Medical Image Analysis 17 (2013) 101–112

Contents lists available at SciVerse ScienceDirect

Medical Image Analysis

journal homepage: www.elsevier.com/locate/media

Segmentation with area constraints

Marc Niethammer[a,b,*], Christopher Zach[c]

[a] Department of Computer Science, University of North Carolina (UNC), Chapel Hill, USA
[b] Biomedical Research Imaging Center, School of Medicine, University of North Carolina (UNC), Chapel Hill, USA
[c] Microsoft Research, Cambridge, UK

ARTICLE INFO

Article history:
Received 6 October 2011
Received in revised form 27 July 2012
Accepted 10 September 2012
Available online 28 September 2012

Keywords:
Segmentation
Area-constraint
Branch and bound
Alternating direction method of multipliers

ABSTRACT

Image segmentation approaches typically incorporate weak regularity conditions such as boundary length or curvature terms, or use shape information. High-level information such as a desired area or volume, or a particular topology are only implicitly specified. In this paper we develop a segmentation method with explicit bounds on the segmented area. Area constraints allow for the soft selection of meaningful solutions, and can counteract the shrinking bias of length-based regularization. We analyze the intrinsic problems of convex relaxations proposed in the literature for segmentation with size constraints. Hence, we formulate the area-constrained segmentation task as a mixed integer program, propose a branch and bound method for exact minimization, and use convex relaxations to obtain the required lower energy bounds on candidate solutions. We also provide a numerical scheme to solve the convex subproblems. We demonstrate the method for segmentations of vesicles from electron tomography images.

1. Introduction

Image segmentation is a fundamental task in image analysis. Consequentially, a large number of segmentation methods have been developed ranging from local thresholding to methods using statistical models of shape variation (Pham et al., 2000; Sonka et al., 2008). The simplest available segmentation methods rely on local pixel-by-pixel segmentation decisions such as Otsu thresholding or methods based on clustering. These fully-local decisions are often not sufficient and because they neglect spatial dependencies, they are sensitive to noise and not directly applicable if an object is defined by its boundary surface only (e.g., if only the cell membrane or a cell membrane surrogate is imaged, but an image of the entire cell is desired). To overcome these limitations, nonlocal approaches have been proposed based on intelligent local merging decisions or by formulating optimization problems incorporating spatial dependencies. The former class of methods encompasses region growing approaches such as the popular watershed segmentation (Sonka et al., 2008). The latter class of methods includes active-contours and -surfaces (Sapiro, 2001) as well as general parametric models which may use statistical information on shape and/or appearance (Cootes et al., 2001; Pizer et al., 2003).

When the object segmentation task is highly structured (i.e., expected shape variations are reasonably small and the approximate number and location of the objects are known) shape- or atlas-based segmentation methods are highly successful (Rohlfing et al., 2005). However, for less structured cases these methods are not applicable. In microscopy, for example, images often contain hundreds or thousands of cells, cell nuclei, or organelles, with possibly large variations in shape and a priori unknown locations. While local thresholding or active-contour-type models may be applied in such cases, they are often too generic, too sensitive to noise, or require the judicial placement of seed points to assure an appropriate segmentation result to avoid over- or under-segmentations.

If shape- or atlas-based segmentation methods are too restrictive, and if general purpose segmentation methods such as active contours, region-growing or thresholding are not restrictive enough for a particular segmentation task, the question of how to incorporate additional domain information into a segmentation that lies between these two extremes arises. A possible option is to use information about simple geometric properties. In this paper we explore an approach for a segmentation with constraints on the segmentation area. Such a method can counteract potential leakage or shrinkage biases in a principled way. Such biases can be observed, for example, for active contour (Sapiro, 2001) or graph cut (Boykov and Funka-Lea, 2006) segmentations when boundary regularity is encouraged by penalizing a weighted length of the segmentation boundary. Area constraints may not be appropriate for all biomedical segmentation tasks; however there are a large number of problems in which reasonable area or volume intervals are known a priori. Our objective in this paper is not to perform an

[*] Corresponding author at: Department of Computer Science, University of North Carolina (UNC), Chapel Hill, USA.
E-mail address: mn@cs.unc.edu (M. Niethammer).

1361-8415/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.media.2012.09.002

actual study for a particular biological problem, but rather to demonstrate the behavior of a segmentation method with area-constraints on realistic image data. We use electron tomography datasets of synaptic vesicles and of double-membrane vesicles (DMVs) implicated in the SARS-coronavirus (severe acute respiratory syndrome coronavirus) replication (Knoops et al., 2008).

Many recent segmentation approaches are formulated such that the optimization problems become convex so that globally optimal solutions can be obtained (Appleton and Talbot, 2006; Bresson et al., 2007) or so that they can be solved with discrete solution methods, such as graph-cuts (Boykov and Funka-Lea, 2006). While area-constraints can formally easily be added to the optimization problems for segmentation, solving the problems is hard. However, if finding a globally optimal solution is not of concern and a good initial guess for a solution is available, one can resort to standard methods from constrained optimization. For example, a curve evolution approach with an area penalty can be used (Ayed et al., 2008). Proposed numerical solution approaches to obtain a global optimum or a good approximation.

