**Noname manuscript No.**
(will be inserted by the editor)

# Branch-and-Mincut: Global Optimization for Image Segmentation with High-Level Priors

**Victor Lempitsky · Andrew Blake · Carsten Rother**

the date of receipt and acceptance should be inserted later

**Abstract** Efficient global optimization techniques such as graph cut exist for energies corresponding to binary image segmentation from low-level cues. However, introducing a high-level prior such as a shape prior or a color-distribution prior into the segmentation process typically results in an energy that is much harder to optimize. The main contribution of the paper is a new global optimization framework for a wide class of such energies. The framework is built upon two powerful techniques: graph cut and branch-and-bound. These techniques are unified through the derivation of lower bounds on the energies. Being computable via graph cut, these bounds are used to prune branches within a branch-and-bound search.

We demonstrate that the new framework can compute globally optimal segmentations for a variety of segmentation scenarios in a reasonable time on a modern CPU. These scenarios include unsupervised segmentation of an object undergoing 3D pose change, category-specific shape segmentation, and the segmentation under intensity/color priors defined by Chan-Vese and GrabCut functionals.

V. Lempitsky
Yandex, Moscow, Russia
E-mail: victorlempitsky@gmail.com

A. Blake
Microsoft Research, Cambridge, UK
E-mail: Andrew.Blake@microsoft.com

C. Rother
Microsoft Research, Cambridge, UK
E-mail: carrot@microsoft.com

## 1 Introduction

Binary image segmentation is often posed as a graph partition problem. This is because efficient graph algorithms such as st-mincut permit fast global optimization of the functionals measuring the quality of the segmentation. As a result, difficult image segmentation problems can be solved efficiently, robustly, and independently of initialization. Yet, while graphs can represent energies based on localized low-level cues, they are much less suitable for representing *non-local* cues and priors describing the foreground or the background segment as a whole.

Consider, for example, the situation when the shape of the foreground segment is known *a priori* to be similar to a particular template (segmentation with shape priors). Graph methods can incorporate such a prior for a single pre-defined and pre-located shape template [20, 30]. However, once the pose of the template is allowed to change, the relative position of each graph edge with respect to the template becomes unknown, and the *non-local* property of shape similarity becomes hard to express with local edge weights. Another example would be the segmentation with non-local color priors, when the color of the foreground and/or background is known a priori to be described by some parametric distribution (e.g. a mixture of the Gaussians as in the case of GrabCut [38]). If the parameters of these distributions are allowed to change, such a non-local prior depending on the segment as a whole becomes very hard to express with the local edge weights.

An easy way to circumvent the aforementioned difficulties is to alternate the graph partitioning with the re-estimation of non-local parameters (such as the template pose or the color distribution). A number of approaches [8,26,38,25] follow this path. Despite the use

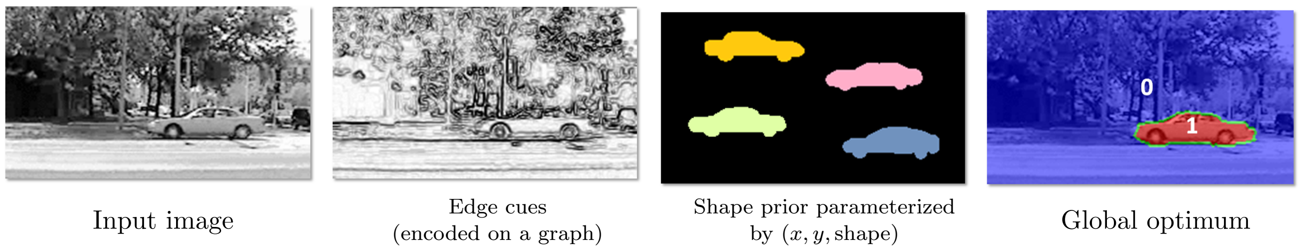


**Fig. 1** Image segmentation with a shape prior is one of motivating applications for Branch-and-Mincut. Here, the image-specific edge cues and cathegory-level shape prior can be combined within the same energy function. Edge cues are encoded by weighting the edges in the image grid graph, while shape prior is given by a combinatorily large set of plausible shapes at all possible locations in the image (four of them are shown in different colors in the image). By computing the global optimum of the resulting functional, Branch-and-Mincut finds the binary segmentation, which is close to one of the plausible shapes and is consistent with the edge cues.

of the global graph cut optimization inside the loop, local search over the prior parameters turns these approaches into local optimization techniques akin to variational segmentation [10,12,35,44]. As a result, these approaches may get stuck in local optima, which in many cases correspond to poor solutions.

The goal of this paper is to introduce a new framework for computing *globally* optimal segmentations under non-local priors. Such priors are expressed by replacing fixed-value edge weights with edge weights depending on *non-local parameters*. The global minimum of the resulting energy that depends on both the graph partition and the non-local parameters is then found using the branch-and-bound tree search. Within the branch-and-bound, lower bounds over tree branches are efficiently evaluated by computing minimal cuts on a graph (hence the name *Branch-and-Mincut*).

The main advantage of the proposed framework is that the globally optimal segmentation can be obtained for a broad family of functionals depending on non-local parameters. Although the worst case complexity of our method is large (essentially, the same as the exhaustive search over the space of non-local parameters), we demonstrate that our framework can obtain globally optimal image segmentation in a matter of seconds on a modern CPU. Test scenarios include globally optimal segmentation with shape priors where the template shape is allowed to deform and to appear in various poses as well as image segmentation by the optimization of the Chan-Vese [10] and the GrabCut [38] functionals. In all cases, bringing in high-level non-local knowledge allows to solve difficult segmentation problems, where local cues (considered by most current global optimization approaches) were highly ambiguous.

## 2 Related Work

### 2.1 Related work: Optimization.

Our framework is built upon two powerful optimization paradigms: graph cuts and branch-and-bound. Graph cut optimization employs the fact that a submodular quadratic function of boolean variables can be efficiently minimized via minimum st-cut computation in the associated graph [4,24,29]. This idea has been successfully applied to binary image segmentation [5] and quickly gained popularity. As discussed above, the approach [5] still has significant limitations, as the high-level knowledge such as shape or color priors are hard to express with fixed local edge weights. These limitations are overcome in our framework, which allows the edge weights to vary according to a high-level prior.

In the restricted case, when unary energy potentials are allowed to vary and depend on a single scalar non-local parameter monotonically, efficient algorithms known as *parametric maxflow* have been suggested [21], and their use in computer vision have been recently investigated in [28]. While the structure of the energy function considered in our framework is related to those optimized by parametric mincut/max-flow, our framework is more general (at a price of having higher worst-case complexity). Thus, it allows both unary and pairwise energy terms to depend non-monotonically on a single or multiple non-local parameters. Such generality gives our framework flexibility in incorporating various high-level priors while retaining the globality of the optimization.

On the other hand, branch-and-bound optimization is a classical approach to non-convex optimization. Over the recent years, it has found numerous applications in the computer vision field. Its usage for solving structure-and-motion problems [2] and in fast category-level object detection [31] has proved particularly fruitful. Of particular relevance here, are the works [22,19] that use tree search over shape hierarchies for the detection of pedestrians [22] and general object tracking [19].

### 2.2 Related work: Segmentation with Priors.

Segmentation with shape priors has attracted a lot of interest over the years. Historically, most approaches use either local continuous optimization [44,10,35,12] or iterated minimization alternating graph cut and search over non-local parameter space [38,8,26]. Unfortunately, both groups of methods are prone to getting stuck in poor local minima. Therefore, in recent years, a number of global optimization algorithms have also been suggested.

These global-optimization approaches solve the segmentation under shape priors via different graph algorithms, namely dynamic programming on a specially constructed graphs [18,41,17], min-ratio cycle search [40], and generalized graph spectral clustering [42]. Some other works [15,17] incorporate shape priors into globally-optimal segmentation by introducing multiple labels corresponding to different object parts, and introducing energy terms that ensure that pixels belonging to different object parts are arranged in a particular way. Each of these approaches is based on an elegant idea, and successfully formulates the segmentation under shape prior as an optimization task of polynomial (in graph size) complexity. Thus, the worst-case complexity of these methods is typically better than the complexity of our method. However, these methods have less flexibility, and the resulting shape priors that they impose are either "unimodal" [18,40,41] (i.e. the target segmentation has to be similar to a single shape prototype), or, on the opposite, they force the segmentation to belong to a quite general shape class [15,17] (e.g. be approximately rectangular-shaped [42]), which, depending on application, may be less desirable.

In contrast to that, our method provides a great flexibility in devising high-level priors. In the case of shape priors one simply needs to provide an arbitrary number of prototypical shapes, so that the branch-and-bound procedure can be used to search over this set. This flexibility is based on a generality of the proposed approach. Based on similar motivation, another group [13] presented (simultaneously and independently with our conference paper [34]) a framework that also utilizes branch-and-bound optimization for the segmentaion with shape-priors. While at the high level the two approaches are similar (both are based on the fusion of branch-and-bound and convex optimization), there are significant dissimilarities. Thus, the approach [13] operate with the relaxed notion of shape (while our segmentations stay integer) and their bounds are based on the Lipshitz-continuity (with sufficiently small Lipshitz constant) of the shape manifold inside the prior, which is not required in our case (as our prior may be given by a discrete set of shapes). In the latter aspect, our approach to formulating shape priors is more similar to [12], which uses local optimization.

