# *Interactive Graph Cuts* for Optimal Boundary & Region Segmentation of Objects in N-D Images

Yuri Y. Boykov Marie-Pierre Jolly

Imaging and Visualization Department
Siemens Corporate Research
755 College Road East, Princeton, NJ 08540, USA
yuri@scr.siemens.com

## Abstract

*In this paper we describe a new technique for general purpose interactive segmentation of N-dimensional images. The user marks certain pixels as "object" or "background" to provide hard constraints for segmentation. Additional soft constraints incorporate both boundary and region information. Graph cuts are used to find the globally optimal segmentation of the N-dimensional image. The obtained solution gives the best balance of boundary and region properties among all segmentations satisfying the constraints. The topology of our segmentation is unrestricted and both "object" and "background" segments may consist of several isolated parts. Some experimental results are presented in the context of photo/video editing and medical image segmentation. We also demonstrate an interesting Gestalt example. A fast implementation of our segmentation method is possible via a new max-flow algorithm in [2].*

## 1. Introduction

Interactive segmentation is becoming more and more popular to alleviate the problems inherent to fully automatic segmentation which seems to never be perfect. Our goal is a general purpose interactive segmentation technique that divides an image into two segments: "object" and "background". A user imposes certain hard constraints for segmentation by indicating certain pixels (seeds) that absolutely have to be part of the object and certain pixels that have to be part of the background. Intuitively, these hard constraints provide clues on what the user intends to segment. The rest of the image is segmented automatically by computing a global optimum among all segmentations satisfying the hard constraints. The cost function is defined in terms of boundary and region properties of the segments. These properties can be viewed as soft constraints for segmentation. A globally optimal segmentation can be very efficiently recomputed when the user adds or removes any hard constraints (seeds). This allows the user to get any desired segmentation results quickly via very intuitive interactions. Our method applies to N-D images (volumes).

One of the main advantages of our interactive segmentation method is that it provides a globally optimal solution for an N-dimensional segmentation when the cost function is clearly defined. Some earlier techniques (snakes [14, 4], deformable templates [21], shortest path [15], ratio regions [5], and other [13]) can do that only in 2D applications when a segmentation boundary is a 1D curve. Many techniques either don't have a clear cost function at all (region growing, split and merge, see Chapter 10 in [11]) or compute only an approximate solution (e.g. a local minimum) that can be arbitrarily far from the global optimum (region competition [22], level set methods [16], normalized cuts [18]). Global properties of such segmentations may be difficult to analyze or predict. Imperfections in the result might come from deficiencies at the minimization stage. In contrast, imperfections of a globally optimal solution are directly related to the definition of the cost function. Thus, the segmentation can be controlled more reliably.

It is also important that the cost function that we use as a soft constraint for segmentation is general enough to include both region and boundary properties of segments. In fact, our cost function is similar to one in [9] obtained in a context of MAP-MRF estimation. Consider an arbitrary set of data elements $\mathcal{P}$ and some neighborhood system represented by a set $\mathcal{N}$ of all unordered[1] pairs $\{p, q\}$

[1]For simplicity, we present the case of *unordered* pairs of neighbors $\{p, q\}$. If pairs of neighbors are *ordered* then we have two distinct pairs $(p, q)$ and $(q, p)$. This would allow different (directed) discontinuity penalties for two cases when $p \in$ "object", $q \in$ "background" and when $p \in$ "background", $q \in$ "object". To set such directed costs one should have prior information on properties of the boundary from object to background. Algorithmically, generalization from unordered to ordered neighbors means switching from undirected (presented here) to directed graphs.

of neighboring elements in $\mathcal{P}$. For example, $\mathcal{P}$ can contain pixels (or voxels) in a 2D (or 3D) grid and $\mathcal{N}$ can contain all unordered pairs of neighboring pixels (voxels) under a standard 8- (or 26-) neighborhood system. Let $A = (A_1, \ldots, A_p, \ldots, A_{|\mathcal{P}|})$ be a binary vector whose components $A_p$ specify assignments to pixels $p$ in $\mathcal{P}$. Each $A_p$ can be either "obj" or "bkg" (abbreviations of "object" and "background"). Vector $A$ defines a segmentation. Then, the soft constraints that we impose on boundary and region properties of $A$ are described by the cost function $E(A)$:

$$E(A) = \lambda \cdot R(A) + B(A) \tag{1}$$

where

$$R(A) = \sum_{p \in \mathcal{P}} R_p(A_p) \tag{2}$$

$$B(A) = \sum_{\{p,q\} \in \mathcal{N}} B_{\{p,q\}} \cdot \delta(A_p, A_q) \tag{3}$$

and

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

The coefficient $\lambda \geq 0$ in (1) specifies a relative importance of the region properties term $R(A)$ versus the boundary properties term $B(A)$. The regional term $R(A)$ assumes that the the individual penalties for assigning pixel $p$ to "object" and "background", correspondingly $R_p(\text{"obj"})$ and $R_p(\text{"bkg"})$, are given. For example, $R_p(\cdot)$ may reflect on how the intensity of pixel $p$ fits into a known intensity model (e.g. histogram) of the object and background.

