What Energy Functions Can Be Minimized via Graph Cuts?

Vladimir Kolmogorov, *Member*, *IEEE*, and Ramin Zabih, *Member*, *IEEE*

Abstract—In the last few years, several new algorithms based on graph cuts have been developed to solve energy minimization problems in computer vision. Each of these techniques constructs a graph such that the minimum cut on the graph also minimizes the energy. Yet, because these graph constructions are complex and highly specific to a particular energy function, graph cuts have seen limited application to date. In this paper, we give a characterization of the energy functions that can be minimized by graph cuts. Our results are restricted to functions of binary variables. However, our work generalizes many previous constructions and is easily applicable to vision problems that involve large numbers of labels, such as stereo, motion, image restoration, and scene reconstruction. We give a precise characterization of what energy functions can be minimized using graph cuts, among the energy functions that can be written as a sum of terms containing three or fewer binary variables. We also provide a general-purpose construction to minimize such an energy function. Finally, we give a necessary condition for any energy function of binary variables to be minimized by graph cuts. Researchers who are considering the use of graph cuts to optimize a particular energy function can use our results to determine if this is possible and then follow our construction to create the appropriate graph. A software implementation is freely available.

Index Terms—Energy minimization, optimization, graph algorithms, minimum cut, maximum flow, Markov Random Fields.

1 Introduction and Overview

Many of the problems that arise in early vision can be naturally expressed in terms of energy minimization. The computational task of minimizing the energy is usually quite difficult as it generally requires minimizing a nonconvex function in a space with thousands of dimensions. If the functions have a very restricted form, they can be solved efficiently using dynamic programming [2]. However, researchers typically have needed to rely on general purpose optimization techniques such as simulated annealing [3], [16], which requires exponential time in theory and is extremely slow in practice.

In the last few years, however, a new approach has been developed based on graph cuts. The basic technique is to construct a specialized graph for the energy function to be minimized such that the minimum cut on the graph also minimizes the energy (either globally or locally). The minimum cut, in turn, can be computed very efficiently by max flow algorithms. These methods have been successfully used for a wide variety of vision problems, including image restoration [9], [10], [18], [21], stereo and motion [4], [9], [10], [20], [24], [27], [32], [35], [36], image synthesis [29], image segmentation [8], voxel occupancy [39], multicamera scene reconstruction [28], and medical imaging [5], [6], [25], [26]. The output of these algorithms is generally a solution with some interesting theoretical quality guarantee. In some cases [9], [18], [20], [21], [35], it is the global minimum, in other cases, a local minimum in a strong sense [10] that is within a known factor of the global minimum. The experimental results produced by these algorithms are also quite good. For example, two recent evaluations of stereo algorithms using real imagery with dense ground truth [37], [41] found that the best overall performance was due to algorithms based on graph cuts.

Minimizing an energy function via graph cuts, however, remains a technically difficult problem. Each paper constructs its own graph specifically for its individual energy function and, in some of these cases (especially [10], [27], [28]), the construction is fairly complex. One consequence is that researchers sometimes use heuristic methods for optimization, even in situations where the exact global minimum can be computed via graph cuts. The goal of this paper is to precisely characterize the class of energy functions that can be minimized via graph cuts and to give a general-purpose graph construction that minimizes any energy function in this class. Our results play a key role in [28], provide a significant generalization of the energy minimization methods used in [4], [5], [6], [10], [18], [26], [32], [39], and show how to minimize an interesting new class of energy functions.

In this paper, we only consider energy functions involving binary-valued variables. At first glance, this restriction seems severe since most work with graph cuts considers energy functions with variables that have many possible values. For example, the algorithms presented in [10] use graph cuts to address the standard pixel labeling problem that arises in early vision, including stereo, motion, and image restoration. In the pixel-labeling problem, the variables represent individual pixels and the possible values for an individual variable represent, e.g., its possible displacements or intensities. However, as we discuss in Section 2, many of the graph cut methods that handle multiple possible values work by repeatedly minimizing an energy function involving only binary variables. As we will see, our results generalize many graph cut algorithms [4], [5], [6], [10], [18], [26], [39] and can be easily applied to

• *The authors are with the Computer Science Department, Cornell University, Ithaca, NY 14853. E-mail: {vnk, rdz}@cs.cornell.edu.*

Manuscript received 2 May 2002; revised 17 Mar. 2003; accepted 16 June 2003.
Recommended for acceptance by M.A.T. Figueiredo, E.R. Hancock, M. Pelillo, and J. Zerubia.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 118731.

problems like pixel labeling, even though the pixels have many possible labels.

1.1 Summary of Our Results

In this paper, we focus on two classes of energy functions. Let $\{x_1, \ldots, x_n\}$, $x_i \in \{0, 1\}$ be a set of binary-valued variables. We define the class $\mathcal{F}^2$ to be functions that can be written as a sum of functions of up to two binary variables at a time,

$$E(x_1, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i, x_j). \quad (1)$$

We define the class $\mathcal{F}^3$ to be functions that can be written as a sum of functions of up to three binary variables at a time,

$$E(x_1, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i, x_j) + \sum_{i<j<k} E^{i,j,k}(x_i, x_j, x_k). \quad (2)$$

Obviously, the class $\mathcal{F}^2$ is a strict subset of the class $\mathcal{F}^3$. However, to simplify the discussion, we will begin by focusing on $\mathcal{F}^2$. Note that there is no restriction on the signs of the energy functions or of the individual terms.

The main results in this paper are:

- A precise characterization of the class of functions in $\mathcal{F}^3$ that can be minimized using graph cuts.
- A general-purpose graph construction for minimizing any function in this class.
- A necessary condition for any function of binary variables to be minimized via graph cuts.

The rest of the paper is organized as follows: In Section 2, we describe how graph cuts can be used to minimize energy functions and discuss the importance of energy functions with binary variables in the context of two example vision problems, namely, stereo and multicamera scene reconstruction. In Section 3, we formalize the relationship between graphs and energy functions and provide a precise definition of the problem that we wish to solve. Section 4 contains our main theorem for the class of functions $\mathcal{F}^2$ and shows how this result can be used for both stereo and multicamera scene reconstruction. Section 5 gives our results for the broader class $\mathcal{F}^3$ and shows how this result can be used for multicamera scene reconstruction. A necessary condition for an arbitrary function of binary variables to be minimized via graph cuts is presented in Section 6. We discuss some related work in the theory of submodular functions in Section 7 and give a summary of our graph constructions in Section 8. Software can be downloaded from http://www.cs.cornell.edu/~rdz/graphcuts.html that automatically constructs the appropriate graph and minimizes the energy. An NP-hardness result and a proof of one of our theorems are deferred to the Appendix.

2 Minimizing Energy Functions via Graph Cuts

Many vision problems, especially in early vision, can naturally be formulated in terms of energy minimization. Energy minimization has a long history in computer vision (see [34] for several examples). The classical use of energy minimization is to solve the pixel-labeling problem, which is a generalization of such problems as stereo, motion, and image restoration. The input is a set of pixels $\mathcal{P}$ and a set of labels $\mathcal{L}$. The goal is to find a labeling f (i.e., a mapping from $\mathcal{P}$ to $\mathcal{L}$) which minimizes some energy function.

A standard form of the energy function is

$$E(f) = \sum_{p \in \mathcal{P}} D_p(f_p) + \sum_{p,q \in \mathcal{N}} V_{p,q}(f_p, f_q), \quad (3)$$

where $\mathcal{N} \subset \mathcal{P} \times \mathcal{P}$ is a neighborhood system on pixels.[1] $D_p(f_p)$ is a function derived from the observed data that measures the cost of assigning the label f_p to the pixel p. $V_{p,q}(f_p, f_q)$ measures the cost of assigning the labels f_p, f_q to the adjacent pixels p, q and is used to impose spatial smoothness. At the borders of objects, adjacent pixels should often have very different labels and it is important that E not overpenalize such labelings. This requires that V be a nonconvex function of $|f_p - f_q|$. Such an energy function is called *discontinuity-preserving*. Energy functions of the form (3) can be justified on Bayesian grounds using the well-known Markov Random Fields (MRF) formulation [16], [31].

Energy functions like E are extremely difficult to minimize, however, as they are nonconvex functions in a space with many thousands of dimensions. They have traditionally been minimized with general-purpose optimization techniques (such as simulated annealing) that can minimize an arbitrary energy function. As a consequence of their generality, however, such techniques require exponential time and are extremely slow in practice. In the last few years, however, efficient algorithms have been developed for these problems based on graph cuts.[2]

2.1 Graph Cuts

Suppose $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a directed graph with nonnegative edge weights that has two special vertices (terminals), namely, the source s and the sink t. An s-t-cut (which we will refer to informally as a cut) $C = S, T$ is a partition of the vertices in $\mathcal{V}$ into two disjoint sets S and T such that $s \in S$ and $t \in T$. The cost of the cut is the sum of costs of all edges that go from S to T:

$$c(S, T) = \sum_{u \in S, v \in T, (u,v) \in \mathcal{E}} c(u, v).$$

The minimum s-t-cut problem is to find a cut C with the smallest cost. Due to the theorem of Ford and Fulkerson [14], this is equivalent to computing the maximum flow from the source to sink. There are many algorithms that solve this problem in polynomial time with small constants [1], [7], [17].

It is convenient to note a cut $C = S, T$ by a labeling f mapping from the set of the vertices $\mathcal{V} - \{s, t\}$ to $\{0, 1\}$, where $f(v) = 0$ means that $v \in S$ and $f(v) = 1$ means that $v \in T$. We will use this notation later.

Note that a cut is a *binary* partition of a graph viewed as a labeling; it is a binary-valued labeling. While there are generalizations of the minimum s-t-cut problem that involve more than two terminals (such as the multiway cut problem [12]), such generalizations are NP-hard.

1. Without loss of generality, we can assume that $\mathcal{N}$ contains only ordered pairs p, q for which $p < q$ since we can combine two terms $V_{p,q}$ and $V_{q,p}$ into one term.

2. It is interesting to note that some similar techniques have been developed by algorithms researchers working on the task assignment problem [30], [33], [40].

