# Fast Approximate Energy Minimization via Graph Cuts

Yuri Boykov
Siemens Research
Princeton, New Jersey

Olga Veksler
NEC Research
Princeton, New Jersey

Ramin Zabih
Computer Science Department
Cornell University

**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. These tasks are naturally stated in terms of energy minimization. However, the huge computational expense involved in minimizing the energy has limited the appeal of this approach. In this paper we address the problem of efficiently minimizing a large class of energy functions that occur in early vision. Minimizing these energy functions can be viewed as computing the maximum *a posteriori* extimate of a first-order Markov Random Field under arbitrary independent noise.

Since global minimization of these energy functions is NP-hard in general, we search for a local minimum. However our solutions are a local minimum even when very large moves are allowed. One move we consider is an $\alpha$-expansion: for a label $\alpha$, this move assigns an arbitrary set of pixels the label $\alpha$. Our expansion move algorithm requires the smoothness term to be a metric. It generates a labeling such that there is no expansion move that decreases the energy. Furthermore, we show that this labeling is within a known factor of the global minimum. Another move we consider is an $\alpha$-$\beta$-swap: for a pair of labels $\alpha, \beta$, this move swaps the labels between an arbitrary set of pixels labeled $\alpha$ and another arbitrary set labeled $\beta$. Our swap move algorithm does not require the smoothness term to obey the triangle. It generates a labeling such that there is no swap move that decreases the energy. Both our algorithms allow important cases of discontinuity preserving energy functions. The algorithms rely on graph cuts as a powerful optimization technique.

An important special case arises when the smoothness term is the Potts model, which penalizes any pair of different labels equally. Here, the task of minimizing the energy function is equivalent to solving a known combinatorial optimization problem called the minimum cost multiway cut. Minimizing this energy function is NP-hard; however, our algorithm results in a solution which is within a a factor of 2 of the global minimum.

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.

# 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 set $\mathcal{L}$.[1] 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

$$E(f) \quad = \quad E_{smooth}(f) \quad + \quad E_{data}(f).$$

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

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

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 $E_{smooth}$ is a critical issue, and many different functions have been proposed. For example, in some regularization-based approaches [18, 26], $E_{smooth}$ makes $f$ smooth everywhere. This leads to poor results at object boundaries. Energy functions that do

[1] In our approach we assume that $\mathcal{L}$ is finite.

not have this problem are called *discontinuity-preserving*. A large number of discontinuity-preserving energy functions have been proposed (see for example [23, 17, 32]).

The major difficulty with energy minimization for early vision lies in the enormous computational costs. Typically these energy functions have many local minima (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 MRF's (see the appendix). We consider energies of the form

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

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 $E_{smooth}$ only pairs of pixels interact directly (as opposed to triples and larger groupls of pixels directly influencing each other), but this is still a dramatic step from independence. Due to computational complexity, most methods have this limitation. Note that for each distinct pair of pixels $\{p, q\}$ we can design its own interaction function $V_{p,q}$ independent of any other pair of interacting pixels. This turns out to be important in many applications, see section (8.2). To simplify the notation, we will frequently drop the subscript $p, q$ for $V$ in equation (1).

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 potentials $V$: semi-metric and metric. $V$ is called a *metric* on the space of labels $\mathcal{L}$ if for any labels $\alpha, \beta, \gamma \in \mathcal{L}$ it satisfies the properties:

$$V(\alpha, \beta) \quad = \quad 0 \qquad \Leftrightarrow \qquad \alpha \quad = \quad \beta, \tag{2}$$

$$V(\alpha, \beta) \quad = \quad V(\beta, \alpha) \geq 0. \tag{3}$$

$$V(\alpha, \beta) \quad \leq \quad V(\alpha, \gamma) \quad + \quad V(\gamma, \beta) \tag{4}$$

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 potentials.[2] An example of discontinuity preserving semi-metric is $V(\alpha, \beta) = min(K, (\alpha-\beta)^2)$ One example metric is the truncated $L_1$ distance $V(\alpha, \beta) = min(K, |\alpha-\beta|)$, where $K$ is some constant. Another important discontinuity preserving metric is the Potts

[2]Informally, a function $V(x, y)$ is discontinuity preserving if $\sup_{x,y\in\mathcal{R}} V(x, y) < K$ for some constant K. Thus, the maximum possible penalty for assigning different labels to neighboring pixels can be bounded.

interaction penalty $V(\alpha, \beta) = T(\alpha \neq \beta)$, 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 $V$'s. Our second algorithm, described in section 5, is based on the more interesting $\alpha$-expansion moves but works only for metric $V$'s. We show that this method produces a solution within a known factor of the global minimum of $E$. In section 7 we describe an interesting special case of our algorithms which arise from the Potts interaction penalty, and we prove that the Potts energy minimization problem is NP-hard. Experimental data is presented in section 8.

# 2 Related work

## 2.1 Global Optimization

There have been numerous attempts to design fast algorithms for energy minimization. Simulated annealing was popularized in computer vision by [15], and is widely used since 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. In practice, annealing is inefficient partly because at each step it changes the value of a single pixel.

An alternative to simulated annealing is to use methods that have optimality guarantees in certain cases. Continuation methods, such as graduated non-convexity [7], is 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 (Parisi [24]) are used. Geiger and Yuille [14] provide 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 to energy functions which are essentially one-dimensional. 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.

Graph cuts can be used to find the global minimum for some two-dimensional 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. Ferrari et al. [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 [19], [9], and [29] 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-dimentional which rules out motion estimation, for example. Most importantly these functions require $V$ to be convex, and hence are not discontinuity preserving.

## 2.2 Local optimization

Due to the inefficiency of computing the global minimum, many authors have opted for a local minimum. One problem with this approach is that 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.[3] Another issue is that local minimization techniques are naturally sensitive to the initial estimate.

An example of a local method is Iterated Conditional Modes (ICM), which is a greedy technique introduced by Besag in [4]. For each site, the label which gives the largest increase of the energy function is chosen, until the iteration converges to a local minimum.

If the energy minimization problem is phrased in continuous terms, variational methods can be applied. These methods were popularized by Horn [18]. Variational techniques use the Euler equations, which are guaranteed to hold at a local minimum (although they may also hold elsewhere). 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

[3] In the 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.

descent which gives the solution satisfying the constraints is used.

# 3 Overview of our algorithms

The most important property of our methods is that they produce a local minimum even when large moves are allowed. In this section, we discuss the moves we allow, which are best described in terms of partitions. Then we sketch the algorithms and list their basic properties.

## 3.1 Partitions and move spaces

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

Given a pair of labels $\alpha, \beta$, a move from a partition $\mathbf{P}$ (labeling $f$) to a new partition $\mathbf{P}'$ (labeling $f'$) is called an $\alpha$-$\beta$ *swap* if $\mathcal{P}_l = \mathcal{P}'_l$ for any label $l \neq \alpha, \beta$. This means that the only difference between $\mathbf{P}$ and $\mathbf{P}'$ is that some pixels that were labeled $\alpha$ in $\mathbf{P}$ are now labeled $\beta$ in $\mathbf{P}'$, and some pixels that were labeled $\beta$ in $\mathbf{P}$ are now labeled $\alpha$ in $\mathbf{P}'$. 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 $\beta$.

Given a label $\alpha$, a move from a partition $\mathbf{P}$ (labeling $f$) to a new partition $\mathbf{P}'$ (labeling $f'$) is called an $\alpha$-*expansion* if $\mathcal{P}_\alpha \subset \mathcal{P}'_\alpha$ and $\mathcal{P}'_l \subset \mathcal{P}_l$ for any label $l \neq \alpha$. In other words, an $\alpha$-expansion move allows any set of image pixels to change their labels to $\alpha$.

Recall that ICM and annealing use *standard* moves, where a move from $f$ to $f'$ is standard if it changes the label of just one pixel. Note that a move which gives an arbitrary 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 local minimums 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 naïvely. In contrast

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 1: Our swap move algorithm (top) and expansion move algorithm (bottom).