The term $B(A)$ comprises the "boundary" properties of segmentation $A$. Coefficient $B_{\{p,q\}} \geq 0$ should be interpreted as a penalty for a discontinuity between $p$ and $q$. Normally, $B_{\{p,q\}}$ is large when pixels $p$ and $q$ are similar (e.g. in their intensity) and $B_{\{p,q\}}$ is close to zero when the two are very different. The penalty $B_{\{p,q\}}$ can also decrease as a function of distance between $p$ and $q$. Costs $B_{\{p,q\}}$ may be based on local intensity gradient, Laplacian zero-crossing, gradient direction, and other criteria (e.g. [15]).

The algorithm in [9] computes a globally optimal binary segmentation minimizing the energy similar to (1) when no hard constraints are placed. In contrast, our goal is to compute the global minimum of (1) among all segmentations that satisfy additional hard constraints imposed by a user.

There are different types of hard constraints that were proposed in the past for interactive segmentation methods. For example, for intelligent scissors [15] and live wire [6], the user has to indicate certain pixels where the segmentation boundary should pass. The segmentation boundary is then computed as the "shortest path" between the marked pixels according to some energy function based on image gradient. One difficulty with such hard constraints is that the user inputs have to be very accurately positioned at the desired boundary. Moreover, this approach can not be easily generalized to 3D data.

We consider hard constraints that indicate segmentation regions rather than the boundary. We assume that some pixels were marked as internal and some as external for the given object of interest. The subsets of marked pixels will be referred to as "object" and "background" seeds. The segmentation boundary can be anywhere but it has to separate the object seeds from the background seeds. Note that the seeds can be loosely positioned inside the object and background regions. Our segmentation technique is quite stable and normally produces the same results regardless of particular seed positioning within the same image object.

Obviously, the hard constraints by themself are not enough to obtain a good segmentation. A segmentation method decides how to segment unmarked pixels. [10] and [17] use the same type of hard constraints as us but they do not employ a clear cost function and segment unmarked pixels based on variations of "region growing". Since the properties of segmentation boundary are not optimized, the results are prone to "leaking" where the boundary between objects is blurry. In contrast, we combine the hard constraints as above with energy (1) that incorporates region and boundary properties of segments.

In this paper we show how to generalize the algorithm in [9] to combine soft constraints encoded in (1) with user defined hard constraints. We also show how the optimal segmentations can be efficiently recomputed if the hard constraints are added or changed. Note that initial segmentation may not be perfect. After reviewing it, the user can specify additional object and background seeds depending on the observed results. To incorporate these new seeds the algorithm can efficiently adjust the current segmentation without recomputing the whole solution from scratch.

Our technique is based on powerful graph cut algorithms from combinatorial optimization [7, 8]. Our implementation uses a new version of "max-flow" algorithm reported in [2]. In the next section we explain the terminology for graph cuts and provide some background information on previous computer vision techniques relying on graph cuts. The details of our segmentation method and its correctness are shown in Section 3. Section 4 provides a number of examples where we apply our technique to photo/video-editing and to segmentation of medical images/volumes. We demonstrate that with a few simple and intuitive manipulations a user can always segment an object of interest as precisely as (s)he wants. Note that some additional experimental results can be found in our preliminary work [1] which can be seen as a restricted case where only boundary term is present in energy (1). Information on possible extensions and future work is given in Section 5.

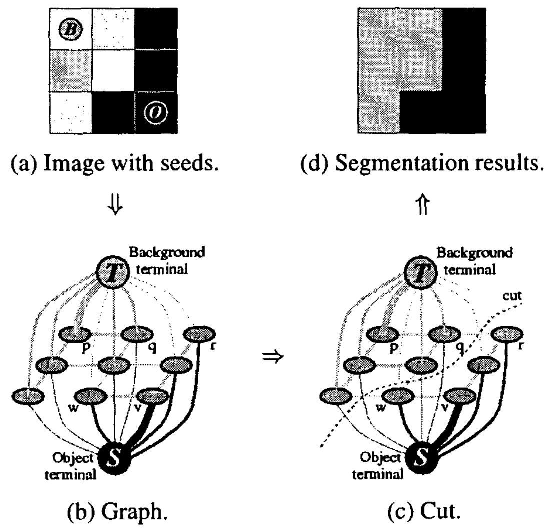


**Figure 1. A simple 2D segmentation example for a $3 \times 3$ image. The seeds are $\mathcal{O} = \{v\}$ and $\mathcal{B} = \{p\}$. The cost of each edge is reflected by the edge's thickness. The regional term (2) and hard constraints (4,5) define the costs of t-links. The boundary term (3) defines the costs of n-links. Inexpensive edges are attractive choices for the minimum cost cut.**