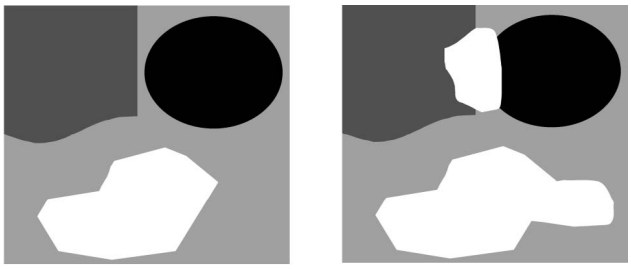

Fig. 1. An example of an expansion move. The labeling on the right is a white-expansion move from the labeling on the left.

2.2 Energy Minimization Using Graph Cuts

In order to minimize E using graph cuts, a specialized graph is created such that the minimum cut on the graph also minimizes E (either globally or locally). The form of the graph depends on the exact form of V and on the number of labels.

In certain restricted situations, it is possible to efficiently compute the global minimum. If there are only two labels, [18] showed how to compute the global minimum of E. This is also possible for an arbitrary number of labels as long as the labels are consecutive integers and V is the L_1 distance. The construction is due to [20] and is a modified version of [36]. This construction has been further generalized to handle an arbitrary convex V [22].

However, a convex V is not discontinuity preserving and optimizing an energy function with such a V leads to oversmoothing at the borders of objects. The ability to find the global minimum efficiently, while theoretically of great value, does not overcome this drawback. This is clearly visible in the relatively poor performance of such algorithms on the stereo benchmarks described in [37].

Moreover, efficient global energy minimization algorithms for even the simplest class of discontinuity-preserving energy functions almost certainly do not exist. Consider $V(f_p, f_q) = T[f_p \neq f_q]$, where the indicator function $T[\cdot]$ is 1 if its argument is true and otherwise 0. This smoothness term, sometimes called the Potts model, is clearly discontinuity-preserving. Yet, it is known to be NP-hard to minimize [10].

However, graph cut algorithms have been developed that compute a local minimum in a strong sense [10]. As we shall see, these methods minimize an energy function with nonbinary variables by repeatedly minimizing an energy function with binary variables.[3]

2.3 The Expansion Move Algorithm

One of the most effective algorithms for minimizing discontinuity-preserving energy functions is the expansion move algorithm, which was introduced in [10]. This algorithm can be used whenever V is a metric on the space of labels; this includes several important discontinuity-preserving energy functions (see [10] for more details).

Consider a labeling f and a particular label α. Another labeling f' is defined to be an α-expansion move from f if $f'_p \neq \alpha$ implies $f'_p = f_p$. This means that the set of pixels assigned the label α has increased when going from f to f'. An example of an expansion move is shown in Fig. 1.

The expansion move algorithm cycles through the labels α in some order (fixed or random) and finds the lowest energy α-expansion move from the current labeling. If this expansion move has lower energy than the current labeling, then it becomes the current labeling. The algorithm terminates with a labeling that is a local minimum of the energy with respect to expansion moves; more precisely, there is no α-expansion move, for any label α, with lower energy. It is also possible to prove that such a local minimum lies within a multiplicative factor of the global minimum [10] (the factor is at least 2 and depends only on V).

2.3.1 Applications of the Expansion Move Algorithm

Energy functions like that of (3) arise commonly in early vision, so it is straightforward to apply the expansion move algorithm to many vision problems. For example, the expansion move algorithm can be used for the standard 2-camera stereo problem, as well as for multicamera scene reconstruction.

Stereo. For the stereo problem, the labels are disparities and the data term $D_p(f_p)$ is some function of the intensity difference between the pixel p in the primary image and the pixel $p + f_p$ in the secondary image. On the stereo problem, the expansion move algorithm and other closely related energy minimization algorithms give very strong empirical performance. They have been used as a critical component of several algorithms [4], [27], [28], [32], including the top two algorithms (and five of the top six) according to a recent published survey [37] that used real data with dense ground truth.

Multicamera scene reconstruction. It is also possible to use the expansion move algorithm for multicamera scene reconstruction, as reported in [28]. In their problem formulation, the energy function consists entirely of terms with two arguments (in other words, D_p does not exist). The set of pixels $\mathcal{P}$ includes all pixels from all cameras and the labels correspond to planes. Thus, a pair $\langle p, f_p \rangle$ specifies a 3D point and a labeling f gives the nearest scene element visible in each pixel in each camera.

The expansion move algorithm can be straightforwardly applied to this problem as long as the energy is of the right form. Note that an α-expansion move increases the set of pixels labeled α in all cameras at once, rather than in a single preferred image.

The actual energy function consists of three terms. All the terms are in the same form as V in that they depend on pairs of labels f_p, f_q. One term enforces the visibility constraint. While this constraint is very important, it has no analogue in the simple pixel labeling problem formulation and we will not discuss it here. The other terms are analogous to the standard stereo data term and smoothness term.

In the multicamera scene reconstruction problem, the data term enforces photoconsistency. Let I be a set of pairs of "nearby" 3D points. These points will come from different cameras, but they will share the same depth (i.e., the points are of the form $\langle p, f_p \rangle, \langle q, f_q \rangle$ where $f_p = f_q$ and p, q are pixels from different cameras). Then, the data term is

$$\sum_{\{\langle (p,l) \rangle, \langle (q,l) \rangle \in I\}} D_{p,q,l}(f_p, f_q), \tag{4}$$

where

$$D_{p,q,l}(f_p, f_q) = D(p, q) \cdot T[f_p = f_q = l].$$

We have rewritten each term in the sum as a function of f_p, f_q (since p, q do not depend on f). The function D will be

3. This is somewhat analogous to the $\mathcal{Z}\pi M$ algorithm [13], although that method produces a global minimum of a function with nonbinary variables by repeatedly minimizing a function with binary variables.

some function of the intensity difference between p and q, typically, a monotonically increasing function.

The smoothness term, on the other hand, involves a single camera at a time. It is defined to be

$$\sum_{\{p,q\}\in\mathcal{N}} V_{p,q}(f_p, f_q), \tag{5}$$

where $\mathcal{N}$ is a neighborhood system on pixels in a single camera.

2.4 Energy Functions of Binary Variables

The key subproblem in the expansion move algorithm is to compute the lowest energy labeling within a single α-expansion of f. This subproblem is solved efficiently with a single graph cut [10], using a somewhat intricate graph construction.

It is important to note that this subproblem is an energy minimization problem over binary variables, even though the overall problem that the expansion move algorithm is solving involves nonbinary variables. This is because each pixel will either keep its old value under f or acquire the new label α.

Formally, any labeling f' within a single α-expansion of the initial labeling f can be encoded by a binary vector $x = \{x_p | p \in \mathcal{P}\}$, where $f'(p) = f(p)$ if $x_p = 0$ and $f'(p) = \alpha$ if $x_p = 1$. Let us denote the labeling defined by a binary vector x as f^x. Since the energy function E is defined over all labelings, it is obviously also defined over labelings specified by binary vectors. The key step in the expansion move algorithm is therefore to find the minimum of $E(f^x)$ over all binary vectors x.

The importance of energy functions of binary variables does not arise simply from the expansion move algorithm. Instead, it results from the fact that a graph cut effectively assigns one of two possible values to each vertex of the graph. So, in a certain sense, any energy minimization construction based on graph cuts relies on intermediate binary variables.

It is, of course, possible to use graph cuts in such a manner that variables in the original problem do not correspond in a one-to-one manner with vertices in the graph. This kind of complex transformation lies at the heart of the graph cut algorithms that compute a global minimum [20], [22]. However, the NP-hardness result given in [10] shows that (unless P = NP) such result cannot be achieved for even the simplest discontinuity-preserving energy function.

3 Representing Energy Functions with Graphs

Let us consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with terminals s and t, thus $\mathcal{V} = \{v_1, \ldots, v_n, s, t\}$. Each cut on $\mathcal{G}$ has some cost; therefore, $\mathcal{G}$ represents the energy function mapping from all cuts on $\mathcal{G}$ to the set of nonnegative real numbers. Any cut can be described by n binary variables $x_1, \ldots, x_n$ corresponding to vertices in $\mathcal{G}$ (excluding the source and the sink): $x_i = 0$ when $v_i \in S$ and $x_i = 1$ when $v_i \in T$. Therefore, the energy E that $\mathcal{G}$ represents can be viewed as a function of n binary variables: $E(x_1, \ldots, x_n)$ is equal to the cost of the cut defined by the configuration $x_1, \ldots, x_n$ $(x_i \in \{0, 1\})$. Note that the configuration that minimizes E will not change if we add a constant to E.

We can efficiently minimize E by computing the minimum s-t-cut on $\mathcal{G}$. This naturally leads to the question: What is the class of energy functions E for which we can construct a graph that represents E? We can also generalize our construction. Above, we used each vertex (except the source and the sink) for encoding one binary variable. Instead, we can specify a subset $\mathcal{V}_0 = \{v_1, \ldots, v_k\} \subset \mathcal{V} - \{s, t\}$ and introduce variables only for the vertices in this set. Then, there may be several cuts corresponding to a configuration $x_1, \ldots, x_k$. If we define the energy $E(x_1, \ldots, x_k)$ as the minimum among the costs of all such cuts, then the minimum s-t-cut on $\mathcal{G}$ will again yield the configuration which minimizes E.

We will summarize the graph constructions that we allow in the following definition.

Definition 3.1. *A function E of n binary variables is called* graph-representable *if there exists a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with terminals s and t and a subset of vertices $\mathcal{V}_0 = \{v_1, \ldots, v_n\} \subset \mathcal{V} - \{s, t\}$ such that, for any configuration $x_1, \ldots, x_n$, the value of the energy $E(x_1, \ldots, x_n)$ is equal to a constant plus the cost of the minimum s-t-cut among all cuts $C = S, T$ in which $v_i \in S$, if $x_i = 0$, and $v_i \in T$, if $x_i = 1$ $(1 \le i \le n)$. We say that E is* exactly represented *by $\mathcal{G}$, $\mathcal{V}_0$ if this constant is zero.*

The following lemma is an obvious consequence of this definition.