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 methods to find the optimal $\alpha$-$\beta$-swap or $\alpha$-expansion given a labeling $f$. This is the key step in each algorithm which enables us to efficiently compute a local minimum. Once these methods are available, the efficient algorithms to find a local minimum are easy to design. For example, we can easily perform fastest descent as follows: given a labeling $f$, for each pair of labels $\{\alpha, \beta\}$ (or for each label $\alpha$) find the optimal $\alpha$-$\beta$-swap (or $\alpha$-expansion) and take the best of these swaps (or expansions). However since finding the optimal $\alpha$-$\beta$-swap and $\alpha$-expansion are the most expensive steps of our algorithm, we want to minimize the number of these steps. We therefore took a slightly different approach which retains a fastest descent flavor to ensure rapid convergence. Given a labeling $f$, once the optimal $\alpha$-$\beta$-swap is found for one pair of labels, we take this move if it decreases the energy (instead of searching for the optimal swap over all pairs of labels). Our algorithms are summarized in figure 1.

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 $|\mathcal{L}|^2$ iterations, and a cycle in the expansion algorithm takes $|\mathcal{L}|$ 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 $\mathcal{P}$ we can easily prove termination in $O(|\mathcal{P}|)$ 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.

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 the next two sections.

### 3.3 Graph cuts

Before describing the key step 3.1 of the swap and the expansion algorithms, we will review graph cuts. Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ 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 $\mathcal{G}(\mathcal{C}) = \langle \mathcal{V}, \mathcal{E} - \mathcal{C} \rangle$. 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 the graphs with 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 $\mathbf{P}$) 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 algorithm given at the top of Figure 1. Our technique is based on computing a labeling corresponding to a minimum cut on a graph $\mathcal{G}_{\alpha\beta} = \langle \mathcal{V}_{\alpha\beta}, \mathcal{E}_{\alpha\beta} \rangle$. The structure of this graph is dynamically determined by the current partition $\mathbf{P}$ and by the labels $\alpha, \beta$.

This section is organized as follows. First we describe the construction of $\mathcal{G}_{\alpha\beta}$ for a given $f$ (or $\mathbf{P}$). We show that cuts $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$ correspond in a natural way to labelings $f^{\mathcal{C}}$ which are within one $\alpha$-$\beta$ swap move of $f$. Theorem 4.4 shows that the cost of a cut is $|\mathcal{C}| = E(f^{\mathcal{C}})$ plus

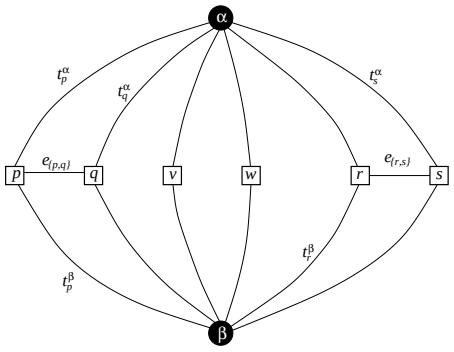



Figure 2: An example of the graph $\mathcal{G}_{\alpha\beta}$ for a 1D image. The set of pixels in the image is $\mathcal{P}_{\alpha\beta} = \mathcal{P}_\alpha \cup \mathcal{P}_\beta$ where $\mathcal{P}_\alpha = \{p, r, s\}$ and $\mathcal{P}_\beta = \{q, \ldots, w\}$.

a constant. A corollary from this theorem states our main result that the desired labeling $\hat{f}$ equals $f^{\mathcal{C}}$ where $\mathcal{C}$ is a minimum cut on $\mathcal{G}_{\alpha\beta}$.

The structure of the graph is illustrated in Figure 2. For legibility, this figure shows the case of a 1D image. For any image the structure of $\mathcal{G}_{\alpha\beta}$ will be as follows. The set of vertices includes the two terminals $\alpha$ and $\beta$, as well as image pixels $p$ in the sets $\mathcal{P}_\alpha$ and $\mathcal{P}_\beta$ (that is $f_p \in \{\alpha, \beta\}$). Thus, the set of vertices $\mathcal{V}_{\alpha\beta}$ consists of $\alpha$, $\beta$, and $\mathcal{P}_{\alpha\beta} = \mathcal{P}_\alpha \cup \mathcal{P}_\beta$. Each pixel $p \in \mathcal{P}_{\alpha\beta}$ is connected to the terminals $\alpha$ and $\beta$ by edges $t_p^\alpha$ and $t_p^\beta$, respectively. For brevity, we will refer to these edges as $t$-links (terminal links). Each pair of pixels $\{p, q\} \subset \mathcal{P}_{\alpha\beta}$ which are neighbors (i.e. $\{p, q\} \in \mathcal{N}$) is connected by an edge $e_{\{p,q\}}$ which we will call an $n$-link (neighbor link). The set of edges $\mathcal{E}_{\alpha\beta}$ thus consists of $\bigcup_{p\in\mathcal{P}_{\alpha\beta}} \{t_p^\alpha, t_p^\beta\}$ (the $t$-links) and $\bigcup_{\substack{\{p,q\}\in\mathcal{N} \\ p,q\in\mathcal{P}_{\alpha\beta}}} e_{\{p,q\}}$ (the $n$-links). The weights assigned to the edges are

| **edge** | **weight** | **for** |
|---|---|---|
| $t_p^\alpha$ | $D_p(\alpha) + \sum_{\substack{q\in\mathcal{N}_p \\ q\notin\mathcal{P}_{\alpha\beta}}} V(\alpha, f_q)$ | $p \in \mathcal{P}_{\alpha\beta}$ |
| $t_p^\beta$ | $D_p(\beta) + \sum_{\substack{q\in\mathcal{N}_p \\ q\notin\mathcal{P}_{\alpha\beta}}} V(\beta, f_q)$ | $p \in \mathcal{P}_{\alpha\beta}$ |
| $e_{\{p,q\}}$ | $V(\alpha, \beta)$ | $\{p,q\}\in\mathcal{N}$ $p,q\in\mathcal{P}_{\alpha\beta}$ |

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 $f^{\mathcal{C}}$ corresponding to a cut $\mathcal{C}$ on $\mathcal{G}_{\alpha\beta}$,

$$f_p^{\mathcal{C}} = \begin{cases} \alpha & \text{if } t_p^{\alpha} \in \mathcal{C} \text{ for } p \in \mathcal{P}_{\alpha\beta} \\ \beta & \text{if } t_p^{\beta} \in \mathcal{C} \text{ for } p \in \mathcal{P}_{\alpha\beta} \\ f_p & \text{for } p \in \mathcal{P}\,, p \notin \mathcal{P}_{\alpha\beta}\,. \end{cases} \tag{5}$$

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 $f^{\mathcal{C}}$ 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 $e_{\{p,q\}}$:*

$$\begin{array}{llll} a) & \textit{If} & t_p^{\alpha}, t_q^{\alpha} \in \mathcal{C} & \textit{then} \quad e_{\{p,q\}} \notin \mathcal{C}. \\ b) & \textit{If} & t_p^{\beta}, t_q^{\beta} \in \mathcal{C} & \textit{then} \quad e_{\{p,q\}} \notin \mathcal{C}. \\ c) & \textit{If} & t_p^{\beta}, t_q^{\alpha} \in \mathcal{C} & \textit{then} \quad e_{\{p,q\}} \in \mathcal{C}. \\ d) & \textit{If} & t_p^{\alpha}, t_q^{\beta} \in \mathcal{C} & \textit{then} \quad e_{\{p,q\}} \in \mathcal{C}. \end{array}$$

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 3.

The next lemma is a consequence of property 4.2 and equation 5.