## 2. Graph Cuts and Computer Vision

First, we describe the basic terminology that pertains to graph cuts in the context of our segmentation method. An undirected graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ is defined as a set of nodes (vertices $\mathcal{V}$) and a set of undirected edges ($\mathcal{E}$) that connect these nodes. An example of a graph that we use in this paper is shown in Figure 1(b). Each edge $e \in \mathcal{E}$ in the graph is assigned a nonnegative weight (cost) $w_e$. There are also two special nodes called terminals. A cut is a subset of edges $C \subset \mathcal{E}$ such that the terminals become separated on the induced graph $\mathcal{G}(C) = \langle \mathcal{V}, \mathcal{E} \backslash C \rangle$. It is normal in combinatorial optimization to define the cost of a cut as the sum of the costs of the edges that it severs

$$|C| = \sum_{e \in C} w_e.$$

Graph cut formalism is well suited for segmentation of images. In fact, it is completely appropriate for N-dimensional volumes. The nodes of the graph can represent pixels (or voxels) and the edges can represent any neighborhood relationship between the pixels. A cut partitions the nodes in the graph. As illustrated in Figure 1 (c-d), this partitioning corresponds to a segmentation of an underlying image or volume. A minimum cost cut generates a segmentation that is optimal in terms of properties that are built into the edge weights.

Our technique is based on a well-known combinatorial optimization fact that a globally minimum cut of a graph with two terminals can be computed efficiently in low-order polynomial time [7, 8, 2]. In Section 3 we show how to set a two terminal graph so that the minimum cut would give a segmentation that minimizes (1) among all segmentations satisfying the given hard constraints.

It should be noted that graph cuts were used for image segmentation before. In [20] the image is optimally divided into $K$ parts to minimize the maximum cut between the segments. In this formulation, however, the segmentation is strongly biased to very small segments. Shi and Malik [18] try to solve this problem by normalizing the cost of a cut. The resulting optimization problem is NP-hard and they use an approximation technique. In [9, 3, 12, 19] graph cuts are applied to minimize certain energy functions used in image restoration, stereo, 3D object reconstruction, and other problems in computer vision.

A fast implementation of theoretically polynomial graph cut algorithms can be an issue. The most straight-forward implementations of the standard graph cut algorithms, e.g. max-flow [7] or push-relabel [8], can be slow. The experiments in [2] compare several well-known "tuned" versions of these standard algorithms in the context of graph based methods in vision. The same paper also describes a new version of the max-flow algorithm that (on typical in vision examples) significantly outperformed the standard techniques. Our implementation of the interactive segmentation method of this paper uses the new graph cut algorithm from [2].

## 3. Segmentation Technique

In this section we provide algorithmic details about our segmentation technique. Assume that $\mathcal{O}$ and $\mathcal{B}$ denote the subsets of pixels marked as "object" and "background" seeds. Naturally, the subsets $\mathcal{O} \subset \mathcal{P}$ and $\mathcal{B} \subset \mathcal{P}$ are such that $\mathcal{O} \cap \mathcal{B} = \emptyset$. Remember that our goal is to compute the global minimum of (1) among all segmentations $A$ satisfying hard constraints

$$\forall p \in \mathcal{O}, \quad A_p = \text{"obj"} \tag{4}$$
$$\forall p \in \mathcal{B}, \quad A_p = \text{"bkg"}. \tag{5}$$

The general work flow is shown in Figure 1. Given an image (Figure 1(a)) we create a graph with two terminals (Figure 1(b)). The edge weights reflect the parameters in the regional (2) and the boundary (3) terms of the cost function, as well as the known positions of seeds in the image.

The next step is to compute the globally optimal minimum cut (Figure 1(c)) separating two terminals. This cut gives a segmentation (Figure 1(d)) of the original image. In the simplistic example of Figure 1 the image is divided into exactly one "object" and one "background" regions. In general, our segmentation method generates binary segmentation with arbitrary topological properties. Examples in Section 4 illustrate that object and background segments may consist of several isolated connected blobs in the image.

