# A New Algorithm for Energy Minimization with Discontinuities

Yuri Boykov, Olga Veksler, and Ramin Zabih

Computer Science Department, Cornell University, Ithaca, NY 14853
`yura@cs.cornell.edu`, `olga@cs.cornell.edu`, `rdz@cs.cornell.edu`

**Abstract.** Many tasks in computer vision involve assigning a label (such as disparity) to every pixel. These tasks can be formulated as energy minimization problems. In this paper, we consider a natural class of energy functions that permits discontinuities. Computing the exact minimum is NP-hard. We have developed a new approximation algorithm based on graph cuts. The solution it generates is guaranteed to be within a factor of 2 of the energy function's global minimum. Our method produces a local minimum with respect to a certain move space. In this move space, a single move is allowed to switch an arbitrary subset of pixels to one common label. If this common label is $\alpha$ then such a move *expands* the domain of $\alpha$ in the image. At each iteration our algorithm efficiently chooses the *expansion* move that gives the largest decrease in the energy. We apply our method to the stereo matching problem, and obtain promising experimental results. Empirically, the new technique outperforms our previous algorithm [6] both in terms of running time and output quality.

## 1 Energy minimization in early vision

Many early vision problems require estimating some spatially varying quantity (such as intensity, disparity or texture) from noisy measurements. Such quantities tend to be piecewise smooth; they vary smoothly at most points, but change dramatically at object boundaries. Every pixel $p \in \mathcal{P}$ must be assigned a label in some 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

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

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, depending upon the exact vision problem. 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 the image restoration problem, for example, usually $D_p(f_p) = (f_p - i_p)^2$, 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 regularization-based vision [2,12,15], $E_{smooth}$ 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 [11,14,19]). Geman and Geman's seminal paper [9] gave a Bayesian interpretation of many energy functions, and proposed a discontinuity-preserving energy function based on Markov Random Fields (MRF's).

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. There have been numerous attempts to design fast algorithms for energy minimization; we will review this area in section 2. However, as a practical matter the computational problem remains unresolved.

In this paper we address a class of discontinuity-preserving energy functions. Let the neighborhood system $\mathcal{N}$ denote the set of pairs of adjacent pixels in $\mathcal{P}$. We consider functions of the form

$$E_P(f) \quad = \quad \sum_{\{p,q\}\in\mathcal{N}} u_{\{p,q\}} \cdot \delta(f_p \neq f_q) \quad + \quad \sum_{p\in\mathcal{P}} D_p(f_p), \tag{1}$$

where

$$\delta(f_p \neq f_q) \quad = \quad \begin{cases} 1 & \text{if } f_p \neq f_q, \\ 0 & \text{otherwise.} \end{cases}$$

We allow $D_p$ to be an arbitrary function, as long as it is non-negative and finite.[1]

This energy function is in some sense the simplest energy that preserves discontinuities. The smoothness term provides a penalty $u_{\{p,q\}} \geq 0$ for assigning different labels to two adjacent pixels $\{p,q\}$. This penalty does not depend on the labels assigned, as long as they are different. Such energy functions naturally arise from a particular MRF that we call a generalized Potts model; this derivation is given in [6]. We will therefore refer to the energy function $E_P$ given in equation (1) as the *Potts energy.*

In early vision, there are a few energy functions whose global minimum can be rapidly computed [6,10,13]. Unfortunately, we have shown in [6] that minimizing the Potts energy is NP-hard, so it very likely requires exponential time. In this paper we introduce a new approximation algorithm for this energy minimization problem, and apply it to several vision problems. The key properties of our algorithm are that it produces a local minimum in a certain move space, and that the resulting labeling is guaranteed to be within a factor of 2 of the global minimum of the Potts energy.

[1]Our results do not require that $D_p$ be finite, but this assumption simplifies the presentation considerably.

We begin with a brief survey of energy minimization methods in computer vision. In section 3 we give an overview of our approach to energy minimization. We define *expansion* moves and prove that a local minimum of the Potts energy with respect to such moves is within a factor of two of the global minimum. Section 4 gives the details of a graph cut technique that efficiently computes the expansion move producing the largest decrease in the Potts energy. Intuitively, this gives the direction of the steepest descent from a current solution. By using this technique iteratively we follow the "fastest" way into a local minimum of the Potts energy with respect to expansion move space. In section 5 we provide some experimental results on the stereo matching problem.

Note that in our earlier work [6] we presented a similar greedy descent algorithm for approximate Potts energy minimization based on *swap* moves. Strictly speaking, expansion moves and swap moves are not directly comparable since there are swap moves that are not expansion moves and vice versa. However, we have both theoretical and experimental evidence that the expansion move algorithm is superior. Theoretically, we will show in section 3.3 that a local minimum in terms of expansion moves is within a factor of 2 of the global minimum, while no such result is available for the swap move algorithm. Experimentally, the results in section 5 suggest that the expansion moves algorithm leads to a better and faster optimization of the Potts energy.