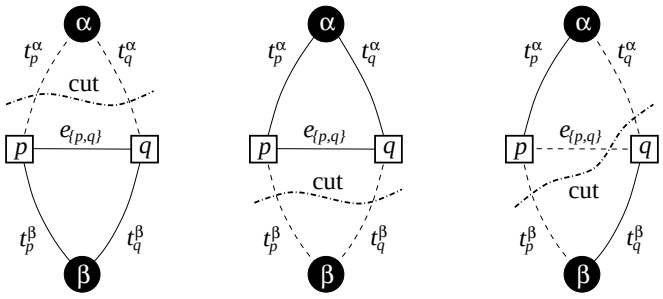
**Lemma 4.3** *For any cut $\mathcal{C}$ and for any $n$-link $e_{\{p,q\}}$*

$$|\mathcal{C} \cap e_{\{p,q\}}| \;=\; V(f_p^{\mathcal{C}}, f_q^{\mathcal{C}}).$$

PROOF: There are four cases with similar structure; we will illustrate one the case where $t_p^{\alpha}, t_q^{\beta} \in \mathcal{C}$. In this case, $e_{\{p,q\}} \in \mathcal{C}$ and, therefore, $|\mathcal{C} \cap e_{\{p,q\}}| = |e_{\{p,q\}}| = V(\alpha, \beta)$. By (5), $f_p^{\mathcal{C}} = \alpha$ and $f_q^{\mathcal{C}} = \beta$. ■

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

Lemmas 4.1 and 4.3 plus property 4.2 yield




Property 4.2(a) Property 4.2(b) Property 4.2(c,d)

Figure 3: 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 $\mathcal{G}(\mathcal{C}) = \langle \mathcal{V}, \mathcal{E} - \mathcal{C} \rangle$.

**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 to be cut. We now compute the cost of a cut $\mathcal{C}$, which is

$$|\mathcal{C}| \;=\; \sum_{p \in \mathcal{P}_{\alpha\beta}} |\mathcal{C} \cap \{t_p^\alpha, t_p^\beta\}| \;+\; \sum_{\substack{\{p,q\} \in \mathcal{N} \\ \{p,q\} \subset \mathcal{P}_{\alpha\beta}}} |\mathcal{C} \cap e_{\{p,q\}}|. \tag{6}$$

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

$$\begin{aligned} |\mathcal{C} \cap \{t_p^\alpha, t_p^\beta\}| \;&=\; \begin{cases} |t_p^\alpha| & \text{if } t_p^\alpha \in \mathcal{C} \\ |t_p^\beta| & \text{if } t_p^\beta \in \mathcal{C} \end{cases} \\ &=\; D_p(f_p^{\mathcal{C}}) + \sum_{\substack{q \in \mathcal{N}_p \\ q \notin \mathcal{P}_{\alpha\beta}}} V(f_p^{\mathcal{C}}, f_q). \end{aligned}$$

Lemma 4.3 gives the second term in (6). Thus, the total cost of a cut $\mathcal{C}$ is

$$\begin{aligned} |\mathcal{C}| \;&=\; \sum_{p \in \mathcal{P}_{\alpha\beta}} D_p(f_p^{\mathcal{C}}) + \sum_{p \in \mathcal{P}_{\alpha\beta}} \sum_{\substack{q \in \mathcal{N}_p \\ q \notin \mathcal{P}_{\alpha\beta}}} V(f_p^{\mathcal{C}}, f_q) \\ &\quad + \sum_{\substack{\{p,q\} \in \mathcal{N} \\ \{p,q\} \subset \mathcal{P}_{\alpha\beta}}} V(f_p^{\mathcal{C}}, f_q^{\mathcal{C}}) \\ &=\; \sum_{p \in \mathcal{P}_{\alpha\beta}} D_p(f_p^{\mathcal{C}}) \;+\; \sum_{\substack{\{p,q\} \in \mathcal{N} \\ p \text{ or } q \in \mathcal{P}_{\alpha\beta}}} V(f_p^{\mathcal{C}}, f_q^{\mathcal{C}}). \end{aligned}$$

This can be rewritten as $|\mathcal{C}| = E(f^{\mathcal{C}}) - K$ where

$$K \;=\; \sum_{p \notin \mathcal{P}_{\alpha\beta}} D_p(f_p) \;+\; \sum_{\substack{\{p,q\} \in \mathcal{N} \\ \{p,q\} \cap \mathcal{P}_{\alpha\beta} = \emptyset}} V(f_p, f_q)$$

is the same constant for all cuts $\mathcal{C}$. ■

**Corollary 4.5** *The optimal $\alpha$-$\beta$ swap from $f$ is $\hat{f} = f^{\mathcal{C}}$ where $\mathcal{C}$ is the minimum cut on $\mathcal{G}_{\alpha\beta}$.*

# 5 Finding the optimal expansion move

Given an input labeling $f$ (partition $\mathbf{P}$) 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 algorithm given at the bottom of Figure 1. 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 $\mathcal{G}_\alpha = \langle \mathcal{V}_\alpha, \mathcal{E}_\alpha \rangle$. The structure of this graph is determined by the current partition $\mathbf{P}$ 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 $\mathcal{G}_\alpha$ for a given $f$ (or $\mathbf{P}$) and $\alpha$. We show that cuts $\mathcal{C}$ on $\mathcal{G}_\alpha$ correspond in a natural way to labelings $f^{\mathcal{C}}$ which are within one $\alpha$-expansion move 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 $|\mathcal{C}| = E(f^{\mathcal{C}})$. A corollary from this theorem states our main result that the desired labeling $\hat{f}$ is $f^{\mathcal{C}}$ where $\mathcal{C}$ is a minimum cut on $\mathcal{G}_\alpha$.

The structure of the graph is illustrated in Figure 4. For legibility, this figure shows the case of 1D image. The set of vertices includes the two terminals $\alpha$ and $\bar{\alpha}$, as well as all image pixels $p \in \mathcal{P}$. In addition, for each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ separated in the current partition (i.e. $f_p \neq f_q$) we create an *auxiliary vertex* $a_{\{p,q\}}$. Auxiliary nodes are introduced at the boundaries between partition sets $\mathcal{P}_l$ for $l \in \mathcal{L}$. Thus, the set of vertices is

$$\mathcal{V}_\alpha \;=\; \{\; \alpha \,,\, \bar{\alpha} \,,\, \mathcal{P} \,,\, \bigcup_{\substack{\{p,q\} \in \mathcal{N} \\ f_p \neq f_q}} a_{\{p,q\}} \;\}.$$

Each pixel $p \in \mathcal{P}$ is connected to the terminals $\alpha$ and $\bar{\alpha}$ by $t$-links $t_p^\alpha$ and $t_p^{\bar{\alpha}}$, correspondingly. Each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ which are not separated by the partition $\mathbf{P}$ (i.e. $f_p = f_q$) is connected by an $n$-link $e_{\{p,q\}}$. For each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ such

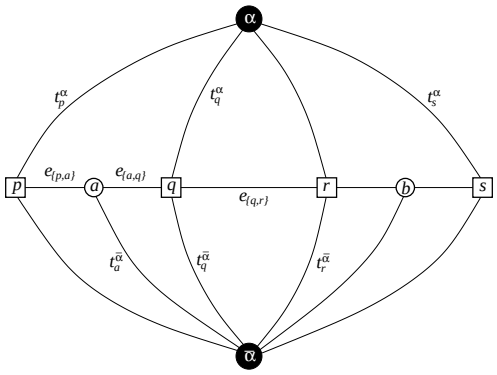



Figure 4: An example of $\mathcal{G}_\alpha$ for a 1D image. The set of pixels in the image is $\mathcal{P} = \{p, q, r, s\}$ and the current partition is $\mathbf{P} = \{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_\alpha\}$ where $\mathcal{P}_1 = \{p\}$, $\mathcal{P}_2 = \{q, r\}$, and $\mathcal{P}_\alpha = \{s\}$. 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 $\mathcal{P}_l$.