Below we describe the details of the graph and prove that the obtained segmentation is optimal. To segment a given image we create a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ with nodes corresponding to pixels $p \in \mathcal{P}$ of the image. There are two additional nodes: an "object" terminal (a source $S$) and a "background" terminal (a sink $T$). Therefore,

$$\mathcal{V} = \mathcal{P} \cup \{S, T\}.$$

The set of edges $\mathcal{E}$ consists of two types of undirected edges: *n-links* (neighborhood links) and *t-links* (terminal links). Each pixel $p$ has two t-links $\{p, S\}$ and $\{p, T\}$ connecting it to each terminal. Each pair of neighboring pixels $\{p, q\}$ in $\mathcal{N}$ is connected by an n-link. Without introducing any ambiguity, an n-link connecting a pair of neighbors $p$ and $q$ will be denoted by $\{p, q\}$. Therefore,

$$\mathcal{E} = \mathcal{N} \bigcup_{p \in \mathcal{P}} \{\{p, S\}, \{p, T\}\}.$$

The following table gives weights of edges in $\mathcal{E}$

| edge | weight (cost) | for |
|---|---|---|
| $\{p, q\}$ | $B_{\{p,q\}}$ | $\{p, q\} \in \mathcal{N}$ |
| $\{p, S\}$ | $\lambda \cdot R_p(\text{"bkg"})$ | $p \in \mathcal{P},\ p \notin \mathcal{O} \cup \mathcal{B}$ |
| | $K$ | $p \in \mathcal{O}$ |
| | $0$ | $p \in \mathcal{B}$ |
| $\{p, T\}$ | $\lambda \cdot R_p(\text{"obj"})$ | $p \in \mathcal{P},\ p \notin \mathcal{O} \cup \mathcal{B}$ |
| | $0$ | $p \in \mathcal{O}$ |
| | $K$ | $p \in \mathcal{B}$ |

where

$$K = 1 + \max_{p \in \mathcal{P}} \sum_{q:\ \{p,q\} \in \mathcal{N}} B_{\{p,q\}}.$$

The graph $\mathcal{G}$ is now completely defined. We draw the segmentation boundary between the object and the background by finding the minimum cost cut on the graph $\mathcal{G}$. The minimum cost cut $\hat{C}$ on $\mathcal{G}$ can be computed exactly in polynomial time via algorithms for two terminal graph cuts (see Section 2) assuming that the edge weights specified in the table above are non-negative[2].

Below we state exactly how the minimum cut $\hat{C}$ defines a segmentation $\hat{A}$ and prove this segmentation is optimal. We need one technical lemma. Assume that $\mathcal{F}$ denotes a set of all *feasible* cuts $C$ on graph $\mathcal{G}$ such that

- $C$ severs exactly one t-link at each $p$
- $\{p, q\} \in C$ iff $p, q$ are t-linked to different terminals
- if $p \in \mathcal{O}$ then $\{p, T\} \in C$
- if $p \in \mathcal{B}$ then $\{p, S\} \in C$.

**Lemma 1** *The minimum cut on $\mathcal{G}$ is feasible, i.e. $\hat{C} \in \mathcal{F}$.*

PROOF: $\hat{C}$ severs et least one t-link at each pixel since it is a cut that separates the terminals. On the other hand, it can not sever both t-links. In such a case it would not be minimal since one of the t-links could be returned. Similarly, a minimum cut should sever an n-link $\{p, q\}$ if $p$ and $q$ are connected to the opposite terminals just because any cut must separate the terminals. If $p$ and $q$ are connected to the same terminal then $\hat{C}$ should not sever unnecessary n-link $\{p, q\}$ due to its minimality. The last two properties are true for $\hat{C}$ because the constant $K$ is larger than the sum of all n-links costs for any given pixel $p$. For example, if $p \in \mathcal{O}$ and $\hat{C}$ severs $\{p, S\}$ (costs $K$) then we would construct a smaller cost cut by restoring $\{p, S\}$ and severing all n-links from $p$ (costs less then $K$) as well as the opposite t-link $\{p, T\}$ (zero cost). ∎

For any feasible cut $C \in \mathcal{F}$ we can define a unique corresponding segmentation $A(C)$ such that

$$A_p(C) = \begin{cases} \text{"obj"}, & \text{if } \{p, T\} \in C \\ \text{"bkg"}, & \text{if } \{p, S\} \in C. \end{cases} \tag{6}$$

The definition above is coherent since any feasible cut severs exactly one of the two t-links at each pixel $p$. The lemma showed that a minimum cut $\hat{C}$ is feasible. Thus, we can define a corresponding segmentation $\hat{A} = A(\hat{C})$. The next theorem completes the description of our algorithm.

**Theorem 1** *The segmentation $\hat{A} = A(\hat{C})$ defined by the minimum cut $\hat{C}$ as in (6) minimizes (1) among all segmentations satisfying constraints (4,5).*

[2]In fact, our technique does require that the penalties $B_{\{p,q\}}$ be non-negative. The restriction that $R_p(\cdot)$ should be non-negative is easy to avoid. For any pixel $p$ we can always add a large enough positive constant to both $R_p(\text{"obj"})$ and $R_p(\text{"bkg"})$ to make sure that they become non-negative. This operation can not change the minimum $A$ of $E(A)$ in (1) since it is equivalent to adding a constant to the whole energy function.

