases starting from a constant labeling (where all pixels have the same label) is good enough.

In Section 3.1 we discuss the moves we allow, which are best described in terms of partitions. In Section 3.2 we sketch the algorithms and list their basic properties. The main computational step of our algorithms is based on graph cuts, which we summarize in Section 3.3.

3.1 Partitions and move spaces

Any labeling \(f\) can be uniquely represented by a partition of image pixels \(P = \{P_l \mid l \in \mathcal{L}\}\) where \(P_l = \{p \in P \mid f_p = l\}\) is a subset of pixels assigned label \(l\). Since there is an obvious one to one correspondence between labelings \(f\) and partitions \(P\), we can use these notions interchangeably.

Given a pair of labels \(\alpha, \beta\), a move from a partition \(P\) (labeling \(f\)) to a new partition \(P'\) (labeling \(f'\)) is called an \(\alpha\)-\(\beta\) swap if \(P_l = P'_l\) for any label \(l \neq \alpha, \beta\). This means that the only difference between \(P\) and \(P'\) is that some pixels that were labeled \(\alpha\) in \(P\) are now labeled \(\beta\) in \(P'\), and some pixels that were labeled \(\beta\) in \(P\) are now labeled \(\alpha\) in \(P'\). A special case of an \(\alpha\)-\(\beta\) swap is a move that gives the label \(\alpha\) to some set of pixels that used to be labeled \(\beta\). One example of \(\alpha\)-\(\beta\) swap move is shown in Figure 2(c).

Given a label \(\alpha\), a move from a partition \(P\) (labeling \(f\)) to a new partition \(P'\) (labeling \(f'\)) is called an \(\alpha\)-expansion if \(P_\alpha \subset P'_\alpha\) and \(P'_l \subset P_l\) for any label \(l \neq \alpha\). In other words, an \(\alpha\)-expansion move allows any set of image pixels to change their labels to \(\alpha\). An example of an \(\alpha\)-expansion move is shown in Figure 2(d).

Recall that ICM and annealing use standard moves allowing only one pixel to change its intensity. An example of a standard move is given in Figure 2(b). Note that a move which assigns a given label \(\alpha\) to a single pixel is both an \(\alpha\)-\(\beta\) swap and an \(\alpha\)-expansion. As a consequence, a standard move is a special case of both a \(\alpha\)-\(\beta\) swap and an \(\alpha\)-expansion.

3.2 Algorithms and properties

We have developed two minimization algorithms. The swap algorithm finds a local minimum when swap moves are allowed and the expansion algorithm finds a local minimum when the
expansion moves are allowed. Finding such a local minimum is not a trivial task. Given a labeling $f$, there is an exponential number of swap and expansion moves. Therefore, even checking for a local minimum requires exponential time if it is performed naively. In contrast checking for a local minimum when only the standard moves are allowed is easy since there is only a linear number of standard moves given any labeling $f$.

We have developed efficient graph based methods to find the optimal $\alpha$-$\beta$-swap or $\alpha$-expansion given a labeling $f$ (see Sections 4 and 5). This is the key step in our algorithms. Once these methods are available, it is easy to design some variations of the “fastest descent” technique that can efficiently find the corresponding local minimums. Our algorithms are summarized in Figure 3.

The structure of the algorithms is quite similar. We will call a single execution of steps 3.1–3.2 an iteration, and an execution of steps 2–4 a cycle. In each cycle, the algorithm performs an iteration for every label (expansion algorithm) or for every pair of labels (swap algorithm), in a certain order that can be fixed or random. A cycle is successful if a strictly better labeling is found at any iteration. The algorithms stop after the first unsuccessful cycle since no further improvement is possible. Obviously, a cycle in the swap algorithm takes $|L|^2$ iterations, and a cycle in the expansion algorithm takes $|L|$ iterations.

These algorithms are guaranteed to terminate in a finite number of cycles. In fact, under the assumptions that $V$ and $D_p$ in equation (1) are constants independent of the image size $P$ we can easily prove termination in $O(|P|)$ cycles [33]. These assumptions are quite reasonable in practice. However, in the experiments we report in Section 8, the algorithm stops after a few cycles, and most of the improvements occur during the first cycle.
1. Start with an arbitrary labeling $f$
2. Set success := 0
3. For each pair of labels $\{\alpha, \beta\} \subset \mathcal{L}$
   3.1. Find $\hat{f} = \arg\min E(f')$ among $f'$ within one $\alpha$-$\beta$ swap of $f$
   3.2. If $E(\hat{f}) < E(f)$, set $f := \hat{f}$ and success := 1
4. If success = 1 goto 2
5. Return $f$

1. Start with an arbitrary labeling $f$
2. Set success := 0
3. For each label $\alpha \in \mathcal{L}$
   3.1. Find $\hat{f} = \arg\min E(f')$ among $f'$ within one $\alpha$-expansion of $f$
   3.2. If $E(\hat{f}) < E(f)$, set $f := \hat{f}$ and success := 1
4. If success = 1 goto 2
5. Return $f$

Figure 3: Our swap algorithm (top) and expansion algorithm (bottom).

We use graph cuts to efficiently find $\hat{f}$ for the key part of each algorithm in step 3.1. Step 3.1 uses a single minimum cut on a graph whose size is $O(|\mathcal{P}|)$. The graph is dynamically updated after each iteration. The details of this minimum cut are quite different for the swap and the expansion algorithms, and are described in details in Sections 4 and 5.

### 3.3 Graph cuts

Before describing the key step 3.1 of the swap and the expansion algorithms, we will review graph cuts. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted graph with two distinguished vertices called the terminals. A cut $\mathcal{C} \subset \mathcal{E}$ is a set of edges such that the terminals are separated in the induced graph $\mathcal{G}(\mathcal{C}) = (\mathcal{V}, \mathcal{E} - \mathcal{C})$. In addition, no proper subset of $\mathcal{C}$ separates the terminals in $\mathcal{G}(\mathcal{C})$. The cost of the cut $\mathcal{C}$, denoted $|\mathcal{C}|$, equals the sum of its edge weights.

The minimum cut problem is to find the cut with smallest cost. This problem can be solved efficiently by computing the maximum flow between the terminals, according to a theorem due to Ford and Fulkerson [13]. There are a large number of fast algorithms for this problem (see [1], for example). The worst case complexity is low-order polynomial; however, for graphs with the special structure that we build, the running time is nearly linear in practice.
4 Finding the optimal swap move

Given an input labeling $f$ (partition $P$) and a pair of labels $\alpha, \beta$, we wish to find a labeling $\hat{f}$ that minimizes $E$ over all labelings within one $\alpha$-$\beta$ swap of $f$. This is the critical step in the swap move algorithm given at the top of Figure 3. Our technique is based on computing a labeling corresponding to a minimum cut on a graph $G_{\alpha \beta} = \langle V_{\alpha \beta}, E_{\alpha \beta} \rangle$. The structure of this graph is dynamically determined by the current partition $P$ and by the labels $\alpha, \beta$.

This section is organized as follows. First we describe the construction of $G_{\alpha \beta}$ for a given $f$ (or $P$). We show that cuts $C$ on $G_{\alpha \beta}$ correspond in a natural way to labelings $f^C$ which are within one $\alpha$-$\beta$ swap move of $f$. Theorem 4.4 shows that the cost of a cut is $|C| = E(f^C)$ plus a constant. A corollary from this theorem states our main result that the desired labeling $\hat{f}$ equals $f^C$ where $C$ is a minimum cut on $G_{\alpha \beta}$.