- are limited to problems with small numbers of variables (Ji, 2004),
- or require long computation times (Dahl and Flatberg, 2007),
- use solution heuristics (Kernighan and Lin, 1970),
- or use various forms of relaxations of the original problem to facilitate computations: e.g., spectral relaxations (Olsson et al., 2008), semidefinite programming (Keuchel et al., 2003; Lisser and Rendl, 2003; Hager et al., 2009), or variational inference approximations (Kropotov et al., 2010).

Approaches have generally focused on equality constraints (i.e., exact size) in formulation (Lim et al., 2010; Eriksson et al., 2011; Falkner et al., 1994; Ayed et al., 2008) or for testing (Hager et al., 2009). However, equality constraints have only limited applicability when the exact object size is not known beforehand or when it is a desired measurement (as is frequently the case in biomedical imaging), because it would bias the segmentation towards the chosen area. We therefore formulate the segmentation problem with inequality constraints on the segmentation area.

Section 2 introduces the area-constrained segmentation problem. Section 3 outlines our solution approach. Sections 4–6 discuss its numerical solution. Segmentation results on real electron tomography images demonstrate the utility of the method in Section 7. The paper concludes with a summary and a discussion of future work.

2. Optimization problem

Our objective is a binary segmentation of an image into foreground and background. Without loss of generality, we consider two-dimensional images here.[1] Markov random field models with Gibbs energies using first and second order cliques have been particularly popular for image segmentation (Li, 2009) and can be exactly minimized under certain conditions (Kolmogorov and Zabin, 2004) for example by using graph cuts. Solutions are typically based on the minimal cut theorem (Ford and Fulkerson, 1956) relating the minimum cut in a graph to the maximum flow through the graph. Hence, by forming an appropriate graph and solving the maximum flow problem the segmentation solution can be obtained. The segmentation algorithm we analyze and extend is the partial differential equation formulation of the maximum flow problem (Appleton and Talbot, 2006), which has equally broad application for image segmentation.

As is customary for image segmentation methods based on energies of Gibbs-type, we allow the optional specification of seed points or areas which explicitly enforce a particular labeling (foreground or background) for the seeds. Our segmentation formulation is an extension of the convex formulation of the active contour method and related segmentation methods such as the Chan-Vese segmentation model (Bresson et al., 2007; Chan and Vese, 2001).

To avoid the segmentation of very small structures which likely represent noise and noisy boundaries, almost all energy-based methods penalize the (weighted) length of the boundary curve separating foreground from background. Most commonly, the length of the boundary curve is added to the segmentation energy. This introduces a well-known shrinking bias towards shorter boundary curves and therefore frequently leads to undersegmentations. Our goal is to add constraints on the segmentation area, to counteract the shrinking bias and to allow "tuning" of the segmentation algorithm to the expected size of the objects to be segmented.

While our area-constrained extension is developed in the context of maximum-flow-based segmentation, the solution strategy itself is generic and expected to be applicable also to other segmentation models. For example, a similar solution strategy might be useful for segmentation methods which do not penalize boundary curve length directly, but instead a ratio between boundary length and enclosed areas (Grady and Schwartz, 2006; Shi and Malik, 2000). Such methods do not exhibit the same shrinking bias, but also lack control over the obtained segmentation area.

The general optimization problem for area-constrained segmentation we consider is

$$\mathrm{argmin}_u E(u), \quad \text{s.t. } A(u) \in [A_l, A_u], \tag{1}$$

where

$$u = \{u_s | s \in \mathscr{X}\}; u_s \in \{0, 1\},$$

and

$$\begin{cases} u_s = 0, s \in \mathscr{T}, \\ u_s = 1, s \in \mathscr{S}. \end{cases}$$

Here, u is an indicator function denoting foreground ($u = 1$) and background ($u = 0$) classes of the segmentation defined over the set of spatial locations $\mathscr{X}$, where s denotes a spatial location, u_s a value at location s and u is the union of these values for all $s \in \mathscr{X}$, i.e., $u \in \{0,1\}^{|\mathscr{X}|}$, where $|\mathscr{X}|$ denotes the number of considered spatial locations. $E(u)$ is an energy function encoding the desired properties of a segmentation, $A(u)$ indicates the area covered by a segmentation u, $\mathscr{S}$ and $\mathscr{T}$ denote the foreground and background seed sets respectively,[2] and A_l and A_u are lower and upper bounds for the segmentation area. This is an integer program (Nemhauser and Wolsey, 1988) since $u_s \in \{0,1\}$.

For image segmentation the number of variables, u_s, corresponds to the number of pixels. A direct solution of problem (1) with integer-programming methods is typically computationally tractable only for very small images. For segmentations without area-constraints various relaxations of the original labeling problem have therefore been proposed. As an illustrative example, consider the continuous maximum flow approach (Appleton and Talbot, 2006) in which

$$E_{mf}(u) = \sum_s g_s \|\nabla_s u\| + \rho_s u_s, \ \text{s.t.} \quad u_s \in [0, 1], \tag{2}$$

[1] The overall algorithm and its analysis extends to higher dimensions. We use two-dimensional terminology in the remainder of the paper only to simplify the presentation, e.g., boundary curve instead of boundary surface.

[2] The foreground and background seeds may be absorbed into E, but keeping them separate will be useful for the branch and bound solution approach.