PROOF: Using the table of edge weights, definition of feasible cuts $\mathcal{F}$, and equation (6) one can show that a cost of any $C \in \mathcal{F}$ is

$$\begin{aligned} |C| &= \sum_{p \notin \mathcal{O} \cup \mathcal{B}} \lambda \cdot R_p(A_p(C)) + \\ & \quad \sum_{\{p,q\} \in \mathcal{N}} B_{\{p,q\}} \cdot \delta(A_p(C), A_q(C)) \\ &= E(A(C)) - \sum_{p \in \mathcal{O}} \lambda R_p(\text{"obj"}) - \sum_{p \in \mathcal{B}} \lambda R_p(\text{"bkg"}). \end{aligned}$$

Therefore, $|C| = E(A(C)) - const(C)$. Note that for any $C \in \mathcal{F}$ assignment $A(C)$ satisfies constraints (4,5). In fact, equation (6) gives a one-to-one correspondence between the set of all feasible cuts in $\mathcal{F}$ and the set $\mathcal{H}$ of all assignments $A$ that satisfy hard constraints (4,5). Then,

$$\begin{aligned} E(\hat{A}) &= |\hat{C}| + const = \min_{C \in \mathcal{F}} |C| + const = \\ &= \min_{C \in \mathcal{F}} E(A(C)) = \min_{A \in \mathcal{H}} E(A) \end{aligned}$$

and the theorem is proved. ■

To conclude this section we would like to show that our algorithm can efficiently adjust the segmentation to incorporate any additional seeds that the user might interactively add. To be specific, assume that a max-flow algorithm is used to determine the minimum cut on $\mathcal{G}$. The max-flow algorithm gradually increases the flow sent from the source $S$ to the sink $T$ along the edges in $\mathcal{G}$ given their capacities (weights). Upon termination the maximum flow saturates[3] the graph. The saturated edges correspond to the minimum cost cut on $\mathcal{G}$ giving us an optimal segmentation.

Assume now that an optimal segmentation is already computed for some initial set of seeds. A user adds a new "object" seed to pixel $p$ that was not previously assigned any seed. We need to change the costs for two t-links at $p$

| t-link | initial cost | new cost |
|---|---|---|
| $\{p, S\}$ | $\lambda R_p(\text{"bkg"})$ | K |
| $\{p, T\}$ | $\lambda R_p(\text{"obj"})$ | 0 |

and then compute the maximum flow (minimum cut) on the new graph. In fact, we can start from the flow found at the end of initial computation. The only problem is that reassignment of edge weights as above reduces capacities of some edges. If there is a flow through such an edge then we may break the flow consistency. Increasing an edge capacity, on the other hand, is never a problem. Then, we can solve the problem as follows.

To accommodate the new "object" seed at pixel $p$ we increase the t-links weights according to the table

[3]See [7] or [2] for details.

| t-link | initial cost | add | new cost |
|---|---|---|---|
| $\{p, S\}$ | $\lambda R_p(\text{"bkg"})$ | $K + \lambda R_p(\text{"obj"})$ | $K + c_p$ |
| $\{p, T\}$ | $\lambda R_p(\text{"obj"})$ | $\lambda R_p(\text{"bkg"})$ | $c_p$ |

These new costs are consistent with the edge weight table for pixels in $\mathcal{O}$ since the extra constant $c_p$ at both t-links of a pixel does not change the optimal cut[4]. Then, a maximum flow (minimum cut) on a new graph can be efficiently obtained starting from the previous flow without recomputing the whole solution from scratch.

Note that the same trick can be done to adjust the segmentation when a new "background" seed is added or when a seed is deleted. One has to figure the right amounts that have to be added to the costs of two t-links at the corresponding pixel. The new costs should be consistent with the edge weight table plus or minus the same constant.

## 4. Examples

We demonstrate our general-purpose segmentation method in several examples including photo/video editing and medical data processing. We show original data and segments generated by our technique for a given set of seeds. Our actual interface allows a user to enter seeds via mouse operated brush of red (for object) or blue (for background) color. Due to limitations of this B&W publication we show seeds as strokes of white (object) or black (background) brush. In addition, these strokes are marked by the letters "O" and "B". For the purpose of clarity, we employ different methods for the presentation of segmentation results in our examples below.