The structure of the graph is illustrated in Figure 4. For legibility, this figure shows the case of a 1D image. For any image the structure of $G_{\alpha \beta}$ will be as follows. The set of vertices includes the two terminals $\alpha$ and $\beta$, as well as image pixels $p$ in the sets $P_\alpha$ and $P_\beta$ (that is $f_p \in \{\alpha, \beta\}$). Thus, the set of vertices $V_{\alpha \beta}$ consists of $\alpha$, $\beta$, and $P_{\alpha \beta} = P_\alpha \cup P_\beta$. Each pixel $p \in P_{\alpha \beta}$ is connected to the terminals $\alpha$ and $\beta$ by edges $t^\alpha_p$ and $t^\beta_p$, respectively. For brevity, we will refer to these edges as $t$-links (terminal links). Each pair of pixels $\{p, q\} \subset P_{\alpha \beta}$ which are neighbors (i.e. $\{p, q\} \in N$) is connected by an edge $e_{\{p, q\}}$ which we will call an $n$-link (neighbor link). The set of edges $E_{\alpha \beta}$ thus consists of $\bigcup_{p \in P_{\alpha \beta}} \{t^\alpha_p, t^\beta_p\}$ (the $t$-links) and $\bigcup_{\{p, q\} \in N} e_{\{p, q\}}$ (the $n$-links). The weights assigned to the edges are...
Any cut $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$ must sever (include) exactly one $t$-link for any pixel $p \in \mathcal{P}_{\alpha\beta}$: if neither $t$-link were in $\mathcal{C}$, there would be a path between the terminals; while if both $t$-links were cut, then a proper subset of $\mathcal{C}$ would be a cut. Thus, any cut leaves each pixel in $\mathcal{P}_{\alpha\beta}$ with exactly one $t$-link. This defines a natural labeling $f^C$ corresponding to a cut $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$:

$$f_p^C = \begin{cases} 
\alpha & \text{if } t^\alpha_p \in \mathcal{C} \text{ for } p \in \mathcal{P}_{\alpha\beta} \\
\beta & \text{if } t^\beta_p \in \mathcal{C} \text{ for } p \in \mathcal{P}_{\alpha\beta} \\
f_p & \text{for } p \in \mathcal{P}, p \notin \mathcal{P}_{\alpha\beta}. 
\end{cases}$$ (6)

In other words, if the pixel $p$ is in $\mathcal{P}_{\alpha\beta}$ then $p$ is assigned label $\alpha$ when the cut $\mathcal{C}$ separates $p$ from the terminal $\alpha$; similarly, $p$ is assigned label $\beta$ when $\mathcal{C}$ separates $p$ from the terminal $\beta$. If $p$ is not in $\mathcal{P}_{\alpha\beta}$ then we keep its initial label $f_p$. This implies

**Lemma 4.1** A labeling $f^C$ corresponding to a cut $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$ is one $\alpha$-$\beta$ swap away from the initial labeling $f$. 

It is easy to show that a cut $\mathcal{C}$ severs an $n$-link $e_{\{p,q\}}$ between neighboring pixels on $\mathcal{G}_{\alpha\beta}$ if and only if $\mathcal{C}$ leaves the pixels $p$ and $q$ connected to different terminals. Formally

**Property 4.2** For any cut $\mathcal{C}$ and for any $n$-link $e_{\{p,q\}}$:

a) If $t^\alpha_p, t^\alpha_q \in \mathcal{C}$ then $e_{\{p,q\}} \notin \mathcal{C}$.

b) If $t^\beta_p, t^\beta_q \in \mathcal{C}$ then $e_{\{p,q\}} \notin \mathcal{C}$.

c) If $t^\beta_p, t^\alpha_q \in \mathcal{C}$ then $e_{\{p,q\}} \in \mathcal{C}$.

d) If $t^\alpha_p, t^\beta_q \in \mathcal{C}$ then $e_{\{p,q\}} \in \mathcal{C}$.

Properties (a) and (b) follow from the requirement that no proper subset of $\mathcal{C}$ should separate the terminals. Properties (c) and (d) also use the fact that a cut has to separate the terminals. These properties are illustrated in figure 5.

The next lemma is a consequence of property 4.2 and equation 6.

**Lemma 4.3** For any cut $\mathcal{C}$ and for any $n$-link $e_{\{p,q\}}$

$$|\mathcal{C} \cap e_{\{p,q\}}| = V(f_p^C, f_q^C).$$
Figure 5: Properties of a cut $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$ for two pixels $p, q \in \mathcal{N}$ connected by an $n$-link $e_{\{p,q\}}$. Dotted lines show the edges cut by $\mathcal{C}$ and solid lines show the edges remaining in the induced graph $\mathcal{G}(\mathcal{C}) = (\mathcal{V}, \mathcal{E} - \mathcal{C})$.

**Proof:** There are four cases with similar structure; we will illustrate one the case where $t^\alpha_p, t^\beta_q \in \mathcal{C}$. In this case, $e_{\{p,q\}} \in \mathcal{C}$ and, therefore, $|\mathcal{C} \cap e_{\{p,q\}}| = |e_{\{p,q\}}| = V(\alpha, \beta)$. By (6), $f^C_p = \alpha$ and $f^C_q = \beta$.

Note that this proof assumes that $V$ is a semi-metric, i.e. that equations 2 and 3 hold.

**Theorem 4.4** There is a one to one correspondence between cuts $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$ and labelings that are one $\alpha$-$\beta$ swap from $f$. Moreover, the cost of a cut $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$ is $|\mathcal{C}| = E(f^\mathcal{C})$ plus a constant.

**Proof:** The first part follows from the fact that the severed $t$-links uniquely determine the labels assigned to pixels $p$ and the $n$-links that must be cut. We now compute the cost of a cut $\mathcal{C}$, which is

$$|\mathcal{C}| = \sum_{p \in \mathcal{P}_{\alpha\beta}} |\mathcal{C} \cap \{t^\alpha_p, t^\beta_p\}| + \sum_{\{p,q\} \in \mathcal{N} \atop \{p,q\} \subset \mathcal{P}_{\alpha\beta}} |\mathcal{C} \cap e_{\{p,q\}}|.$$  \hspace{1cm} (7)

Note that for $p \in \mathcal{P}_{\alpha\beta}$ we have

$$|\mathcal{C} \cap \{t^\alpha_p, t^\beta_p\}| = \begin{cases} |t^\alpha_p| & \text{if } t^\alpha_p \in \mathcal{C} \\ |t^\beta_p| & \text{if } t^\beta_p \in \mathcal{C} \end{cases} = D_p(f^\mathcal{C}_p) + \sum_{q \in \mathcal{N} \atop q \notin \mathcal{P}_{\alpha\beta}} V(f^\mathcal{C}_p, f_q).$$
Lemma 4.3 gives the second term in (7). Thus, the total cost of a cut \( C \) is

\[
|C| = \sum_{p \in P_{\alpha \beta}} D_p(f_p^C) + \sum_{p \in P_{\alpha \beta}} \sum_{q \in N \setminus P_{\alpha \beta}} V(f_p^C, f_q) + \sum_{(p,q) \in N : p \text{ or } q \in P_{\alpha \beta}} V(f_p^C, f_q^C).
\]

This can be rewritten as

\[
|C| = E(f^C) - K
\]

where

\[
K = \sum_{p \notin P_{\alpha \beta}} D_p(f_p) + \sum_{(p,q) \in N : (p,q) \notin P_{\alpha \beta}} V(f_p, f_q)
\]

is the same constant for all cuts \( C \).

Corollary 4.5 The lowest energy labeling within a single \( \alpha-\beta \) swap move from \( f \) is \( \hat{f} = f_C \), where \( C \) is the minimum cut on \( G_{\alpha \beta} \).

5 Finding the optimal expansion move

Given an input labeling \( f \) (partition \( P \)) and a label \( \alpha \), we wish to find a labeling \( \hat{f} \) that minimizes \( E \) over all labelings within one \( \alpha \)-expansion of \( f \). This is the critical step in the expansion move algorithm given at the bottom of Figure 3. In this section we describe a technique that solves the problem assuming that each \( V \) is a metric, and thus satisfies the triangle inequality (4). Our technique is based on computing a labeling corresponding to a minimum cut on a graph \( G_{\alpha} = (V_{\alpha}, E_{\alpha}) \). The structure of this graph is determined by the current partition \( P \) and by the label \( \alpha \). As before, the graph dynamically changes after each iteration.