## 2 Related work

Energy minimization is quite popular in computer vision, and a wide variety of methods have been used. An exhaustive survey is beyond the scope of this paper; however, we will briefly describe the energy minimization methods that are most prevalent in vision.

### 2.1 Global energy minimization

The problem of finding the global minimum of an arbitrary energy function is obviously intractable (it includes the Potts energy minimization problem as a special case). As a consequence, any general-purpose energy minimization algorithm will require exponential time to find the global minimum, unless P=NP.

Simulated annealing was popularized for vision by [9], and is the only general-purpose global energy minimization method in widespread use. With certain annealing schedules, annealing can be guaranteed to find the global minimum. Unfortunately, the schedules that lead to this guarantee are extremely slow. In practice, annealing is inefficient partly because at each step it changes the value of a single pixel.

Graph cuts can be used to find the global minimum of certain energy functions. These algorithms permit $D_p$ to be arbitrary. [10] addressed the case of $|L| = 2$. This result was generalized by [6, 13] to handle label sets of arbitrary size, when the smoothness energy is of the form $\sum_{\{p,q\}\in\mathcal{N}} |f_p - f_q|$. This smoothness energy, unfortunately, leads to oversmoothing at object boundaries. In addition, there must be a natural isomorphism between the label set $L$ and the integers $\{1, 2, \ldots, k\}$. This rules out some significant problems such as motion.

Another alternative is to use methods that have optimality guarantees in certain cases. Continuation methods, such as GNC [4], are the best-known example. These methods involve approximating an intractable (non-convex) energy function by a sequence of energy functions, beginning with a tractable (convex) approximation. At every step in the approximation, a local minimum is found using the solution from the previous step as the starting point. There are circumstances where these methods are known to compute the optimal solution (see [4] 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.

## 2.2 Local energy minimization

Due to the inefficiency of computing a global minimum, many authors have opted for a local minimum. One problem with this 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 energy function, or to the fact the answer is far from the global minimum. There is no obvious way to tell which of these is the problem.[2] Another issue is that local minimization techniques are naturally sensitive to the initial estimate.

There are several ways in which a local minimum can be computed. By phrasing the energy minimization problem in continuous terms, variational methods can be applied. These methods were popularized by Horn [12]. Variational techniques use the Euler equations, which are guaranteed to hold at a local minimum (although they may also hold elsewhere). A number of methods have been proposed to speed up the convergence of the resulting numerical problems, including (for example) multigrid techniques [18]. To apply these algorithms to actual imagery, of course, requires discretization. An alternative is to use discrete relaxation methods; this has been done by many authors, including [7, 16, 17].

It is important to note that a local minimum is defined relative to a set of allowed moves. Most existing minimization algorithms find a local minimum relative to "small" moves, which typically are defined in terms of the $L_2$ distance. To be precise, they attempt to compute a labeling $f$ such that $f = \arg\min_{|f-f'|<\epsilon} E_P(f')$, for some small $\epsilon$.

In [6] we described an algorithm for approximate minimization of the Potts energy based on *swap* moves. For a fixed pair of labels $\alpha, \beta$, this move swaps the labels between a subset of pixels labeled $\alpha$ and another subset labeled $\beta$. The

[2]In the special cases where the global minimum can be rapidly computed, it is possible to separate these issues. For example, [10] points out that the global minimum of an Ising energy function is not necessarily the desired solution for image restoration. [5, 10] analyze the performance of simulated annealing in cases with a known global minimum.

algorithm in [6] is based on a graph cut technique that efficiently computes the best $\alpha, \beta$-swap move from a current solution. By iterating over all distinct pairs $\alpha, \beta$ this technique enables the steepest descent search of the local minimum of the Potts energy with respect to swap moves. The properties of such a local minimum are based on the strength of swap moves.

In this paper we describe a new algorithm based on *expansion* moves. The structure of the algorithm is similar to [6]. It is still the greediest descent into a local minimum with respect to a certain move space. However, there are several important differences. Most importantly, the new algorithm produces a local minimum that is guaranteed to be within a factor of 2 from the global minimum. Such a bound is not available for the swap move algorithm in [6]. Moreover, the steepest descent in the new move space requires iterating over distinct labels, not pairs of labels. Altogether this suggest that the new algorithm can potentially produce faster and better solutions. The data we present in section 5 supports this conclusion.

## 3 The expansion move algorithm

Here we describe the algorithm for approximate minimization of the Potts energy $E_P$ based on expansion moves. In this section, we discuss the expansion moves, which are best described in terms of partitions. We sketch the algorithm and list its basic properties. Then we introduce the notion of a graph cut, which is the basis for our method.

### 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 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 $\alpha$.

Note that a move which gives an arbitrary label $\alpha$ to a single pixel is an $\alpha$-expansion. As a consequence, the standard move space used in annealing is a special case of our move space.

### 3.2 Algorithm and properties

The structure of the expansion move algorithm is shown in figure 1. 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 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 algorithm stops after the first unsuccessful cycle since no further improvement is possible.

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

Fig. 1: The expansion move algorithm.