Lemma 3.2. *Suppose the energy function E is graph-representable by a graph $\mathcal{G}$ and a subset $\mathcal{V}_0$. Then, it is possible to find the exact minimum of E in polynomial time by computing the minimum s-t-cut on $\mathcal{G}$.*

In this paper, we will give a complete characterization of the classes $\mathcal{F}^2$ and $\mathcal{F}^3$ in terms of graph representability and show how to construct graphs for minimizing graph-representable energies within these classes. Moreover, we will give a necessary condition for all other classes that must be met for a function to be graph-representable. Obviously, it would be sufficient to consider only the class $\mathcal{F}^3$ since $\mathcal{F}^2 \subset \mathcal{F}^3$. However, the condition for $\mathcal{F}^2$ is simpler, so we will consider it separately.

Note that the energy functions we consider can be negative, as can the individual terms in the energy functions. However, the graphs that we construct must have nonnegative edge weights. All previous work that used graph cuts for energy minimization dealt with nonnegative energy functions and terms.

4 The Class $\mathcal{F}^2$

Our main result for the class $\mathcal{F}^2$ is the following theorem.

Theorem 4.1 ($\mathcal{F}^2$ Theorem). *Let E be a function of n binary variables from the class $\mathcal{F}^2$, i.e.,*

$$E(x_1, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i, x_j). \tag{6}$$

Then, E is graph-representable if and only if each term $E^{i,j}$ satisfies the inequality

$$E^{i,j}(0,0) + E^{i,j}(1,1) \le E^{i,j}(0,1) + E^{i,j}(1,0). \tag{7}$$

We will call functions satisfying the condition of (7) *regular*.[4] This theorem thus states that regularity is a necessary and sufficient condition for graph-representability, at least in $\mathcal{F}^2$.

4. The representation of a function E as a sum in (6) is not unique. However, we will show in Section 5 that the definition of regularity does not depend on this representation.

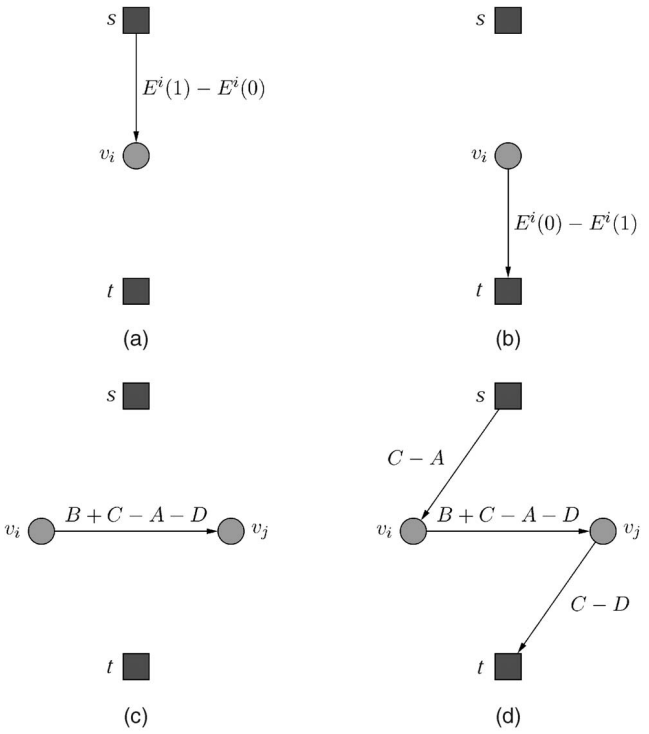

Fig. 2. Graphs that represent some functions in $\mathcal{F}^2$. (a) Graph for E^i, where $E^i(0) > E^i(1)$. (b) Graph for E^i, where $E^i(0) \not> E^i(0)$. (c) Third edge for $E^{i,j}$. (d) Complete graph for $E^{i,j}$ if $C > A$ and $C > D$.

In this section, we will give a constructive proof that regularity is a sufficient condition by describing how to build a graph that represents an arbitrary regular function in $\mathcal{F}^2$. The other half of the theorem is proven in much more generality in Section 6, where we show that a regularity is a necessary condition for any function to be graph-representable.

Regularity is thus an extremely important property as it allows energy functions to be minimized using graph cuts. Moreover, without the regularity constraint, the problem becomes intractable. Our next theorem shows that minimizing an arbitrary nonregular function is NP-hard, even if we restrict our attention to $\mathcal{F}^2$.

Theorem 4.2 (NP-Hardness). *Let E^2 be a nonregular function of two variables. Then, minimizing functions of the form*

$$E(x_1, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{(i,j)\in\mathcal{N}} E^2(x_i, x_j),$$

where E^i are arbitrary functions of one variable and $\mathcal{N} \subset \{(i,j) | 1 \le i < j \le n\}$, is NP-hard.

The proof of this theorem is deferred to the Appendix. Note that this theorem implies the intractability of minimizing the entire class of nonregular functions in $\mathcal{F}^2$. It thus allows for the existence of nonregular functions in $\mathcal{F}^2$ that can be minimized efficiently.[5]

4.1 Graph Construction for $\mathcal{F}^2$

In this section, we will prove the constructive part of the $\mathcal{F}^2$ theorem. Let E be a regular function of the form given in the theorem. We will construct a graph for each term separately and then "merge" all graphs together. This is justified by the additivity theorem, which is given in the Appendix.

TABLE 1

$$E^{i,j} = \begin{array}{|c|c|}\hline E^{i,j}(0,0) & E^{i,j}(0,1) \\ \hline E^{i,j}(1,0) & E^{i,j}(1,1) \\ \hline \end{array} = \begin{array}{|c|c|}\hline A & B \\ \hline C & D \\ \hline \end{array}$$

TABLE 2

$$\begin{array}{|c|c|}\hline A & B \\ \hline C & D \\ \hline \end{array} = A + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline C-A & C-A \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & D-C \\ \hline 0 & D-C \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & B+C-A-D \\ \hline 0 & 0 \\ \hline \end{array}$$

The graph will contain $n + 2$ vertices: $\mathcal{V} = \{s, t, v_1, \ldots, v_n\}$. Each nonterminal vertex v_i will encode the binary variable x_i. For each term of E, we will add one or more edges that we describe next.

First, consider a term E^i depending on one variable x_i. If $E^i(0) < E^i(1)$, then we add the edge (s, v_i) with the weight $E^i(1) - E^i(0)$ (Fig. 2a). Otherwise, we add the edge (v_i, t) with the weight $E^i(0) - E^i(1)$ (Fig. 2b). It easy to see that, in both cases, the constructed graph represents the function E^i (but with different constants: $E^i(0)$ in the former case and $E^i(1)$ in the latter).

Now, consider a term $E^{i,j}$ depending on two variables x_i, x_j. It is convenient to represent it in Table 1. We can rewrite it as it is expressed in Table 2. The first term is a constant function, so we don't need to add any edges for it. The second and the third terms depend only on one variable x_i and x_j, respectively. Therefore, we can use the construction given above. To represent the last term, we add an edge (v_i, v_j) with the weight $B + C - A - D$ (Fig. 2c). Note that this weight is nonnegative due to the regularity of E.

A complete graph for $E^{i,j}$ will contain three edges. One possible case ($C - A > 0$ and $D - C < 0$) is illustrated in Fig. 2d.

Note that we did not introduce any additional vertices for representing binary interactions of binary variables. This is in contrast to the construction in [10] which added auxiliary vertices for representing energies that we just considered. Our construction yields a smaller graph and, thus, the minimum cut can potentially be computed faster.

4.2 Example: Applying Our Results for Stereo and Multicamera Scene Reconstruction

Our results give a generalization of a number of previous algorithms [4], [5], [6], [10], [18], [26], [32], [39] in the following sense. Each of these methods used a graph cut algorithm that was specifically constructed to minimize a certain form of energy function. The class of energy functions that we show how to minimize is much larger and includes the techniques used in all of these methods as special cases.

To illustrate the power of our results, we return to the use of the expansion move algorithm for stereo and multicamera scene reconstruction described in Section 2.3.1. Recall that the key subproblem is to find the minimum energy labeling within a single α-expansion of f, which is equivalent to minimizing the energy over binary variables x_i.

5. Gortler et al. have investigated a particularly simple example, namely, functions that can be made regular by interchanging the senses of 0 and 1 for some set of variables.

TABLE 3

$E^{i,j}(0,0)$	$E^{i,j}(0,1)$	=	$V(f_p, f_q)$	$V(f_p, \alpha)$
$E^{i,j}(1,0)$	$E^{i,j}(1,1)$		$V(\alpha, f_q)$	$V(\alpha, \alpha)$

4.2.1 Stereo

The paper that proposed the expansion move algorithm [10] showed how to solve the key subproblem with graph cuts as long as D_p is nonnegative and V is a metric on the space of labels. This involved an elaborate graph construction and several associated theorems.

Using our results, we can recreate the proof that such an energy function can be solved in just a few lines. All that we need is to prove that E is regular if D_p is nonnegative and V is a metric. Obviously, the form of D_p doesn't matter; we simply have to show that if V is a metric the individual terms satisfy the inequality in (7). Table 3 considers two neighboring pixels p, q, with associated binary variables i, j. If V is a metric, by definition, for any labels α, β, γ $V(\alpha, \alpha) = 0$ and $V(\beta, \alpha) + V(\alpha, \gamma) \geq V(\beta, \gamma)$. This gives the inequality of (7) and shows that E can be minimized using graph cuts.

4.2.2 Multicamera Scene Reconstruction

We can also show that the expansion move algorithm can be used for a different energy function, namely, the one proposed for multicamera scene reconstruction in [28]. We need to show that the smoothness (5) and the data (4) energy functions are regular. The argument for the smoothness term is identical to the one we just gave for stereo because the smoothness term is regular.

The data term for multicamera scene reconstruction is perhaps the best demonstration of the utility of our results. The function in the data term is clearly not a metric on the space of labels (for example, it is entirely possible that $D_{p,q,l}(\beta, \beta) \neq 0$ for some label β). Prior to our results, the only way to use the expansion move algorithm for this energy function would be to create an elaborate graph construction that is specific to this particular energy function. Worse, it was not a priori obvious that such a construction existed.

Using our results, it is simple to show that the data term can be made regular and the expansion move algorithm can be applied. Let us write out the sum in (4) and consider a single term $D(p, q) \cdot T[f_p = f_q = l]$, which imposes a penalty if p and q both have the label l. Recall that we are seeking the lowest energy labeling within a single α-expansion of f. We will have two binary variables which are 0 if the associated pixel keeps its original label in f and 1 if it acquires the new label α.