This section is organized as follows. First we describe the construction of \( G_{\alpha} \) for a given \( f \) (or \( P \)) and \( \alpha \). We show that cuts \( C \) on \( G_{\alpha} \) correspond in a natural way to labelings \( f_C \) which are within one \( \alpha \)-expansion of \( f \). Then, based on a number of simple properties, we define a class of elementary cuts. Theorem 5.4 shows that elementary cuts are in one to one correspondence with labelings that are within one \( \alpha \)-expansion of \( f \), and also that the cost of an elementary cut is \( |C| = E(f^C) \). A corollary from this theorem states our main result that the desired labeling \( \hat{f} \) is \( f_C \) where \( C \) is a minimum cut on \( G_{\alpha} \).

The structure of the graph is illustrated in Figure 6. For legibility, this figure shows the case of a 1D image. The set of vertices includes the two terminals \( \alpha \) and \( \bar{\alpha} \), as well as all image pixels \( p \in P \). In addition, for each pair of neighboring pixels \( \{p,q\} \in N \) separated in
Figure 6: An example of $\mathcal{G}_\alpha$ for a 1D image. The set of pixels in the image is $\mathcal{P} = \{p, q, r, s\}$ and the current partition is $\mathcal{P} = \{P_1, P_2, P_\alpha\}$ where $P_1 = \{p\}, P_2 = \{q, r\},$ and $P_\alpha = \{s\}$. Two auxiliary nodes $a = a_{\{p,q\}}, b = a_{\{r,s\}}$ are introduced between neighboring pixels separated in the current partition. Auxiliary nodes are added at the boundary of sets $P_l$.

The current partition (i.e. such that $f_p \neq f_q$), we create an auxiliary vertex $a_{\{p,q\}}$. Auxiliary nodes are introduced at the boundaries between partition sets $P_l$ for $l \in \mathcal{L}$. Thus, the set of vertices is

$$V_\alpha = \{ \alpha, \bar{\alpha}, \mathcal{P}, \bigcup_{(p,q) \in \mathcal{N}} a_{\{p,q\}} \}.$$  

Each pixel $p \in \mathcal{P}$ is connected to the terminals $\alpha$ and $\bar{\alpha}$ by $t$-links $t^\alpha_p$ and $t^\bar{\alpha}_p$, respectively. Each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ which are not separated by the partition $\mathcal{P}$ (i.e. such that $f_p = f_q$) is connected by an $n$-link $e_{\{p,q\}}$. For each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ such that $f_p \neq f_q$ we create a triplet of edges $\mathcal{E}_{\{p,q\}} = \{e_{\{p,a\}}, e_{\{a,q\}}, t^\bar{\alpha}_a\}$ where $a = a_{\{p,q\}}$ is the corresponding auxiliary node. The edges $e_{\{p,a\}}$ and $e_{\{a,q\}}$ connect pixels $p$ and $q$ to $a_{\{p,q\}}$ and the $t$-link $t^\bar{\alpha}_a$ connects the auxiliary node $a_{\{p,q\}}$ to the terminal $\bar{\alpha}$. So we can write the set of all edges as

$$\mathcal{E}_\alpha = \bigcup_{p \in \mathcal{P}} \{t^\alpha_p, t^\bar{\alpha}_p\}, \bigcup_{(p,q) \in \mathcal{N}, \ f_p \neq f_q} \mathcal{E}_{\{p,q\}}, \bigcup_{(p,q) \in \mathcal{N}, \ f_p = f_q} e_{\{p,q\}}.$$  

The weights assigned to the edges are
As in Section 4, any cut $C$ on $G_{\alpha}$ must sever (include) exactly one $t$-link for any pixel $p \in \mathcal{P}$. This defines a natural labeling $f^C$ corresponding to a cut $C$ on $G_{\alpha}$. Formally,

$$f^C_p = \begin{cases} 
\alpha & \text{if } t^\alpha_p \in C \\
f_p & \text{if } t^\alpha_p \in C \end{cases} \quad \forall p \in \mathcal{P}. \quad (8)$$

In other words, a pixel $p$ is assigned label $\alpha$ if the cut $C$ separates $p$ from the terminal $\alpha$, while $p$ is assigned its old label $f_p$ if $C$ separates $p$ from $\bar{\alpha}$. Note that for $p \not\in \mathcal{P}_{\alpha}$ the terminal $\bar{\alpha}$ represents labels assigned to pixels in the initial labeling $f$. Clearly we have

**Lemma 5.1** A labeling $f^C$ corresponding to a cut $C$ on $G_{\alpha}$ is one $\alpha$-expansion away from the initial labeling $f$.

It is also easy to show that a cut $C$ severs an $n$-link $e_{\{p,q\}}$ between neighboring pixels $\{p, q\} \in \mathcal{N}$ such that $f_p = f_q$ if and only if $C$ leaves the pixels $p$ and $q$ connected to different terminals. In other words, property 4.2 holds when we substitute “$\bar{\alpha}$” for “$\beta$”. We will refer to this as property 4.2($\bar{\alpha}$). Analogously, we can show that property 4.2 and equation (8) establish lemma 4.3 for the $n$-links $e_{\{p,q\}}$ in $G_{\alpha}$.

Consider now the set of edges $E_{\{p,q\}}$ corresponding to a pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ such that $f_p \neq f_q$. In this case, there are several different ways to cut these edges even when the pair of severed $t$-links at $p$ and $q$ is fixed. However, a minimum cut $C$ on $G_{\alpha}$ is guaranteed to sever the edges in $E_{\{p,q\}}$ depending on what $t$-links are cut at the pixels $p$ and $q$.

The rule for this case is described in property 5.2 below. Assume that $a = a_{\{p,q\}}$ is an auxiliary node between the corresponding pair of neighboring pixels.

<table>
<thead>
<tr>
<th>edge</th>
<th>weight</th>
<th>for</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t^\alpha_p$</td>
<td>$\infty$</td>
<td>$p \in \mathcal{P}_{\alpha}$</td>
</tr>
<tr>
<td>$t^\alpha_p$</td>
<td>$D_p(f_p)$</td>
<td>$p \not\in \mathcal{P}_{\alpha}$</td>
</tr>
<tr>
<td>$t^\alpha_p$</td>
<td>$D_p(\alpha)$</td>
<td>$p \in \mathcal{P}$</td>
</tr>
<tr>
<td>$e_{{p,q}}$</td>
<td>$V(f_p, \alpha)$</td>
<td>${p, q} \in \mathcal{N}, f_p \neq f_q$</td>
</tr>
<tr>
<td>$e_{{a,q}}$</td>
<td>$V(\alpha, f_q)$</td>
<td>${p, q} \in \mathcal{N}, f_p = f_q$</td>
</tr>
</tbody>
</table>

As in Section 4, any cut $C$ on $G_{\alpha}$ must sever (include) exactly one $t$-link for any pixel $p \in \mathcal{P}$. This defines a natural labeling $f^C$ corresponding to a cut $C$ on $G_{\alpha}$. Formally,
Property 5.2

Property 5.2(a) If \( t^α_p, t^α_q \in C \) then \( C \cap E \{p, q\} = \emptyset \).

Property 5.2(b) If \( t^α_p, t^α_q \in C \) then \( C \cap E \{p, q\} = e_{p,a} \).

Property 5.2(c) If \( t^α_p, t^α_q \in C \) then \( C \cap E \{p, q\} = e_{a,q} \).

Property 5.2(d) If \( t^α_p, t^α_q \in C \) then \( C \cap E \{p, q\} = e_{a,q} \).

Property (a) results from the fact that no subset of \( C \) is a cut. The others follow from the minimality of \(|C|\) and the fact that \(|e_{p,a}|, |e_{a,q}|\) and \(|t^α_a|\) satisfy the triangle inequality so that cutting any one of them is cheaper than cutting the other two together. These properties are illustrated in Figure 7.