The algorithm have a number of important properties.

- Obviously, a cycle takes $|\mathcal{L}|$ iterations. Note that a cycle in the swap move algorithm [6] takes $|\mathcal{L}|^2$ iterations.
- The algorithm is guaranteed to terminate in a finite number of cycles, although there is no bound beyond the trivial one of $|\mathcal{L}|^{|\mathcal{P}|}$. Nevertheless, in the applications we have considered the algorithm stops after a few cycles. Moreover, most of the improvements occur during the first cycle, as we will show in section 5.
- Once the algorithm has terminated, the energy of the resulting labeling is a local minimum with respect to an expansion move.

### 3.3 Optimality Guarantees

We now show that if $f^*$ is a local minimum in terms of expansion moves, then $E_P(f^*) \leq 2 \cdot E_P(f^o)$, where $f^o$ is the optimal solution minimizing the Potts energy $E_P$. Let $\mathbf{P}^o = \{\mathcal{P}^o_\alpha \mid \alpha \in \mathcal{L}\}$ be a partition corresponding to $f^o$ so that

$$\mathcal{P}^o_\alpha = \big\{p \in \mathcal{P} \mid f^o_p = \alpha\big\}$$

is a set of pixels assigned to $\alpha$ in the optimal solution. We can produce a labeling $f^\alpha$ within one $\alpha$-expansion move from $f^*$ as follows:

$$f^\alpha_p = \begin{cases} \alpha & \text{if } p \in \mathcal{P}^o_\alpha \\ f^*_p & \text{if } p \notin \mathcal{P}^o_\alpha \end{cases}$$

The key observation is that since $f^*$ is a local minimum in the expansion move space then for any $\alpha \in \mathcal{L}$

$$E_P(f^*) \leq E_P(f^\alpha). \tag{2}$$

For a given label $\alpha \in \mathcal{L}$ we can split the Potts energy of any labeling $f$ into three terms $E_P(f) = E^\alpha_{in}(f) + E^\alpha_{bd}(f) + E^\alpha_{ex}(f)$ where

$$\begin{aligned}
E^\alpha_{in}(f) &= \sum_{\substack{\{p,q\}\in\mathcal{N}\\ p,q\in\mathcal{P}^o_\alpha}} u_{\{p,q\}} \cdot \delta(f_p \neq f_q) \;+\; \sum_{p\in\mathcal{P}^o_\alpha} D_p(f_p)\\
E^\alpha_{bd}(f) &= \sum_{\substack{\{p,q\}\in\mathcal{N}\\ p\in\mathcal{P}^o_\alpha, q\notin\mathcal{P}^o_\alpha}} u_{\{p,q\}} \cdot \delta(f_p \neq f_q)\\
E^\alpha_{ex}(f) &= \sum_{\substack{\{p,q\}\in\mathcal{N}\\ p,q\notin\mathcal{P}^o_\alpha}} u_{\{p,q\}} \cdot \delta(f_p \neq f_q) \;+\; \sum_{p\notin\mathcal{P}^o_\alpha} D_p(f_p)
\end{aligned}$$

correspond to the parts of the Potts energy $E_P(f)$ concentrated at the pixels inside $\mathcal{P}^o_\alpha$, at the boundary of $\mathcal{P}^o_\alpha$, and at the pixels outside of $\mathcal{P}^o_\alpha$, correspondingly.

Since $f^*_p = f^\alpha_p$ for any $p \notin \mathcal{P}^o_\alpha$ then $E^\alpha_{ex}(f^*) = E^\alpha_{ex}(f^\alpha)$. Thus, (2) implies that for any $\alpha \in \mathcal{L}$

$$E^\alpha_{in}(f^*) + E^\alpha_{bd}(f^*) \leq E^\alpha_{in}(f^\alpha) + E^\alpha_{bd}(f^\alpha). \tag{3}$$

Since $f^\alpha_p = f^o_p = \alpha$ for any $p \in \mathcal{P}^o_\alpha$ then $E^\alpha_{in}(f^\alpha) = E^\alpha_{in}(f^o)$. Moreover,

$$E^\alpha_{bd}(f^\alpha) \leq \sum_{\substack{\{p,q\}\in\mathcal{N}\\ p\in\mathcal{P}^o_\alpha, q\notin\mathcal{P}^o_\alpha}} u_{\{p,q\}} = E^\alpha_{bd}(f^o).$$

Therefore, (3) implies that for any $\alpha \in \mathcal{L}$

$$E^\alpha_{in}(f^*) + E^\alpha_{bd}(f^*) \leq E^\alpha_{in}(f^o) + E^\alpha_{bd}(f^o). \tag{4}$$

Summing up inequality (4) over all labels $\alpha \in \mathcal{L}$ we obtain

$$E_P(f^*) + \sum_{\{p,q\}\in B} u_{\{p,q\}} \cdot \delta(f^*_p \neq f^*_q) \leq E_P(f^o) + \sum_{\{p,q\}\in B} u_{\{p,q\}} \tag{5}$$

where $B = \{\{p,q\} \in \mathcal{N} \mid f^o_p \neq f^o_q\}$ is a set of all pairs of neighboring pixels disconnected in the optimal solution $f^o$. Note that the summations on both sides of (5) show up because each pair of pixels in $B$ is encountered twice when summing up the terms in (4) over $\alpha \in \mathcal{L}$. Finally, since $\sum_{\{p,q\}\in B} u_{\{p,q\}} \leq E_P(f^o)$ then (5) implies that $E_P(f^*) \leq 2E_P(f^o)$.