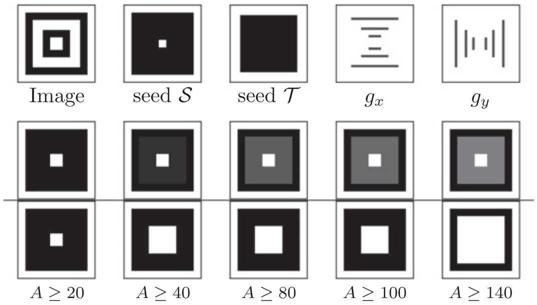

Fig. 1. Three squares segmentation experiment (top): image to be segmented, foreground seed set $\mathscr{S}$, background seed set $\mathscr{T}$ and the cost g in the x and y direction ($g_x = 1/(1 + 50I_x)$ and $g_y = 1/(1 + 50I_y)$) respectively (from top left to top right). The cost is illustrated with directional dependence, because we use the 1-norm, $g_s\|\nabla u_s\|_1$, to discretize the weighted total-variation term. Linear programming (LP) solution for different lower area bounds (middle); solution of integer program (bottom). The integer program is by construction binary and is able to capture all three concentric squares (with respect to the gradient magnitude). The LP solution is blind to the middle square and immediately "bleeds out" into the biggest square once the desired area is larger than the smallest square at the center.

subject to the same seedpoint constraints as in (1) is minimized. Here, $g > 0$ denotes an edge-weighting term and ρ a regional bias which allows for the integration of local likelihoods of an element s to belong to the foreground or the background.[3] The key difference in structure is to allow $u_s \in [0,1]$ which renders the optimization problem convex, because it is now defined over a convex domain, $u \in [0,1]^{|S|}$. A globally optimal solution can then efficiently be obtained. For the continuous maximum flow problem, the optimal solution will be *essentially binary* regardless the convex relaxation. This means that a minimizer, u^* of (2) may not necessarily be binary, but any thresholded $u_s^{\theta*} = \mathbb{1}_{[\theta,1]}(u_s^*)$ is binary and globally optimal with $E(u^*) = E(u^{\theta*})$ for $\theta \in (0,1)$ (Appleton and Talbot, 2006). Here, $\mathbb{1}_S(x)$ is the indicator function which returns 1 if x is in S and 0 otherwise.

Unfortunately, this relaxed solution is no longer guaranteed to be essentially binary when adding the area-constraint by inequality constraints $\sum_s u_s \geqslant A_l$, $\sum_s u_s \leqslant A_u$. The "area" of a segmentation is defined as $A(u) := \sum_s u_s$. In addition, the segmentation method can become "blind" to the true optimal integer solution as illustrated in Section 2.1, which precludes the possibility of finding good approximate solutions for the integer program by thresholding the relaxed solution. A different solution strategy is therefore needed for area-constrained segmentation. We resort to branch and bound (see Section 6).

2.1. Problem with the relaxed solution

Assume there are strong gradients along the boundary of concentric, non-intersecting shapes. For example, several circles with increasing radii or squares with increasing side lengths with small weights g. Assume that the weights are chosen such that all discontinuities of the resulting segmentation (for the original problem and its relaxation) occur only at these shape boundaries. This can always be achieved by assigning sufficiently large weights outside the desired boundaries. These concentric shapes are indexed by a scale parameter r, e.g. the radius of a circle or the diagonal of a square. The shape itself is not important, but only that the area of the shape is $c_A r^2$ and its circumference is $c_L r$ for suitable constants c_A and c_L. We drop these constants without loss of generality in what follows. We would like the segmentation to snap into successively larger shapes when increasing the lower bound on the area. The following counter-example shows that this cannot be assured and therefore the convex relaxation (with $u_s \in [0,1]$) is insufficient to obtain solutions to the area-constrained segmentation problem.

Consider three concentric shapes with scales $r_1 < r_2 < r_3$ (see Fig. 1) and a sufficiently large seed region within the inner-most shape so that the unconstrained problem results in the segmentation of the smallest shape. The segmentation energy (2) is proportional to r_i (for $\rho_s = 0$) and the area is proportional to r_i^2. Without loss of generality, set $r_1 = 1$. To obtain the middle shape from the segmentation, we enforce $A_l \in (r_1^2, r_2^2) = (1, r_2^2)$. The segmentation energy of the desired shape (for an integer-solution) is $E_{int} = r_2$. Under the relaxed segmentation model the optimal solution needs to occur at r_2 or r_3. Since smaller values for u will lead to smaller overall energy values, the optimal relaxed solution will have $A(u) = A_l$. Therefore, for a jump at r_i the uniform fractional value for u which fulfills the area constraint exactly will be

$$u = \frac{A_l - r_1^2}{r_i^2 - r_1^2} = \frac{A_l - 1}{r_i^2 - 1}, \quad i \in \{2, 3\}.$$

The energy values (for a jump from 1 to u at $r_1 = 1$ and from u to 0 at r_i) are

$$\widehat{E}_i = r_1(1 - u) + r_i u = \frac{r_i^2 - 1 + (A_l - 1)(r_i - 1)}{r_i^2 - 1} = \frac{r_i + A_l}{r_i + 1}.$$

But then $A_l > 1$ and $r_3 > r_2$ by assumption leads to $\widehat{E}_3 < \widehat{E}_2$. This shows that the middle shape cannot be recovered by thresholding and the fractional solution has a lower energy than the solution for the integer program. Fig. 1 illustrates the difference between the relaxed and integer solutions for successively larger lower bounds on the area for concentric squares with foreground seeds at the center of the image and background seeds at the image boundary. As predicted, the relaxed solution is blind to the middle square and simply uniformly increases the fractional values of u with increasing A. In contrast, the integer solution is able to capture all three squares.