Lemma 5.3 If \( \{p, q\} \in \mathcal{N} \) and \( f_p \neq f_q \) then the minimum cut \( C \) on \( \mathcal{G}_α \) satisfies

\[
|C \cap E \{p, q\}| = V(f^C_p, f^C_q).
\]

Proof: The equation follows from property 5.2, equation (8), and the edge weights. For example, if \( t^α_p, t^α_q \in C \) then \(|C \cap E \{p, q\}| = |t^α_a| = V(f_p, f_q)\). At the same time, (8) implies that \( f^C_p = f_p \) and \( f^C_q = f_q \). Note that the right penalty \( V \) is imposed whenever \( f^C_p \neq f^C_q \), due to the auxiliary pixel construction.

Property 4.2(\( \bar{α} \)) holds for any cut, and property 5.2 holds for a minimum cut. However, there can be other cuts besides the minimum cut that satisfy both properties. We will define an elementary cut on \( \mathcal{G}_α \) to be a cut that satisfies properties 4.2(\( \bar{α} \)) and 5.2.
**Theorem 5.4** Let $G_\alpha$ be constructed as above given $f$ and $\alpha$. Then there is a one to one correspondence between elementary cuts on $G_\alpha$ and labelings within one $\alpha$-expansion of $f$. Moreover, for any elementary cut $C$ we have $|C| = E(f^C)$.

**Proof:** We first show that an elementary cut $C$ is uniquely determined by the corresponding labeling $f^C$. The label $f_p^C$ at the pixel $p$ determines which of the $t$-links to $p$ is in $C$. Property 4.2($\bar{\alpha}$) shows which $n$-links $e_{\{p,q\}}$ between pairs of neighboring pixels $\{p,q\}$ such that $f_p = f_q$ should be severed. Similarly, property 5.2 determines which of the links in $E_{\{p,q\}}$ corresponding to $\{p,q\} \in \mathcal{N}$ such that $f_p \neq f_q$ should be cut.

The cost of an elementary cut $C$ is

$$|C| = \sum_{p \in \mathcal{P}} |C \cap \{t^\alpha_p, t^\bar{\alpha}_p\}| + \sum_{\{p,q\} \in \mathcal{N}} |C \cap e_{\{p,q\}}| + \sum_{\{p,q\} \in \mathcal{N}} |C \cap E_{\{p,q\}}|. \quad (9)$$

It is easy to show that for any pixel $p \in \mathcal{P}$ we have $|C \cap \{t^\alpha_p, t^\bar{\alpha}_p\}| = D_p(f_p^C)$. Lemmas 4.3 and 5.3 hold for elementary cuts, since they were based on properties 4.2 and 5.2. Thus, the total cost of a elementary cut $C$ is

$$|C| = \sum_{p \in \mathcal{P}} D_p(f_p^C) + \sum_{\{p,q\} \in \mathcal{N}} V(f_p^C, f_q^C) = E(f^C).$$

Therefore, $|C| = E(f^C)$.

Our main result is a simple consequence of this theorem, since the minimum cut is an elementary cut.

**Corollary 5.5** The lowest energy labeling within a single $\alpha$ expansion move from $f$ is $\hat{f} = f^C$, where $C$ is the minimum cut on $G_\alpha$.

### 6 Optimality properties

Here we discuss optimality properties of our algorithms. In Section 6.1 we show that any local minimum generated by our expansion moves algorithm is within a known factor from the global optimum. This algorithm works in case of metric $V$. The swap move algorithm can be applied to a wider class of semi-metric $V$’s but, unfortunately, it does not have any guaranteed optimality properties. In Section 6.2 we show that a provably good solution can be obtained even for semi-metric $V$ by approximating such $V$’s with a simple Potts metric.
6.1 The expansion move algorithm

We now prove that a local minimum when expansion moves are allowed is within a known factor of the global minimum. This factor, which can be as small as 2, will depend on $V$. Specifically, let $c = \max_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta) / \min_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta)$ be the ratio of the largest non-zero value of $V$ to the smallest non-zero value of $V$. Note that $c$ is well defined since $V(\alpha, \beta) \neq 0$ for $\alpha \neq \beta$ according to the metric properties (2) and (3). If $V_{p,q}$’s are different for neighboring pairs $p, q$ then $c = \max_{p,q \in N} \left( \frac{\max_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta)}{\min_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta)} \right)$.

**Theorem 6.1** Let $\hat{f}$ be a local minimum when the expansion moves are allowed and $f^*$ be the globally optimal solution. Then $E(\hat{f}) \leq 2cE(f^*)$.

**Proof:** Let us fix some $\alpha \in \mathcal{L}$ and let

$$\mathcal{P}_\alpha = \{ p \in \mathcal{P} \mid f^*_p = \alpha \}. \quad (10)$$

We can produce a labeling $f^\alpha$ within one $\alpha$-expansion move from $\hat{f}$ as follows:

$$f^\alpha_p = \begin{cases} \alpha & \text{if } p \in \mathcal{P}_\alpha \\ \hat{f}_p & \text{otherwise} \end{cases} \quad (11)$$

The key observation is that since $\hat{f}$ is a local minimum if expansion moves are allowed,

$$E(\hat{f}) \leq E(f^\alpha). \quad (12)$$

Let $S$ be a set consisting of any number of pixels in $\mathcal{P}$ and any number of pairs of neighboring pixels in $\mathcal{N}$. We define $E(f|S)$ to be a restriction of the energy of labeling $f$ to the set $S$:

$$E(f|S) = \sum_{p \in S} D_p(f_p) + \sum_{\{p,q\} \in S} V(f_p, f_q).$$

Let $I^\alpha$ be the set of pixels and pairs of neighboring pixels contained inside $\mathcal{P}_\alpha$. Also, let $B^\alpha$ be the set of pairs of neighboring pixels on the boundary of $\mathcal{P}_\alpha$ and $O^\alpha$ be the set of pixels and pairs of neighboring pixels contained outside of $\mathcal{P}_\alpha$. Formally,

$$I^\alpha = \mathcal{P}_\alpha \cup \{ \{p, q\} \in \mathcal{N} : p \in \mathcal{P}_\alpha, q \in \mathcal{P}_\alpha \},$$

$$B^\alpha = \{ \{p, q\} \in \mathcal{N} : p \in \mathcal{P}_\alpha, q \notin \mathcal{P}_\alpha \},$$

$$O^\alpha = (\mathcal{P} - \mathcal{P}_\alpha) \cup \{ \{p, q\} \in \mathcal{N} : p \notin \mathcal{P}_\alpha, q \notin \mathcal{P}_\alpha \}. $$
The following three facts hold:

\[ E(f^\alpha|O^\alpha) = E(\hat{f}|O^\alpha), \]  
(13)

\[ E(f^\alpha|I^\alpha) = E(f^*|I^\alpha), \]  
(14)

\[ E(f^\alpha|B^\alpha) \leq cE(f^*|B^\alpha). \]  
(15)

Equations (13) and (14) are obvious from the definitions in (11) and (10). Equation (15) holds because for any \( \{p, q\} \in B^\alpha \) we have \( V(f_p^\alpha, f_q^\alpha) \leq cV(f_p^*, f_q^*) \neq 0. \)

Since \( I^\alpha \cup B^\alpha \cup O^\alpha \) includes all pixels in \( P \) and all neighboring pairs of pixels in \( N \), we can expand both sides of (12) to get:

\[ E(\hat{f}|I^\alpha) + E(\hat{f}|B^\alpha) + E(\hat{f}|O^\alpha) \leq E(f^\alpha|I^\alpha) + E(f^\alpha|B^\alpha) + E(f^\alpha|O^\alpha) \]

Using (13), (14) and (15) we get from the equation above:

\[ E(\hat{f}|I^\alpha) + E(\hat{f}|B^\alpha) \leq E(f^*|I^\alpha) + cE(f^*|B^\alpha). \]  
(16)