The generality of the proposed optimization framework allowed us to attack image segmentation under priors imposed by Chan-Vese [10] and GrabCut [38] functionals (which are quite different from the shape priors). Recent related work here includes [43], where they proposed a highly-efficient branch-and-bound scheme devised specifically for the Chan-Vese functional, and [39], where they formulated a functional similar to GrabCut that is amenable for global graph cut optimization.

The preliminary version of this work has appeared in [34], and this paper extends it by providing more experimental detail, more illustrations of the theoretical part of the framework, and more detailed explanation of the construction of the shape priors. It also provides a new simpler version of our algorithm for a restricted class of shape priors.

## 3 Optimization Framework

In this section, we discuss our global energy optimization framework for obtaining image segmentations under non-local priors[1]. In the next sections, we detail how it can be used for the segmentation with non-local shape priors (Section 4) and non-local intensity/color priors (Section 5).

### 3.1 Energy Formulation

Firstly, we introduce notation and give the general form of the energy that can be optimized in our framework. Below, we consider the pixel-wise segmentation of the image. We denote the pixel set as $\mathcal{V}$ and use letters $p$ and $q$ to denote individual pixels. We also denote the set of edges connecting adjacent pixels as $\mathcal{E}$ and refer to individual edges as to the pairs of pixels (e.g. $p, q$). In our experiments, the set of edges consisted of all 8-connected pixel pairs in the raster.

The segmentation of the image is given by its $0\!-\!1$ labeling $\mathbf{x} \in 2^{\mathcal{V}}$, where individual pixel labels $x_p$ take the values 1 for the pixels classified as the foreground and 0 for the pixels classified as the background. Finally, we denote the non-local parameter as $\omega$ and allow it to vary over a discrete, possibly very large, set

[1] The C++ code for this framework is available at the webpage of the first author.

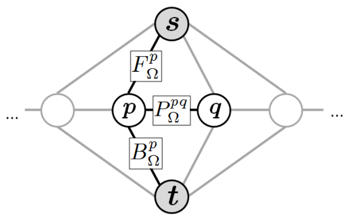


**Fig. 2** The fragment of the network graph realizing $L(\Omega)$ (edge weights are shown in boxes). The traditional graph cut construction [5] is shown, where "t-links" and "n-links" are used.

$\Omega$. The general form of the energy function that can be handled within our framework is then given by:

$$E(\mathbf{x},\omega) = C(\omega) + \sum_{p\in\mathcal{V}} F^p(\omega)\cdot x_p + \sum_{p\in\mathcal{V}} B^p(\omega)\cdot(1-x_p) + \sum_{p,q\in\mathcal{E}} P^{pq}(\omega)\cdot|x_p - x_q| \quad (1)$$

Here, $C(\omega)$ is a constant potential, which does not depend directly on the segmentation $\mathbf{x}$; $F^p(\omega)$ and $B^p(\omega)$ are the unary potentials defining the cost for assigning the pixel $p$ to the foreground and to the background respectively; $P^{pq}(\omega)$ is the pairwise potential defining the cost of assigning adjacent pixels $p$ and $q$ to different segments. In our experiments, the pairwise potentials were taken non-negative to ensure the tractability of $E(\mathbf{x},\omega)$ as the function of $\mathbf{x}$ for graph cut optimization [29].

All potentials in our framework depend on the non-local parameter $\omega \in \Omega$. In general, we assume that $\Omega$ is a discrete set, which may be large (e.g. millions of elements) and should have some structure (although, it need not be linearly or partially ordered). For the segmentation with shape priors, $\Omega$ will correspond to the product space of various poses and deformations of the template, while for the segmentation with color priors $\Omega$ will correspond to the set of parametric color distributions.

## 3.2 Lower Bound

Our approach optimizes the energy (1) exactly, finding its global minimum using branch-and-bound tree search [11], which utilizes the lower bound on (1) derived as follows:

$$\min_{x\in 2^{\mathcal{V}},\omega\in\Omega} E(\mathbf{x},\omega) =$$

$$\min_{x\in 2^{\mathcal{V}}} \min_{\omega\in\Omega} \Bigg[ C(\omega) + \sum_{p\in\mathcal{V}} F^p(\omega)\cdot x_p + \sum_{p\in\mathcal{V}} B^p(\omega)\cdot(1-x_p) + \sum_{p,q\in\mathcal{E}} P^{pq}(\omega)\cdot|x_p - x_q| \Bigg] \geq$$

$$\min_{\mathbf{x}\in 2^{\mathcal{V}}} \Bigg[ \min_{\omega\in\Omega} C(\omega) + \sum_{p\in\mathcal{V}} \min_{\omega\in\Omega} F^p(\omega)\cdot x_p + \sum_{p\in\mathcal{V}} \min_{\omega\in\Omega} B^p(\omega)\cdot(1-x_p) + \sum_{p,q\in\mathcal{E}} \min_{\omega\in\Omega} P^{pq}(\omega)\cdot|x_p - x_q| \Bigg] =$$

$$\min_{x\in 2^{\mathcal{V}}} \Bigg[ C_\Omega + \sum_{p\in\mathcal{V}} F^p_\Omega\cdot x_p + \sum_{p\in\mathcal{V}} B^p_\Omega\cdot(1-x_p) + \sum_{p,q\in\mathcal{E}} P^{pq}_\Omega\cdot|x_p - x_q| \Bigg] = L(\Omega)\,. \quad (2)$$

Here, $C_\Omega$, $F^p_\Omega$, $B^p_\Omega$, $P^{pq}_\Omega$ denote the minima of $C(\omega)$, $F^p(\omega)$, $B^p(\omega)$, $P^{pq}(\omega)$ over $\omega \in \Omega$ referred below as *aggregated potentials*. $L(\Omega)$ denotes the derived lower bound for $E(\mathbf{x},\omega)$ over $2^{\mathcal{V}} \otimes \Omega$. The inequality in (2) is essentially the Jensen inequality for the minimum operation.

The proposed lower bound possesses three properties crucial to the Branch-and-Mincut framework:
**Monotonicity.** For the nested domains of non-local parameters $\Omega_1 \subset \Omega_2$ the inequality $L(\Omega_1) \geq L(\Omega_2)$ holds (the proof is given in the Appendix).
**Computability.** The key property of the derived lower bound is the ease of its evaluation. Indeed, this bound equals the minimum of a submodular quadratic pseudo-boolean function. Such function can be realized on a network graph such that each configuration of the binary variables is in one-to-one correspondence with an *st*-cut of the graph having the weight equal to the value of the function (plus a constant $C_\Omega$) [4,24,29] (see Fig. 2). The minimal *st*-cut corresponding to the minimum of $L(\Omega)$ then can be computed in a low-polynomial of $|\mathcal{V}|$ time e.g. with the popular algorithm [7].
**Tightness.** For a singleton $\Omega$ the bound is *tight*: $L(\{\omega\}) = \min_{x\in 2^{\mathcal{V}}} E(\mathbf{x},\omega)$. In such case, the minimal *st*-cut also yields the segmentation $\mathbf{x}$ optimal for this $\omega$ (thus, we set $x_p = 0$ *iff* the respective vertex belongs to the *s*-component of the minimal cut found as a result of maxflow).

Note, that the fact that the lower bound (2) may be evaluated via *st*-mincut gives rise to a whole family

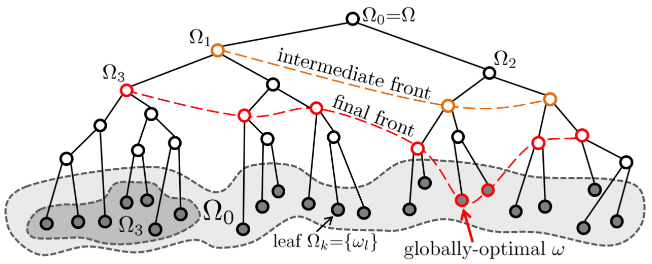


**Fig. 3** *Best-first branch-and-bound* optimization on the tree of nested regions finds the globally-optimal $\omega$ by the top-down propagation of the *active front* (see text for details). At the moment when the lowest lower bound of the front is observed at leaf node, the process terminates with the global minimum found without traversing the whole tree.

of looser, but cheaper, lower bounds. Indeed, the minimal cut on a network graph is often found by pushing *flows* until the flow becomes maximal (and equal to the weight of the mincut) [7]. Thus, the sequence of intermediate flows provides a sequence of the increasing lower bounds on (1) converging to the bound (2) (**flow bounds**). If some upper bound on the minimum value is imposed, the process may be terminated earlier without computing the full maxflow/mincut. This happens when the new flow bound exceeds the given upper bound. In this case it may be concluded that the value of the global minimum is greater than the imposed upper bound and a branch can be pruned without completing the maxflow computation.

## 3.3 Branch-and-Bound Optimization

Finding the global minimum of (1) is, in general, a very difficult problem. Indeed, since the potentials can depend arbitrarily on the non-local parameter spanning arbitrary discrete set $\Omega$, in the worst-case any optimization has to search exhaustively over $\Omega$. In practice, however, any segmentation problem would have some specifically-structured space $\Omega$. This structure can be efficiently exploited by the branch-and-bound search detailed below.