There are two cases that must be analyzed separately: $l \neq \alpha$ and $l = \alpha$. Consider the case $l \neq \alpha$. If $f_p \neq l$ or $f_q \neq l$, then the term is zero for any values of the binary variables. If $f_p = f_q = l$, then the penalty is only imposed when the binary variables are both 0. We can graphically show this in Table 4.

TABLE 4

$E^{i,j}(0,0)$	$E^{i,j}(0,1)$	=	$D(p,q)$	0
$E^{i,j}(1,0)$	$E^{i,j}(1,1)$		0	0

TABLE 5

$E^{i,j}(0,0)$	$E^{i,j}(0,1)$	=	0	0
$E^{i,j}(1,0)$	$E^{i,j}(1,1)$		0	$D(p,q)$

If $l = \alpha$, then, if either f_p or f_q are α, the penalty imposed is independent of one of the binary variables and it is easy to see that this is regular. If $l = \alpha$ and neither f_p nor f_q are α, then the penalty is only imposed when the binary variables are both 1. Graphically, this can be shown in Table 5. Overall, the energy function is regular if $D(p, q)$ is nonpositive. Recall that $D(p, q)$ is a photoconsistency term. It is typically a monotonically increasing function of the intensity difference between p and q. By simply subtracting a large constant, we can ensure that D is nonpositive and, thus, apply our construction.

5 The Class $\mathcal{F}^3$

Before stating our theorem for the class $\mathcal{F}^3$, we will extend the definition of regularity to arbitrary functions of binary variables. This requires a notion of *projections*.

Suppose we have a function E of n binary variables. If we fix m of these variables, then we get a new function E' of $n - m$ binary variables; we will call this function a *projection* of E. The notation for projections is as follows:

Definition 5.1. *Let $E(x_1, \ldots, x_n)$ be a function of n binary variables and let I, J be a disjoint partition of the set of indices $\{1, \ldots, n\}$: $I = \{i(1), \ldots, i(m)\}$, $J = \{j(1), \ldots, j(n-m)\}$. Let $\alpha_{i(1)}, \ldots, \alpha_{i(m)}$ be binary constants. A* projection *$E' = E[x_{i(1)} = \alpha_{i(1)}, \ldots, x_{i(m)} = \alpha_{i(m)}]$ is a function of $n - m$ variables defined by*

$$E'(x_{j(1)}, \ldots, x_{j(n-m)}) = E(x_1, \ldots, x_n),$$

where $x_i = \alpha_i$ for $i \in I$. We say that we fix the variables $x_{i(1)}, \ldots, x_{i(m)}$.

Now, we can give a generalized definition of regularity.

Definition 5.2.

- *All functions of one variable are regular.*
- *A function E of two variables is called regular if $E(0,0) + E(1,1) \leq E(0,1) + E(1,0)$.*
- *A function E of more than two variables is called regular if all projections of E of two variables are regular.*

For the class $\mathcal{F}^2$, this definition is equivalent to the previous one. A proof of this fact is given in Section 5.3.

Now, we are ready to formulate our main theorem for $\mathcal{F}^3$.

Theorem 5.3 ($\mathcal{F}^3$ Theorem). *Let E be a function of n binary variables from $\mathcal{F}^3$, i.e.,*

$$E(x_1, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i, x_j) + \sum_{i<j<k} E^{i,j,k}(x_i, x_j, x_k). \quad (8)$$

Then, E is graph-representable if and only if E is regular.

TABLE 6

$$E^{i,j,k} = \begin{array}{|c|c|}\hline E^{i,j}(0,0,0) & E^{i,j}(0,0,1) \\ \hline E^{i,j}(0,1,0) & E^{i,j}(0,1,1) \\ \hline E^{i,j}(1,0,0) & E^{i,j}(1,0,1) \\ \hline E^{i,j}(1,1,0) & E^{i,j}(1,1,1) \\ \hline \end{array} = \begin{array}{|c|c|}\hline A & B \\ \hline C & D \\ \hline E & F \\ \hline G & H \\ \hline \end{array}$$

5.1 Graph Construction for $\mathcal{F}^3$

Consider a regular function E of the form given in (8). The regularity of E does not necessarily imply that each term in (8) is regular. However, we will give an algorithm for converting any regular function in $\mathcal{F}^3$ to the form (8) where each term is regular (see the regrouping theorem in the next section). Note that this was not necessary for the class $\mathcal{F}^2$ since, for any representation of a regular function in the form (6), each term in the sum is regular.

Here, we will give graph construction for a regular function E of the form as in the $\mathcal{F}^3$ theorem assuming that each term of E is regular. The graph will contain $n+2$ vertices: $\mathcal{V} = \{s, t, v_1, \ldots, v_n\}$ as well as some additional vertices described below. Each nonterminal vertex v_i will encode the binary variable x_i. For each term of E, we will add one or more edges and, possibly, an additional (unique) vertex that we describe next. Again, our construction is justified by the additivity theorem given in the Appendix.

Terms depending on one and two variables were considered in the previous section, so we will concentrate on a term $E^{i,j,k}$ depending on three variables x_i, x_j, x_k. It is convenient to represent it in Table 6.

Let us denote $P = (A + D + F + G) - (B + C + E + H)$. We consider two cases:

Case 1. $P >= 0$. $E^{i,j,k}$ can be rewritten as shown in Table 7, where $P_1 = F - B$, $P_2 = G - E$, $P_3 = D - C$, $P_{23} = B + C - A - D$, $P_{31} = B + E - A - F$, $P_{12} = C + E - A - G$. $E^{i,j,k}$ is regular, therefore, by considering projections $E^{i,j,k}[x_1 = 0]$, $E^{i,j,k}[x_2 = 0]$, $E^{i,j,k}[x_3 = 0]$, we conclude that P_{23}, P_{31}, P_{12} are nonnegative.

The first term is a constant function, so we don't need to add any edges for it. The next three terms depend only on one variable (x_i, x_j, x_k, respectively), so we can use the construction given in the previous section. The next three terms depend on two variables; the nonnegativity of P_{23}, P_{31}, P_{12} implies that they are all regular. Therefore, we can again use the construction of the previous section. (Three edges will be added: (v_j, v_k), (v_k, v_i), (v_i, v_j) with weights P_{23}, P_{31}, P_{12}, respectively).

To represent the last term, we will add an auxiliary vertex u_{ijk} and four edges $(v_i, u_{ijk}), (v_j, u_{ijk}), (v_k, u_{ijk}), (u_{ijk}, t)$ with the weight P. This is shown in Fig. 3a. Let us prove these four edges exactly represent the function shown in Table 8. If $x_i = x_j = x_k = 1$, then the cost of the minimum cut is 0 (the minimum cut is $S = \{s\}$, $T = \{v_i, v_j, v_k, u_{ijk}, t\}$. Suppose at least one of the variables x_i, x_j, x_k is 0; without loss of generality, we can assume that $x_i = 0$, i.e., $v_i \in \mathcal{S}$. If $u_{ijk} \in S$, then the edge (u_{ijk}, t) is cut; if $u_{ijk} \in T$, the edge (v_i, u_{ijk}) is cut yielding the cost P. Hence, the cost of the minimum cut is at least P. However, if $u_{ijk} \in S$, the cost of the cut is exactly P, no matter where the vertices v_i, v_j, v_k are. Therefore, the cost of the minimum cut will be 0 if $x_i = x_j = x_k = 1$ and P otherwise, as desired.

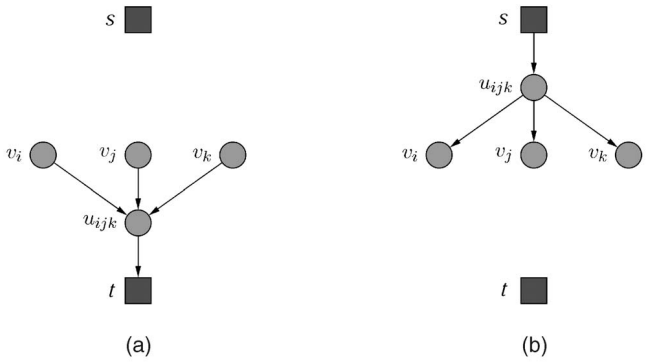

Fig. 3. Graphs for functions in $\mathcal{F}^3$. (a) Edge weights are all P. (b) Edge weights are all $-P$.

TABLE 8

$$\begin{array}{|c|c|}\hline P & P \\ \hline P & P \\ \hline P & P \\ \hline P & 0 \\ \hline \end{array} = P + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline 0 & -P \\ \hline \end{array}$$

Case 2. $P < 0$. This case is similar to the Case 1. $E^{i,j,k}$ can be rewritten as shown in Table 9, where $P_1 = C - G$, $P_2 = B - D$, $P_3 = E - F$, $P_{32} = F + G - E - H$, $P_{13} = D + G - C - H$, $P_{21} = D + F - B - H$. By considering projections $E^{i,j,k}[x_1 = 1]$, $E^{i,j,k}[x_2 = 1]$, $E^{i,j,k}[x_3 = 1]$, we conclude that P_{32}, P_{13}, P_{21} are nonnegative.

As in the previous case, all terms except the last are regular and depend on at most two variables, so we can use the construction of the previous section. We will have, for example, edges (v_k, v_j), (v_i, v_k), (v_j, v_i) with weights P_{32}, P_{13}, P_{21}, respectively.

To represent the last term, we will add an auxiliary vertex u_{ijk} and four edges (u_{ijk}, v_i), (u_{ijk}, v_j), (u_{ijk}, v_k), (s, u_{ijk}) with the weight $-P$. This is shown in Fig. 3b.