Our current implementation actually makes a double use of the seeds entered by a user. First of all, they provide the hard constraints for the segmentation process as discussed in Section 3. In addition, we use intensities of pixels (voxels) marked as seeds to get histograms for "object" and "background" intensity distributions: $\Pr(I|\mathcal{O})$ and $\Pr(I|\mathcal{B})$[5]. Then, we use these histograms to set the regional penalties $R_p(\cdot)$ as negative log-likelihoods:

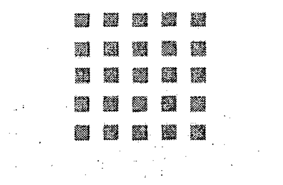
$$\begin{aligned} R_p(\text{"obj"}) &= -\ln \Pr(I_p|\mathcal{O}) \\ R_p(\text{"bkg"}) &= -\ln \Pr(I_p|\mathcal{B}). \end{aligned}$$

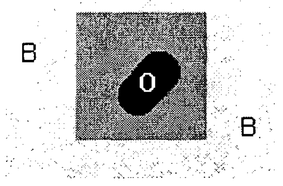
Such penalties are motivated by the underlying MAP-MRF formulation [9, 3].

To set the boundary penalties we use an *ad-hoc* function

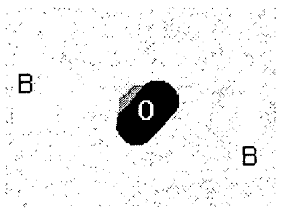
$$B_{\{p,q\}} \propto exp\left(-\frac{(I_p - I_q)^2}{2\sigma^2}\right) \cdot \frac{1}{dist(p,q)}.$$

[4]Note that the minimum cut severs exactly one of two t-links at pixel p.
[5]Alternatively, the histograms can be learned from historic data.

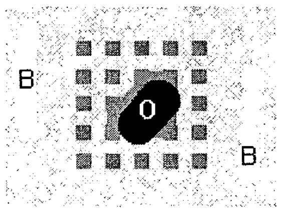

(a) Original image



(b) Result for $\lambda = 7$—$43$



(c) Result for $\lambda = 0$



(d) Result for $\lambda = 60$

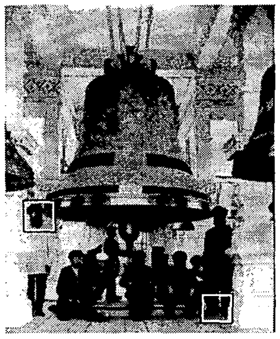
**Figure 2. Synthetic Gestalt example. The segmentation results in (b-d) are shown for various levels $\lambda$ of relative importance of "region" versus "boundary" in (1). Note that the result in (b) corresponds to a wide range of $\lambda$.**

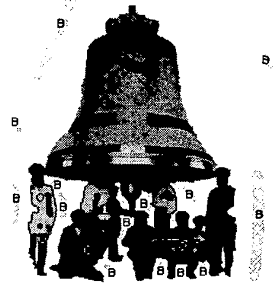
This function penalizes a lot for discontinuities between pixels of similar intensities when $|I_p - I_q| < \sigma$. However, if pixels are very different, $|I_p - I_q| > \sigma$, then the penalty is small. Intuitively, this function corresponds to the distribution of noise among neighboring pixels of an image. Thus, $\sigma$ can be estimated as "camera noise".

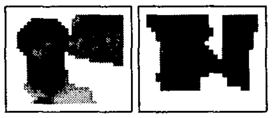
Note that we use an 8-neighborhood system in 2D examples and 26-neighborhood system in 3D examples. All running times are given for 333MHz Pentium III. Our implementation uses a new "max-flow" algorithm from [2].

## 4.1. Gestalt Example

Figure 2 illustrates some interesting properties of our cost function (1). Isolated dark blocks that stay relatively close to each other in (a) are segmented as one object in (b). This may correspond to the psychological perception of the original image (a). Only an optimal solution that combines both boundary and region properties can generate such result. Our segmentation does not leak through the numerous "holes" as a simple region growing would. As shown in (c), the segmentation based on boundary properties only "shrinks" the object. The opposite extreme when the region properties strongly dominate the boundary forces is shown in (d). In this case the method creates isolated segments for individual blocks. The bright space between the blocks has a better fit with the "background" histogram and the super-strong regional term forces the "object" segment out.


(a) Original B&W photo


(b) Segmentation results


(c) Details of segmentation with regional term

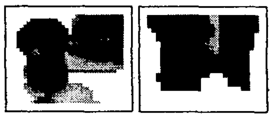

(d) Details of segmentation without regional term

**Figure 3. Segmentation of a photograph.**

## 4.2. Photo and Video Editing

In Figure 3 we segmented a bell with a group of people from a photograph. The user can start with a few "object" and "background" seeds loosely positioned inside and, correspondingly, outside the object(s) of interest. By reviewing the results of initial segmentation the user will see what areas are segmented incorrectly. Then (s)he can put additional seeds[6] into the troubled places and efficiently recompute the optimal segmentation as explained at the end of Section 3. This process of adding seeds gives more clues to the algorithm and may be continued until the user likes the results.