We assume that the discrete domain $\Omega$ can be hierarchically clustered and the binary tree of its subregions $T_\Omega = \{\Omega = \Omega_0, \Omega_1, \dots \Omega_N\}$ can be constructed (binarity of the tree is not essential). Each non-leaf node corresponding to the subregion $\Omega_k$ then has two children corresponding to the subregions $\Omega_{ch1(k)}$ and $\Omega_{ch2(k)}$ such that $\Omega_{ch1(k)} \subset \Omega_k$, $\Omega_{ch2(k)} \subset \Omega_k$. Here, $ch1(\cdot)$ and $ch2(\cdot)$ map the index of the node to the indices of its children. Also, leaf nodes of the tree are in one-to-one correspondence with singleton subsets $\Omega_l = \{\omega_t\}$.

Given such tree, the global minimum of (1) can be efficiently found using the *best-first* branch-and-bound search [11] (Fig. 4). This algorithm propagates a *front* of nodes in the top-down direction (Fig. 3). During the search, the front contains a set of tree nodes, such that each top-down path from the root to a leaf contains exactly one active vertex. In the beginning, the front contains the tree root $\Omega_0$. At each step the active node with the smallest lower bound (2) is removed from the active front, while two of its children are added to the active front (by monotonicity property they have higher or equal lower bounds). Thus, an active front moves towards the leaves making local steps that increase the lowest lower bound of all active nodes. Note, that at each moment, this lowest lower bound of the front constitutes a lower bound on the global optimum of (1) over the whole domain.

At some moment of time, the active node with the smallest lower bound turns out to be a leaf $\{\omega'\}$. Let $\mathbf{x}'$ be the optimal segmentation for $\omega'$ (found via minimum $st$-cut). Then, $E(x', \omega') = L(\omega')$ (tightness property) is by assumption the lowest bound of the front and hence a lower bound on the global optimum over the whole domain. Consequently, $(x', \omega')$ is a global minimum of (1) and the search terminates without traversing the whole tree. In our experiments, the number of the traversed nodes was typically very small (two-three orders of magnitude smaller then the size of the full tree). Therefore, the algorithm was able to find global minima much faster than exhaustive search over $\Omega$.

In order to further accelerate the search, we exploit the coherency between the mincut problems solved at different nodes. Indeed, the maximum flow as well as auxiliary structures such as shortest path trees computed for one graph may be "reused" in order to accelerate the computation of the minimal $st$-cut on another similar graph [5, 27]. For some applications, this trick may give an order of magnitude speed-up for the evaluation of lower bounds.

In addition to the best-first branch-and-bound search we also tried the *depth-first* branch-and-bound [11]. When problem-specific heuristics are available that give good initial solutions, this variant may lead to moderate (up to a factor of 2) time savings. Interestingly, the depth-first variant of the search, which maintains upper bounds on the global optimum, may benefit significantly from the use of flow bounds discussed above. Nevertheless, we stick with the best-first branch-and-bound for the final experiments due to its relative simplicity (no need for initialization heuristics).

In the rest of the paper we detail how the general framework developed above may be used within different segmentation scenarios.

**Algorithm 1** Branch-And-Mincut

```
Require: domain $\Omega_0$,
    problem-specific function $GetAggregPotentials$,
    problem-specific function $GetChildrenSubdomains$
Ensure: $(\mathbf{x}, \omega)$ = global minimum of the functional
    defined by $GetAggregPotentials$
 1: $Front = \emptyset$ // initializing the priority queue
 2: $[C_0, \{F_0^p\}, \{B_0^p\}, \{P_0^{pq}\}] = GetAggregPotentials(\Omega_0)$
 3: $LB_0 = GetMaxFlowValue(\{F_0^p\}, \{B_0^p\}, \{P_0^{pq}\}) + C_0$
 4: $Front.InsertWithPriority(\Omega_0, -LB_0)$
 5: loop // advancing front
 6:   $\Omega = Front.PullHighestPriorityElement()$
 7:   if $IsSingleton(\Omega)$ then // global minimum found
 8:     $\omega = \Omega$
 9:     $[C, \{F^p\}, \{B^p\}, \{P^{pq}\}] = GetAggregPotentials(\omega)$
10:     $\mathbf{x} = FindMinimumViaMincut(\{F^p\}, \{B^p\}, \{P^{pq}\})$
11:     return $(\mathbf{x}, \omega)$
12:   end if
13:   $[\Omega_1, \Omega_2] = GetChildrenSubdomains(\Omega)$
14:   $[C_1, \{F_1^p\}, \{B_1^p\}, \{P_1^{pq}\}] = GetAggregPotentials(\Omega_1)$
15:   $LB_1 = GetMaxFlowValue(\{F_1^p\}, \{B_1^p\}, \{P_1^{pq}\}) + C_1$
16:   $Front.InsertWithPriority(\Omega_1, -LB_1)$
17:   $[C_2, \{F_2^p\}, \{B_2^p\}, \{P_2^{pq}\}] = GetAggregPotentials(\Omega_2)$
18:   $LB_2 = GetMaxFlowValue(\{F_2^p\}, \{B_2^p\}, \{P_2^{pq}\}) + C_2$
19:   $Front.InsertWithPriority(\Omega_2, -LB_2)$
20: end loop
```

**Fig. 4** The pseudocode for the Branch-and-Mincut algorithm (the version based on best-first branch-and-bound search). The way to compute the aggregated potentials are determined by a problem-specific function *GetAggregPotentials*, which for a singleton $\Omega = \{\omega\}$ should return the set of potentials defining the functional itself (for that $\omega$). The hierarchical clustering of the domain is determined by another problem-specific function *GetChildrenSubdomains*.

## 4 Segmentation with Shape Priors

### 4.1 The Framework for Shape Priors

We start with the segmentation with shape priors. The success of such segmentation crucially depends on the way shape prior is defined. Earlier works have often defined this prior as a Gaussian distribution of some geometrical shape statistics (e.g. control point positions or level set functions) [44,35]. In reality, however, pose variance and deformations specific to the object of interest typically lead to highly non-Gaussian, multi-modal prior distributions. For better modeling of prior distributions, [12] suggested the use of non-parametric kernel densities. Our approach to shape modeling (Fig. 5) is similar in spirit, as it also uses exemplar-based prior. Arguably, it is more direct, since it involves the distances between the binary segmentations themselves, rather than their level set functions. Our approach to shape modeling is also closely related to [22] that used

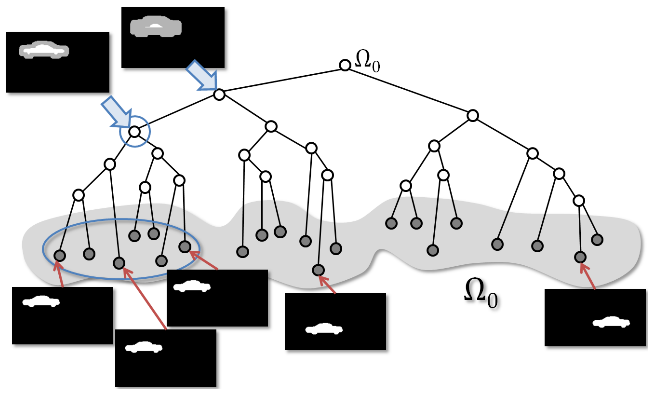


**Fig. 5** Cartoonish visualization of the hierarchy tree in the case of the segmentation with shape priors. Each node in the tree corresponds to a set of shapes. The set of leaves correspond to a set of binary masks defining the shape prior. The bitmaps show the potentials $B_\Omega$ and $F_\Omega$ corresponding to each node (black $= B_\Omega(\cdot) = 0, F_\Omega(\cdot) = 1$, white $= B_\Omega(\cdot) = 1, F_\Omega(\cdot) = 0$, gray $= B_\Omega(\cdot) = 0, F_\Omega(\cdot) = 0$). Note that the amount of gray pixels (corresponding to *looseness* of the mask defined below increases from leaves towards the root. A hierarchical clustering should then cluster together leaves with similar masks.

shape hierarchies to detect or track objects in image edge maps.

We assume that the prior is defined by the set of exemplar binary segmentations $\{\mathbf{y}^\omega | \omega \in \Omega\}$, where $\Omega$ is a discrete set indexing the exemplar segmentations. Then the following term introduces a joint prior over the segmentation and the non-local parameter into the segmentation process:

$$E_{\text{prior}}(\mathbf{x}, \omega) = \rho(\mathbf{x}, \mathbf{y}^\omega) = \sum_{p \in \mathcal{V}} (1 - y_p^\omega) \cdot x_p + \sum_{p \in \mathcal{V}} y_p^\omega \cdot (1 - x_p) \; , \tag{3}$$

where $\rho$ denotes the Hamming distance between segmentations. This term clearly has the form (1) and therefore its combinations with other terms of this form can be optimized within our framework. Being optimized over the domain $2^{\mathcal{V}} \otimes \Omega$, this term would encourage the segmentation $\mathbf{x}$ to be close in the Hamming distance to one of the exemplar shapes. Note, that the Hamming distance in the continuous limit may be interpreted as the $L1$-distance between shapes. It is relatively straightforward to modify the term (3) to replace the Hamming distance with discrete approximations of other distances ($L2$, truncated $L1$ or $L2$, data-driven Mahalonobis distance, etc.).