### 3.4 Graph cuts

The key part of the algorithm is step 3.1, where graph cuts are used to efficiently find $\hat{f}$. 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 $\mathcal{C}$ is denoted by $|\mathcal{C}|$ and equals the sum of its edge weights.

The minimum cut problem is to find the cut with smallest cost. This problem can be solved very efficiently by computing the maximum flow between the terminals, according to a theorem due to Ford and Fulkerson [8]. There are a large number of fast algorithms for this problem (see [1], for example). The worst case complexity is low-order polynomial; however, in practice the running time is nearly linear.

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 next section describes the details of our graph cut technique that allows efficient implementation of step 3.1.

## 4 Finding the optimal expansion move

Given an input labeling $f$ (partitioning $\mathbf{P}$) and a label $\alpha$, we wish to find a labeling $\hat{f}$ that minimizes $E_P$ over all labelings within one $\alpha$-expansion of $f$. This is the critical step in the algorithm given at the bottom of figure 1. 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 partitioning $\mathbf{P}$ and by the label $\alpha$. 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 1 shows that elementary cuts are in one to one correspondence with the set of labelings that are within one $\alpha$-expansion of $f$, and also that the cost of an elementary cut is $|\mathcal{C}| = E_P(f^{\mathcal{C}})$. 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 $\mathcal{G}_\alpha$.

The structure of the graph is illustrated in Figure 2. For legibility, this figure shows the case of 1D image. In fact, the structure of $\mathcal{G}_\alpha$ will be the same for any 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

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

Each pixel $p \in \mathcal{P}$ is connected to the terminals $\alpha$ and $\bar{\alpha}$ by edges $t_p^\alpha$ and $t_p^{\bar{\alpha}}$, correspondingly. For brevity, we will refer to these edges as $t$-links (terminal

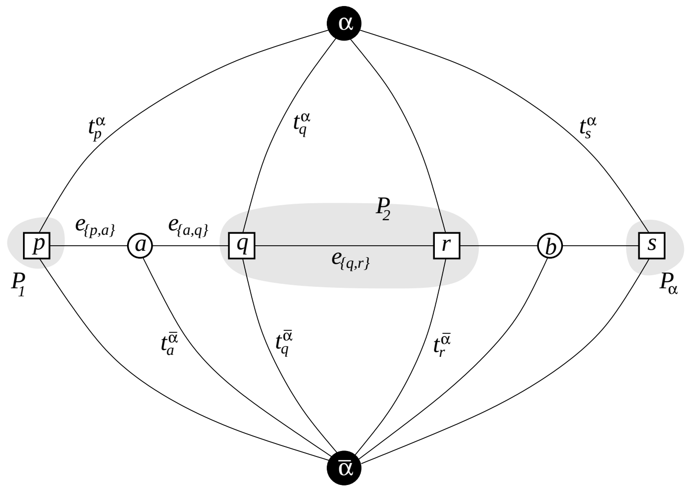


Fig. 2: An example of the graph $\mathcal{G}_\alpha$ for a 1D image. The set of pixels in the image is $\mathcal{P} = \{p, q, r, s\}$ and the current partition is $\mathbf{P} = \{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_\alpha\}$ where $\mathcal{P}_1 = \{p\}$, $\mathcal{P}_2 = \{q, r\}$, and $\mathcal{P}_\alpha = \{s\}$. There are two auxiliary nodes $a = a_{\{p,q\}}$, $b = a_{\{r,s\}}$ introduced between neighboring pixels separated in the current partition. Auxiliary nodes are added at the boundary of sets $\mathcal{P}_l$.

links). Each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ which is not separated by the partition $\mathbf{P}$ (i.e. $f_p = f_q$) is connected by an edge $e_{\{p,q\}}$ which we will call an $n$-link (neighborhood link). For each pair of neighboring pixels $\{p, q\} \in \mathcal{N}$ such that $f_p \neq f_q$ we create a triplet of edges

$$\mathcal{E}_{\{p,q\}} = \left\{ e_{\{p,a\}},\ e_{\{a,q\}},\ t_a^{\bar\alpha} \right\}$$

where $a = a_{\{p,q\}}$ is the corresponding auxiliary node. The $n$-links $e_{\{p,a\}}$ 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 $\bar\alpha$. Finally, we can write the set of all edges as

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

The weights assigned to the edges are shown in the table below.