To get the bound on the total energy, we need to sum equation (16) over all labels \( \alpha \in \mathcal{L} \):

\[ \sum_{\alpha \in \mathcal{L}} (E(\hat{f}|I^\alpha) + E(\hat{f}|B^\alpha)) \leq \sum_{\alpha \in \mathcal{L}} (E(f^*|I^\alpha) + cE(f^*|B^\alpha)) \]  
(17)

Let \( B = \bigcup_{\alpha \in \mathcal{L}} B^\alpha \). Observe that for every \( \{p, q\} \in B \), the term \( V(\hat{f}_p, \hat{f}_q) = E(\hat{f}|\{p, q\}) \) appears twice on the left side of (17), once in \( E(\hat{f}|B^\alpha) \) for \( \alpha = f_p^\ast \) and once in \( E(\hat{f}|B^\alpha) \) for \( \alpha = f_q^\ast \). Similarly every \( V(f_p^\ast, f_q^\ast) = E(f^*|\{p, q\}) \) appears \( 2c \) times on the right side of (17). Therefore equation (17) can be rewritten to get the bound of \( 2c \):

\[ E(\hat{f}) + E(\hat{f}|B) \leq E(f^*) + (2c - 1)E_B(f^*) \leq 2cE(f^*). \]

Note that Kleinberg and Tardos [22] develop an algorithm for minimizing \( E \) which also has optimality properties. For the Potts model \( V \) discussed in the next section, their algorithm has a bound of 2, which is the same bound as we have.\(^4\) For a general metric \( V \), they have a bound of \( O(\log k \log \log k) \) where \( k \) is the number of labels. However, their algorithm uses linear programming, which is impractical for the large number of variables occurring in computer vision.

19
Figure 8: The image consists of three pixels $\mathcal{P} = \{1, 2, 3\}$. There are two pairs of neighbors $\mathcal{N} = \{\{1, 2\}, \{2, 3\}\}$. The set of labels is $\mathcal{L} = \{a, b, c\}$. $D_p$ is shown in (c). $V(a, b) = V(b, c) = \frac{K}{2}$ and $V(a, c) = K$. It is easy to see that configuration in (a) is a local minimum with the energy of $K$, while the optimal configuration (b) has energy 4.

6.2 Approximating a semi-metric

A local minimum when the swap moves are allowed can be arbitrarily far from the global minimum. This is illustrated by an example in Figure 8.

In fact, we can use the expansion algorithm to get an answer within a factor of $2c$ from the optimum of energy (1) even when $V$ is a semi-metric. Here $c$ is the same as in Theorem 6.1. This $c$ is still well defined for a semi-metric. Suppose that penalty $V$ inside the definition of energy $E$ in (1) is a semi-metric. Let $r$ be any real number in the interval $[m, M]$ where

\[ m = \min_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta) \quad \text{and} \quad M = \max_{\alpha \neq \beta \in \mathcal{L}} V(\alpha, \beta). \]

Define a new energy based on the Potts interaction model

\[ E_P(f) = \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{\{p,q\} \in \mathcal{N}} r \cdot T(f_p \neq f_q). \]

**Theorem 6.2** If $\hat{f}$ is a local minimum of $E_P$ given the expansion moves and $f^*$ is the global minimum of $E(f)$ then $E(\hat{f}) \leq 2cE(f^*)$.

**Proof:** Suppose $f^o$ is the global minimum of $E_P$. Then

\[ \frac{r}{M} E(\hat{f}) \leq E_P(\hat{f}) \leq 2E_P(f^o) \leq 2E_P(f^*) \leq 2\frac{r}{m} E(f^*) \]

where the second inequality follows from Theorem 6.1. Note that $c = M/m$. 

4In fact, it can be shown that any algorithm that is within a factor of 2 for the Potts model is within a factor of $2c$ for an arbitrary metric $V$.

5Actually we just need $V(\alpha, \beta) \geq 0$ and $V(\alpha, \beta) = 0 \iff \alpha = \beta$. 

20
Thus to find an answer within a fixed factor from the global minimum for a semi-metric $V$, one can take a local minimum $\hat{f}$ given the expansion moves for $E_P$ as defined above. Note that that such an $\hat{f}$ is not a local minimum of $E(f)$ given the expansion moves. In practice however we find that local minimum given the swap moves gives empirically better results than using $\hat{f}$. In fact, the estimate $\hat{f}$ can be used as a good starting point for the swap algorithm. In this case the swap move algorithm will also generate a local minimum whose energy is within a known factor from the global minimum.

7 The Potts model

An interesting special case of the energy in equation (1) arises when $V$ is given by the Potts model [27]

$$E_P(f) = \sum_{\{p,q\} \in \mathcal{N}} u_{\{p,q\}} \cdot T(f_p \neq f_q) + \sum_{p \in \mathcal{P}} D_p(f_p).$$

(18)

In this case, discontinuities between any pair of labels are penalized equally. This is in some sense the simplest discontinuity preserving model and it is especially useful when the labels are unordered or the number of labels is small. The Potts interaction penalty $V_{p,q} = u_{\{p,q\}} \cdot T(f_p \neq f_q)$ is a metric; in this case $c = 1$ and our expansion algorithm gives a solution that is within a factor of 2 of the global minimum. Note that by definition $c \geq 1$, so this is the energy function with the best bound.

Interestingly, the Potts model energy minimization problem is closely related to a known combinatorial optimization problem called the multiway cut problem. In this section we investigate this relationship and its consequences. We will first show (Section 7.1) that the Potts model energy minimization problem can be reduced to the multiway cut problem. More precisely, we prove that the global minimum of the Potts model energy $E_P$ can be computed by finding the minimum cost multiway cut on an appropriately constructed graph. We then prove (Section 7.2) that if we could efficiently compute the global minimum of $E_P$ we could also solve a certain class of multiway cut problems that are known to be NP-hard. This in turn implies that minimizing $E_P$ is NP-hard, and so is minimizing the energy in (1).

The multiway cut problem is defined on a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ with non-negative edge weights, with a set of terminal vertices $\mathcal{L} \subset \mathcal{V}$. A subset of the edges $\mathcal{C} \subset \mathcal{E}$ is called a multiway cut if the terminals are completely separated in the induced graph $\mathcal{G}(\mathcal{C}) = \langle \mathcal{V}, \mathcal{E} - \mathcal{C} \rangle$. We will also require that no proper subset of $\mathcal{C}$ separates the terminals in $\mathcal{G}(\mathcal{C})$. The cost of the multiway cut $\mathcal{C}$ is denoted by $|\mathcal{C}|$ and equals the sum of its edge weights. The multiway cut problem is to find the minimum cost multiway cut [11]. Also in [11] they show
that the multiway cut problem is NP-complete. Note that the multiway cut problem is a generalization of the standard two-terminal graph cut problem described in Section 3.3.

7.1 The Potts model and the multiway cut problem

We now show that the problem of minimizing the Potts energy $E_P(f)$ can be solved by computing a minimum cost multiway cut on a certain graph. We take $\mathcal{V} = \mathcal{P} \cup \mathcal{L}$. This means that $\mathcal{G}$ contains two types of vertices: $p$-vertices (pixels) and $l$-vertices (labels). Note that $l$-vertices will serve as terminals for our multiway cut problem. Two $p$-vertices are connected by an edge if and only if the corresponding pixels are neighbors in the neighborhood system $\mathcal{N}$. The set $\mathcal{E}_\mathcal{N}$ consists of the edges between $p$-vertices, which we will call $n$-links. Each $n$-link $\{p, q\} \in \mathcal{E}_\mathcal{N}$ is assigned a weight $w_{\{p,q\}} = u_{\{p,q\}}$.

Each $p$-vertex is connected by an edge to each $l$-vertex. An edge $\{p, l\}$ that connects a $p$-vertex with a terminal (an $l$-vertex) will be called a $t$-link and the set of all such edges will be denoted by $\mathcal{E}_\mathcal{T}$. Each $t$-link $\{p, l\} \in \mathcal{E}_\mathcal{T}$ is assigned a weight $w_{\{p,l\}} = K_p - D_p(l)$, where $K_p > \max_l D_p(l)$ is a constant that is large enough to make the weights positive. The edges of the graph are $\mathcal{E} = \mathcal{E}_\mathcal{N} \cup \mathcal{E}_\mathcal{T}$. Figure 9(a) shows the structure of the graph $\mathcal{G}$.