The full segmentation energy then may be defined by adding a standard contrast-sensitive edge term [5]:

$$E_{\text{shape}}(\mathbf{x}, \omega) = E_{\text{prior}}(\mathbf{x}, \omega) + \sum_{p,q \in \mathcal{E}} \lambda \frac{e^{-\frac{||K_p - K_q||}{\sigma}}}{|p - q|} \cdot |x_p - x_q| \; , \tag{4}$$

where $||K_p - K_q||$ denote the *SAD* (L1) distance between RGB colors of the pixels $p$ and $q$ in the image ($\lambda$ and $\sigma$ were fixed throughout the experiments described in this section), $|p-q|$ denotes the distance between the centers of the pixels $p$ and $q$ (being either 1 or $\sqrt{2}$ for the 8-connected grid). The functional (4) thus incorporates the shape prior with edge-contrast cues.

Note the three properties of our approach to segmentation with shape priors. Firstly, since any shapes can be included in $\Omega_{\text{shape}}$, general 3D pose transformations and deformations can be handled. Secondly, the segmentations can have general varying topology not restricted to segments with single-connected boundaries. Thirdly, our framework is general enough to introduce other terms in the segmentation process (e.g. regional terms used in a standard graph cut segmentation [5]). These properties of our approach are demonstrated within the experiments below.

## 4.2 Constructing the Shape Prior

In this subsection we discuss the practical implementation of the framework introduced above. The main challenge here is that the set $\Omega_{\text{shape}}$ in the framework introduced above could be huge, e.g. tens of millions exemplars. Therefore, representation and hierarchical clustering of the exemplar segmentations $y^\omega, \omega \in \Omega$ can be challenging. Fortunately, this is accomplishable in many cases when the translation invariance is exploited. Below, we consider two cases: a more general one, when the set $\Omega_{\text{shape}}$ can be defined as a set of segmentations produced by translating multiple template shapes, and a more specific case, when $\Omega_{\text{shape}}$ is produced by translating a single template.

**Multiple templates + translation.** We start with a more general case when the shape prior is given by a set of multiple templates, whereas each template can be located anywhere within the image.

In this case, the aggregated potentials for each node of the tree should be precomputed and stored in memory; translation invariance of the aggregated potentials should be exploited to make this feasible. We assume that the set $\Omega_{\text{shape}}$ is factorized into the Cartesian product of two sets $\Omega_{\text{shape}} = \Delta \otimes \Theta$. The factor set $\Delta$ indexes the set of all exemplar segmentations $y^\delta$ centered at the origin (this set would typically correspond to the variations in scale, orientation as well as non-rigid deformations). The factor set $\Theta$ then corresponds to the shift transformations and ensures the translation invariance of the prior. Any exemplar segmentation $y^\omega, \omega = \delta \otimes \theta$ is then defined as some exemplar segmentation $y^\delta$ centered at the origin and then translated by the shift $\theta$.

Being much smaller than $\Omega_{\text{shape}}$, both factor sets can be clustered in hierarchy trees. For the factor set $\Delta$ we used agglomerative bottom-up clustering [36] (a complete linkeage algorithm as available in Matlab that uses the Hamming distance between the exemplar segmentations). In accordance with the notations above, we consider the resulting clustering tree $T_\Delta$:

$$T_\Delta = \{\Delta = \Delta_0, \Delta_1, \ldots \Delta_N\}\,. \tag{5}$$

Each non-leaf node corresponding to the subtree $\Delta_k$ then has two children corresponding to the subtree $\Delta_{ch1(k)}$ and $\Delta_{ch2(k)}$ such that $\Delta_{ch1(k)} \subset \Delta_k$, $\Delta_{ch2(k)} \subset \Delta_k$. Here, $ch1(\cdot)$ and $ch2(\cdot)$ map the index of the node to the indices of its children. Also, leaf nodes of the tree are in one-to-one correspondence with singleton subsets $\Delta_l = \{\delta_t\}$.

Hierarchical clustering for the set of shifts is easier, as a simple top-down subdivision may be used. We assume that overall set of shifts is a rectangle ranging from $(0,0)$-shift to a $(2^{K_1}-1, 2^{K_2}-1)$. To build the clustering tree, we then recursively split along the "longer" dimension. This leads to a tree $T_\Theta$:

$$T_\Theta = \{\Theta = \Theta_0, \Theta_1, \ldots \Theta_N\}\,, \tag{6}$$

with the same operators $ch1(\cdot)$ and $ch2(\cdot)$ as above. Importantly, each nodeset $\Theta_t$ is essentially a 2D rectangle of shifts with the width and height uniquely determined by the level of $\Theta_t$ in the tree $T_\Theta$.

Our goal is then to merge the two resulting trees $T_\Delta$ and $T_\Theta$ into a combined tree $T_\Omega$. Each nodeset $\Omega_t$ in the combined tree is defined by a pair $\Delta_{p(t)} \otimes \Theta_{q(t)}$, i.e. by all shapes contained in the subtree $\Delta_{p(t)}$ translated by all shifts in the subtree $\Theta_{q(t)}$ (here, $p(t)$ and $q(t)$ denote the mappings from the nodeset indices in $T_\Omega$ to the nodeset indices in $T_\Delta$ and $T_\Theta$). We can then define the *looseness* of a nodeset $\Omega_t$ as the number of pixels that change their mask value under different shapes in $\Omega_t$:

$$\begin{aligned}\Lambda(\Omega_t) = |\,\{p\,|\,\exists\,\delta_1,\theta_1 :\\ y_p^{\delta_1\otimes\theta_1} = 0 \wedge \exists\,\delta_2,\theta_2 : y_p^{\delta_2\otimes\theta_2} = 1\}\,|\end{aligned} \tag{7}$$

In other words, if we consider the two *aggregated masks* $F_{\Omega_t} = \{F^p_{\Omega_t}\}$ and $B_{\Omega_t} = \{B^p_{\Omega_t}\}$ (derived from (3)), then the looseness is defined as the number of pixels $p$ that are zero in both masks ($F^p_{\Omega_t} = 0$ and $B^p_{\Omega_t} = 0$).

Given this definition of looseness the combined tree is built in a recursive top-down fashion. We start by creating a root nodeset $\Omega_0 = \Delta_0 \otimes \Theta_0$. Given a nodeset $\Omega_t = \Delta_{p(t)} \otimes \Theta_{q(t)}$, we then consider two possible splits:

- Split along the shape dimension into $\Delta_{ch1(p(t))} \otimes \Theta_{q(t)}$ and $\Delta_{ch2(p(t))} \otimes \Theta_{q(t)}$.

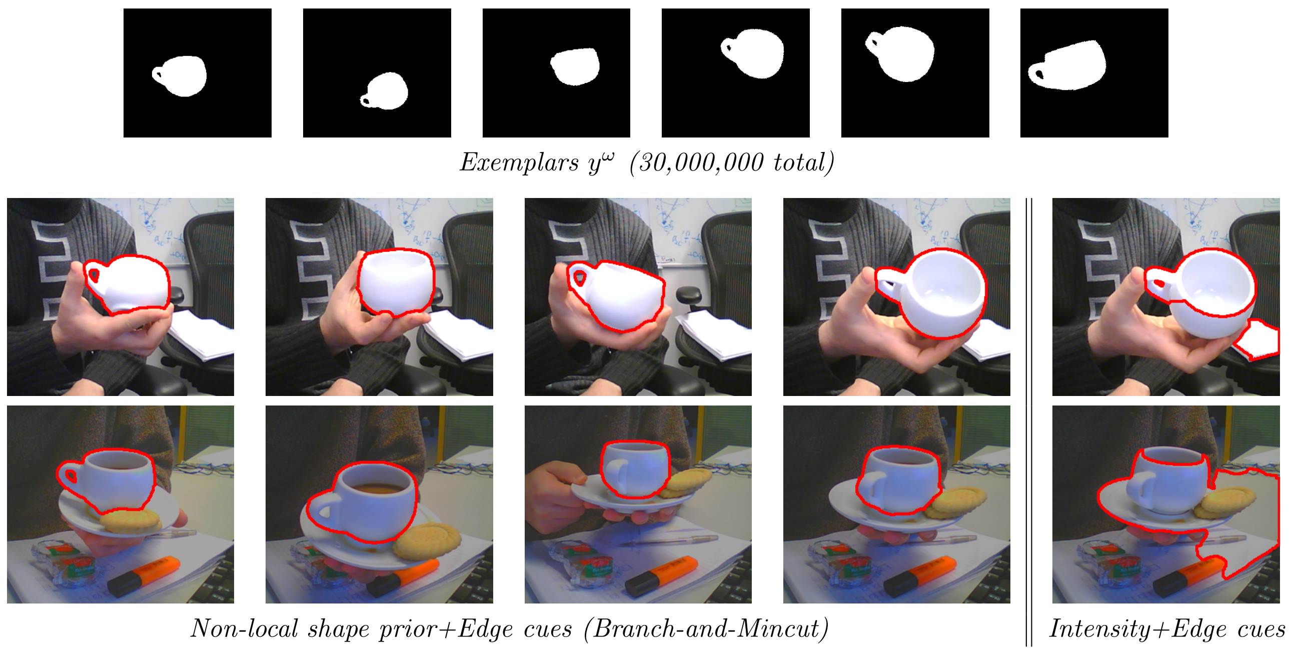

Exemplars $y^\omega$ (30,000,000 total)
Non-local shape prior+Edge cues (Branch-and-Mincut)
Intensity+Edge cues

**Fig. 6** Using the shape prior constructed from the set of exemplars (left column) our approach can accomplish segmentation of an object undergoing general 3D pose changes within two differently illuminated sequences (two middle columns). Note the varying topology of the segmentations. For comparison, we give the results of a standard graph cut segmentation (right column): even with parameters tuned specifically to the test images, separation is very inaccurate.