TABLE 7

$$\begin{array}{|c|c|}\hline A & B \\ \hline C & D \\ \hline E & F \\ \hline G & H \\ \hline \end{array} = A + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & 0 \\ \hline P_1 & P_1 \\ \hline P_1 & P_1 \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline P_2 & P_2 \\ \hline 0 & 0 \\ \hline P_2 & P_2 \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & P_3 \\ \hline 0 & P_3 \\ \hline 0 & P_3 \\ \hline 0 & P_3 \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & P_{23} \\ \hline 0 & 0 \\ \hline 0 & P_{23} \\ \hline 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & 0 \\ \hline P_{31} & 0 \\ \hline P_{31} & 0 \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline P_{12} & P_{12} \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline 0 & -P \\ \hline \end{array}$$

TABLE 9

$$\begin{array}{|c|c|} \hline A & B \\ \hline C & D \\ \hline E & F \\ \hline G & H \\ \hline \end{array} = H + \begin{array}{|c|c|} \hline P_1 & P_1 \\ \hline P_1 & P_1 \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|} \hline P_2 & P_2 \\ \hline 0 & 0 \\ \hline P_2 & P_2 \\ \hline 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|} \hline P_3 & 0 \\ \hline P_3 & 0 \\ \hline P_3 & 0 \\ \hline P_3 & 0 \\ \hline \end{array} + \begin{array}{|c|c|} \hline 0 & 0 \\ \hline P_{32} & 0 \\ \hline 0 & 0 \\ \hline P_{32} & 0 \\ \hline \end{array} + \begin{array}{|c|c|} \hline 0 & P_{13} \\ \hline 0 & P_{13} \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|} \hline 0 & 0 \\ \hline 0 & 0 \\ \hline P_{21} & P_{21} \\ \hline 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|} \hline P & 0 \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array}$$

Note that, in both cases, we added an auxiliary vertex u_{ijk}. It is easy to see that this is necessary since graphs without auxiliary vertices can only represent functions in $\mathcal{F}^2$. Each edge represents a function of at most two variables, so the whole graph represents a function that is a sum of terms of at most two variables.

5.2 Constructive Proof of the Regrouping Theorem

Now, we will show how to convert a regular function in $\mathcal{F}^3$ to the form given in (8) where each term is regular.

Theorem 5.4 (regrouping). *Any regular function E from the class $\mathcal{F}^3$ can be rewritten as a sum of terms such that each term is regular and depends on at most three variables.*

We will assume E given as

$$E(x_1, \ldots, x_n) = \sum_{i<j<k} E^{i,j,k}(x_i, x_j, x_k),$$

where i, j, k are indices from the set $\{1, \ldots, n\}$ (we omitted terms involving functions of one and two variables since they can be viewed as functions of three variables).

We begin by giving a definition.

Definition 5.5. *The functional π will be a mapping from the set of all functions (of binary variables) to the set of real numbers which is defined as follows. For a function $E(x_1, \ldots, x_n)$*

$$\pi(E) = \sum_{x_1 \in \{0,1\}, \ldots, x_n \in \{0,1\}} \left(\Pi_{i=1}^{n} (-1)^{x_i}\right) E(x_1, \ldots, x_n).$$

For example, for a function E of two variables $\pi(E) = E(0,0) - E(0,1) - E(1,0) + E(1,1)$. Note that a function E of two variables is regular if and only if $\pi(E) \leq 0$.

It is trivial to check the following lemma.

Lemma 5.6. *The functional π has the following properties.*

- *π is linear, i.e., for a scalar c and two functions E', E'' of n variables $\pi(E' + E'') = \pi(E') + \pi(E'')$ and $\pi(c \cdot E') = c \cdot \pi(E')$.*
- *If E is a function of n variables that does not depend on at least one of the variables, then $\pi(E) = 0$.*

We will prove the theorem using the following lemma and a trivial induction argument.

Definition 5.7. *Let $E^{i,j,k}$ be a function of three variables. The functional $N(E^{i,j,k})$ is defined as the number of projections of two variables of $E^{i,j,k}$ with positive values of the functional π.*

Note that $N(E^{i,j,k}) = 0$ exactly when $E^{i,j,k}$ is regular.

Lemma 5.8. *Suppose the function E of n variables can be written as*

$$E(x_1, \ldots, x_n) = \sum_{i<j<k} E^{i,j,k}(x_i, x_j, x_k),$$

where some of the terms are not regular. Then, it can be written as

$$E(x_1, \ldots, x_n) = \sum_{i<j<k} \tilde{E}^{i,j,k}(x_i, x_j, x_k),$$

where

$$\sum_{i<j<k} N\left(\tilde{E}^{i,j,k}\right) < \sum_{i<j<k} N\left(E^{i,j,k}\right).$$

Proof. For simplicity of notation, let us assume that the term $E^{1,2,3}$ is not regular and $\pi(E^{1,2,3}[x_3 = 0]) > 0$ or $\pi(E^{1,2,3}[x_3 = 1]) > 0$ (we can ensure this by renaming indices). Let

$$C_k = \max_{\alpha_k \in \{0,1\}} \pi(E^{1,2,k}[x_k = \alpha_k]) \qquad k \in \{4, \ldots, n\}$$

$$C_3 = -\sum_{k=4}^{n} C_k.$$

Now, we will modify the terms $E^{1,2,3}$, ..., $E^{1,2,n}$ as follows:

$$\tilde{E}^{1,2,k} \equiv E^{1,2,k} - R[C_k] \qquad k \in \{3, \ldots, n\},$$

where $R[C]$ is the function of two variables x_1 and x_2 defined by Table 10 (other terms are unchanged: $\tilde{E}^{i,j,k} \equiv E^{i,j,k}$, $(i,j) \neq (1,2)$). We have

$$E(x_1, \ldots, x_n) = \sum_{i<j<k} \tilde{E}^{i,j,k}(x_i, x_j, x_k)$$

since $\sum_{k=3}^{n} C_k = 0$ and $\sum_{k=3}^{n} R[C_k] \equiv 0$.

If we consider $R[C]$ as a function of n variables and take a projection of two variables where the two variables that are not fixed are x_i and x_j $(i < j)$, then the functional π will be C, if $(i,j) = (1,2)$, and 0 otherwise (since, in the latter case, a projection actually depends on, at most, one variable). Hence, the only projections of two variables that could have changed their value of the functional π are $\tilde{E}^{1,2,k}[x_3 = \alpha_3, \ldots, x_n = \alpha_n], k \in \{3, \ldots, n\}$, if we treat $\tilde{E}^{1,2,k}$ as functions of n variables, or $\tilde{E}^{1,2,k}[x_k = \alpha_k]$, if we treat $\tilde{E}^{1,2,k}$ as functions of three variables.

TABLE 10

$$R[C] = \begin{array}{|c|c|} \hline 0 & 0 \\ \hline 0 & C \\ \hline \end{array}$$

TABLE 11

$E^{i,j}(0,0,0)$	$E^{i,j}(0,0,1)$	$D(p,q,r)$	0
$E^{i,j}(0,1,0)$	$E^{i,j}(0,1,1)$	0	0
$E^{i,j}(1,0,0)$	$E^{i,j}(1,0,1)$	0	0
$E^{i,j}(1,1,0)$	$E^{i,j}(1,1,1)$	0	0

TABLE 12

$E^{i,j}(0,0,0)$	$E^{i,j}(0,0,1)$	0	0
$E^{i,j}(0,1,0)$	$E^{i,j}(0,1,1)$	0	0
$E^{i,j}(1,0,0)$	$E^{i,j}(1,0,1)$	0	0
$E^{i,j}(1,1,0)$	$E^{i,j}(1,1,1)$	0	$D(p,q,r)$

First, let us consider terms with $k \in \{4,\dots,n\}$. We have $\pi(E^{1,2,k}[x_k = \alpha_k]) \leq C_k$, thus

$$\pi(\tilde{E}^{1,2,k}[x_k = \alpha_k]) = \pi(E^{1,2,k}[x_k = \alpha_k]) - \pi(R[C_k][x_k = \alpha_k]) \leq C_k - C_k = 0.$$

Therefore, we did not introduce any nonregular projections for these terms.

Now, let us consider the term $\pi(\tilde{E}^{1,2,3}[x_3 = \alpha_3])$. We can write

$$\pi(\tilde{E}^{1,2,3}[x_3 = \alpha_3]) = \pi(E^{1,2,3}[x_3 = \alpha_3]) - \pi(R[C_3][x_3 = \alpha_3])$$
$$= \pi(E^{1,2,3}[x_3 = \alpha_3]) - (-\sum_{k=4}^{n} C_k) = \sum_{k=3}^{n} \pi(E^{1,2,k}[x_k = \alpha_k]),$$

where $\alpha_k = \arg\max_{\alpha \in \{0,1\}} \pi(E^{1,2,k}[x_k = \alpha])$, $k \in \{4,\dots,n\}$. The last expression is just $\pi(E[x_3 = \alpha_3, \dots, x_n = \alpha_n])$ and is nonpositive since E is regular by assumption. Hence, values $\pi(\tilde{E}^{1,2,3}[x_3 = 0])$ and $\pi(\tilde{E}^{1,2,3}[x_3 = 1])$ are both nonpositive and, therefore, the number of nonregular projections has decreased. □

The complexity of a single step is $O(n)$ since it involves modifying at most n terms. Therefore, the complexity of the whole algorithm is $O(mn)$, where m is the number of terms in (8) since, in the beginning, $\sum_{i<j<k} N(E^{i,j,k})$ is at most $6m$ and each step decreases it by at least one.

5.3 The Two Definitions of Regularity Are Equivalent

We now prove that the definition of regularity 5.2 is equivalent to the previous definition of regularity for the class $\mathcal{F}^2$. Note that the latter one is formulated not only as a property of the function E but also as a property of its representation as a sum in (6). We will show equivalence for an arbitrary representation, which will imply that the definition of regularity for the class $\mathcal{F}^2$ depends only on E but not on the representation.

Let us consider a graph-representable function E from the class $\mathcal{F}^2$:

$$E(x_1,\dots,x_n) = \sum_i E^i(x_i) + \sum_{i<j} E^{i,j}(x_i, x_j).$$

Consider a projection where the two variables that are not fixed are x_i and x_j. By Lemma 5.6, the value of the functional π of this projection is equal to $\pi(E^{i,j})$ (all other terms yield zero). Therefore, this projection is regular if and only if $E^{i,j}(0,0) + E^{i,j}(1,1) \leq E^{i,j}(0,1) + E^{i,j}(1,0)$, which means that the two definitions are equivalent.