2.2. Proposed formulation

While the relaxed solution is not suitable for area-constrained segmentation by itself it can be used to obtain lower bounds for the integer program. Instead of directly enforcing $u \in \{0,1\}$ we formulate the optimization problem as a mixed integer nonlinear program (MINLP) (Hijazi et al., 2009)

$$\begin{aligned}
&\min\nolimits_u E_{minlp}(u),\\
&E_{minlp}(u) = \sum_s g_s\|\nabla u_s\| + \rho_s u_s,\\
&\text{s.t. } A(u) \in [A_l, A_u]; u_s \in [0,1];\\
&\begin{cases} u_s = 0, s \in \mathscr{T}, \\ u_s = 1, s \in \mathscr{S}, \end{cases}\\
&u_s = b_k, s \in \mathscr{B}_k, b_k \in \{0,1\}, \forall k\\
&u \text{ essentially binary},
\end{aligned} \tag{3}$$

which augments the maximal flow formulation (2) by selection variables b_k and areas $\mathscr{B}_k$, which allow selection of additional

[3] This is a very general energy form which can express many highly popular segmentation models, such as active contour and surface models, Chan-Vese segmentation, and segmentation models with general region-based likelihoods. For example, any energy of the form $E(C) = \int_{\Omega_1} -log(p_1(d(x)|\theta_1)dx + \int_{\Omega_2} -log(p_2(d(x)|\theta_2)dx + \int_C g(s)ds$ can be written in the form of Eq. (2). Here, C is the boundary curve separating the foreground region, Ω_1, from the background region, Ω_2; $d(x)$ denotes available data at spatial location x, θ_i are given parameters, which typically parameterize the likelihoods p_i, $g(s) > 0$ and s denotes arc-length. Hence, ρ_s in Eq. (2) can be interpreted as the logarithm of the likelihood ratios in the foreground and the background regions at location s.

foreground and background seeds. For practical segmentation problems full control over all pixels is in many cases not necessary. Instead, it is desirable to obtain a good approximation to the original optimization problem while controlling the computational complexity of the method. Hence, we replace the control of individual pixels by the control of coarse selection areas $\mathscr{B}_k$. Since we solve this problem by branch-and-bound the resulting reduction in the number of integer variables reduces the effort to compute the solution drastically (because it reduces the size of the branch and bound tree). The original integer program (1) with the non-relaxed maximum flow energy is recovered if the $\mathscr{B}_k$ correspond to individual image pixels $s \notin \mathscr{S} \cup \mathscr{T}$. If desired, maximal flow formulations with direction-dependent costs could be used (Zach et al., 2009b,a). For formulations with only a lower area bound, the last condition is replaced by $u_s \geqslant b_k$ (and similarly for only an upper bound).

The essentially binary property is a consequence of the underlying continuous max-flow solutions and means that given an optimal (not-necessarily binary) u an equally optimal solution can be found by thresholding u for any $\theta \in (0,1)$. I.e., we only want to accept values for the selection variables b_k which result in essentially binary solutions and therefore indicate that they were selected appropriately to avoid the problems discussed in Section 2.1.

Intuitively, the maximal flow approach yields an essentially binary solution even when an area constraint is present if it is sufficiently constrained by seed points. For example, imposing a lower area constraint for a max-flow-based segmentation is trivial if the number of foreground seed points is larger or equal to the lower area bound. Then the constraint is essentially inactive and "invisible" to the segmentation algorithm. The difficulty lies in finding these seed points (without requiring a user to provide close to the final segmentation as input). Integer programming solves this problem by intelligent pixel-by-pixel searching. However, even though the existing search methods (Nemhauser and Wolsey, 1988) avoid the combinatorial explosion inherent to a brute-force approach, search trees will still get extremely large even for moderately small problems unless special problem structure can be exploited. We control the combinatorial explosion instead by an appropriate, coarse choice of selection areas.

We would like the solution to be robust to the choice of the selection areas $\mathscr{B}_k$. The solution boundaries are expected to be located close to where their cost is low, i.e., where g_s is small. Hence, we try to avoid placing the boundaries of selection regions there and let the remaining pixels not covered by any selection region snap into the best boundary location. We use homogeneous image regions for the $\mathscr{B}_k$, which can be derived from super-pixels (Vedaldi and Soatto, 2008; Comaniciu and Meer, 2002) or from an image oversegmentation using a watershed method (Vincent and Soille, 1991). Formulation (3) is more general than a direct super-pixel segmentation (e.g., one can use seed regions covering only a subset of the image to guide the segmentation while having complete representational freedom close to putative segmentation boundaries). To define the $\mathscr{B}_k$, we use quick-shift (Vedaldi and Soatto, 2008) to find super-pixels and erode them so that they do not touch the potential segmentation boundaries. Quick-shift is an efficient mode-seeking algorithm based on medoid shift (conceptually similar to the popular mean-shift segmentation algorithms Comaniciu and Meer, 2002). It provides a tuning parameter to control under- and over-fragmentation of modes and can therefore be used to indirectly control the number of selection regions to be detected. Our method is not dependent on quick-shift, and other clustering methods such as mean-shift or k-means (Jain et al., 1999) could be substituted. Fig. 2 shows an illustration of the selection regions. In addition to the max-flow method addressed in this paper we expect this approach of facilitating area-constraints through selection regions also to be generally useful for other segmentation methods.