It is easy to see that there is a one-to-one correspondence between multiway cuts and labelings. A multiway cut $C$ corresponds to the labeling $f^C$ which assigns the label $l$ to all pixels $p$ which are $t$-linked to the $l$-vertex in $\mathcal{G}(C)$. An example of a multiway cut and the corresponding image partition (labeling) is given in Figure 9(b).

**Theorem 7.1** If $C$ is a multiway cut on $\mathcal{G}$, then $|C| = E_P(f^C)$ plus a constant.

The proof of theorem 7.1 is given in [9].

**Corollary 7.2** If $C$ is a minimum cost multiway cut on $\mathcal{G}$, then $f^C$ minimizes $E_P$.

While the multiway cut problem is known to be NP-complete if there are more than 2 terminals, there is a fast approximation algorithm [11]. This algorithm works as follows. First, for each terminal $l \in \mathcal{L}$ it finds an isolating two-way minimum cut $C(l)$ that separates $l$ from all other terminals. This is just the standard graph cut problem. Then the algorithm generates a multiway cut $C = \bigcup_{l \neq l_{\text{max}}} C(l)$ where $l_{\text{max}} = \arg \max_{l \in \mathcal{L}} |C(l)|$ is the terminal with the largest cost isolating cut. This “isolation heuristic” algorithm produces a cut which is optimal to within a factor of $2 - \frac{2}{|\mathcal{L}|}$. However, the isolation heuristic algorithm suffers from two problems that limits its applicability to our energy minimization problem.

- The algorithm will assign many pixels a label that is chosen essentially arbitrarily. Note that the union of all isolating cuts $\bigcup_{l \in \mathcal{L}} C(l)$ may leave some vertices disconnected
Figure 9: An example of the graph $G = \langle V, E \rangle$ with terminals $L = \{1, \ldots, k\}$ is given in (a). The pixels $p \in P$ are shown as white squares. Each pixel has an $n$-link to its four neighbors. Each pixel is also connected to all terminals by $t$-links (some of the $t$-links are omitted from the drawing for legibility). The set of vertices $V = P \cup L$ includes all pixels and terminals. The set of edges $E = E_N \cup E_T$ consists of all $n$-links and $t$-links. In (b) we show an induced graph $G(C) = \langle V, E - C \rangle$ corresponding to some multiway cut $C$. A multiway cut corresponds to a unique partition (labeling) of image pixels.
from any terminal. The multiway cut \( C = \bigcup_{l \neq l_{\text{max}}} C(l) \) connects all those vertices to the terminal \( l_{\text{max}} \).

- While the multiway cut \( C \) produced is close to optimal, this does not imply that the resulting labeling \( f^C \) is close to optimal. Formally, let us write theorem 7.1 as \( |C| = E_P(C) + K \) (the constant \( K \) results from the \( K_p \)'s, as described in [9]). The isolation heuristic gives a solution \( \hat{C} \) such that \( |\hat{C}| \leq 2|C^*| \), where \( C^* \) is the minimum cost multiway cut. Thus, \( E_P(\hat{C}) + K \leq 2(E_P(C^*) + K) \), so \( E_P(\hat{C}) \leq 2E_P(C^*) + K \). As a result, the isolation heuristic algorithm does not produce a labeling whose energy is within a constant factor of optimal. Note that the \( K \) used in the construction given in [9] is so large that this bound is nearly meaningless.

7.2 Minimizing the Potts energy is NP-hard

In the previous section we showed that the problem of minimizing the energy \( E_P(f) \) in (18) over all possible labelings \( f \) can be solved by computing a minimum multiway cut on a certain graph. In this section we make the reduction in the opposite direction. Specifically, for an arbitrary fixed graph \( G = \langle V, E \rangle \) we will construct an instance of minimizing \( E_P(f) \) where the optimal labeling \( f^* \) determines a minimum multiway cut on \( G \). This will prove that a polynomial-time method for finding \( f^* \) would provide a polynomial-time algorithm for finding the minimum cost multiway cut, which is known to be NP-hard [11]. This NP-hardness proof is based on a construction due to Jon Kleinberg.

The energy minimization problem we address takes as input a set of pixels \( P \), a neighborhood relation \( \mathcal{N} \) and a label set \( \mathcal{L} \), as well as a set of weights \( u_{\{p,q\}} \) and a function \( D_p(l) \). The problem is to find the labeling \( f^* \) that minimizes the energy \( E_P(f) \) given in equation (18).

Let \( G = \langle V, E \rangle \) be an arbitrary weighted graph with terminal vertices \( \{t_1, \ldots, t_k\} \subset V \) and edge weights \( w_{\{p,q\}} \). We will do the energy minimization using \( P = V, \mathcal{N} = E, \) and \( u_{\{p,q\}} = w_{\{p,q\}} \). The label set will be \( \mathcal{L} = \{1, \ldots, k\} \). Let \( K \) be a constant such that \( K > E_P(f^*) \); for example, we can select \( K \) to be the sum of all \( w_{\{p,q\}} \). Our function \( D_p(l) \) will force \( f^*(t_j) = j \); if \( p = t_j \) is a terminal vertex,

\[
D_p(l) = \begin{cases} 
0 & l = j, \\
K & \text{otherwise.}
\end{cases}
\]

For a non-terminal vertex \( p \) all labels are equally good,

\[
\forall l \quad D_p(l) = 0.
\]
We will define a labeling $f$ to be feasible if the set of pixels labeled $j$ by $f$ forms a connected component that includes $t_j$. Feasible labelings obviously correspond one-to-one with multiway cuts.

**Theorem 7.3** The labeling $f^*$ is feasible, and the cost of a feasible labeling is the cost of the corresponding multiway cut.

**Proof:** To prove that $f^*$ is feasible, suppose that there were a set $S$ of pixels that $f^*$ labeled $j$ which were not part of the component containing $t_j$. We could then obtain a labeling with lower energy by switching this set to the label of some pixel on the boundary of $S$. The energy of a feasible labeling $f$ is

$$\sum_{\{p,q\} \in \mathcal{N}} u_{\{p,q\}} \cdot T(f(p) \neq f(q)),$$

which is the cost of the multiway cut corresponding to $f$.

This shows that minimizing the Potts model energy $E_P(f)$ on an arbitrary $\mathcal{P}$ and $\mathcal{N}$ is intractable. In computer vision, however, $\mathcal{P}$ is usually a planar grid, and combinatorial problems that are intractable on arbitrary graphs sometimes become tractable on the plane or grid.

We now sketch a proof that the energy minimization problem is intractable even when restricted to a planar grid. The reduction is from a special case of the multiway cut problem, where $\mathcal{G}$ is a planar graph with degree 11 and all the edges have weight 1, which is shown to be NP-hard in [11]. We first must embed $\mathcal{G}$ in a grid of pixels, which happens in two stages. In the first stage we convert $\mathcal{G}$ into a planar graph of degree 4. In the second stage we embed this graph in the grid by using a method given in [20]. This embedding can be done in polynomial time; after it is done, each vertex $v \in \mathcal{G}$ corresponds to a connected set of pixels $S(v)$ in the grid, and the adjacency relationships among vertices in $\mathcal{G}$ has been preserved.

The proof now proceeds along the same lines as theorem 7.3, except for three subtleties. First, we need to ensure that for every vertex $v$ all pixels in $S(v)$ are given the same label. We address this by making the edge weights $K$ between adjacent pixels in $S(v)$. Second, when we embed $\mathcal{G}$ in the grid, there will be gaps. We can solve this by adding additional “grid pixels”, which $D$ forces to have the extra label 0 ($D$ will prevent non-grid pixels from having label 0 by making $D_p(0) = K$). We take the edge weights between grid pixels and non-grid pixels to be one. The cost of a feasible labeling is now the cost of the corresponding multiway cut plus a constant. Third, the constant $K > E_P(f^*)$ must be now chosen more carefully.
8 Experimental results

We applied our algorithms to image restoration and visual correspondence and achieved promising results. Due to space considerations, we omit the results for image restoration. For these results we refer the interested reader to [33].