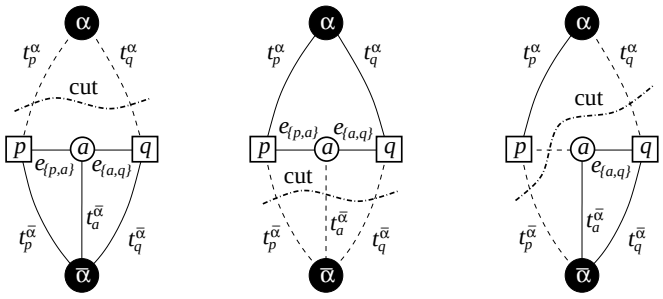
that $f_p \neq f_q$ we create a triplet of edges $\mathcal{E}_{\{p,q\}} = \left\{e_{\{p,a\}},\ e_{\{a,q\}},\ t_a^{\bar\alpha}\right\}$ where $a = a_{\{p,q\}}$ is the corresponding auxiliary node. The edges $e_{\{p,a\}}$ and $e_{\{a,q\}}$ connect pixels $p$ and $q$ to $a_{\{p,q\}}$ and the $t$-link $t_a^{\bar\alpha}$ connects the auxiliary node $a_{\{p,q\}}$ to the terminal $\bar\alpha$. So we can write the set of all edges as

$$\mathcal{E}_\alpha \ = \ \{ \ \bigcup_{p\in\mathcal{P}} \{t_p^\alpha, t_p^{\bar\alpha}\} \ , \ \bigcup_{\substack{\{p,q\}\in\mathcal{N}\\ f_p\neq f_q}} \mathcal{E}_{\{p,q\}} \ , \ \bigcup_{\substack{\{p,q\}\in\mathcal{N}\\ f_p= f_q}} e_{\{p,q\}} \ \}.$$

The weights assigned to the edges are

| **edge** | **weight** | **for** |
|---|---|---|
| $t_p^{\bar\alpha}$ | $\infty$ | $p \in \mathcal{P}_\alpha$ |
| $t_p^{\bar\alpha}$ | $D_p(f_p)$ | $p \notin \mathcal{P}_\alpha$ |
| $t_p^\alpha$ | $D_p(\alpha)$ | $p \in \mathcal{P}$ |
| $e_{\{p,a\}}$ | $V(f_p, \alpha)$ | $\{p,q\} \in \mathcal{N},\ f_p \neq f_q$ |
| $e_{\{a,q\}}$ | $V(\alpha, f_q)$ | |
| $t_a^{\bar\alpha}$ | $V(f_p, f_q)$ | |
| $e_{\{p,q\}}$ | $V(f_p, \alpha)$ | $\{p,q\} \in \mathcal{N},\ f_p = f_q$ |

As in section 4, any cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ must sever (include) exactly one $t$-link for any pixel




Property 5.2(a) Property 5.2(b) Property 5.2(c,d)

Figure 5: Properties of a minimum cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ for two pixel $p, q \in \mathcal{N}$ such that $f_p \neq f_q$. Dotted lines show the edges cut by $\mathcal{C}$ and solid lines show the edges in the induced graph $\mathcal{G}(\mathcal{C}) = \langle \mathcal{V}, \mathcal{E} - \mathcal{C} \rangle$.

$p \in \mathcal{P}$. This defines a natural labeling $f^{\mathcal{C}}$ corresponding to a cut $\mathcal{C}$ on $\mathcal{G}_\alpha$. Formally,

$$f_p^{\mathcal{C}} = \begin{cases} \alpha & \text{if } \; t_p^\alpha \in \mathcal{C} \\ f_p & \text{if } \; t_p^{\bar{\alpha}} \in \mathcal{C} \end{cases} \qquad \forall p \in \mathcal{P}. \tag{7}$$

In other words, a pixel $p$ is assigned label $\alpha$ if the cut $\mathcal{C}$ separates $p$ from the terminal $\alpha$ and, $p$ is assigned its old label $f_p$ if $\mathcal{C}$ separates $p$ from $\bar{\alpha}$. Note that for $p \notin \mathcal{P}_\alpha$ the terminal $\bar{\alpha}$ represents labels assigned to pixels in the initial labeling $f$. Clearly we have

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

It is also easy to show that a cut $\mathcal{C}$ 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 $\mathcal{C}$ 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 1($\bar{\alpha}$). Analogously, we can show that Property 4.2 and equation (7) establish Lemma 4.3 for the $n$-links $e_{\{p,q\}}$ in $\mathcal{G}_\alpha$.

Consider now the set of edges $\mathcal{E}_{\{p,q\}}$ 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 $\mathcal{C}$ on $\mathcal{G}_\alpha$ is guaranteed to sever the edges in $\mathcal{E}_{\{p,q\}}$ 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.

**Property 5.2** *A minimum cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ satisfies:*

$$\begin{array}{llll} a) & \textit{If} \quad t_p^\alpha, t_q^\alpha \in \mathcal{C} & \textit{then} & \mathcal{C} \cap \mathcal{E}_{\{p,q\}} = \emptyset. \\ b) & \textit{If} \quad t_p^{\bar{\alpha}}, t_q^{\bar{\alpha}} \in \mathcal{C} & \textit{then} & \mathcal{C} \cap \mathcal{E}_{\{p,q\}} = t_a^{\bar{\alpha}}. \\ c) & \textit{If} \quad t_p^{\bar{\alpha}}, t_q^{\alpha} \in \mathcal{C} & \textit{then} & \mathcal{C} \cap \mathcal{E}_{\{p,q\}} = e_{\{p,a\}}. \\ d) & \textit{If} \quad t_p^{\alpha}, t_q^{\bar{\alpha}} \in \mathcal{C} & \textit{then} & \mathcal{C} \cap \mathcal{E}_{\{p,q\}} = e_{\{a,q\}}. \end{array}$$

Property (a) results from the fact that no subset of $\mathcal{C}$ is a cut. The others follow from the minimality of $|\mathcal{C}|$ and the fact that $|e_{\{p,a\}}|$, $|e_{\{a,q\}}|$ and $|t_a^{\bar{\alpha}}|$ 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 5.

**Lemma 5.3** *If $\{p,q\} \in \mathcal{N}$ and $f_p \neq f_q$ then the minimum cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ satisfies $|\mathcal{C} \cap \mathcal{E}_{\{p,q\}}| = V(f_p^{\mathcal{C}}, f_q^{\mathcal{C}})$.*

PROOF: The equation follows from property 5.2, equation (7), and the edge weights. For example, if $t_p^{\bar{\alpha}}, t_q^{\bar{\alpha}} \in \mathcal{C}$ then $|\mathcal{C} \cap \mathcal{E}_{\{p,q\}}| = |t_a^{\bar{\alpha}}| = V(f_p, f_q)$. At the same time, (7) implies that $f_p^{\mathcal{C}} = f_p$ and $f_q^{\mathcal{C}} = f_q$. ■

Note that the right penalty $V$ is imposed whenever $f_p^{\mathcal{C}} \neq f_q^{\mathcal{C}}$, due to the auxiliary pixel construction.

Property 1($\bar{\alpha}$) 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 $\mathcal{G}_\alpha$ to be a cut that satisfies Properties 1($\bar{\alpha}$) and 5.2.

**Theorem 5.4** *Let $\mathcal{G}_\alpha$ be constructed as above given $f$ and $\alpha$. Then there is a one to one correspondence between elementary cuts on $\mathcal{G}_\alpha$ and labelings within one $\alpha$-expansion of $f$. Moreover, for any elementary cut $\mathcal{C}$ we have $|\mathcal{C}| = E(f^{\mathcal{C}})$.*

PROOF: We first show that an elementary cut $\mathcal{C}$ is uniquely determined by the corresponding labeling $f^{\mathcal{C}}$. The label $f_p^{\mathcal{C}}$ at the pixel $p$ determines which of the $t$-links to $p$ is in $\mathcal{C}$. 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 $\mathcal{E}_{\{p,q\}}$ corresponding to $\{p,q\} \in \mathcal{N}$ such that $f_p \neq f_q$ should be cut.