5.4 Example: Multicamera Scene Reconstruction

We now show an example of function from the class $\mathcal{F}^3$ that is potentially useful in vision. Consider the multicamera scene reconstruction problem described earlier. Similar to the data term depending on pairs of pixels, we can define a term depending on triples of pixels.[6] Let I_3 be a set of triples of "nearby" 3D points. These points will come from three different cameras, but they will share the same depth (i.e., the points are of the form $\langle p, f_q\rangle, \langle q, f_q\rangle, \langle r, f_r\rangle$, where $f_p = f_q = f_r$ and p, q, r are pixels from different cameras). The new data term will be

$$\sum_{\{\langle p,l\rangle,\langle q,l\rangle,\langle r,l\rangle \in I_3\}} D_{p,q,r,l}(f_p, f_q, f_r), \tag{9}$$

where

$$D_{p,q,r,l}(f_p, f_q, f_r) = D(p,q,r) \cdot T[f_p = f_q = f_r = l].$$

This term can be motivated as follows: If three pixels have similar intensities, then it is more likely that they see the same scene element than if only two pixels have similar intensities.

We now show the expansion move algorithm can be used for minimizing this new term, i.e., that the resulting energy function is regular. The proof proceeds similarly to the proof in Section 4.2.2. Consider a single term $D(p,q,r) \cdot T[f_p = f_q = f_r = l]$, which imposes a penalty if p, q, and r have the label l. We will have three binary variables which are 0 if the associated pixel keeps its original label in f and 1 if it acquires the new label α.

Again, there are two cases that must be analyzed separately: $l \neq \alpha$ and $l = \alpha$. Consider the case $l \neq \alpha$. If at least one of labels f_p, f_q, f_r is not l, then the term is zero for any values of binary variables. If $f_p = f_q = f_r = l$, then the penalty is only imposed when the binary variables are all 0. Graphically, we show this in Table 11. Now, consider the case $l = \alpha$. If at least one of the labels f_p, f_q, f_r is α then the penalty imposed is independent of one of the binary variables; therefore, this case reduces to a data term depending on two variables, which has been analyzed earlier. Suppose that all labels f_p, f_q, f_r are different from α. Then, the penalty is only imposed when the binary variables are all 1, which is written graphically in Table 12. We get the result similar to the previous one: The energy function is regular if $D(p,q,r)$ is nonpositive.

6 More General Classes of Energy Functions

Finally, we give a necessary condition for graph representability for arbitrary functions of binary variables.

6. We thank Rick Szeliski for discussions that have led to the development of this term.

TABLE 13

$$\begin{array}{|c|c|}\hline 0&0\\\hline 0&A\\\hline\end{array}=\bar{E}(0,0)+\begin{array}{|c|c|}\hline 0&-\bar{E}(0,1)\\\hline 0&-\bar{E}(0,1)\\\hline\end{array}+\begin{array}{|c|c|}\hline 0&0\\\hline -\bar{E}(1,0)&-\bar{E}(1,0)\\\hline\end{array}+\begin{array}{|c|c|}\hline \bar{E}(0,0)&\bar{E}(0,1)\\\hline \bar{E}(1,0)&\bar{E}(1,1)\\\hline\end{array}$$

Theorem 6.1 (regularity). *Let E be a function of binary variables. If E is not regular, then E is not graph-representable.*

This theorem will imply the corresponding directions of the $\mathcal{F}^2$ and $\mathcal{F}^3$ theorems (Theorems 4.1 and 5.3).

Definition 6.2. *Let $\mathcal{G}=(\mathcal{V},\mathcal{E})$ be a graph, $v_1,\ldots,v_k$ be a subset of vertices $\mathcal{V}$, and $\alpha_1,\ldots,\alpha_k$ be binary constants whose values are from $\{0,1\}$. We will define the graph $\mathcal{G}[x_1=\alpha_1,\ldots,x_k=\alpha_k]$ as follows: Its vertices will be the same as in $\mathcal{G}$ and its edges will be all edges of $\mathcal{G}$ plus additional edges corresponding to vertices $v_1,\ldots,v_k$: for a vertex v_i, we add the edge (s,v_i) if $\alpha_i=0$ or (v_i,t) if $\alpha_i=1$, with an infinite capacity.*

It should be obvious that these edges enforce constraints $f(v_1)=\alpha_1,\ \ldots,\ f(v_k)=\alpha_k$ in the minimum cut on $\mathcal{G}[x_1=\alpha_1,\ldots,x_k=\alpha_k]$, i.e., if $\alpha_i=0$, then $v_i\in S$ and if $\alpha_i=1$, then $v_i\in T$. (If, for example, $\alpha_i=0$ and $v_i\in T$, then the edge (s,v_i) must be cut yielding an infinite cost, so it would not the minimum cut.)

Now, we can give a definition of graph representability which is equivalent to Definition 3.1. This new definition will be more convenient for the proof.

Definition 6.3. *We say that the function E of n binary variables is exactly represented by the graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ and the set $\mathcal{V}_0\subset\mathcal{V}$ if for any configuration $\alpha_1,\ldots,\alpha_n$ the cost of the minimum cut on $\mathcal{G}[x_1=\alpha_1,\ldots,x_k=\alpha_k]$ is $E(\alpha_1,\ldots,\alpha_n)$.*

Lemma 6.4. *Any projection of a graph-representable function is graph-representable.*

Proof. Let E be a graph-representable function of n variables and the graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ and the set $\mathcal{V}_0$ represents E. Suppose that we fix variables $x_{i(1)},\ldots,x_{i(m)}$. It is straightforward to check that the graph $\mathcal{G}[x_{i(1)}=\alpha_{i(1)},\ldots,x_{i(m)}=\alpha_{i(m)}]$ and the set $\mathcal{V}_0'=\mathcal{V}_0-\{v_{i(1)},\ldots,v_{i(m)}\}$ represents the function $E'=E[x_{i(1)}=\alpha_{i(1)},\ldots,x_{i(m)}=\alpha_{i(m)}]$. □

This lemma implies that it suffices to prove Theorem 3.1 only for energy functions of two variables.

Let $\bar{E}(x_1,x_2)$ be a graph-representable function of two variables. Let us prove that $\bar{E}$ is regular, i.e., that $A\le 0$, where $A=\pi(\bar{E})=\bar{E}(0,0)+\bar{E}(1,1)-\bar{E}(0,1)-\bar{E}(1,0)$. Suppose this is not true: $A>0$.

In Table 13, all the functions on the right side are graph-representable, therefore, by the additivity theorem (see the Appendix), the function $\bar{E}$ is graph-representable as well, where, in Table 14, consider a graph $\mathcal{G}$ and a set $\mathcal{V}_0=\{v_1,v_2\}$ representing E. This means that there is a constant K such that $\mathcal{G},\mathcal{V}_0$ exactly represent $E'(x_1,x_2)=E(x_1,x_2)+K$ (Table 15).

The cost of the minimum s-t-cut on $\mathcal{G}$ is K (since this cost is just the minimum entry in the table for E'); hence, $K\ge 0$. Thus, the value of the maximum flow from s to t in $\mathcal{G}$ is K. Let $\mathcal{G}^0$ be the residual graph obtained from $\mathcal{G}$ after pushing the flow K. Let $E^0(x_1,x_2)$ be the function exactly represented by $\mathcal{G}^0,\mathcal{V}_0$.

By the definition of graph representability, $E'(\alpha_1,\alpha_2)$ is equal to the value of the minimum cut (or maximum flow) on the graph $\mathcal{G}[x_1=\alpha_1,x_2=\alpha_2]$. The following sequence of operations shows one possible way to push the maximum flow through this graph.

- First, we take the original graph $\mathcal{G}$ and push the flow K; then we get the residual graph $\mathcal{G}^0$. (This is equivalent to pushing flow through $\mathcal{G}[x_1=\alpha_1,x_2=\alpha_2]$, where we do not use edges corresponding to the constraints $x_1=\alpha_1$ and $x_2=\alpha_2$).
- Then, we add edges corresponding to these constraints; we get the graph $\mathcal{G}^0[x_1=\alpha_1,x_2=\alpha_2]$.
- Finally, we push the maximum flow possible through the graph $\mathcal{G}^0[x_1=\alpha_1,x_2=\alpha_2]$; the amount of this flow is $E^0(\alpha_1,\alpha_2)$ according to the definition of graph representability.

The total amount of flow pushed during all steps is $K+E^0(\alpha_1,\alpha_2)$; thus,

$$E'(\alpha_1,\alpha_2)=K+E^0(\alpha_1,\alpha_2)$$

or

$$E(\alpha_1,\alpha_2)=E^0(\alpha_1,\alpha_2).$$

We have proven that E is exactly represented by $\mathcal{G}^0$, $\mathcal{V}_0$.

The value of the minimum cut/maximum flow on $\mathcal{G}^0$ is 0 (it is the minimum entry in the table for E); thus, there is no augmenting path from s to t in $\mathcal{G}^0$. However, if we add the edges (v_1,t) and (v_2,t), then there will be an augmenting path from s to t in $\mathcal{G}^0[x_1=\alpha_1,x_2=\alpha_2]$ since $E(1,1)=A>0$. Hence, this augmenting path will contain at least one of these edges and, therefore, either v_1 or v_2 will be in the path. Let P be the part of this path going from the source until v_1 or v_2 is first encountered. Without loss of generality, we can assume that it will be v_1. Thus, P is an augmenting path from s to v_1 which does not contain edges that we added, namely, (v_1,t) and (v_2,t).

Finally, let us consider the graph $\mathcal{G}^0[x_1=1,x_2=0]$ which is obtained from $\mathcal{G}^0$ by adding edges (v_1,t) and (s,v_2) with infinite capacities. There is an augmenting path $\{P,(v_1,t)\}$ from the source to the sink in this graph; hence, the minimum cut/maximum flow on it is greater than zero or $E(1,0)>0$. We get a contradiction.

TABLE 14

$$E=\begin{array}{|c|c|}\hline 0&0\\\hline 0&A\\\hline\end{array}$$

TABLE 15

$$E'=\begin{array}{|c|c|}\hline K&K\\\hline K&K+A\\\hline\end{array}$$

7 Related Work

There is an interesting relationship between regular functions and submodular functions.[7] Let $\mathcal{S}$ be a finite set and $g : 2^{\mathcal{S}} \rightarrow \mathcal{R}$ be a real-valued function defined on the set of all subsets of $\mathcal{S}$. g is called *submodular* if for any $X, Y \subset \mathcal{S}$

$$g(X) + g(Y) \geq g(X \cup Y) + g(X \cap Y).$$

See [15], for example, for a discussion of submodular functions. An equivalent definition of submodular functions is that g is called submodular if, for any $X \subset \mathcal{S}$ and $i, j \in \mathcal{S} - X$,

$$g(X \cup \{j\}) - g(X) \geq g(X \cup \{i, j\}) - g(X \cup \{i\}).$$

Obviously, functions of subsets X of $\mathcal{S} = \{1, \ldots, n\}$ can be viewed as functions of n binary variables $(x_1, \ldots, x_n)$; the indicator variable x_i is 1 if i is included in X and 0 otherwise. Then, it is easy to see that the second definition of submodularity reduces to the definition of regularity. Thus, submodular functions and regular functions are essentially the same. We use different names to emphasize a technical distinction between them: Submodular functions are functions of subsets of $\mathcal{S}$ while regular functions are functions of binary variables. From our experience, the second point of view is much more convenient for vision applications.