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

Any cut $\mathcal{C}$ on the graph $\mathcal{G}_\alpha$ must sever (include) exactly one $t$-link for any pixel $p \in \mathcal{P}$: 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 includes either $t_p^{\alpha}$ or $t_p^{\bar\alpha}$ for each pixel $p \in \mathcal{P}$. This defines a natural labeling $f^{\mathcal{C}}$ corresponding to a cut $\mathcal{C}$ on $\mathcal{G}_\alpha$. Formally,

$$f_p^{\mathcal{C}} = \begin{cases} \alpha & \text{if } t_p^{\alpha} \in \mathcal{C} \\ f_p & \text{if } t_p^{\bar\alpha} \in \mathcal{C} \end{cases} \qquad \forall p \in \mathcal{P}. \tag{6}$$

In other words, a pixel $p$ is assigned label $\alpha$ if the cut $\mathcal{C}$ severs $t$-link $t_p^{\alpha}$, while $p$ is assigned its old label $f_p$ if $\mathcal{C}$ severs $t_p^{\bar\alpha}$. The terminal $\alpha$ stands for the new label and the terminal $\bar\alpha$ stands for the old labels assigned to pixels in the initial labeling $f$.

**Lemma 1.** *A cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ corresponds to a labeling $f^{\mathcal{C}}$ which is one $\alpha$-expansion away from the original labeling $f$.*

*Proof.* A cut $\mathcal{C}$ cannot sever $t$-links $t_p^{\bar\alpha}$ for any pixel $p \in \mathcal{P}_\alpha$ due to the infinite cost. Thus, $f_p^{\mathcal{C}} = \alpha$ for any $p \in \mathcal{P}_\alpha$. For any pixel $p \notin \mathcal{P}_\alpha$ the value of $f_p^{\mathcal{C}}$ can be either $\alpha$ or $f_p$. □

It is 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. Formally, for any cut $\mathcal{C}$

**Property 1.** If $t_p^{\bar\alpha}, t_q^{\bar\alpha} \in \mathcal{C}$ then $e_{\{p,q\}} \notin \mathcal{C}$.
**Property 2.** If $t_p^{\alpha}, t_q^{\alpha} \in \mathcal{C}$ then $e_{\{p,q\}} \notin \mathcal{C}$.
**Property 3.1** If $t_p^{\bar\alpha}, t_q^{\alpha} \in \mathcal{C}$ then $e_{\{p,q\}} \in \mathcal{C}$.
**Property 3.2** If $t_p^{\alpha}, t_q^{\bar\alpha} \in \mathcal{C}$ then $e_{\{p,q\}} \in \mathcal{C}$.

The first two properties follow from the requirement that no proper subset of $\mathcal{C}$ should separate the terminals. Properties 3.1 and 3.2 also use the fact that a cut has to separate the terminals.

These properties are illustrated in Figure 3. The following lemma is a consequence of properties 1–3 above and equation 6.

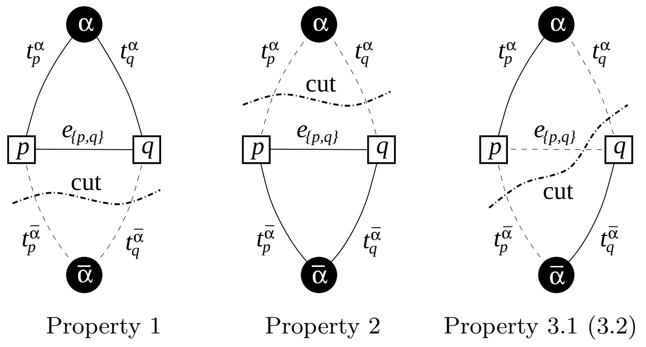


Fig. 3: Properties of a cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ for two pixels $p, q \in \mathcal{N}$ such that $f_p = f_q$. Dotted lines show the edges cut by $\mathcal{C}$ and solid lines show the edges remaining in the induced graph $\mathcal{G}_\alpha(\mathcal{C}) = \langle \mathcal{V}_\alpha, \mathcal{E}_\alpha - \mathcal{C} \rangle$.

**Lemma 2.** *If $\{p, q\} \in \mathcal{N}$ and $f_p = f_q$ then any cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ satisfies*

$$|\mathcal{C} \cap e_{\{p,q\}}| = u_{\{p,q\}} \cdot \delta(f_p^{\mathcal{C}} \neq f_q^{\mathcal{C}}).$$

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 properties 4–6 below. Assume that $a = a_{\{p,q\}}$ is an auxiliary node between the corresponding pair of neighboring pixels. Then a minimum cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ satisfies the following properties.

**Property 4.** If $t_p^{\bar\alpha}, t_q^{\bar\alpha} \in \mathcal{C}$ then $\mathcal{C} \cap \mathcal{E}_{\{p,q\}} = t_a^{\bar\alpha}$.
**Property 5.** If $t_p^{\alpha}, t_q^{\alpha} \in \mathcal{C}$ then $\mathcal{C} \cap \mathcal{E}_{\{p,q\}} = \emptyset$.
**Property 6.1** If $t_p^{\bar\alpha}, t_q^{\alpha} \in \mathcal{C}$ then $\mathcal{C} \cap \mathcal{E}_{\{p,q\}} = e_{\{p,a\}}$.
**Property 6.2** If $t_p^{\alpha}, t_q^{\bar\alpha} \in \mathcal{C}$ then $\mathcal{C} \cap \mathcal{E}_{\{p,q\}} = e_{\{a,q\}}$.