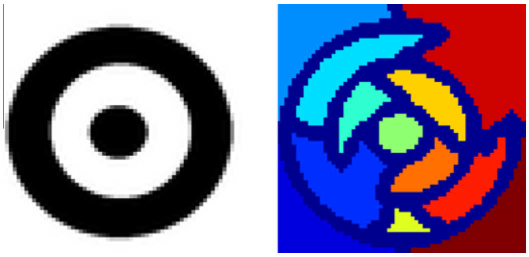

Fig. 2. Illustration of selection regions for a concentric circle example. Left: original image. Right: automatically determined selection regions using quick-shift followed by an erosion. Different colors represent different regions. Dark-blue indicates regions not covered by the selection regions, for which pixels are not controlled by selection regions and can therefore faithfully represent segmentation boundaries. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Note that when only an upper or a lower bound on the area are present, the segmentation can be robust even to selection areas crossing the integer-programming-optimal solution because we replace the equality constraint $u_s = b_k$ by an inequality ($u_s \geqslant b_k$ or $u_s \leqslant b_k$ respectively) effectively resulting in "don't-care" selection areas. Segmentation boundaries can pass *through* such "don't care" selection areas if desired. Specifically, if only a lower-bound is imposed, then the selection regions drive the actual segmentation values u_s through the inequality $u_s \geqslant b_k$. Therefore setting a selection region to 1 forces the segmentation value u_s to be 1 for $s \in \mathscr{B}_k$. However, setting a selection region to 0 amounts to leaving the segmentation values u_s free. Hence, the solution will neither be forced to 0 or 1 in such an area and can be completely determined pixel-wise by the underlying image. The segmentation will be robust to selection regions that cross segmentation boundaries as long as there are a sufficient number of selection regions on the inside of an object that can be set to 1 so that the segmentation naturally "snaps" into the desired location. However, when we enforce a lower and an upper bound, we need to be able to increase *and* decrease the natural size of an unconstrained segmentation by setting regions to 1 or 0 respectively. In this case, selection regions crossing segmentation boundaries will matter because they have to be set to either 0 or 1. Consequentially, enforcing an upper and a lower bound may produce results which are worse than enforcing only a lower bound. This problem could easily be avoided by moving from binary to ternary selection variables (0: set to zero, 1: set to one, 2: do not care). This would leave the overall approach intact, but would result in a slightly different branch and bound implementation, which is not our focus here.

3. Outline of solution approach

Solving the MINLP (3) involves the computation of the optimal binary selection variables, b_k. A brute-force approach enumerating all possible combinations $\{b_k\} \in \{0,1\}^{|b|}$ (where $|b|$ denotes the number of selection areas) is prohibitive for all but the most simple general integer programming problems. We therefore use branch and bound (Nemhauser and Wolsey, 1988) to solve (3), which determines the optimal values of the selection variables b_k by building a search tree. Evaluation of the full search tree (feasible only for small problems) is avoided by guiding the search towards promising solution candidates and discarding branches which can provably not lead to an optimal solution.