The cost of an elementary cut $\mathcal{C}$ is

$$\begin{aligned} |\mathcal{C}| &= \sum_{p \in \mathcal{P}} |\mathcal{C} \cap \{t_p^\alpha, t_p^{\bar{\alpha}}\}| \\ &+ \sum_{\substack{\{p,q\} \in \mathcal{N} \\ f_p = f_q}} |\mathcal{C} \cap e_{\{p,q\}}| \quad + \sum_{\substack{\{p,q\} \in \mathcal{N} \\ f_p \neq f_q}} |\mathcal{C} \cap \mathcal{E}_{\{p,q\}}|. \end{aligned} \tag{8}$$

It is easy to show that for any pixel $p \in \mathcal{P}$ we have $|\mathcal{C} \cap \{t_p^\alpha, t_p^{\bar{\alpha}}\}| = D_p(f_p^{\mathcal{C}})$. 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 $\mathcal{C}$ is

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

Therefore, $|\mathcal{C}| = E(f^{\mathcal{C}})$. ■

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

**Corollary 5.5** *The optimal $\alpha$ expansion from $f$ is $\hat{f} = f^{\mathcal{C}}$ where $\mathcal{C}$ is the minimum cut on $\mathcal{G}_\alpha$.*

# 6 Optimality properties

## 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 $V$.

**Theorem 6.1** *Let $\hat{f}$ be a minimum when the expansion moves are allowed, $f^*$ be the optimal solution, and $c$ be the ratio of the largest nonzero value of $V$ to the smallest nonzero value of $V$, i.e.*

$$c = \frac{\max_{l_1 \neq l_2 \in \mathcal{L}} V(l_1, l_2)}{\min_{l_1 \neq l_2 \in \mathcal{L}} V(l_1, l_2)}.^4$$

*Then $E(\hat{f}) \leq 2cE(f^*)$.*

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

$$\mathcal{P}_\alpha = \left\{ p \in \mathcal{P} \mid f_p^* = \alpha \right\}$$

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

$$f_p^\alpha = \begin{cases} \alpha & \text{if } p \in \mathcal{P}_\alpha \\ \hat{f}_p & \text{otherwise} \end{cases}$$

[4]Note that $c$ is well defined since $V(\alpha, \beta) > 0$ for $\alpha \neq \beta$. If $V_{p,q}$'s are different for neighboring pairs $p, q$ then $c = \max_{p,q \in \mathcal{N}} \left( \frac{\max_{l_1 \neq l_2 \in \mathcal{L}} V(l_1, l_2)}{\min_{l_1 \neq l_2 \in \mathcal{L}} V(l_1, l_2)} \right)$.

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

$$E(\hat{f}) \leq E(f^{\alpha}). \tag{9}$$

Let $S$ be a set of pixels and pairs of neighboring pixels. If $p$, $q$ denote pixels, define $E_S(f)$ as a restriction of the energy to the set $S$, i.e.:

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

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

$$I = \{p \mid p \in \mathcal{P}_{\alpha}\} \cup \{\{p,q\} \mid \{p,q\} \in \mathcal{N} \text{ and } \{p,q\} \subset \mathcal{P}_{\alpha}\},$$

$$B = \{\{p,q\} \mid \{p,q\} \in \mathcal{N} \text{ and } \{p,q\} \cap \mathcal{P}_{\alpha} \neq \emptyset \text{ and } \{p,q\} \cap (\mathcal{P} - \mathcal{P}_{\alpha}) \neq \emptyset\},$$

$$O = \{p \mid p \notin \mathcal{P}_{\alpha}\} \cup \{\{p,q\} \mid \{p,q\} \in \mathcal{N} \text{ and } \{p,q\} \subset (\mathcal{P} - \mathcal{P}_{\alpha})\}.$$

The following three facts hold:

$$E_O(f^{\alpha}) = E_O(\hat{f}), \tag{10}$$

$$E_I(f^{\alpha}) = E_I(f^*), \tag{11}$$

$$E_B(f^{\alpha}) \leq cE_B(f^*). \tag{12}$$

Equations 10 and 11 are obvious, and equation 12 holds because for any $\{p,q\} \in B$ we have $V(f_p^{\alpha}, f_q^{\alpha}) \leq cV(f_p^*, f_q^*)$.

Since $I \cup B \cup O$ is the set of all pixels and all neighboring pairs of pixels, we can expand both sides of (9) to get:

$$E_I(\hat{f}) + E_B(\hat{f}) + E_O(\hat{f}) \leq E_I(f^{\alpha}) + E_B(f^{\alpha}) + E_O(f^{\alpha})$$

Using fact (10), (11) and (12) we get from the equation above:

$$E_I(\hat{f}) + E_B(\hat{f}) \leq E_I(f^*) + cE_B(f^*). \tag{13}$$

To get the bound on the total energy, we need to sum equation (13) over all labels $\alpha \in \mathcal{L}$:

$$\sum_{\alpha \in \mathcal{L}} \left(E_I(\hat{f}) + E_B(\hat{f})\right) \leq \sum_{\alpha \in \mathcal{L}} \left(E_I(f^*) + cE_B(f^*)\right) \tag{14}$$

(a) Local minimum (b) Global minimum (c) Values of $D_p$

Figure 6: The image consists of pixels 1, 2 and 3. Pixel 1 and 2 are neighbors and pixel 2 and 3 are neighbors. There are three possible labels $a$, $b$, and $c$. $D_p$ 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 is in (b) with the energy of 4.

Let $B = \bigcup_{\alpha \in \mathcal{L}} B$. Observe that for every $\{p,q\} \in B$, $E_{\{p,q\}}(\hat{f})$ appears twice on the left side of (13), once in $E_{B^\alpha}(\hat{f})$ for $\alpha = f_p^*$ and once in $E_{B^\alpha}(\hat{f})$ for $\alpha = f_q^*$. Similarly every $E_{\{p,q\}}(f^*)$ appears 2c times on the right side of (13). Therefore equation (14) can be rewritten to get the bound of $2c$:

$$E(\hat{f}) + E_B(\hat{f}) \leq E(f^*) + (2c-1)E_B(f^*) \leq 2cE(f^*).$$

■

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.[5] For a general metric $V$, they have a bound of $O(\log k \log\log k)$. However, their algorithm uses linear programming, which is impractical for the large number of variables occurring in computer vision.

## 6.2 Approximating a semi-metric

A local minimum when the swap moves are allowed can be arbitrary far from the global minimum. This is illustrated by an example in figure 6.

However we can use the expansion algorithm to get an answer within a factor of $2c$ from the optimum for the energy function (1) when $V$ is a semi-metric[6]. Here $c$ is defined as in theorem 6.1, which is well defined for a semi-metric. Suppose $E = \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{\{p,q\} \in \mathcal{N}} V(f_p, f_q)$ with a semi-metric $V$, and let $r = \max_{\alpha,\beta} V(\alpha,\beta)$. Define a new energy $E_P(f) = \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{\{p,q\} \in \mathcal{N}} r \cdot \delta(f_p \neq f_q)$. If $\hat{f}$ is a local minimum of $E_P$ given the expansion moves and $f^o$ is the global minimum of $E(f)$ then we have the following theorem:

**Theorem 6.2** $E(\hat{f}) \leq 2cE(f^o)$.

[5] In fact, any algorithm that is within a factor of 2 for the Potts model is within a factor of $2c$ for an arbitrary metric $V$.

[6] Actually we just need that $V(\alpha,\beta) \geq 0$ and $V(\alpha,\beta) = 0 \Leftrightarrow \alpha = \beta$.

PROOF: Let $f^*$ be the global minimum of $E_P$. Then

$$E(\hat{f}) \;\leq\; E_P(\hat{f}) \;\leq\; 2E_P(f^*) \;\leq\; 2E_P(f^o) \;\leq\; 2cE(f^o)$$

■

The theorem holds for $r \in [\min_{\alpha\neq\beta} V(\alpha,\beta)\ ,\ \max_{\alpha,\beta} V(\alpha,\beta)]$. 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 $\hat{f}$. The estimate $\hat{f}$ can be used as a good starting point for the swap algorithm.

# 7 The Potts model

An interesting special case of the energy in equation (1) arises when $V$ is given by the Potts model. Potts model was first described by Potts in [27] for statistical mechanics?

$$E_P(f) \quad = \quad \sum_{\{p,q\}\in\mathcal{N}} u_{\{p,q\}} \cdot \delta(f_p \neq f_q) + \sum_{p\in\mathcal{P}} D_p(f_p), \tag{15}$$

In this case, the discontinuities between any pair of labels are penalized equally. This model in some sense is the simplest discontinuity-preserving model and it is especially useful when the labels are unordered or the number of labels is small.