– Split along the shift dimension into $\Delta_{p(t)} \otimes \Theta_{ch1(q(t))}$ and $\Delta_{p(t)} \otimes \Theta_{ch2(q(t))}$.

Then the split that minimizes the sum of loosenesses is preferred, the resulting two nodesets are added into the combined tree and the recursion proceeds. At some point, splitting along one of the dimensions becomes impossible as the respective leaf is reached in one of the two factor trees. In this case, we naturally split along the other dimension. The recursion stops when the leaf level is reached within both the shape and the shift trees.

The algorithm described above is still inefficient as it needs to visit each of the many million nodes in the combined tree during the preprocessing stage. One may notice however, that the translational invariance of the shift tree $T_\Theta$ may be exploited. Indeed, consider the two shift nodesets $\Theta_m$ and $\Theta_n$ on the same level of the $T_\Theta$ tree (e.g. the siblings of the same parent). Being on the same level, they represent the same set of translations just shifted with respect to each other. Therefore, given any shape nodeset $\Delta_t$ from the shape tree $T_\Delta$, the product sets $\Delta_t \otimes \Theta_m$ and $\Delta_t \otimes \Theta_n$ will have the same looseness measure and their aggregated masks will differ only by a global translation.

Consequently, when one splits along the shift dimension, there is no need to track both subtrees $\Delta_{p(t)} \otimes \Theta_{ch1(q(t))}$ and $\Delta_{p(t)} \otimes \Theta_{ch2(q(t))}$ in the recursion, as they will be identical in terms of the aggregated masks (up to a global translation) as well as in terms of looseness and the order of further splits along different dimensions. Thus, the practical implementation of the tree construction algorithm does not follow the second branch when the split is performed along the shift dimension. As such, only the shift sets with the top-left corners at zero will be considered during the pass.

Thus, the algorithm essentially makes a top-down pass over the shape tree $T_\Delta$ and augments each nodeset $\Delta_t$ there with the information about which range of layers of the shift tree $T_\Theta$ are paired with this shape node. In other words, the rule recorded at the shape nodeset $\Delta_t$ is "if the level $lev(\Delta_t)$ of the shift tree is not reached, split along the shift dimension, otherwise split along the shape dimension (and recurse down the shape tree)", where $lev(\Delta_t)$ is a "critical" level for this particular shape nodeset. During the top-down construction pass, the algorithm also precomputes and stores the respective aggregated masks $F_{\Omega_s}$ and $B_{\Omega_s}$. Once again, only the masks corresponding to the products with the shift sets with the top-left corner at zero are precomputed and stored.

At runtime the constructed trees and shape masks are loaded to memory. Then, the full product tree $T_\Omega$ is traversed. Consider some point of the branch-and-bound process, when each element of the front $\Omega_t$ corresponds to a pair $\Delta_{p(t)} \otimes \Theta_{q(t)}$. To evaluate the bound (2) at the node, the aggregated potentials are retrieved as the precomputed masks based on the shape component $\Delta_{p(t)}$ and the size (tree level) of the shift component $\Theta_{q(t)}$. The retrieved mask is then shifted according to the top-left point of $\Theta_{q(t)}$. Likewise, when there is a need to propagate the front at node $\Delta_{p(t)} \otimes \Theta_{q(t)}$, then one of the two splits is performed according to the information precomputed and stored at $\Delta_{p(t)}$ (if the level $lev(\Delta_{p(t)})$ in the shift tree is reached then split along the shape dimension, otherwise split along the shift dimension).

**Single template + translation.** In the case when the set $\Delta$ contains a single shape $\delta_0$ with the corresponding shape mask $y^0$, it is possible to avoid the precomputation and storage of multiple templates. Instead, only an integral image of $y^0$ has to be precomputed and stored in memory. Consider a sub-tree $t$ defined by a set of shifts $\Theta_t$ (note that the mapping $q(\cdot)$ in the case of single object is identity). Consider then a pixel $p$ of the image and the task of computing the value $F^p_{\Omega_t}$. Denote with $R(p, \Theta_t)$ a rectangle defined by shifting pixel $p$ by all shifts $d$, where $-d \in \Theta_t$. In other words, $R(p, \Theta_t)$ is a "backprojection" of $p$ under shifts in $\Theta_t$. Then $F^p_{\Omega_t}$ can be computed as:

$$F^p_{\Omega_t} = \begin{cases} 1, \text{if } R(p,\Theta_t) \cap y^0 = \emptyset \\ 0, \text{otherwise} \end{cases} \tag{8}$$

Likewise, $B^p_{\Omega_t}$ can be computed as:

$$B^p_{\Omega_t} = \begin{cases} 1, \text{if } R(p,\Theta_t) \cap y^0 = R(p,\Theta_t) \\ 0, \text{otherwise} \end{cases} \tag{9}$$

Expressions (8) and (9) can be evaluated in a constant number of operations independent of the size of $\Theta_t$ using the integral image computed on the shape mask $y^0$. Thus, it is enough to check whether the sum over $R(p, \Theta_t)$ is equal 0 (in the case of (8)) or whether this sum is equal to the size of $R(p, \Theta_t)$ (in the case of (9)). Note that similar integral-image-based approach can be employed when the set $\Omega$ can be defined as a single shape $\delta_0$ under a range of translations as well as scalings (it is possible to have different scaling factors along horizontal and vertical image dimensions). This is because in this more general case, the "back-projection" of a pixel into $y^0$ is still a rectangle.

## 4.3 Experiments with Shape Priors

The experiments below demonstrate the capability of branch-and-mincut to compute globally optimal segmentations under shape priors. In all cases, we used the more general implementation of the framework (multiple templates + translation).

**Single object+3D pose changes.** In our first experiment, we constructed a shape prior for a single object (a coffee cup) undergoing 3D pose changes. We obtained a set of outlines using "blue-screening". We then normalized these outlines (by centering at the origin, resizing to a unit scale and orienting the principle axes with the coordinate axes). After that we clustered the normalized outlines using k-means. A representative of each cluster was then taken into the exemplar set. After that we added scale variations, in-plane rotations, and translations. As a result, we got a set $\{y^\omega | \omega \in \Omega_{\text{shape}}\}$ containing about 30,000,000 exemplar shapes (while the set $\Delta$ contained about 1900 shapes).

The results of the global optimization of the functional (4) for the frames from the two sequences containing clutter and camouflage are shown in Fig. 6. On average, we observed that segmenting 312x272 image took about 30 seconds of an Intel-2.40 GHz CPU and less than 1 Gb of RAM. The proportion of the nodes of the tree traversed by the active front was on average about 1 : 5000 (computed on the frames of two videos with 500 frames). Thus, branch-and-bound tree search used in our framework improved very considerably over exhaustive search, which would have to traverse all leaves (1 : 2 of the tree).

As a baseline algorithm, we considered the segmentation with a "standard" graph cut functional, replacing non-local shape prior term with a local intensity-based term $\sum_{p \in \mathcal{V}}(I - I_p) \cdot x_p$, adjusting the constant $I$ for each frame so that it gives the best results. However, since the intensity distributions of the cup and the backgrounds overlapped significantly, the segmentations were grossly erroneous (Fig. 6 – right column).

**Object class+translation invariance.** In the second experiment, we performed the segmentation with shape priors on UIUC car dataset [1] (the version without scale variations), containing 170 images with cars in uncontrolled environment (city streets). The prior set $\Delta$ was built by manual segmentation of 60 training images coming with the dataset. The set of shifts $\Theta$ was defined by the varying size of test images. While the test image sizes varied from 110x75 to 360x176, the size of $\Omega_{\text{shape}}$ varied from 18,666 to 2,132,865. We computed the globally optimal segmentations under the constructed prior using the energy (4).

Using the bounding boxes of the cars provided with the dataset, we found that in 6.5% of the images the global minima corresponded to clutter rather than cars. To provide a baseline for localization accuracy based on edge cues and a set of shape templates, we considered Chamfer matching (as e.g. in [22]). For the comparison we used the same set of templates, which were matched against truncated Canny-based chamfer distance (with

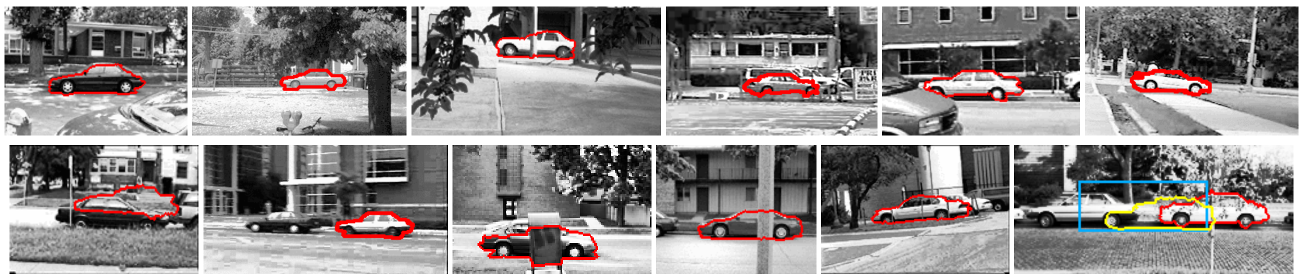


**Fig. 7** Results of the global optimization of (10) on some of the 170 UIUC car images including 1 of the 2 cases where localization failed (bottom left). In the case of the bottom right image, the global minimum of (4) (yellow) and the result of our feature-based car detector (blue) gave erroneous localization, while the global minimum of their combination (10) (red) represented an accurate segmentation.

optimally tuned truncation and Canny sensitivity parameters). In this way, the optimal localization failed (i.e. corresponded to clutter rather than a car) in 12.4% of the images.