For the MINLP energy (3) we introduce the *relaxed* MINLP energy as

$$E_{relaxed}(u) = \sum_s g_s \|\nabla u_s\| + \rho_s u_s,$$

$$\text{s.t.} \quad A(u) \in [A_l, A_u]; u_s \in [0,1];$$

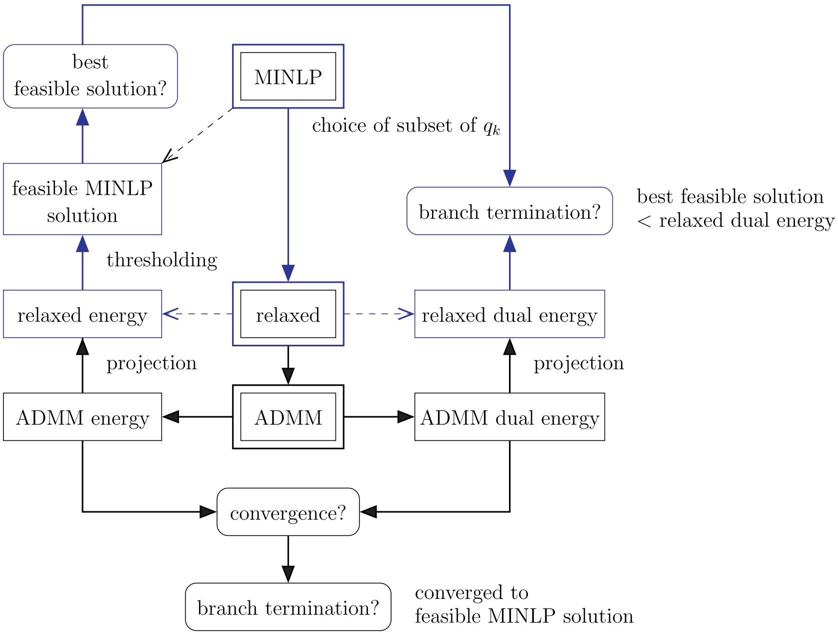

Fig. 3. Relation between the different optimization problems and branch and bound. Blue color indicates path for a non-ADMM-based solution. The goal is to solve the MINLP problem. The branch and bound solver selects (and sets) a subset of selection variables b_k and leaves the remaining ones free. To obtain lower energy bounds and feasible solution to MINLP (given the selected b_k) we use a relaxed formulation. The relaxed formulation is solved by ADMM. We can compute a primal and a dual energy from ADMM. From the ADMM energies we can obtain finite primal and dual relaxed energies by projections. We can obtain a feasible MINLP solution from the projected solution from which the relaxed energy was obtained by thresholding. A current candidate branch is terminated if the relaxed dual energy is larger than the best feasible MINLP solution or if ADMM converged to a feasible MINLP solution. Note that the termination criteria can be checked *before* ADMM convergence. The branch and bound solver builds a search tree for all possible choices for the b_k, but never evaluates branches which can be discarded. Dashed lines indicate possible (but difficult paths), solid lines indicate the proposed approach. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$\begin{cases} u_s = 0, & s \in \mathscr{T}, \\ u_s = 1, & s \in \mathscr{S}, \end{cases}$$

$$u_s = b_k, \quad s \in \mathscr{B}_k, \quad b_k \in \{0,1\}, \quad k \in \mathscr{K}$$

where we dropped the essentially binary condition and removed some of the selection regions. Here, $\mathscr{K}$ is the set containing the indices of the selection regions *which are used in the particular relaxed solution, with all other b_k free.*[4] We have

$$E^*_{relaxed}(p) \leqslant E_{relaxed}(u_r) \leqslant E_{minlp}(u_r)$$

where $E^*_{relaxed}$ denotes the dual energy to $E_{relaxed}$, p is the dual variable to u, and u_r is a feasible candidate solution to the relaxed optimization problem. The first inequality holds, because a dual energy is never larger than the corresponding primal energy. The second inequality holds, because $E_{relaxed}$ removes constraints from E_{minlp} and therefore will either have a smaller energy value than E_{minlp} (if u_r violates some constraints of E_{minlp}) or will be equal to it. If u_r is a feasible solution for E_{minlp} the energy value will be finite, otherwise it will be infinite. Hence, if within the search tree we find a relaxed solution such that $E^*_{relaxed}(p) > E(u^*_{best})$ where u^*_{best} is the current best feasible solution known for E_{minlp} we can prune the search tree for u_r, i.e., we no longer need to look at any solutions for its free selection variables b_k, because they could only cause higher energies. A search branch can further be terminated if it results in a feasible integer solution.

We use the alternating direction method of multipliers (ADMM) (Sections 4 and 5) to compute solution candidates, u_r, to the relaxed problem and show how to compute a dual energy at every iteration step of the optimization algorithm.[5] Section 6 discusses how to use the relaxed dual and primal energies within the branch and bound solution framework and how to obtain finite-valued relaxed dual energies and integer-feasible solutions from the ADMM variables. See Fig. 3 for a graphical overview.

4. Alternating direction method of multipliers

A possible numerical scheme is to perform a standard primal/dual gradient descent/ascent (Reinbacher et al., 2010). While simple, these methods tend to oscillatory behavior and require costly projections at every iteration step to fulfill the area-constraint.[6]

We instead use the alternating direction method of multipliers (ADMM) (Boyd et al., 2010) for the solution of the optimization problem. The basic idea of this method is to split a problem into smaller sub-problems while making use of the method of multipliers developed to solve constrained optimization problems (the augmented Lagrangian approach). This decomposition simplifies the solution process for the area-constrained segmentation problem by breaking it into simpler sub-pieces. It also allows for the computation of a finite-valued dual energy estimate, which serves as a lower bound for the branch and bound algorithm.

4.1. Background on ADMM

We only provide the basic setup for ADMM here for completeness, but refer to Boyd et al., 2010 for details. ADMM optimization problems are of the form

$$\min_{u,w} f(w) + g(u), \quad \text{s.t.} \quad Bw + Cu = c, \tag{4}$$

[4] As for Eq. (3), we allow inequalities for the selection regions when only enforcing lower or upper bounds. The energies change correspondingly.

[5] The iterative solution can be terminated prior to convergence if the dual energy is larger than the best integer-feasible primal energy, $E_{minlp}(u^*_{best})$.