The Potts $V_{p,q} = u_{\{p,q\}} \cdot \delta(f_p \neq f_q)$ is a metric and in this case our expansion algorithm has the approximation of 2. In this section we show that minimizing the energy in (15) is NP-complete. This also implies that minimizing $E(f)$ in equation (1) is NP-hard, since the Potts energy is a special case. The construction is by reducing to and from a multiway cut problem. Multiway cut problem is interesting in its own right and it has good approximation algorithms.

## 7.1 The Potts model and the multiway cut problem

In this section we 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.

Consider a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ with non-negative edge weights, along with a set of terminal vertices $\mathcal{L} \subset \mathcal{V}$. A subset of 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]. The multiway cut problem is a generalization of the standard two-terminal graph cut problem.

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 7 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 $\mathcal{C}$ corresponds to the labeling $f^{\mathcal{C}}$ which assigns the label $l$ to all pixels $p$ which are $t$-linked to the $l$-vertex in $\mathcal{G}(\mathcal{C})$.

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

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

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

When the number of terminals is 2, the multiway cut problem reduces to the standard graph cut problem which can be solved efficiently. Thus when there are just 2 labels, the exact minimum of $E_P(f)$ can be found. This result was first reported by Greig et al. in [16].

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 $\mathcal{C}(l)$ that separates $l$ from all other terminals. This is just the standard graph cut problem. Then the algorithm generates a multiway cut $\mathcal{C} = \cup_{l \neq l_{max}} \mathcal{C}(l)$ where $l_{max} = \arg\max_{l \in \mathcal{L}} |\mathcal{C}(l)|$ 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 $\cup_{l \in \mathcal{L}} \mathcal{C}(l)$ may leave some vertices disconnected

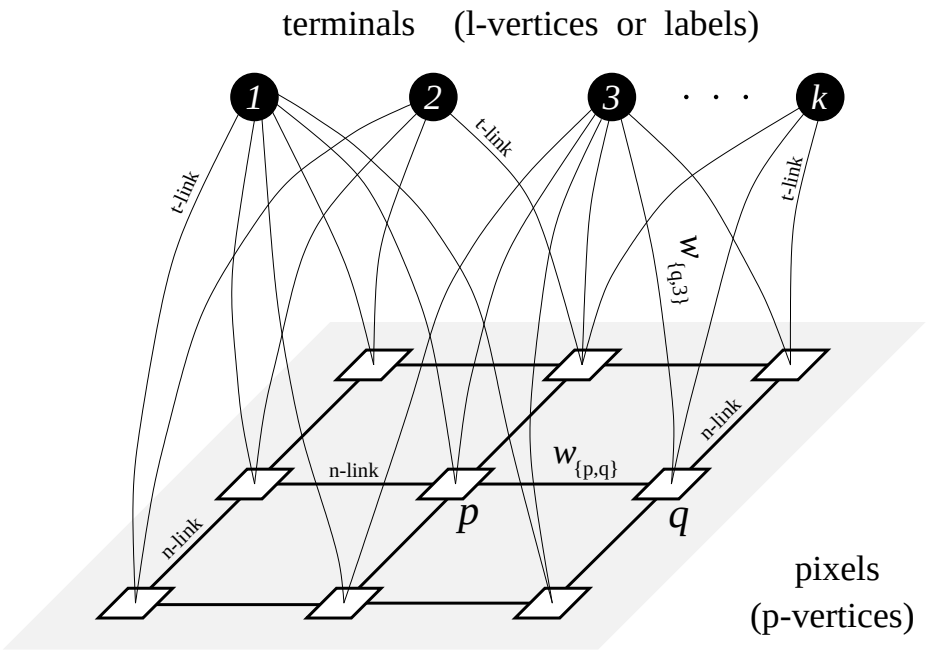



Figure 7: An example of the graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ with terminals $\mathcal{L} = \{1, \ldots, k\}$. The pixels $p \in \mathcal{P}$ are shown as white squares. Each pixel has an $n$-link to its four neighbors. Each pixel is also connected to all terminals by $t$-links (some of the $t$-links are omitted from the drawing for legibility). The set of vertices $\mathcal{V} = \mathcal{P} \cup \mathcal{L}$ includes all pixels and terminals. The set of edges $\mathcal{E} = \mathcal{E}_{\mathcal{N}} \cup \mathcal{E}_{\mathcal{T}}$ consists of all $n$-links and $t$-links.

from any terminal. The multiway cut $\mathcal{C} = \cup_{l \neq l_{max}} \mathcal{C}(l)$ connects all those vertices to the terminal $l_{max}$.

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

## 7.2 Minimizing the Potts energy is NP-hard

We now show that minimizing the Potts energy $E_P(f)$ in (15) is an NP-hard problem.

In the previous section we showed that the problem of minimizing the energy $E_P(f)$ in (15) 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 $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ we will construct an instance of minimizing $E_P(f)$ where the optimal labeling $f^*$ determines a minimum multiway cut on $\mathcal{G}$. 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 $\mathcal{P}$, 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 (15).

Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ be an arbitrary weighted graph with terminal vertices $\{t_1, \ldots, t_k\} \subset \mathcal{V}$ and edge weights $w_{\{p,q\}}$. We will do the energy minimization using $\mathcal{P} = \mathcal{V}$, $\mathcal{N} = \mathcal{E}$, 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

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

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

This shows that minimizing the $E_P(f)$ 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 $D_p(0) = K$) and by taking the edge weights between grid pixels and non-grid pixels to be one. The cost of a feasible labeling will be 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

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 disparity. In stereo the disparity is usually one-dimentional because corresponding points lie along epipolar lines. In motion the disparity is usually a two-dimentional quantity. The disparity varies smoothly everywhere except the object's boundary, and corresponding points are expected to have similar intensities. Thus we can formulate the correspondence problem as the energy minimization problem:

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

Here $\mathcal{P}$ is the set of all pixels in the left 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 for small difference between $I_p$ and $I'_{p+f_p}$ and 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 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

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

Here $Const$ is used to make the measure more robust. For motion we develop a similar technique which can be found in [33].

## 8.2 Static cues

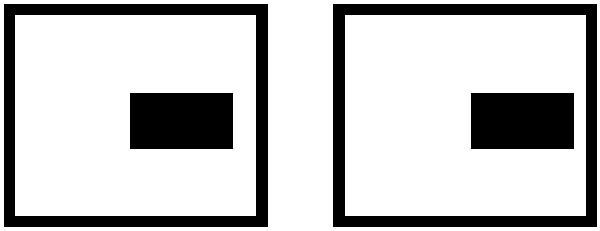
In this section we discuss how to choose different $V_{p,q}$ for each pair of interacting pixels $\{p, q\}$ to take advantage of contextual information. For simplicity we will consider the case of the Potts model, i.e. $V_{p,q} = u_{\{p,q\}} \cdot \delta(f_p \neq f_q)$. The intensities of pixels in the 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 $I(p) \approx I(q)$, where $I(p)$ and $I(q)$ stand for the intensities of pixels $p$ and $q$ in the primary image. Most methods for computing correspondence do not make use of this kind of contextual information. Some exceptions are Poggio et al. [25], Birchfield [5] and Weiss and Adelson [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

$$u_{\{p,q\}} = U(|I_p - I_q|). \tag{16}$$

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 (16) 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.

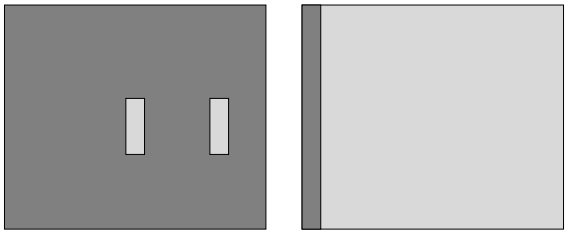
The following example shows the importance of contextual information. Consider the pair of synthetic images below, with a uniformly black rectangle in front of a black and white background.