In this section we present experimental results on visual correspondence for stereo and motion. In visual correspondence we are given two images taken at the same time from different view points for stereo and taken from the same view point but at different times for motion. For each pixel in the first image there is a corresponding pixel in the second image which is a projection along the line of sight of the same real world scene element. The difference in the coordinates of the corresponding points is called the disparity. In stereo the disparity is usually one-dimensional because corresponding points lie along epipolar lines. In motion the disparity is usually a two-dimensional quantity. The disparity varies smoothly everywhere except at object boundaries, and corresponding points are expected to have similar intensities. Thus we can formulate the correspondence problem as the energy minimization problem

\[ E(f) = \sum_{(p,q) \in N} V_{p,q}(f_p, f_q) + \sum_{p \in P} D(I_p - I'_{p+f_p}). \]

Here \( P \) is the set of all pixels in the first image, \( I_p \) is the intensity of pixel \( p \) in the first image, \( I'_q \) is the intensity of pixel \( q \) in the second image, and \( p + f_p \) stands for the pixel with coordinates of \( p \) shifted by disparity \( f_p \). The data penalty \( D \) is small if there is a small difference between \( I_p \) and \( I'_{p+f_p} \); \( D \) will be discussed in more detail in Section 8.1.

For our experiments, we used three energy functions. The first energy function, called \( E_Q \), uses the truncated quadratic \( V(f_p, f_q) = \min(K, |f_p - f_q|^2) \) (for some constant \( K \)) as its smoothness term. This choice of \( V \) does not obey the triangle inequality, so we minimized \( E_Q \) using our swap algorithm. The second \( E_P \) and the third \( E_L \) energy functions use, correspondingly, the Potts model and the truncated \( L_2 \) distance as their smoothness penalty \( V \). Both of these obey the triangle inequality and we minimized \( E_P \) and \( E_L \) with our expansion algorithm.

8.1 Data term

If the pixels \( p \) and \( q \) correspond, they are assumed to have similar intensities \( I_p \) and \( I'_q \). Thus \((I_p - I'_q)^2\) is frequently used as a penalty for deciding that \( p \) and \( q \) correspond. This penalty has a heavy weight unless \( I_p \approx I'_q \). However there are special circumstances when corresponding pixels have very different intensities due to the effects of image sampling.
Suppose that the true disparity is not an integer. If a pixel overlaps a scene patch with high intensity gradient, then the corresponding pixels may have significantly different intensities.

For stereo we use the technique in [6] to develop a $D_p$ that is insensitive to image sampling. First we measure how well $p$ fits into the real valued range of disparities $(d - \frac{1}{2}, d + \frac{1}{2})$ by

$$C_{fwd}(p, d) = \min_{d - \frac{1}{2} \leq x \leq d + \frac{1}{2}} |I_p - I'_{p+x}|.$$  

We get fractional values $I'_{p+x}$ by linear interpolation between discrete pixel values. For symmetry we also measure

$$C_{rev}(p, d) = \min_{p - \frac{1}{2} \leq x \leq p + \frac{1}{2}} |I_x - I'_{p+d}|.$$  

$C_{fwd}(p, d)$ and $C_{rev}(p, d)$ can be computed with just a few comparisons. The final measure is

$$C(p, d) = (\min \{C_{fwd}(p, d), C_{rev}(p, d), Const\})^2,$$  

Here $Const$ is used to make the measure more robust. For motion we developed a similar technique which is described in [33].

8.2 Static cues

In this section we discuss how to choose different $V_{p,q}$ for each pair of interacting pixels $\{p, q\}$ to take advantage of contextual information. For simplicity we will consider the case of the Potts model, i.e. $V_{p,q} = u_{\{p,q\}} \cdot T(f_p \neq f_q)$. The intensities of pixels in the first image contain information that can significantly influence our assessment of disparities without even considering the second image. For example, two neighboring pixels $p$ and $q$ are much more likely to have the same disparity if we know that $I(p) \approx I(q)$. Most methods for computing correspondence do not make use of this kind of contextual information. Some exceptions include [5, 25, 34].

We can easily incorporate contextual information into our framework by allowing $u_{\{p,q\}}$ to vary depending on their intensities $I_p$ and $I_q$. Let

$$u_{\{p,q\}} = U(|I_p - I_q|).$$  

Each $u_{\{p,q\}}$ represents a penalty for assigning different disparities to neighboring pixels $p$ and $q$. The value of the penalty $u_{\{p,q\}}$ should be smaller for pairs $\{p, q\}$ with larger intensity differences $|I_p - I_q|$. In practice we use an empirically selected decreasing function $U(\cdot)$. Note that instead of (19) we could also set the coefficients $u_{\{p,q\}}$ according to an output of an edge detector on the first image. For example, $u_{\{p,q\}}$ can be made small for pairs $\{p, q\}$
where an intensity edge was detected and large otherwise. Segmentation results can also be used.

The following example shows the importance of contextual information. Consider the pair of synthetic images below, with a uniformly white rectangle in front of a black background.

![First image](image1.png) ![Second image](image2.png)

There is a one pixel horizontal shift in the location of the rectangle, and there is no noise. Without noise, the problem of estimating $f$ is reduced to minimizing the smoothness term $E_{\text{smooth}}(f)$ under the constraint that pixel $p$ can be assigned disparity $d$ only if $I_p = I_{p+d}$.

If $u_{\{p,q\}}$ is the same for all pairs of neighbors $\{p,q\}$ then $E_{\text{smooth}}(f)$ is minimized at one of the labeling shown in the picture below. Exactly which labeling minimizes $E_{\text{smooth}}(f)$ depends on the relationship between the height of the square and the height of the background.

![Labeling](labeling.png)

Suppose now that the penalty $u_{\{p,q\}}$ is much smaller if $I_p \neq I_q$ than it is if $I_p = I_q$. In this case the minimum of $E_{\text{smooth}}(f)$ is achieved at the disparity configuration shown in the picture below. This result is much closer to human perception.

![Optimal labeling](optimal_labeling.png)

### 8.3 Real stereo imagery with ground truth

In figure 11 we show results from a real stereo pair with known ground truth, provided by Dr. Y. Ohta and Dr. Y. Nakamura from the University of Tsukuba. The left image of the
Figure 10: Real imagery with ground truth
Figure 11: Performance comparison of expansion and swap algorithms with simulated annealing for the problem in figure 10(a).
real stereo pair is shown in figure 11(a). Figure 11(b) shows the ground truth for this stereo pair.

For this stereo pair we used $E_P$. We compared our results against annealing and normalized correlation. For normalized correlation we chose parameters which give the best statistics. We implemented several different annealing variants, and used the one that gave the best performance. This was the Metropolis sampler with a linearly decreasing temperature schedule. To give it a good starting point, simulated annealing was initialized with the results from normalized correlation. In contrast for our algorithms the starting point is unimportant. The results differ by less than 1% of image pixels from any starting point that we have tried.

Figures 10(c), and (d) show the results of the swap and expansion algorithms for $\lambda = 20$. Figures 10(e) and (f) show the results of normalized correlation and simulated annealing. More detailed analysis of this imagery, together with a comparison of additional algorithms, can be found in [30]. Note, however, that [30] confirms that for this imagery the best previous algorithm is simulated annealing, which outperforms (among others) correlation, robust estimation, scanline-based dynamic programming, and mean-field techniques.

The table below summarizes the errors made by the algorithms. In approximately 20 minutes simulated annealing reduces the total errors normalized correlation makes by about one fifth and it cuts the number of $\pm 1$ errors in half. It makes very little additional progress in the rest of 19 hours that we ran it. Our expansion, swap, and jump algorithms make approximately 3 times fewer $\pm 1$ errors and approximately 5 times fewer total errors compared to normalized correlation.