[6] In (Reinbacher et al., 2010) the projection step is solved iteratively. Our approach requires this projection step only for the *evaluation* of the energy, which is not required at every iteration. We also provide a non-iterative method to solve the problem in Section 6.1.

where $u \in \mathbb{R}^n$, $w \in \mathbb{R}^m$, f and g are functions ($f : \mathbb{R}^m \mapsto \mathbb{R}$, $g : \mathbb{R}^n \mapsto \mathbb{R}$) that do not need to be differentiable, $c \in \mathbb{R}^q$ and B and C are appropriately sized matrices.[7] The ADMM update steps (with step size $\sigma > 0$) are (Boyd et al., 2010)

$$\begin{aligned} w^{k+1} &\leftarrow \operatorname{argmin}_w f(w) + \frac{\sigma}{2}\|Bw + Cu^k - c - z^k\|_2^2 = \operatorname{prox}^B_{\frac{1}{\sigma}f}(-Cu^k + c + z^k), \\ u^{k+1} &\leftarrow \operatorname{argmin}_u g(u) + \frac{\sigma}{2}\|Bw^{k+1} + Cu - c - z^k\|_2^2 = \operatorname{prox}^C_{\frac{1}{\sigma}g}(-Bw^{k+1} + c + z^k), \\ z^{k+1} &\leftarrow z^k - (Bw^{k+1} + Cu^{k+1} - c). \end{aligned} \tag{5}$$

This amounts to first solving for w then for u and finally updating the normalized dual variables, z. The prox operator (Combettes and Pesquet, 2010) is defined as

$$\operatorname{prox}^L_f(y) = \operatorname{argmin}_w f(w) + \frac{1}{2}\|Lw - y\|^2.$$

Note that the update scheme for ADMM can readily be derived from an augmented Lagrangian formulation (Nocedal and Wright, 2006). The augmented Lagrangian corresponding to (4) is

$$\begin{aligned} L_\sigma(w, u, p) &= f(w) + g(u) + p^T(Bw + Cu - c) \\ &\quad + \frac{\sigma}{2}\|Bw + Cu - c\|_2^2, \end{aligned} \tag{6}$$

where p is the Lagrangian multiplier. Making the identification $p = \sigma z$, the ADMM Eq. (5) are simply the augmented Lagrangian update equations for (6) where the update for the primal variables is performed separately, and conveniently written using the prox operator. If f is an indicator function for a set C, i.e., $f(x) = \imath_C\{x\}$, (which is 0 if $x \in C$, ∞ otherwise) the prox operator $\operatorname{prox}_f(y)$ is simply the projection of y on C. For general functions f the prox operator $\operatorname{prox}^L_f(y)$ minimizes f while not moving "too far" from y. See (Combettes and Pesquet, 2010) for a more detailed discussion.

4.2. Background on consensus optimization

For area-constrained segmentation, splitting the problem into more than two sub-problems subject to consistency constraints simplifies the solution because it will allow for decoupling of the spatial regularization of the total variation term, $g_s\|\nabla_s u\|$, the unary potential term, $\rho_s u_s$, and the area constraint. The coupling is then re-introduced through a consistency constraint. Specifically, we will have an optimization problem of the form

$$\min_{u_i} \sum_{i=1}^n f_i(u_i), \quad \text{s.t.} \quad u_i - u = 0, \quad \forall i, \tag{7}$$

where the u_i are all independent variable copies and the consensus variable u (our indicator function) is only present through the consistency constraints (i.e., $g(u) = 0$). At convergence, the constraints will be fulfilled and hence $u_i = u$, $\forall i$. The prox step for u then becomes an averaging step (here, $B = I$, $C = -I$, $c = \mathbf{0}$)

$$\begin{aligned} u_i^{k+1} &\leftarrow \operatorname{prox}_{\frac{1}{\sigma}f}(u^k + z_i^k), \\ u^{k+1} &\leftarrow \frac{1}{n}\sum_{i=1}^n (u_i^{k+1} - z_i^k), \\ z_i^{k+1} &\leftarrow z_i^k - (u_i^{k+1} - u^{k+1}). \end{aligned}$$

This global variable consensus (Boyd et al., 2010) formulation is well suited for parallel processing. Constraints on the consensus variable (u) can be encoded in $g(u)$ and therefore allow the specification of seed points for area-constrained segmentation.

Interestingly, this does not change the overall solution scheme much, since the optimization problem

$$u^{k+1} = \operatorname*{argmin}_u \left(g(u) + \sum_{i=1}^n \frac{\sigma}{2}\|u_i^{k+1} - u - z_i^k\|_2^2 \right)$$

can be rewritten in the two-step form

$$\tilde{u}^{k+1} = \frac{1}{n}\sum_{i=1}^n (u_i^{k+1} - z_i^k), \quad u^{k+1} = \operatorname{prox}_{\frac{1}{n\sigma}g}(\tilde{u}^{k+1}),$$

which replaces the update step for u in Eq. (5). We solve the relaxed area-constraint segmentation problem with this form of ADMM by transforming it to look like (7) as described in Section 5. The consensus variable u then corresponds to our sought-for indicator function u.