Submodular functions have received significant attention in the combinatorial optimization literature. A remarkable fact about submodular functions is that they can be minimized in polynomial time [19], [23], [38]. A function g is assumed to be given as a *value oracle*, i.e., a "black box" which, for any input subset X returns the value $g(X)$.

Unfortunately, algorithms for minimizing arbitrary submodular functions are extremely slow. For example, the algorithm of [23] runs in $O(n^5 \min\{\log nM, n^2 \log n\})$ time, where M is an upper bound on $|g(X)|$ among $X \in 2^{\mathcal{S}}$. Thus, our work can be viewed as identifying an important subclass of submodular functions for which much faster algorithm can be used.

In addition, a relation between submodular functions and certain graph-representable functions called *cut functions* is already known. Using our terminology, cut functions can be defined as functions that can be represented by a graph without the source and the sink and without auxiliary vertices. It is well-known that cut functions are submodular. Cunningham [11] characterizes cut functions and gives a general-purpose graph construction for them.

It can be shown that the set of cut functions is a strict subset of $\mathcal{F}^2$. We allow a more general graph construction; as a result, we can minimize a larger class of functions, namely, regular functions in $\mathcal{F}^3$.

8 Summary of Graph Constructions

We now summarize the graph constructions used for regular functions. Software can be downloaded from http://www.cs.cornell.edu/~rdz/graphcuts.html that takes an energy function as input and automatically constructs the appropriate graph, then minimizes the energy.

7. We thank Yuri Boykov and Howard Karloff for pointing this out.

Recall that, for a function $E(x_1, \ldots, x_n)$, we define

$$\pi(E) = \sum_{x_1 \in \{0,1\}, \ldots, x_n \in \{0,1\}} \left(\Pi_{i=1}^{n} (-1)^{x_i}\right) E(x_1, \ldots, x_n).$$

The notation $edge(v, c)$ will mean that we add an edge (s, v) with the weight c if $c > 0$ or an edge (v, t) with the weight $-c$ if $c < 0$.

8.1 Regular Functions of One Binary Variable

Recall that all functions of one variable are regular. For a function $E(x_1)$, we construct a graph $\mathcal{G}$ with three vertices $\mathcal{V} = \{v_1, s, t\}$. There is a single edge $edge(v_1, E(1) - E(0))$.

8.2 Regular Functions of Two Binary Variables

We now show how to construct a graph $\mathcal{G}$ for a regular function $E(x_1, x_2)$ of two variables. It will contain four vertices: $\mathcal{V} = \{v_1, v_2, s, t\}$. The edges $\mathcal{E}$ are given below:

- $edge(v_1, E(1,0) - E(0,0))$;
- $edge(v_2, E(1,1) - E(1,0))$;
- (v_1, v_2) with the weight $-\pi(E)$.

8.3 Regular Functions of Three Binary Vvariables

We next show how to construct a graph $\mathcal{G}$ for a regular function $E(x_1, x_2, x_3)$ of three variables. It will contain five vertices: $\mathcal{V} = \{v_1, v_2, v_3, u, s, t\}$. If $\pi(E) \geq 0$, then the edges are

- $edge(v_1, E(1,0,1) - E(0,0,1))$;
- $edge(v_2, E(1,1,0) - E(1,0,0))$;
- $edge(v_3, E(0,1,1) - E(0,1,0))$;
- (v_2, v_3) with the weight $-\pi(E[x_1 = 0])$;
- (v_3, v_1) with the weight $-\pi(E[x_2 = 0])$;
- (v_1, v_2) with the weight $-\pi(E[x_3 = 0])$;
- (v_1, u), (v_2, u), (v_3, u), (u, t) with the weight $\pi(E)$.

If $\pi(E) < 0$, then the edges are

- $edge(v_1, E(1,1,0) - E(0,1,0))$;
- $edge(v_2, E(0,1,1) - E(0,0,1))$;
- $edge(v_3, E(1,0,1) - E(1,0,0))$;
- (v_3, v_2) with the weight $-\pi(E[x_1 = 1])$;
- (v_1, v_3) with the weight $-\pi(E[x_2 = 1])$;
- (v_2, v_1) with the weight $-\pi(E[x_3 = 1])$;
- (u, v_1), (u, v_2), (u, v_3), (s, u) with the weight $-\pi(E)$.

Appendix A
Proof of the NP-Hardness Theorem

We now give a proof of the NP-hardness theorem (Theorem 4.2), which shows that, without the regularity constraint, it is intractable to minimize energy functions in $\mathcal{F}^2$.

Proof. Adding functions of one variable does not change the class of functions that we are considering. Thus, we can assume without loss of generality that E^2 has the form that is shown in Table 16. (We can transform an arbitrary function of two variables to this form as follows: We subtract a constant from the first row to make the upper left element zero, then we subtract a constant from the second row to make the bottom left element zero, and, finally, we subtract a constant from the second column to make the upper right element zero.)

These transformations preserve the functional π, so E^2 is nonregular, which means that $A > 0$.

TABLE 16

$$E^2 = \begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & A \\ \hline \end{array}$$

We will prove the theorem by reducing the maximum independent set problem, which is known to be NP-hard, to our energy minimization problem.

Let an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the input to the maximum independent set problem. A subset $\mathcal{U} \subset \mathcal{V}$ is said to be independent if, for any two vertices $u, v \in \mathcal{U}$, the edge (u, v) is not in $\mathcal{E}$. The goal is to find an independent subset $\mathcal{U}^* \subset \mathcal{V}$ of maximum cardinality. We construct an instance of the energy minimization problem as follows: There will be $n = |\mathcal{V}|$ binary variables $x_1, \ldots, x_n$ corresponding to the vertices $v_1, \ldots, v_n$ of $\mathcal{V}$. Let us consider the energy

$$E(x_1, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{(i,j)\in\mathcal{N}} E^2(x_i, x_j),$$

where $E^i(x_i) = -\frac{A}{2n} \cdot x_i$ and $\mathcal{N} = \{(i, j) \,|\, (v_i, v_j) \in \mathcal{E}\}$.

There is a one-to-one correspondence between all configurations $(x_1, \ldots, x_n)$ and subsets $\mathcal{U} \subset \mathcal{V}$: A vertex v_i is in $\mathcal{U}$ if and only if $x_i = 1$. Moreover, the first term of $E(x_1, \ldots, x_n)$ is $-\frac{A}{2n}$ times the cardinality of U (which cannot be less than $-\frac{A}{2}$) and the second term is 0 if U is independent and at least A otherwise. Thus, minimizing the energy yields the independent subset of maximum cardinality. □

APPENDIX B
THE ADDITIVITY THEOREM

We now prove the additivity theorem, which plays an important role in our constructions.

Theorem (additivity). *The sum of two graph-representable functions is graph-representable.*

It is important to note that the proof of this theorem is constructive. The construction has a particularly simple form if the graphs representing the two functions are defined on the same set of vertices (i.e., they differ only in their edge weights). In this case, by simply adding the edge weights together, we obtain a graph that represents the sum of the two functions. If one of the graphs has no edge between two vertices, we can add an edge with weight 0.

Proof. Let us assume for simplicity of notation that E' and E'' are functions of all n variables: $E' = E'(x_1, \ldots, x_n)$, $E'' = E''(x_1, \ldots, x_n)$. By the definition of graph representability, there exist constants K', K'', graphs $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$, $\mathcal{G}'' = (\mathcal{V}'', \mathcal{E}'')$, and the set $\mathcal{V}_0 = \{v_1, \ldots, v_n\}$, $\mathcal{V}_0 \subset \mathcal{V}' - \{s, t\}$, $\mathcal{V}_0 \subset \mathcal{V}'' - \{s, t\}$ such that $E' + K'$ is exactly represented by $\mathcal{G}'$, $\mathcal{V}_0$ and $E'' + K''$ is exactly represented by $\mathcal{G}''$, $\mathcal{V}_0$. We can assume that the only common vertices of $\mathcal{G}'$ and $\mathcal{G}''$ are $\mathcal{V}_0 \cup \{s, t\}$. Let us construct the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as the combined graph of $\mathcal{G}'$ and $\mathcal{G}''$: $\mathcal{V} = \mathcal{V}' \cup \mathcal{V}''$, $\mathcal{E} = \mathcal{E}' \cup \mathcal{E}''$.

Let $\tilde{E}$ be the function that $\mathcal{G}$, $\mathcal{V}_0$ exactly represents. Let us prove that $\tilde{E} \equiv E + (K' + K'')$ (and, therefore, E is graph-representable).