The first property 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 $t_a^{\bar\alpha}$ have identical weights. These properties are illustrated in figure 4.

**Lemma 3.** *If $\{p, q\} \in \mathcal{N}$ and $f_p \neq f_q$ then the minimum cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ satisfies*

$$|\mathcal{C} \cap \mathcal{E}_{\{p,q\}}| = u_{\{p,q\}} \cdot \delta(f_p^{\mathcal{C}} \neq f_q^{\mathcal{C}}).$$

*Proof.* The equation follows from properties 4-6 above and equation (6). Note that $f_p \neq f_q$ implies that $\alpha$ is the only common label that a cut on $\mathcal{G}_\alpha$ can assign to $p$ and $q$ using our convention in (6). That is, $f_p^{\mathcal{C}} = f_q^{\mathcal{C}}$ if and only if both $f_p^{\mathcal{C}} = \alpha$ and $f_q^{\mathcal{C}} = \alpha$. □

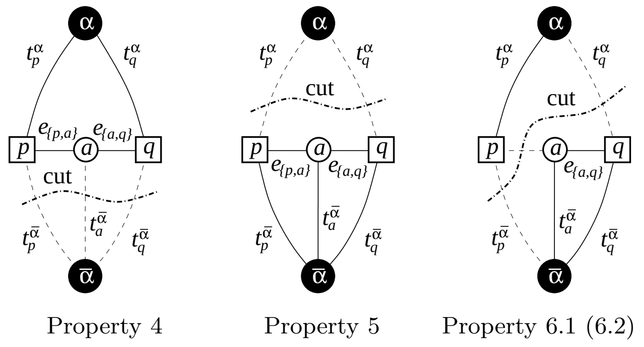


Fig. 4: Properties of a minimum cut $\mathcal{C}$ on $\mathcal{G}_\alpha$ for two pixel $p, q \in \mathcal{N}$ such that $f_p \neq f_q$. Dotted lines show the edges cut by $\mathcal{C}$ and solid lines show the edges in the induced graph $\mathcal{G}_\alpha(\mathcal{C}) = \langle \mathcal{V}_\alpha, \mathcal{E}_\alpha - \mathcal{C}\rangle$.

Note that the penalty $u_{\{p,q\}}$ is imposed whenever $f_p^{\mathcal{C}} \neq f_q^{\mathcal{C}}$. This is exactly what the auxiliary pixel construction was designed for. We had to develop a special trick for the case when the original labels for $p$ and $q$ do not agree ($f_p \neq f_q$) in order to get the same effect that lemma 2 establishes for the simpler situation when $f_p = f_q$.

Properties 1–3 hold for any cut, and properties 4–6 hold for a minimum cut. However, there can be other cuts besides the minimum cut that satisfy all six properties. We will define a *elementary* cut on $\mathcal{G}_\alpha$ to be a cut that satisfies properties 1–6.

**Theorem 1.** *Let the graph $\mathcal{G}_\alpha$ be constructed as above for a given $f$ and $\alpha$. Then there is a one to one correspondence between the set of all elementary cuts on $\mathcal{G}_\alpha$ and the set of all labelings within one $\alpha$-expansion of $f$. Moreover, for any elementary cut $\mathcal{C}$ we have $|\mathcal{C}| = E_P(f^{\mathcal{C}})$.*

*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 $\mathcal{C}$. Properties 1-3 show which $n$-links $e_{\{p,q\}}$ between pairs of neighboring pixels $\{p, q\}$ such that $f_p = f_q$ should be severed. Similarly, properties 4-6 determine 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.

We now compute the cost of an elementary cut $\mathcal{C}$, which is

$$|\mathcal{C}| \;=\; \sum_{p\in\mathcal{P}} |\mathcal{C} \cap \{t_p^\alpha, t_p^{\bar\alpha}\}| \;+\; \sum_{\substack{\{p,q\}\in\mathcal{N}\\ f_p=f_q}} |\mathcal{C} \cap e_{\{p,q\}}| \;+\; \sum_{\substack{\{p,q\}\in\mathcal{N}\\ f_p\neq f_q}} |\mathcal{C} \cap \mathcal{E}_{\{p,q\}}|. \quad (7)$$

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 2 and 3 hold for elementary cuts, since they were based on properties 1-6. These two lemmas give us the second and the third terms in (7). Thus, the total cost of a elementary cut $\mathcal{C}$ is

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

Therefore, $|\mathcal{C}| = E_P(f^{\mathcal{C}})$. □

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

**Corollary 1.** *The optimal $\alpha$-expansion move from $f$ is $\hat{f} = f^{\mathcal{C}}$ where $\mathcal{C}$ is the minimum cut on $\mathcal{G}_\alpha$.*

## 5 Experimental results

In this section we apply our method to the stereo matching problem. We compare our method with simulating annealing, using real image pairs, including one with dense ground truth. For $D_p$ we use the method of [3] to reduce the effects of image sampling. We select $u_{\{p,q\}}$ using the information present in a single image, as described in [6].