Clearly, segmenting images using (4) takes into account the shape prior and edge-contrast cues, but ignores the appearance typical for the object category under consideration. At the same time, there exists a large number of algorithms working with image appearance cues and performing object detection based on these cues (see e.g. [32] and references therein). Typically, such algorithms produce the likelihood of the object presence either as a a function of a bounding box or even in the form of per-pixel "soft segmentation" masks. Both types of the outputs can be added into the functional (1) either via constant potential $C(\Omega)$ or via unary potentials. In this way, such appearance-based detectors can be integrated with shape prior and edge-contrast cues.

As an example of such integration, we devised a simple detector similar in spirit to [32]. The detector looked for the appearance features typical for cars (wheels) using normalized cross-correlation. Each pixel in the image then "voted" for the location of the car center depending on the strength of the response to the detector and the relative position of the wheels with respect to the car center observed on the training dataset. We then added an additional term $C_{\text{vote}}(\omega)$ in our energy (1) that for each $\omega$ equaled minus the accumulated strength of the votes for the center of $y^\omega$:

$$E_{\text{shape\&detect}}(\mathbf{x},\omega) = C_{\text{vote}}(\omega) + E_{\text{prior}}(\mathbf{x},\omega) + \sum_{p,q\in\mathcal{E}} \lambda \frac{e^{-\frac{||K_p-K_q||}{\sigma}}}{|p-q|}\cdot|x_p - x_q| \ , \quad (10)$$

Adding the appearance-based term improved the robustness of the segmentation, as the global optima of (10) corresponded to clutter only in 1.2% of the images. The global minima found for some of the images are shown in Fig. 7. Note, that for our simple detector on its own the most probable bounding box corresponded to clutter on as much as 14.7% of the images.

In terms of the performance, on average (over the 170 test images in the UIUC car dataset), for the functional (10) the segmentation took 1.8 seconds and the proportion of the tree traversed by the active front was 1 : 441. For the functional (4), the segmentation took 6.6 seconds and the proportion of the tree traversed by the active front was 1 : 131. This difference in performance is natural to branch-and-bound methods: the more difficult and ambiguous is the optimization problem, the larger is the portion of the tree that has to be investigated.

## 5 Segmentation with Color/Intensity Priors

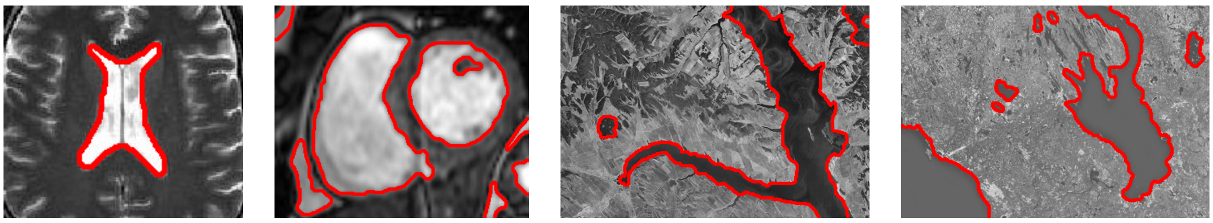
Our framework can also be used to impose non-local priors on the intensity or color distributions of the foreground and background segments, as the examples below demonstrate.

### 5.1 Segmenting Grayscale Images: Chan-Vese Functional

In [10] Chan and Vese have proposed the following popular functional for the variational image segmentation problem:

$$E(S,c^f,c^b) = \mu\int_{\partial S} dl + \nu\int_S dp + \lambda_1\int_S \Big(I(p)-c^f\Big)^2 dp + \lambda_2\int_{\bar S}\Big(I(p)-c^b\Big)^2 dp \ , \quad (11)$$

where $S$ denotes the foreground segment, $\bar S$ denotes the background segment, and $I(p)$ is a grayscale image. The first two terms measure the length of the boundary and the area, the third and the forth terms are the integrals over the fore- and background of the difference between image intensity and the two intensity values



183x162, time=3s, proportion=1:115 | 300x250, time=4s, proportion=1:103 | 385x264, time=16s, proportion=1:56 | 371x255, time=21s, proportion=1:98

**Fig. 8** The global minima of the Chan-Vese functional for medical and aerial images. These global minima were found using our framework in the specified amount of time; specified proportion of the tree was traversed.

$c^f$ and $c^b$, which correspond to the average intensities of the respective regions. Traditionally, this functional is optimized using level set framework [37] converging to one of its local minima.

Below, we show that the discretized version of this functional can be optimized globally within our framework. Indeed, the discrete version of (11) can be written [45,14,34,23,3,16] as (using notation as before):

$$E(\mathbf{x},(c^f,c^b)) = \sum_{p,q\in\mathcal{E}} \frac{\mu}{|p-q|}\cdot|x_p - x_q| + \sum_{p\in\mathcal{V}} \left(\nu + \lambda_1 (I(p)-c^f)^2\right)\cdot x_p + \sum_{p\in\mathcal{V}} \lambda_2 \left(I(p)-c^b\right)^2 \cdot (1-x_p)\ . \tag{12}$$

Here, the first term approximates the first term of (11) multiplied by a constant (the accuracy of the approximation and the constant depend on the size of the pixel neighborhood [6]), and the last two terms express the last three terms of (11) in a discrete setting.

The functional (12) clearly has the form (1) with non-local parameter $\omega = \{c^f, c^b\}$. Discretizing intensities $c^f$and $c^b$ into 255 levels and building a quad-tree over their joint domain, we can apply our framework to find the global minima of (11). Example globally optimal segmentations are shown on Fig. 8.

**Measuring the efficiency of Branch-and-Mincut.** For the particular case of the Chan-Vese functional, it is easy to measure the efficiency of our branch-and-bound procedure for the varying size of the non-local parameter set. To do that, we varied the number of discretization levels, setting it to both higher and lower values than 256. For one of the examples (Fig. 8-left), we then measured the number of calls to the bound made by our procedure as a function of $|\Omega|$. The result (in Fig. 9) verifies the clear sublinear growth of this quantity, thus demonstrating the efficacy of the branch-and-bound in our case.

**Does global optimality matter for the Chan-Vese case?** The recent graph-based algorithms [45,23,16] developed efficient schemes based on the EM-style iterations that alternate the graph cut step with the reestimation of $c^f$ and $c^b$. While these approaches lack the global optimality guarantees provided by our approach (as well as [14,3,9]), the alternation-based approach provides a huge speed advantage. We have implemented the alternation-based scheme for the energy (12) and found out that over several dozens of examples the alternation scheme always converged to the global optimum (centered rectangle half the size of the image was used for the initialization). Needless to say, the global optima did not always correspond to the accurate or meaningful segmentations. One may therefore conclude 1) that the lack of global optimality guarantees for alternation-based schemes do not have any practical significance in the case of the (two-phase) Chan-Vese functional and 2) that the functional itself (not the optimization schemes) is not suitable for some of the segmentation tasks that we have considered.

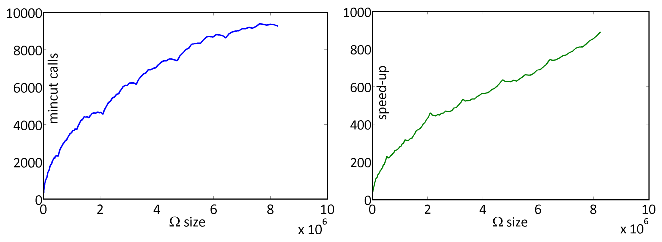


**Fig. 9** As the size of the the non-local parameter set $|\Omega|$ (x-axes) grows, the plots demonstrate the number of calls to the mincut (left) and the ratio between the size of $\Omega$ to the number of mincut call (speed-up – right). Both plots demonstrate that the branch-and-mincut scales sublinearly (verified by a sublinear growth on the left and monotonic growth on the right) with the size of $\Omega$. This experiment was performed for the Chan-Vese segmentation of one of the images.

## 5.2 Segmenting Color Images: GrabCut functional

In [38], the *GrabCut* framework for the interactive color image segmentation based on Gaussian mixtures was proposed. In GrabCut, the segmentation is driven by the following energy:

$$
\begin{aligned}
&E_{\text{GrabCut}}(\mathbf{x},(GM^f,GM^b)) = \\
&\quad\sum_{p\in V} -\log(\mathrm{P}(K_p|\,GM^f))\cdot x_p + \\
&\quad\sum_{p\in V} -\log(\mathrm{P}(K_p|\,GM^b))\cdot(1-x_p) + \\
&\quad\sum_{p,q\in\mathcal{E}} \frac{\lambda_1+\lambda_2\cdot e^{-\frac{||K_p-K_q||^2}{\beta}}}{|p-q|}\cdot|x_p-x_q|\,.
\end{aligned} \tag{13}
$$

Here, $GM^f$ and $GM^b$ are Gaussian mixtures in RGB color space and the first two terms of the energy measure how well these mixtures explain colors $K_p$ of pixels attributed to foreground and background respectively. The third term is the contrast sensitive edge term, ensuring that the segmentation boundary is compact and tends to stick to color region boundaries in the image. In addition to this energy, the user provides supervision in the form of a bounding rectangle and brush strokes, specifying which parts of the image should be attributed to the foreground and to the background.