Naturally, the hope is that the method can quickly identify the right object. The user would not like to keep adding new seeds until the whole image is covered by these seeds. This is no better than a manual segmentation. The performance of an algorithm can be judged by the efforts required from the user. Thus, the results of our segmentation are shown with seeds that we entered to get this segmentation.

Our segmentation algorithm runs in less than a second for most 2D images (up to 512 $\times$ 512) with a fixed set of seeds. When additional seeds are entered the solution is recomputed in the blink of an eye. Thus, the speed evaluation of our method in 2D is mainly concerned with the user efforts. The detailed segmentation in Figure 3(b) is obtained in approximately a minute. Note that in this exam-

[6]Note that adding correcting "object" seeds to pixels that are already segmented as "object" would not change the optimal segmentation. These additional hard constraint are already satisfied by the current solution. The same argument applies to correcting "background" seeds.

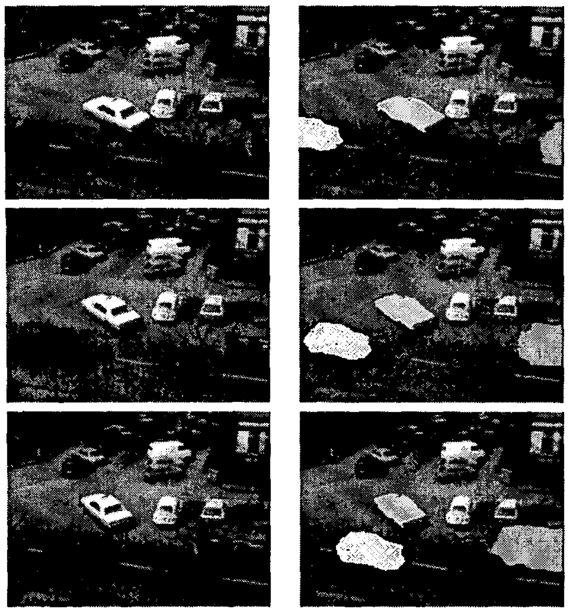


Figure 4. Segmentation of a video sequence.

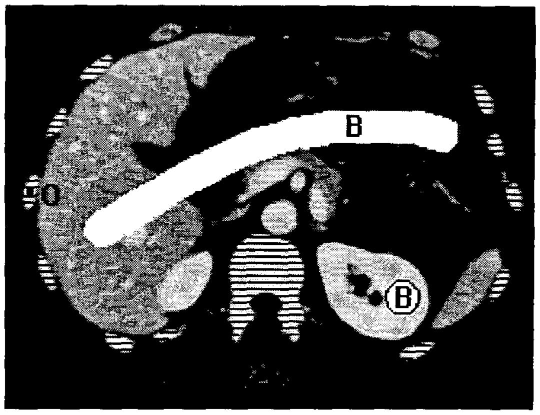


Figure 5. Bone removal in a CT image. Bone segments are marked by horizontal lines.

ple the algorithm created some isolated "background" segments. In fact, the algorithm automatically decided which "background" seeds were grouped together and which were placed into isolated segments. The same is equally true for the "object" seeds. The segments can have any topology.

In many cases the regional term of energy (1) helps to get the right results faster. In Figures 3(c,d) we show some details of segmentation with and without the regional term ($\lambda = 0$) given the same sets of seeds. In (d) the user would spend more time by placing additional "background" seeds to correct imperfections.

The globally minimum two-terminal cut can be computed on any graph. Thus, our technique is valid for segmentation of N-D data. In Figure 4 we segmented moving cars in a video sequence. The sequence of 21 video frames ($255 \times 189$) was treated as a single 3D volume. The necessary seeds were entered in a simple 3D interface where we could browse through individual 2D slices (frames) of the volume. Initial seeds can be entered in just a few representative frames. Note that the boundary, region, and seed information is automatically propagated between the slices since we compute a globally optimum solution directly on the 3D data set. Thus, the whole sequence can be segmented based on seeds placed in just a few frames. For example, entering correcting seeds in one frame can fix imperfections in many adjacent frames. The results in Figure 4 are obtained by placing seeds in 3 out of 21 frames. Each car was segmented in an independent experiment. We do not show seeds to avoid confusion.

The computation of the minimum cut is slower in 3D cases. The initial segmentation might take from 2-3 seconds on smaller volumes ($200 \times 200 \times 10$) to a few minutes on bigger ones ($512 \times 512 \times 50$). Thus, efficient recomputing of an optimal solution when new seeds are added is crucial. Most of the time the new seeds are incorporated in a few seconds even for bigger volumes. Therefore, we can still consider our method as "interactive". The results in Figure 4 can be obtained in approximately 30 seconds including user interactions.