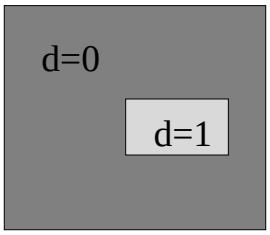
First image Second image

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 $E_{smooth}(f)$ 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 $E_{smooth}(f)$ is minimized at one of the labeling shown in the picture below. Exactly which labeling minimizes $E_{smooth}(f)$ depends on the relationship between the height of the square and the height of the background.



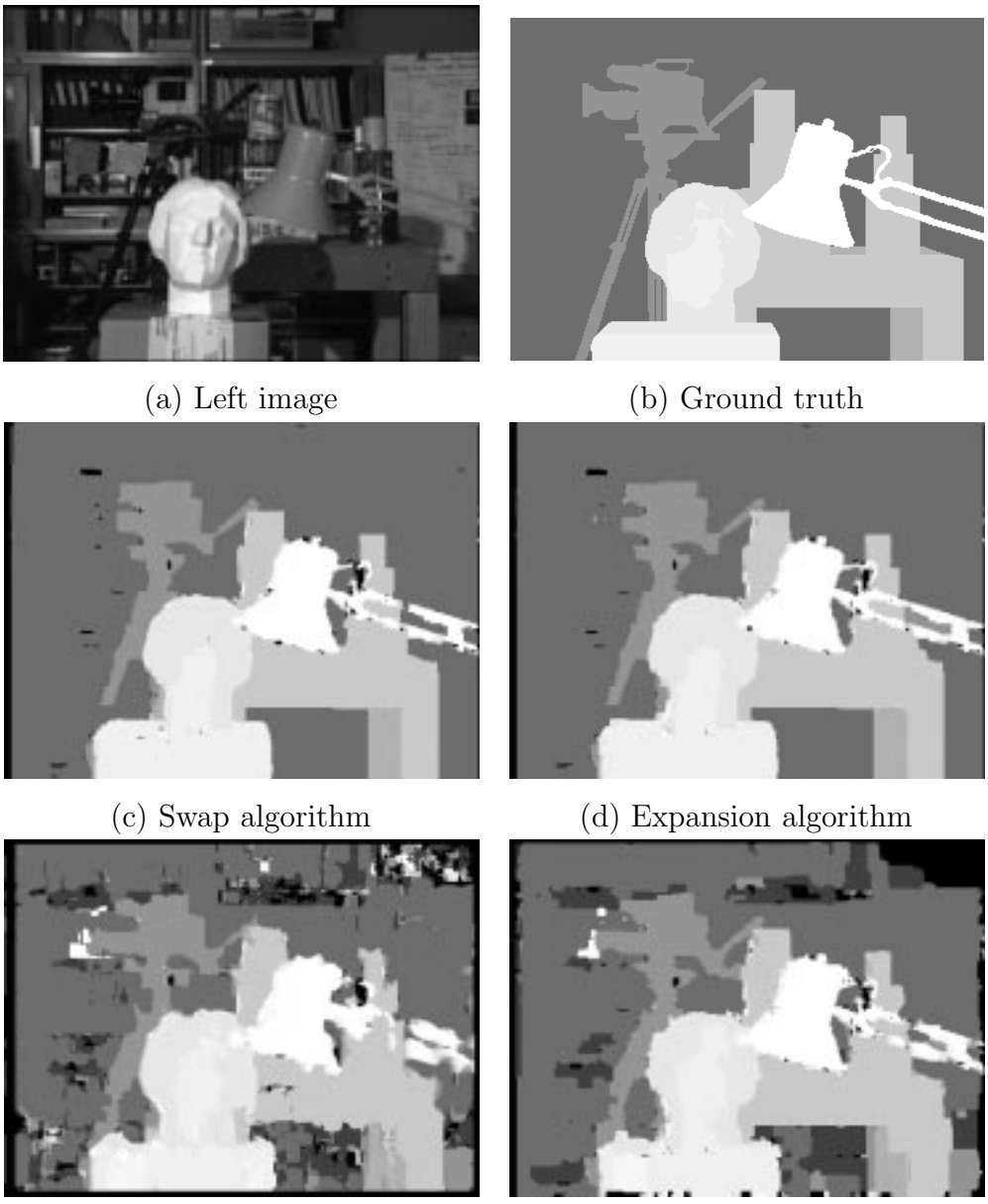
Suppose now that the penalty $u_{\{p,q\}}$ is much smaller if $I_p \neq I_q$ than it is if $I_p = I_q$. In this case the minimum of $E_{smooth}(f)$ is achieved at the disparity configuration shown in the picture below. This result is much closer to human perception.




## 8.3 Real stereo imagery with ground truth

The left image of the real stereo pair with known ground truth is shown in figure 9(a). Figure 9(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



(a) Left image (b) Ground truth

(c) Swap algorithm (d) Expansion algorithm

(e) Normalized correlation (f) Simulated annealing

Figure 8: Real imagery with ground truth

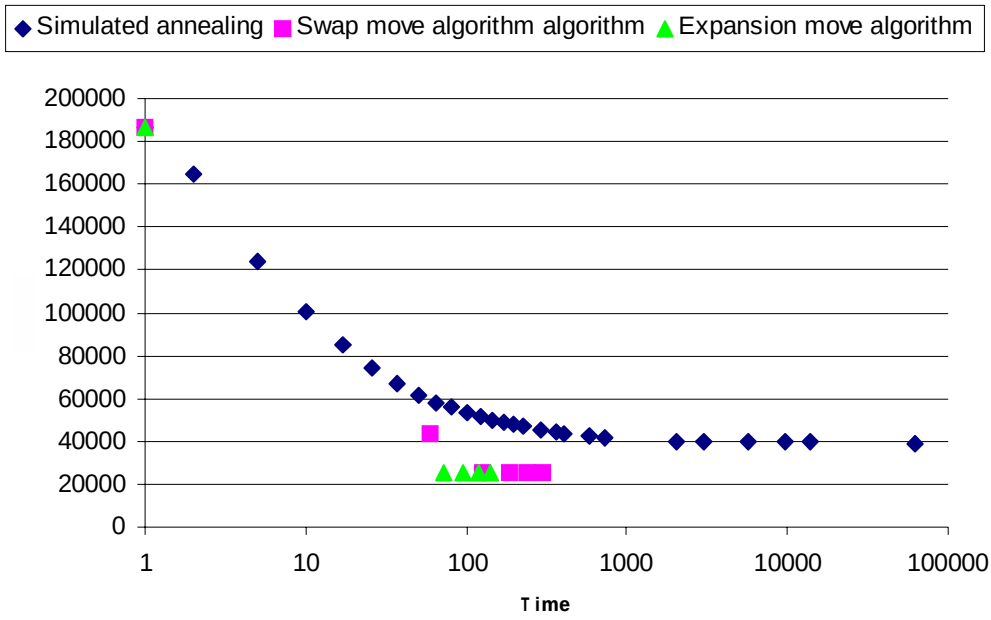



Figure 9: Performance comparison of expansion and swap algorithms with simulated annealing for the problem in figure 8(a).

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 8(c), and (d) show the results of the swap and expansion algorithms for $\lambda = 20$. Figures 8(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 in a random manner. 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 takes 3 times less to converge than the swap algorithm.

| **algorithm** | % **total errors** | % **of errors** $> \pm 1$ | **running time** |
|---|---|---|---|
| expansion algorithm | 7.6 | 2.1 | 106 sec |
| swap algorithm | 7.0 | 2.0 | 300 sec |
| simulated annealing | 20.3 | 5.0 | 1200 sec |
| normalized correlation | 24.7 | 10.0 | 5 sec |

Figure 9 shows the graph of $E_{smooth}$ versus time for our algorithms versus simulated annealing. Notice that the time axis is on the logarithmic scale. We do not show the graph for $E_{data}$ because the difference in the $E_{data}$ term all algorithms achieve is insignificant, as expected from the following argument. Most pixels in real images have nearby pixels with very similar intensities. Thus for most pixels $p$ there are a few disparities $d$ for which $D_p(d)$ is approximately the same and small. For the rest of $d$'s, $D_p(d)$ is quite large. This latter group of disparities are essentially excluded from consideration by energy minimizing algorithms. The remaining choices of $d$ are more or less equally likely. Thus the $E_{data}$ term of the energy function has very similar values for our methods and simulated annealing. Our methods quickly reduce the smoothness energy to around $16,000$, while the best simulated annealing can produce in 19 hours is around $30,000$, nearly twice as bad.