The original method [38] minimizes the energy within EM-style process, alternating between (i) the minimization of (13) over $\mathbf{x}$ given $GM^f$ and $GM^b$ and (ii) refitting the mixtures $GM^f$ and $GM^b$ given $\mathbf{x}$. Despite the use of the global graph cut optimization within the segmentation update step, the whole process yields only a local minimum of (13). In [38], the segmentation is initialized to the provided bounding box and then typically shrinks to one of the local minima.

The energy (13) has the form (1) and therefore can be optimized within Branch-and-Mincut framework, provided that the space of non-local parameters (which in this case is the joint space of the Gaussian mixtures for the foreground and for the background) is discretized and the tree of the subregions is built. In this scenario, however, the dense discretization of the non-local parameter space is infeasible (if the mixtures contain $n$ Gaussians then the space is described by $20n-2$ continuous parameters)[2]. It is possible, nevertheless, to choose a much smaller discrete subset $\Omega$ that is still likely to contain a pair of mixtures corresponding to the lower-energy bassin of attraction and better segmentation.

To construct such $\Omega$, we fit a mixture of $M = 8$ Gaussians $G_1, G_2, ...G_M$ with the support areas $a_1, a_2, ...a_M$ to the *whole* image. The support area $a_i$ here counts the number of pixels $p$ such as $\forall j\ \mathrm{P}(K_p|G_i) \geq \mathrm{P}(K_p|G_j)$. We assume that the components are ordered such that the support areas decrease ($a_i > a_{i+1}$). Then, the Gaussian mixtures we consider are defined by the binary vector $\beta = \{\beta_1,\beta_2\ldots\beta_M\} \in \{0,1\}^M$ specifying which Gaussians should be included into the mixture, so that:

$$
\mathrm{P}(K|\,GM(\beta)) = \sum_i \beta_i a_i \mathrm{P}(K|G_i)\,/\sum_i \beta_i a_i\,. \tag{14}
$$

The overall set $\Omega$ is then defined as $\{0,1\}^{2M}$, where odd bits correspond to the foreground mixture vector $\beta^f$ and even bits correspond to the background mixture vector $\beta^b$. Vectors with all even bits and/or all odd bits equal to zero do not correspond to meaningful mixtures and are therefore assigned an infinite cost. The hierarchy tree is naturally defined by the bit-ordering (the first bit corresponding to subdivision into the first two branches etc.).

Depending on the image and the value of $M$, the solutions found by Branch-and-Mincut framework may have larger or smaller energy (13) than the solutions found by the original EM-style method [38]. This is because Branch-and-Mincut here finds the global optimum over the subset of the domain of (13) while [38] searches locally but within the continuous domain. However, for all 15 images in our experiments, improving Branch-and-Mincut solutions with a few EM-style iterations [38] gave lower energy than the original solution of [38]. In most cases, these additional iterations simply refit the Gaussians properly and change very few pixels near boundary (see Fig. 10).

In terms of performance, for $M = 8$ the segmentation takes on average a few dozen seconds (10s and 40s for the images in Fig. 10) for 300x225 image. The proportion of the tree traversed by an active front is one to several hundred (1:963 and 1:283 for the images in Fig. 10).

This experiment suggests the usefulness of Branch-and-Mincut framework as a mean of obtaining good initial point for local methods, when the domain space is too large for an exact branch-and-bound search.

## 6 Discussion

The Branch-and-Mincut framework presented in this paper finds global optima of a wide class of energies dependent on the image segmentation mask and non-local parameters. The joint use of branch-and-bound and graph cut allows efficient traversal of the solution space. The developed framework is useful within a variety of image segmentation scenarios, including segmentation with non-local shape priors and non-local color/intensity priors.

[2] In fact, the global minimum of the GrabCut functional over the set of all Gaussian mixtures is not well defined, because fitting Gaussian mixture to data without additional regularization can achieve arbitrarily high likelihood/low energy. Restricting the set of all mixtures to a large discrete subset done in our case can thus be regarded as a variant of a necessary regularization on mixture parameters.

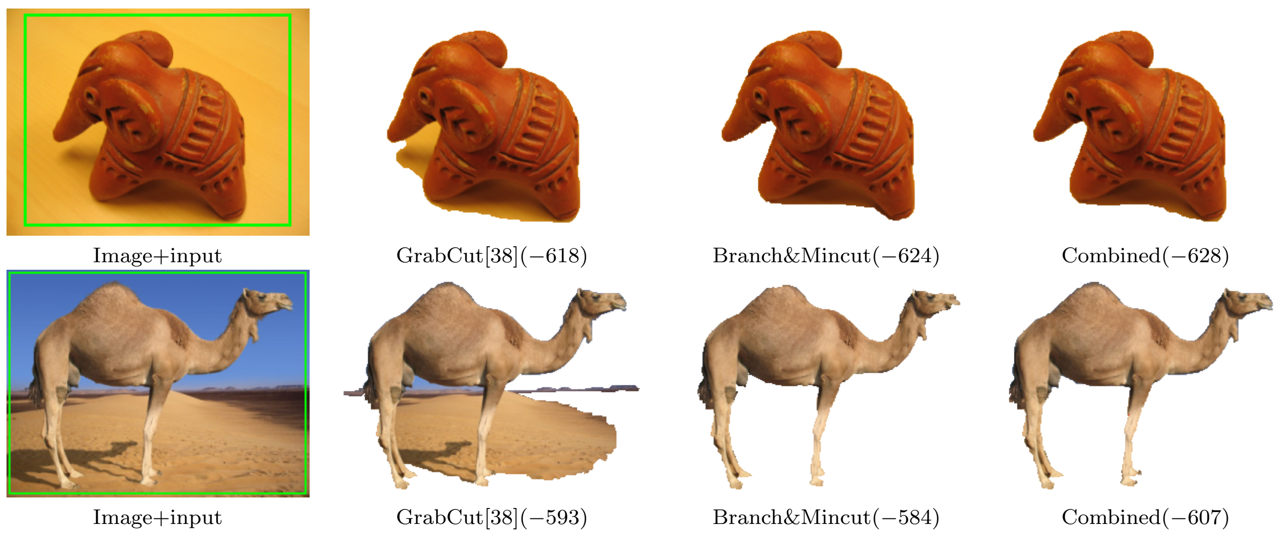


**Fig. 10** Being initialized with the user-provided bounding rectangle (shown in green in the first column) as suggested in [38], EM-style process [38] converges to a local minimum (the second column). Branch-and-Mincut result (the third column) escapes that local minimum and after EM-style improvement lead to the solution with much smaller energy and better segmentation accuracy (the forth column). Energy values are shown in brackets.

One notable characteristic of the presented optimization framework is that it is derivative-free. This makes it applicable to the situations when the derivatives (variations) of the functionals cannot be easily computed (e.g. this allows our method to use shape priors represented as a mere collection of segmentation exemplars). The ignorance about derivatives, of course, mean that in some situations, when the derivatives (variation) of the functionals can be computed, the speed of the proposed framework would be inferior to other optimization methods that use derivatives.

When the speed of Branch-and-Mincut is not sufficient for its practical deployment, it can still be useful to improve and to get insight into the performance of segmentation methods. In general, in the case of poor segmentation results obtained with methods based on local optimization, it is not possible to tell for sure whether the failure is because of inappropriate functional or inappropriate optimization method. In such cases, obtaining global minima with branch-and-mincut allows to find out which of the two reasons really matters. Towards this end, our experiments demonstrate that failures of Chan-Vese functional are always caused by the properties of the functional itself (not the optimization), while for the case of GrabCut, improving the optimization might bring improvement in segmentation accuracy in many situations.

Finally, it is worth noting that the branch-and-bound approach presented in this paper can be extended to other combinatorial optimization methods that are popular in structural image processing, such as multilabel MRF inference or minimum ratio cycles, in a way that the power of the corresponding combinatorial algorithm is exploited "along" the low- (e.g. pixel)-level variables in the energy, while branch-and-bound is used to search "along" the dimensions corresponding to non-local parameter. For the case of tree-based multi-label inference (pictorial structures), we provide such an example in [33].