Our expansion and swap algorithms perform similarly to each other. The observed slight difference in errors is quite insignificant (less than one percent). At each cycle the order of labels to iterate over is chosen randomly. Another run of the algorithms might give slightly different results, where expansion algorithm might do better than the swap algorithm. In general we observed very slight variation between different runs of an algorithm. However the difference in the running time is significant. On average the expansion algorithm converges 3 times as rapidly as the swap algorithm.

<table>
<thead>
<tr>
<th>algorithm</th>
<th>% total errors</th>
<th>% of errors $&gt; \pm 1$</th>
<th>running time</th>
</tr>
</thead>
<tbody>
<tr>
<td>expansion algorithm</td>
<td>7.6</td>
<td>2.1</td>
<td>106 sec</td>
</tr>
<tr>
<td>swap algorithm</td>
<td>7.0</td>
<td>2.0</td>
<td>300 sec</td>
</tr>
<tr>
<td>simulated annealing</td>
<td>20.3</td>
<td>5.0</td>
<td>1200 sec</td>
</tr>
<tr>
<td>normalized correlation</td>
<td>24.7</td>
<td>10.0</td>
<td>5 sec</td>
</tr>
</tbody>
</table>

Figure 11 shows the graph of $E_{\text{smooth}}$ versus time for our algorithms versus simulated annealing. Notice that the time axis is on the logarithmic scale. We do not show the
<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>% of total errors</th>
<th>% of errors $&gt; \pm 1$</th>
<th>Absolute average error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>26.6</td>
<td>4.5</td>
<td>0.40</td>
</tr>
<tr>
<td>5</td>
<td>13.0</td>
<td>4.5</td>
<td>0.27</td>
</tr>
<tr>
<td>10</td>
<td>7.0</td>
<td>2.3</td>
<td>0.15</td>
</tr>
<tr>
<td>20</td>
<td>7.6</td>
<td>2.1</td>
<td>0.15</td>
</tr>
<tr>
<td>30</td>
<td>7.9</td>
<td>2.3</td>
<td>0.17</td>
</tr>
<tr>
<td>50</td>
<td>8.8</td>
<td>2.3</td>
<td>0.18</td>
</tr>
<tr>
<td>100</td>
<td>10.4</td>
<td>2.9</td>
<td>0.21</td>
</tr>
<tr>
<td>500</td>
<td>16.3</td>
<td>8.2</td>
<td>0.37</td>
</tr>
</tbody>
</table>

Figure 12: Table of errors for the expansion algorithm for different values of $\lambda$.

The expansion algorithm gives a convergence curve significantly steeper than the other curves. In fact the expansion algorithm makes 99% of the progress in the first iteration.

The algorithms appear to be quite stable in the choice of parameter $\lambda$. For example the table in figure 12 gives the errors made by the expansion algorithm for different choices of $\lambda$. For small $\lambda$ the algorithm makes a lot of errors because it overemphasizes the data, for large values of $\lambda$ the algorithm makes a lot of errors because it overemphasizes the prior. However for a large interval of $\lambda$ values the results are good.

### 8.4 SRI tree stereo pair

In the well-known SRI stereo pair whose left image is shown in figure 13(a) the number of disparities is larger, and $E_P$ does not work as well. Figure 13(b) and (c) compares the results of minimizing $E_P$ and $E_L$. Notice that there are fewer disparities found in figure 13(b), since the piecewise constant prior tends to produce large regions with the same disparity.
8.5 Motion

Figure 14(a) shows one image of a motion sequence where a cat moves against moving background. This is a difficult sequence because the cat’s motion is non-rigid. We used $E_Q$ for minimization. Figures 14(b) and (c) show the horizontal and vertical motions detected with our swap algorithm. Notice that the cat has been accurately localized. Even the tail and parts of the legs are clearly separated from the background motion.

Figure 15 shows the output from the well-known flower garden sequence. Since the camera motion is nearly horizontal, we have simply displayed the camera motion. We used $E_P$ to achieve these results.
Figure 15: Flower garden sequence

Acknowledgements

We thank J. Kleinberg, D. Shmoys and E. Tardos for providing important input on the content of the paper. This research has been supported by DARPA under contract DAAL01-97-K-0104, by NSF awards CDA-9703470 and IIS-9900115, and by a grant from Microsoft.

Appendix: Bayesian labeling of Markov Random Fields

In this appendix we show that minimization of the energy function in (1) is equivalent to computing the maximum a posteriori estimate of a Markov Random Field.

Let $\mathcal{P}$ be a set of sites, $\mathcal{L}$ a set of labels, and $\mathcal{N} = \{\{p, q\} | p, q \in \mathcal{P}\}$ a neighborhood system on $\mathcal{P}$. Let $F = F_1, \ldots, F_n$ be a set of random variables defined on $\mathcal{P}$. Each $F_p$ takes values in the label set $\mathcal{L}$. A particular realization of the field will be denoted by $f = \{f_p | p \in P\}$, which is also called a configuration of the field $F$. As usual, $P(F_p = f_p)$ will be abbreviated by $P(f_p)$. $F$ is said to be a Markov Random Field (MRF) if:

(i) $P(f) > 0 \quad \forall f \in \mathcal{F}$.
(ii) $P(f_p | f_{\mathcal{P} \setminus \{p\}}) = P(f_p | f_{\mathcal{N}_p})$

where $\mathcal{P} - \{p\}$ denotes set difference, $f_{\mathcal{N}_p}$ denotes all labels of sites in $\mathcal{N}_p$, and $\mathcal{F}$ denotes the set of all possible labelings.

The easiest way to specify an MRF is by the joint distribution using the Hammersley-Clifford theorem [3]. This theorem proves the equivalence between MRFs and Gibbs random fields.

Before defining Gibbs random fields we need to define a clique. A set of sites is called a
clique if each member of the set is a neighbor of all the other members. A Gibbs random field can be specified by the Gibbs distribution:

\[ P(f) = Z^{-1} \cdot \exp \left( - \sum_{c \in C} V_c(f) \right), \]

where \( C \) is the set of all cliques, \( Z \) is the normalizing constant, and \( \{V_c(f)\} \) are functions from a labeling to non-negative reals, called the clique potential functions.

Thus to specify an MRF we need to specify the clique potential functions. We will consider a first order MRF, which means that for all cliques of size larger than two the potential functions are zero, and for the cliques of size two the potential functions are specified by

\[ V_c(f) = V_{p,q}(f_p, f_q). \]

This defines an MRF with the joint distribution:

\[ P(f) = Z^{-1} \cdot \exp \left( - \sum_{\{p,q\} \in N} V_{p,q}(f_p, f_q) \right). \]

In general, the field \( F \) is not directly observable in the experiment. A popular way to estimate its realized configuration \( f \) based on an observation \( d \) is the maximum a posteriori (MAP) estimation. Using Bayes rule, the posterior probability can be written as

\[ p(f|d) = \frac{p(d|f)p(f)}{p(d)}. \]

Thus the MAP estimate \( f^* \) is equal to

\[ \arg \max_{f \in F} p(d|f)p(f) = \arg \min_{f \in F} (- \log p(d|f)p(f)) \]

Assume that the observation \( d_p \) at each pixel is independent and that

\[ p(d_p|l) = C_p \cdot \exp(-D_p(l)) \quad \text{for} \quad l \in \mathcal{L}, \]

where \( C_p \) is the normalizing constant, and \( D_p \) was defined in Section 1. Then the likelihood can be written as

\[ p(d|f) \propto \exp \left( - \sum_{p \in \mathcal{P}} D_p(f_p) \right). \]

Writing out \( p(d) \) and \( p(d|f) \) with the above assumptions, we get

\[ f^* = \arg \max_{f \in F} \sum_{(p,q) \in \mathcal{N}} V_{p,q}(f_p, f_q) + \sum_{p \in \mathcal{P}} D_p(f_p), \]

which is the general form of the energy function we are minimizing.
References