Consider a configuration $x_1, \ldots, x_n$. Let $C' = S', T'$ be the cut on $\mathcal{G}'$ with the smallest cost among all cuts for which $v_i \in S'$ if $x_i = 0$ and $v_i \in T'$ if $x_i = 1$ $(1 \le i \le n)$. According to the definition of graph representability,

$$E'(x_1, \ldots, x_n) + K' = \sum_{u\in S', v\in T', (u,v)\in\mathcal{E}'} c(u, v).$$

Let $C'' = S'', T''$ be the cut on $\mathcal{G}''$ with the smallest cost among all cuts for which $v_i \in S''$ if $x_i = 0$, and $v_i \in T''$ if $x_i = 1$ $(1 \le i \le n)$. Similarly,

$$E''(x_1, \ldots, x_n) + K'' = \sum_{u\in S'', v\in T'', (u,v)\in\mathcal{E}''} c(u, v).$$

Let $S = S' \cup S''$, $T = T' \cup T''$. It is easy to check that $C = S, T$ is a cut on $\mathcal{G}$. Thus,

$$\tilde{E}(x_1, \ldots, x_n) \le \sum_{u\in S, v\in T, (u,v)\in\mathcal{E}} c(u, v)$$

$$= \sum_{u\in S', v\in T', (u,v)\in\mathcal{E}'} c(u, v) + \sum_{u\in S'', v\in T'', (u,v)\in\mathcal{E}''} c(u, v)$$

$$= (E'(x_1, \ldots, x_n) + K') + (E''(x_1, \ldots, x_n) + K'')$$
$$= E(x_1, \ldots, x_n) + (K' + K'').$$

Now, let $C = S, T$ be the cut on $\mathcal{G}$ with the smallest cost among all cuts for which $v_i \in S$ if $x_i = 0$ and $v_i \in T$ if $x_i = 1$ $(1 \le i \le n)$ and let $S' = S \cap \mathcal{V}'$, $T' = T \cap \mathcal{V}'$, $S'' = S \cap \mathcal{V}''$, $T'' = T \cap \mathcal{V}'$. It is easy to see that $C' = S', T'$, and $C'' = S'', T''$ are cuts on $\mathcal{G}'$ and $\mathcal{G}''$, respectively. According to the definition of graph representability,

$$E(x_1, \ldots, x_n) + (K' + K'') = (E'(x_1, \ldots, x_n) + K')$$
$$+ (E''(x_1, \ldots, x_n) + K'') \le \sum_{u\in S', v\in T', (u,v)\in\mathcal{E}'} c(u, v)$$
$$+ \sum_{u\in S'', v\in T'', (u,v)\in\mathcal{E}''} c(u, v)$$
$$= \sum_{u\in S, v\in T, (u,v)\in\mathcal{E}'} c(u, v) = \tilde{E}(x_1, \ldots, x_n).$$

□

ACKNOWLEDGMENTS

The authors are grateful to Olga Veksler and Yuri Boykov for their careful reading of this paper and for valuable comments that greatly improved its readability. They would like to thank Jon Kleinberg and Eva Tardos for helping them understand the relationship between their work and submodular function minimization. They would also like to thank Ian Jermyn and several anonymous reviewers for helping them clarify the paper's presentation. This research was supported by US National Science Foundation grants IIS-9900115 and CCR-0113371 and by a grant from Microsoft Research. A preliminary version of this paper appeared in the *Proceedings of the European Conference on Computer Vision*, May 2002.

REFERENCES

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, 1993.

[2] A. Amini, T. Weymouth, and R. Jain, "Using Dynamic Programming for Solving Variational Problems in Vision," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, no. 9, pp. 855-867, Sept. 1990.

[3] S. Barnard, "Stochastic Stereo Matching Over Scale," *Int'l J. Computer Vision,* vol. 3, no. 1, pp. 17-32, 1989.

[4] S. Birchfield and C. Tomasi, "Multiway Cut for Stereo and Motion with Slanted Surfaces," *Proc. Int'l Conf. Computer Vision,* pp. 489-495, 1999.

[5] Y. Boykov and M.-P. Jolly, "Interactive Organ Segmentation Using Graph Cuts," *Proc. Medical Image Computing and Computer-Assisted Intervention,* pp. 276-286, 2000.

[6] Y. Boykov and M.-P. Jolly, "Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images," *Proc. Int'l Conf. Computer Vision,* pp. 105-112, 2001.

[7] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Computer Vision," *Proc. Int'l Workshop Energy Minimization Methods in Computer Vision and Pattern Recognition,* Lecture Notes in Computer Science, pp. 359-374, Springer-Verlag, Sept. 2001.

[8] Y. Boykov and V. Kolmogorov, "Computing Geodesics and Minimal Surfaces via Graph Cuts," *Proc. Int'l Conf. Computer Vision,* pp. 26-33, 2003.

[9] Y. Boykov, O. Veksler, and R. Zabih, "Markov Random Fields with Efficient Approximations," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 648-655, 1998.

[10] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," *Proc. IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 11, pp. 1222-1239, Nov. 2001.

[11] W.H. Cunningham, "Minimum Cuts, Modular Functions, and Matroid Polyhedra," *Networks,* vol. 15, pp. 205-215, 1985.

[12] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis, "The Complexity of Multiway Cuts," *Proc. ACM Symp. Theory of Computing,* pp. 241-251, 1992.

[13] J. Dias and J. Leitao, "The $\mathcal{Z}\pi M$ Algorithm: A Method for Interferometric Image Reconstruction in SAR/SAS" *IEEE Trans. Image Processing,* vol. 11, no. 4, pp. 408-422, Apr. 2002.

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

[15] S. Fujishige, " Submodular Functions and Optimization," vol. 47, *Annals of Discrete Math.,* North Holland, 1990.

[16] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 6, pp. 721-741, 1984.

[17] A. Goldberg and R. Tarjan, "A New Approach to the Maximum Flow Problem," *J. ACM,* vol. 35, no. 4, pp. 921-940, Oct. 1988.

[18] D. Greig, B. Porteous, and A. Seheult, "Exact Maximum A Posteriori Estimation for Binary Images," *J. Royal Statistical Soc., Series B,* vol. 51, no. 2, pp. 271-279, 1989.

[19] M. Grötschel, L. Lovasz, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, 1988.

[20] H. Ishikawa and D. Geiger, "Occlusions, Discontinuities, and Epipolar Lines in Stereo," *Proc. European Conf. Computer Vision,* pp. 232-248, 1998.

[21] H. Ishikawa and D. Geiger, "Segmentation by Grouping Junctions," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 125-131, 1998.

[22] H. Ishikawa, "Exact Optimization for Markov Random Fields with Convex Priors," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 10, pp. 1333-1336, Oct. 2003.

[23] S. Iwata, L. Fleischer, and S. Fujishige, "A Combinatorial, Strongly Polynomial Algorithm for Minimizing Submodular Functions," *J. ACM,* vol. 48, no. 4, pp. 761-777, July 2001.

[24] J. Kim, V. Kolmogorov, and R. Zabih, "Visual Correspondence Using Energy Minimization and Mutual Information," *Proc. Int'l Conf. Computer Vision,* pp. 1033-1040, 2003.

[25] J. Kim and R. Zabih, "Automatic Segmentation of Contrast-Enhanced Image Sequences," *Proc. Int'l Conf. Computer Vision,* pp. 502-509, 2003.

[26] J. Kim, J. Fisher, A. Tsai, C. Wible, A. Willsky, and W. Wells, "Incorporating Spatial Priors into an Information Theoretic Approach for FMRI Data Analysis," *Proc. Medical Image Computing and Computer-Assisted Intervention,* pp. 62-71, 2000.

[27] V. Kolmogorov and R. Zabih, "Visual Correspondence with Occlusions Using Graph Cuts," *Proc. Int'l Conf. Computer Vision,* pp. 508-515, 2001.

[28] V. Kolmogorov and R. Zabih, "Multi-Camera Scene Reconstruction via Graph Cuts," *Proc. European Conf. Computer Vision,* vol. 3, pp. 82-96, 2002.

[29] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut Textures: Image and Video Synthesis Using Graph Cuts," *ACM Trans. Graphics, Proc. SIGGRAPH 2003,* July 2003.

[30] C.-H. Lee, D. Lee, and M. Kim, "Optimal Task Assignment in Linear Array Networks," *IEEE Trans. Computers,* vol. 41, no. 7, pp. 877-880, July 1992.

[31] S. Li, *Markov Random Field Modeling in Computer Vision.* Springer-Verlag, 1995.

[32] M.H. Lin, "Surfaces with Occlusions from Layered Stereo," PhD thesis, Stanford Univ., Dec. 2002.

[33] I. Milis, "Task Assignment in Distributed Systems Using Network Flow Methods," *Proc. Combinatorics and Computer Science,* Lecture Notes in Computer Science, pp. 396-405, Springer-Verlag, 1996.

[34] T. Poggio, V. Torre, and C. Koch, "Computational Vision and Regularization Theory," *Nature,* vol. 317, pp. 314-319, 1985.

[35] S. Roy, "Stereo without Epipolar Lines: A Maximum Flow Formulation," *Int'l J. Computer Vision,* vol. 1, no. 2, pp. 1-15, 1999.

[36] S. Roy and I. Cox, "A Maximum-Flow Formulation of the n-Camera Stereo Correspondence Problem," *Proc. Int'l Conf. Computer Vision,* 1998.

[37] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *Int'l J. Computer Vision,* vol. 47, pp. 7-42, Apr. 2002.

[38] A. Schrijver, "A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time," *J. Combinatorial Theory,* vol. B 80, pp. 346-355, 2000.

[39] D. Snow, P. Viola, and R. Zabih, "Exact Voxel Occupancy with Graph Cuts," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 345-352, 2000.

[40] H.S. Stone, "Multiprocessor Scheduling with the Aid of Network Flow Algorithms," *IEEE Trans. Software Eng.,* pp. 85-93, 1977.

[41] R. Szeliski and R. Zabih, "An Experimental Comparison of Stereo Algorithms," *Proc. Vision Algorithms: Theory and Practice,* Lecture Notes in Computer Science, B. Triggs, A. Zisserman, and R. Szeliski, eds.,vol. 1883, pp. 1-19, Springer-Verlag, Sept. 1999.

Vladimir Kolmogorov received the MS degree from the Moscow Institute of Physics and Technology in applied mathematics and physics in 1999 and the PhD degree in computer science from Cornell University in January 2004. He is currently a postdoctoral researcher at Microsoft Research, Cambridge, United Kingdom. His research interests are graph algorithms, stereo correspondence, image segmentation, parameter estimation, and mutual information. Two of his papers (written with Ramin Zabih) received a best paper award at the European Conference on Computer Vision, 2002. He is a member of the IEEE and the IEEE Computer Society.

Ramin Zabih attended the Massachusetts Institute of Technology as an undergraduate, where he received BS degrees in computer science and mathematics and the MSc degree in computer science. After earning the PhD degree in computer science from Stanford University in 1994, he joined the faculty at Cornell University, where he is currently an associate professor of computer science. Since 2001, he has held a joint appointment as an associate professor of radiology at Cornell Medical School. His research interests lie in early vision and its applications, especially in medicine. He has served on numerous program committees, including the International Conference on Computer Vision (ICCV) in 1999, 2001, and 2003, and the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 1997, 2000, 2001, and 2003. He coedited a special issue of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* in October 2001. He is a member of the IEEE and the IEEE Computer Society.