## 4.3. Medical Images and Volumes

Figure 5 shows a segmentation that we obtained on a abdominal CT image. Our goal was to remove bones from the image for better volume rendering. The right segmentation was obtained after one "click" on one of the bones and a couple of loose "strokes" in the background. The other bones were correctly segmented out based on regional properties. The boundary term helped to avoid incorrect "bone" segments in the bright parts of liver and other tissues. The results are obtained in a few seconds. Without the regional term in the energy function (1) when $\lambda = 0$ the user would have to add "object" seeds in all isolated bones.

In Figure 6 we demonstrate our segmentation method on 3D medical data. We segmented out cortex, medulla, and collecting system of a kidney from a background. This was done in three steps. First, the kidney was separated from the background. Then we separated the cortex from the rest of the kidney. In the last step the medulla was separated from

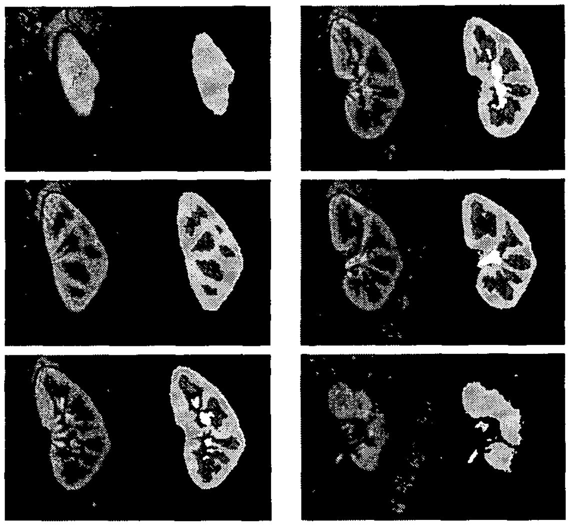


**Figure 6. Kidney in a 3D MRI angio data.**

the collecting system. The results in Figure 6 are shown without the seeds since the process involved three different segmentations. The segmentation can be computed in 3-4 minutes including the efforts of a user.

## 5. Extensions

The process of putting seeds can be automated for certain applications. The seeds should be positioned with a low probability of "false alarm" while the probability of "right detect" is not required to be high. Even very simple recognition techniques based on filters might be good enough if the corresponding threshold is set high. Such filters would have difficulties near the boundaries of objects but they can confidently place many seeds anywhere else. These seeds give hard constraints. Based on additional soft constraints (1) the minimum cut can complete the segmentation where the recognition method failed.

## References

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

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. In *3rd. Intnl. Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*. Springer-Verlag, September 2001, to appear.

[3] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–655, 1998.

[4] L. D. Cohen. On active contour models and ballons. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 53(2):211–218, 1991.

[5] I. J. Cox, S. B. Rao, and Y. Zhong. "Ratio regions": a technique for image segmentation. In *International Conference on Pattern Recognition*, volume II, pages 557–564, 1996.

[6] A. X. Falcão, J. K. Udupa, S. Samarasekera, and S. Sharma. User-steered image segmentation paradigms: Live wire and live lane. *Graphical Models and Image Processing*, 60:233–260, 1998.

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

[8] A. Goldberg and R. Tarjan. A new approach to the maximum flow problem. *Journal of the Association for Computing Machinery*, 35(4):921–940, October 1988.

[9] 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.

[10] L. D. Griffin, A. C. F. Colchester, S. A. Röll, and C. S. Studholme. Hierarchical segmentation satisfying constraints. In *British Machine Vision Conference*, pages 135–144, 1994.

[11] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*. Addison-Wesley Publishing Company, 1992.

[12] H. Ishikawa and D. Geiger. Segmentation by grouping junctions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 125–131, 1998.

[13] I. H. Jermyn and H. Ishikawa. Globally optimal regions and boundaries. In *International Conference on Computer Vision*, volume II, pages 904–910, 1999.

[14] M. Kass, A. Witkin, and D. Terzolpoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 2:321–331, 1988.

[15] E. N. Mortensen and W. A. Barrett. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60:349–384, 1998.

[16] S. Osher and J. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on the Hamilton-Jacobi formulation. *Journal of Computational Physics*, 79:12–49, 1988.

[17] L. J. Reese. Intelligent paint: Region-based interactive image segmentation. Master's thesis, Brigham Young University, 1999.

[18] J. Shi and J. Malik. Normalized cuts and image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 731–737, 1997.

[19] D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 345–352, 2000.

[20] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, November 1993.

[21] A. Yuille and P. Hallinan. Deformable templates. In A. Blake and A. Yuille, editors, *Active Vision*, pages 20–38. MIT Press, 1992.

[22] S. C. Zhu and A. Yuille. Region competition: Unifying snakes, region growing, and Bayes/MDL for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, September 1996.