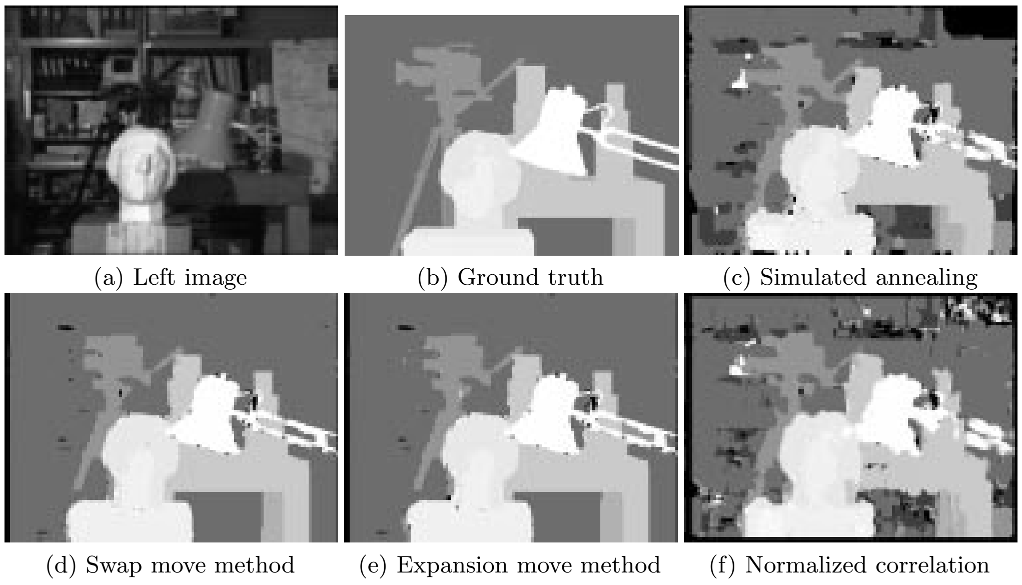
We experimented with several variants of simulated annealing, including both the standard (Metropolis) sampler and the Gibbs sampler. Our comparative data uses the annealing variant and the choice of cooling schedule that best minimized the energy. Simulated annealing is quite sensitive to the starting point, so we initialized it using the results of normalized correlation. Our methods give very similar answers regardless of the starting point, but we used the same starting point as annealing to make the comparison fair. All running times are given in seconds, on a 200 MHz Pentium Pro.

Figure 5(a) shows the left image of a real stereo pair where the ground truth is known at each pixel. We obtained this image pair from the University of Tsukuba Multiview Image Database. The ground truth is shown in figure 5(b). Our results are shown below, both for the expansion move algorithm presented in this paper and the swap move algorithm introduced in [6]. For comparison, we also show the results from simulated annealing, as well as from normalized correlation (using the window size that minimizes the number of errors). Figure 7 shows the performance of algorithms as a function of time, both in terms of energy and in terms of accuracy with respect to the ground truth.

Figure 6 shows the performance of expansion move algorithm on the CMU meter image, along with the results of simulated annealing. The performance in terms of energy is similar to the results shown in figure 7(a).

**Acknowledgements**

We thank J. Kleinberg, D. Shmoys and E. Tardos for insightful remarks on the content of this paper. We are also grateful to Y. Ohta and Y. Nakamura for supplying the ground truth imagery from the University of Tsukuba. This research has been supported by DARPA under contract DAAL01-97-K-0104, and by a grant from Microsoft.