## References

1. Agarwal, S., Awan, A., Roth, D.: Learning to detect objects in images via a sparse, part-based representation. IEEE Trans. Pattern Anal. Mach. Intell. **26**(11), 1475–1490 (2004)
2. Agarwal, S., Chandraker, M.K., Kahl, F., Kriegman, D.J., Belongie, S.: Practical global optimization for multiview geometry. In: ECCV (1), pp. 592–605 (2006)
3. Bae, E., Tai, X.C.: Efficient global minimization for the multiphase chan-vese model of image segmentation. In: EMMCVPR, pp. 28–41 (2009)
4. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete Applied Mathematics **123**(1-3), 155–225 (2002)
5. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In: ICCV, pp. 105–112 (2001)
6. Boykov, Y., Kolmogorov, V.: Computing geodesics and minimal surfaces via graph cuts. In: ICCV, pp. 26–33 (2003)
7. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. **26**(9), 1124–1137 (2004)
8. Bray, M., Kohli, P., Torr, P.H.S.: Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In: ECCV (2), pp. 642–655 (2006)
9. Brown, E.S., Chan, T.F., Bresson, X.: Completely convex formulation of the chan-vese image segmentation model. International Journal of Computer Vision (2011). Doi: 10.1007/s11263-011-0499-y

10. Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Transactions on Image Processing **10**(2), 266–277 (2001)
11. Clausen, J.: Branch and bound algorithms - principles and examples (2003)
12. Cremers, D., Osher, S., Soatto, S.: Kernel density estimation and intrinsic alignment for shape priors in level set segmentation. International Journal of Computer Vision **69**(3), 335–351 (2006)
13. Cremers, D., Schmidt, F.R., Barthel, F.: Shape priors in variational image segmentation: Convexity, lipschitz continuity and globally optimal solutions. In: CVPR (2008)
14. Darbon, J.: A note on the discrete binary mumford-shah model. Proceedings of Computer Vision/Computer Graphics Collaboration Techniques, (MIRAGE 2007), LNCS Series vol. 44182 pp. 283–294 (March 2007)
15. Delong, A., Boykov, Y.: Globally optimal segmentation of multi-region objects. In: ICCV (2009)
16. El-Zehiry, N.Y., Sahoo, P., Elmaghraby, A.: Combinatorial optimization of the piecewise constant mumford-shah functional with application to scalar/vector valued and volumetric image segmentation. Image Vision Comput. **29**(6), 365–381 (2011)
17. Felzenszwalb, P., Veksler, O.: Tiered scene labeling with dynamic programming. In: CVPR (2010)
18. Felzenszwalb, P.F.: Representation and detection of deformable shapes. IEEE Trans. Pattern Anal. Mach. Intell. **27**(2), 208–220 (2005)
19. Freedman, D.: Effective tracking through tree-search. IEEE Trans. Pattern Anal. Mach. Intell. **25**(5), 604–615 (2003)
20. Freedman, D., Zhang, T.: Interactive graph cut based segmentation with shape priors. In: CVPR (1), pp. 755–762 (2005)
21. Gallo, G., Grigoriadis, M.D., Tarjan, R.E.: A fast parametric maximum flow algorithm and applications. SIAM Journal on Computing **18**(1), 30–55 (1989)
22. Gavrila, D., Philomin, V.: Real-time object detection for "smart" vehicles. In: ICCV, pp. 87–93 (1999)
23. Grady, L., Alvino, C.V.: The piecewise smooth mumford-shah functional on an arbitrary graph. IEEE Transactions on Image Processing **18**(11), 2547–2561 (2009)
24. Greig, D.M., Porteous, B.T., Seheult, A.H.: Exact maximum a posteriori estimation for binary images. Journal of the Royal Statistical Society **51**(2) (1989)
25. Huang, R., Pavlovic, V., Metaxas, D.N.: A graphical model framework for coupling mrfs and deformable models. In: CVPR (2), pp. 739–746 (2004)
26. Kim, J., Zabih, R.: A segmentation algorithm for contrast-enhanced images. In: ICCV, pp. 502–509 (2003)
27. Kohli, P., Torr, P.H.S.: Effciently solving dynamic markov random fields using graph cuts. In: ICCV, pp. 922–929 (2005)
28. Kolmogorov, V., Boykov, Y., Rother, C.: Applications of parametric maxflow in computer vision. In: ICCV, pp. 1–8 (2007)
29. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? In: ECCV (3), pp. 65–81 (2002)
30. Kumar, M.P., Torr, P.H.S., Zisserman, A.: Obj cut. In: CVPR (1), pp. 18–25 (2005)
31. Lampert, C.H., Blaschko, M.B., Hofmann, T.: Beyond sliding windows: Object localization by efficient subwindow search. In: CVPR (2008)
32. Leibe, B., Leonardis, A., Schiele, B.: Robust object detection with interleaved categorization and segmentation. IJCV **77**(1-3), 259–289 (2008)
33. Lempitsky, V., Blake, A., Rother, C.: Exact optimization for markov random fields with non-local parameters. In: Advances in Markov Random Fields for Vision and Image Processing. MIT Press (2011)
34. Lempitsky, V.S., Blake, A., Rother, C.: Image segmentation by branch-and-mincut. In: ECCV (4), pp. 15–29 (2008)
35. Leventon, M.E., Grimson, W.E.L., Faugeras, O.D.: Statistical shape influence in geodesic active contours. In: CVPR, pp. 1316–1323 (2000)
36. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
37. Osher, S., Sethian, J.A.: Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. JOURNAL OF COMPUTATIONAL PHYSICS **79**(1), 12–49 (1988)
38. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004)
39. Sara Vicente, V.K., Rother, C.: Joint optimization of segmentation and appearance models. In: ICCV (2009)
40. Schoenemann, T., Cremers, D.: Globally optimal image segmentation with an elastic shape prior. In: ICCV, pp. 1–6 (2007)
41. Schoenemann, T., Schmidt, F.R., Cremers, D.: Image segmentation with elastic shape priors via global geodesics in product spaces. In: British Machine Vision Conference (BMVC). Leeds, UK (2008)
42. Sinop, A.K., Grady, L.: Uninitialized, globally optimal, graph-based rectilinear shape segmentation the opposing metrics method. In: ICCV, pp. 1–8 (2007)
43. Strandmark, P., Kahl, F., Overgaard, N.C.: Optimizing parametric total variation models. In: ICCV (2009)
44. Wang, Y., Staib, L.H.: Boundary finding with correspondence using statistical shape models. In: CVPR, pp. 338–345 (1998)
45. Zeng, X., Chen, W., Peng, Q.: Efficiently solving the piecewise constant mumford-shah model using graph cuts. Tech. rep. (2006)

## Appendix

**Corollary**. The bound (2) is monotonic, i.e. if $\Omega_1 \subset \Omega_2$ then $L(\Omega_1) \geq L(\Omega_2)$.

**Proof.** Let us denote with $A(\mathbf{x}, \Omega)$ the expression within the outer minimum of (2):

$$A(\mathbf{x}, \Omega) = \Bigg[ \min_{\omega \in \Omega} C(\omega) + \sum_{p \in \mathcal{V}} \min_{\omega \in \Omega} F^p(\omega) \cdot x_p + \sum_{p \in \mathcal{V}} \min_{\omega \in \Omega} B^p(\omega) \cdot (1 - x_p) + \sum_{p,q \in \mathcal{E}} \min_{\omega \in \Omega} P^{pq}(\omega) \cdot |x_p - x_q| \Bigg] . \tag{15}$$

Then, (2) reformulates as:

$$L(\Omega) = \min_{\mathbf{x} \in 2^{\mathcal{V}}} A(\mathbf{x}, \Omega) . \tag{16}$$

Assume $\Omega_1 \subset \Omega_2$. Then, for any fixed $\mathbf{x}$, for all pixels $p$ and edges $p, q \in \mathcal{E}$, the following inequalities hold:

$$\min_{\omega \in \Omega_1} C(\omega) \geq \min_{\omega \in \Omega_2} C(\omega) \tag{17}$$

$$\min_{\omega \in \Omega_1} F^p(\omega) \cdot x_p \geq \min_{\omega \in \Omega_2} F^p(\omega) \cdot x_p \tag{18}$$

$$\min_{\omega \in \Omega_1} B^p(\omega) \cdot (1 - x_p) \geq \min_{\omega \in \Omega_2} B^p(\omega) \cdot (1 - x_p) \tag{19}$$

$$\min_{\omega \in \Omega_1} P^{pq}(\omega) \cdot |x_p - x_q| \geq \min_{\omega \in \Omega_2} P^{pq}(\omega) \cdot |x_p - x_q| . \tag{20}$$

This is because, firstly, all values $x_p$, $1-x_p$, and $|x_p-x_q|$ are non-negative (recall that all $x_p$ takes the value of 0 or 1) and, secondly, all minima on the left side are taken over a subset of the domain of the same minima on the right side.

Summing up inequalities (17)–(20) over all pixels $p$ and edges $p, q$ and taking into account the definition (15), we get:

$$\forall \mathbf{x} \qquad A(\mathbf{x}, \Omega_1) \geq A(\mathbf{x}, \Omega_2) \ , \tag{21}$$

i.e. monotonicity holds for any fixed $\mathbf{x}$.

Let $\mathbf{x}_1$ be the segmentation delivering the global optimum of $A(\mathbf{x}, \Omega_1)$: $\mathbf{x}_1 = \arg\min_{\mathbf{x} \in 2^{\mathcal{V}}} A(\mathbf{x}, \Omega_1)$. Let $\mathbf{x}_2$ be the segmentation delivering the global optimum of $A(\mathbf{x}, \Omega_2)$: $\mathbf{x}_2 = \arg\min_{\mathbf{x} \in 2^{\mathcal{V}}} A(\mathbf{x}, \Omega_2)$. Then, from the definition (15) and the monotonicity (21), one gets:

$$L(\Omega_1) = A(\mathbf{x}_1, \Omega_1) \geq A(\mathbf{x}_1, \Omega_2) \geq A(\mathbf{x}_2, \Omega_2) = L(\Omega_2) \ , \tag{22}$$

which concludes the proof.