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.

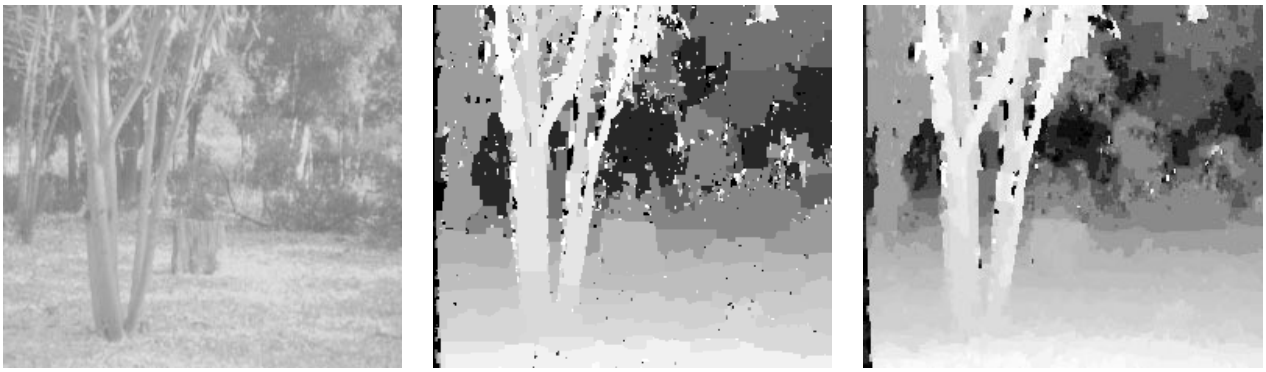
The algorithms appear to be quite stable in the choice of parameter $\lambda$. For example the table in figure 10 gives the errors made by the expansion algorithm for different choices of $\lambda$. For small $\lambda$ the algorithm makes a lot of errors because it overemphasizes the data, for large 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 stereo pair which left image is shown in figure 11(a) the number of disparities is larger, and $E_P$ does not work as well. discussed in chapter 4. For example figure 11(b) and (c)

| $\lambda$ | % **of total errors** | %**of errors** $> \pm 1$ | **Absolute average error** |
|---|---|---|---|
| 1 | 26.6 | 4.5 | 0.40 |
| 5 | 13.0 | 4.5 | 0.27 |
| 10 | 7.0 | 2.3 | 0.15 |
| 20 | 7.6 | 2.1 | 0.15 |
| 30 | 7.9 | 2.3 | 0.17 |
| 50 | 8.8 | 2.3 | 0.18 |
| 100 | 10.4 | 2.9 | 0.21 |
| 500 | 16.3 | 8.2 | 0.37 |

Figure 10: Table of errors for the expansion algorithm for different values of $\lambda$.



(a) Left image (b) Piecewise constant prior Piecewise smooth prior

Figure 11: Results with piecewise constant and piecewise smooth priors.

compares the results of minimizing $E_P$ and $E_L$. Notice that there are fewer disparities found in figure 11(b), since the piecewise constant prior tends to produce large regions with the same disparity.

## 8.5 Motion

Figure 12(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 12(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.

(a) First image (b) Horizontal movement (c) Vertical movement

Figure 12: Moving cat


## 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 MRF's

In this appendix we show that minimization of the energy function in (1) is equivalent to a MAP 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 if:

(i) $P(f) > 0 \qquad \forall f \in \mathcal{F}$.

(ii) $P(f_p | f_{\mathcal{P}-\{p\}}) = P(f_p | f_{N_p})$

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

The easiest way to specify an MRFs is by the joint distribution using Hammersley-Clifford theorem [3]. The 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:

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

where $\mathbf{C}$ is the set of all cliques, $Z$ is the normalizing constant, and $\{V_c(f)\}$ are functions from a labeling to real number, 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:

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

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

$$\arg\max_{f \in \mathcal{F}} p(d|f)p(f) \quad = \quad \arg\min_{f \in \mathcal{F}} \left(-\log p(d|f)p(f)\right)$$

Assume that the observation $d_p$ at each pixel is independent and that

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

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

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

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

# References


[1] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[2] Amir Amini, Terry Weymouth, and Ramesh Jain. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, September 1990.

[3] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, 36:192–236, 1974.

[4] J. Besag. On the statistical analysis of dirty pictures (with discussion). *Journal of the Royal Statistical Society, Series B*, 48(3):259–302, 1986.

[5] S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. *IJCV*, 35(3):1–25, December 1999.

[6] Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, April 1998.

[7] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987.

[8] Andrew Blake. Comparison of the efficiency of deterministic and stochastic algorithms for visual reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(1):2–12, January 1989.

[9] Yuri Boykov, Olga Veksler, and Ramin Zabih. Markov random fields with efficient approximations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–655, 1998.

[10] P.B. Chou and C.M. Brown. The theory and practice of Bayesian image labeling. *International Journal of Computer Vision*, 4(3):185–210, 1990.

[11] E. Dahlhaus, D. S. Johnson, C.H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The complexity of multiway cuts. In *ACM Symposium on Theory of Computing*, pages 241–251, 1992.

[12] P. Ferrari, A. Frigessi, and P. de Sá. Fast approximate maximum a posteriori restoration of multicolour images. *Journal of the Royal Statistical Society, Series B*, 57(3):485–500, 1995.

[13] L. Ford and D. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[14] Davi Geiger and Alan Yuille. A common framework for image segmentation. *International Journal of Computer Vision*, 6(3):227–243, 1991.

[15] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[16] D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51(2):271–279, 1989.

[17] W. Eric L. Grimson and Theo Pavlidis. Discontinuity detection for visual surface reconstruction. *Computer Vision, Graphics and Image Processing*, 30:316–330, 1985.

[18] B. K. P. Horn and B. Schunk. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[19] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *European Conference on Computer Vision*, pages 232–248, 1998.

[20] G. Kant and X. He. Regular edge labeling of 4-connected plane graphs and its applications in graph drawing problems. *Theoretical Computer Science*, 172:175–193, 1997.

[21] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[22] Jon Kleinberg and Eva Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and markov random fields. In *IEEE Symposium on Foundations of Computer Science*, pages 14–24, 1999.

[23] David Lee and Theo Pavlidis. One dimensional regularization with discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):822–829, November 1988.

[24] G. Parisi. *Statistical field theory*. Addison-Wesley, Reading MA, 1988.

[25] T. Poggio, E. Gamble, and J. Little. Parallel integration of vision modules. *Science*, 242:436–440, October 1988. See also E. Gamble and T. Poggio, MIT AI Memo 970.

[26] Tomaso Poggio, Vincent Torre, and Christof Koch. Computational vision and regularization theory. *Nature*, 317:314–319, 1985.

[27] R. Potts. Some generalized order-disorder transformation. *Proceedings of the Cambridge Philosophical Society*, 48:106–109, 1952.

[28] A. Rosenfeld, R.A. Hummel, and S.W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(6):420–433, June 1976.

[29] S. Roy and I. Cox. A maximum-flow formulation of the $n$-camera stereo correspondence problem. In *International Conference on Computer Vision*, 1998.

[30] Rick Szeliski and Ramin Zabih. An experimental comparison of stereo algorithms. In *IEEE Workshop on Vision Algorithms*, September 1999. To appear in *LNCS*.

[31] R.S. Szeliski. Bayesian modeling of uncertainty in low-level vision. *International Journal of Computer Vision*, 5(3):271–302, December 1990.

[32] Demetri Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):413–424, 1986.

[33] Olga Veksler. *Efficient Graph-based Energy Minimization Methods in Computer Vision*. PhD thesis, Cornell University, July 1999.

[34] Y. Weiss and E.H. Adelson. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *CVPR96*, pages 321–326, 1996.