(a) Left image (b) Ground truth (c) Simulated annealing

(d) Swap move method (e) Expansion move method (f) Normalized correlation

Fig. 5: Performance on real imagery with ground truth

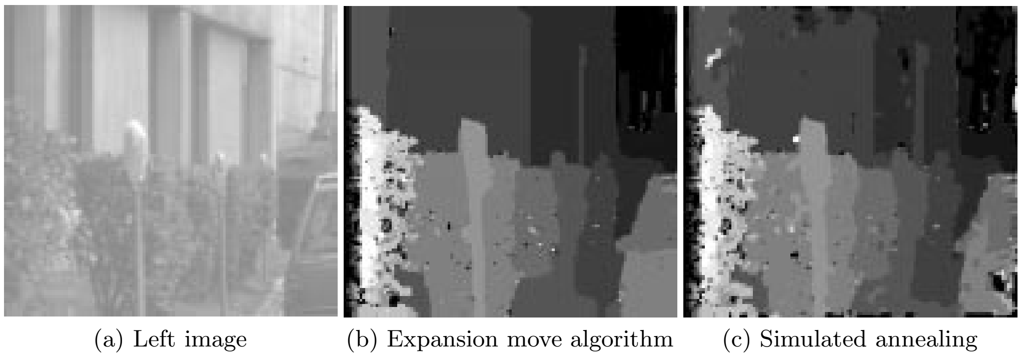
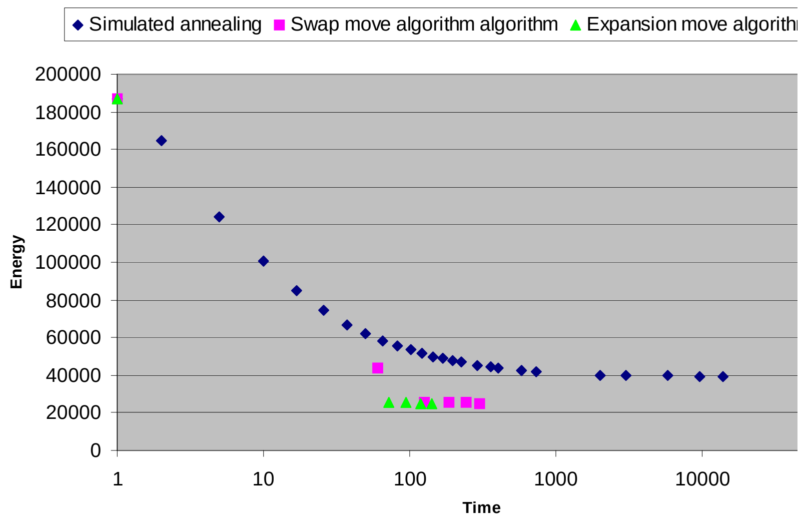

(a) Left image (b) Expansion move algorithm (c) Simulated annealing

Fig. 6: CMU meter imagery results



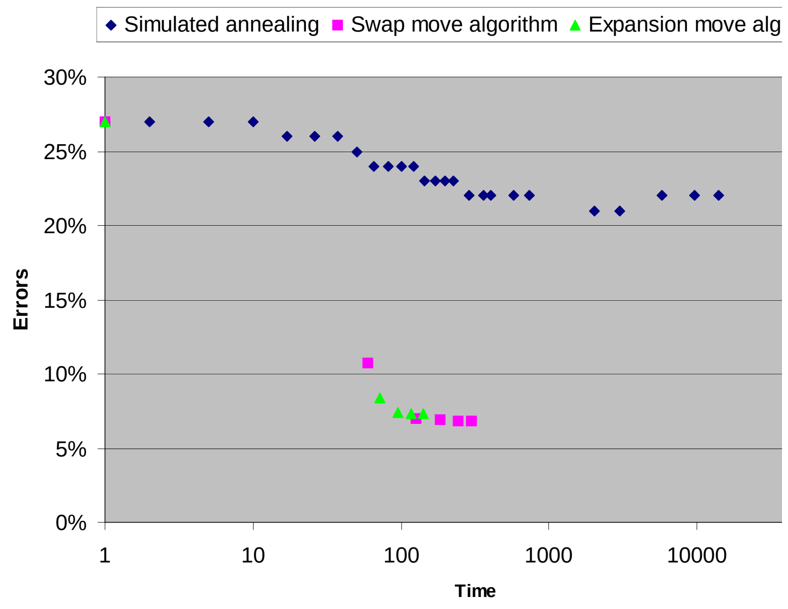


Fig. 7: Performance comparison with simulated annealing, on the imagery shown in figure 5(a). Comparison is done in terms of energy (top) and accuracy with respect to the ground truth (bottom). Each data point for our methods corresponds to a cycle.

## References

1. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
2. Stephen Barnard. Stochastic stereo matching over scale. *International Journal of Computer Vision*, 3(1):17–32, 1989.
3. 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.
4. A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987.
5. 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.
6. Yuri Boykov, Olga Veksler, and Ramin Zabih. Energy minimization with discontinuities. In review. Available from http://www.cs.cornell.edu/home/rdz. An earlier version of this paper appeared in *CVPR '98*.
7. 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.
8. L. Ford and D. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.
9. 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.
10. 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.
11. W. Eric L. Grimson and Theo Pavlidis. Discontinuity detection for visual surface reconstruction. *Computer Vision, Graphics and Image Processing*, 30:316–330, 1985.
12. B. K. P. Horn and B. Schunk. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.
13. H. Ishikawa and D. Geiger. Segmentation by grouping junctions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 125–131, 1998.
14. David Lee and Theo Pavlidis. One dimensional regularization with discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):822–829, November 1988.
15. Tomaso Poggio, Vincent Torre, and Christof Koch. Computational vision and regularization theory. *Nature*, 317:314–319, 1985.
16. 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.
17. R.S. Szeliski. Bayesian modeling of uncertainty in low-level vision. *International Journal of Computer Vision*, 5(3):271–302, December 1990.
18. Demetri Terzopoulos. Image analysis using multigrid relaxation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):129–139, 1986.
19. Demetri Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):413–424, 1986.