5. ADMM for area-constrained segmentation

We assume that the set of selection variables $b = \{b_k\}$ of (3) is split into a set of selection variables with known value (within a branch and bound tree) and a set of free selection variables. We then subsume the determined selection variables in the foreground, $\mathscr{S}$, and background, $\mathscr{T}$, seed sets respectively. Dropping the free selection variables from the formulation results in the relaxed area-constrained problem. For simplicity we use the 1-norm for the gradient term resulting in the energy

$$E(u) = \sum_{(s,t)} c_{st}|u_s - u_t| + \sum_s \rho_s u_s, \tag{8}$$

$$\text{s.t.} \quad A_l \leqslant \sum_s u_s \leqslant A_u; u_s \in [0, 1] \tag{9}$$

$$\begin{cases} u_s = 1, & s \in \mathscr{S}, \\ u_s = 0, & s \in \mathscr{T}, \end{cases} \tag{10}$$

where (s,t) denotes a pair of neighboring pixels (in our case using a four-connected neighborhood) and the weighted total variation term $\sum_s g_s\|\nabla u_s\|_1 = \sum_s g_s(|(u_x)_s| + |(u_y)_s|)$ was discretized as $\sum_{(s,t)} c_{st}|u_s - u_t|$. This is a slightly more general formulation, but includes $\sum_s g_s\|\nabla u_s\|_1$ if the spatial gradients in the x and y directions (u_x and u_y) are discretized using finite differences, the sites s are given by the grid position of individual pixels and c_{st} is set to g_s for all t neighboring s. Note that this formulation is sufficiently general to support area-constrained segmentation for general graph structures. To simplify the solution of (10), we break the problem into the following four energies which need to be minimized:

$$\begin{aligned} E_1(u) &= \sum_s \rho_s u_s + \imath_{[0,1]}(u_s) := \sum_s f_s(u_s), \\ E_2(u) &= \imath\{A_l \leqslant \sum_s u_s \leqslant A_u\} := f_A(u), \\ E_3(u) &= \sum_{(s,t)} c_{st}|u_s - u_t| := \sum_{(s,t)} f_{st}(u_s, u_t), \\ E_4(u) &= \sum_s \imath_{[0,1]}(u_s) + \sum_{s\in\mathscr{S}} \imath\{u_s = 1\} + \sum_{s\in\mathscr{T}} \imath\{u_s = 0\} := g(u), \end{aligned}$$

where $\imath_C\{x\}$ denotes the indicator function and we write for notational simplicity $\imath_{C=\{x:f(x)=0\}}\{x\} = \imath\{f(x) = 0\}$. The energies encode the unary potential term, the area constraint, the pairwise-potential (edge) term, and the seeds, respectively. These problems are simple to solve independently. The consensus form of ADMM then allows us to couple the four easy sub-problems so that we obtain a solution of the original optimization problem (10) at convergence.

Specifically, we use variable copies u^A, u^s, $\bar{u}^s$, $\bar{u}^t$ and the consensus variable u. The energy for the consensus ADMM is then

[7] In the specialization of ADMM for the area-constrained segmentation u will be the sought-for indicator-function and w will hold variable copies of u which simplify the numerical solution.

$$E(u,u^s,u^A,\bar{u}^s,\bar{u}^t)=\sum_s f_s(u^s_s)+f_A(u^A)+\sum_{(s,t)} f_{st}(\bar{u}^s_s,\bar{u}^t_t)+g(u),\qquad \text{s.t.}\quad \begin{cases} u=u^s=u^A, \\ u_s=\bar{u}^s_s\wedge u_t=\bar{u}^t_t, \forall\{s,t\} \end{cases} \tag{11}$$

In ADMM notation of Section 4.1: $f(u^A,u^s,\bar{u}^s,\bar{u}^t)=\sum_s f_s(u^s_s)+f_A(u^A)+\sum_{(s,t)} f_{st}(\bar{u}^s_s,\bar{u}^t_t)$, and g holds the constraints for the consensus variable. The prox operators are easy to compute because they decouple spatially for u^s, u^A, and $(\bar{u}^s,\bar{u}^t)$. The edge variables $\bar{u}^s$ and $\bar{u}^t$ encode the presence of an edge between a source (s) and target (t) node and locally have as many copies as there exist edges (i.e., for a regular grid two copies for s and two for t at each interior pixel to account for edges in the x and y directions). The overall algorithm is given in Algorithm 1. The prox operators are given in Section 5.1. Section 5.2 shows how we can compute the dual energy to (11) using the variables of the ADMM solution scheme.

5.1. Prox operators

Some derivations yield the averaging operator

$$\mathrm{avg}_s(u^A,u^s,\bar{u}^s,\bar{u}^t)=\frac{u^s_s+u^A_s\sum\limits_{t:(s,t)\in\mathscr{E}}\bar{u}^s_s+\sum\limits_{t:(t,s)\in\mathscr{E}}\bar{u}^t_s}{2+|\{t:(s,t)\in\mathscr{E}\}|+|\{t:(t,s)\in\mathscr{E}\}|}, \tag{12}$$

and the prox operators

$$\mathrm{prox}_{\frac{1}{\sigma}f_s}(q_s)=\min\left\{1,\max\left\{0,q_s-\frac{1}{\sigma}\rho_s\right\}\right\}, \tag{13}$$

$$\mathrm{prox}_{\frac{1}{\sigma}f_A}(q_s)=\begin{cases} q_s+\frac{1}{|\mathscr{V}|}(A_l-A_q), & \text{if}\quad A_l>A_q, \\ q_s+\frac{1}{|\mathscr{V}|}(A_u-A_q), & \text{if}\quad A_u<A_q, \\ q_s, & \text{otherwise}, \end{cases} \tag{14}$$

$$\mathrm{prox}_{\frac{1}{\sigma}f_{st}(s,t)}(u,v)=\begin{cases} \left(u+\frac{c_{st}}{\sigma}\quad v-\frac{c_{st}}{\sigma}\right), & \text{if}\quad v-u>\frac{2c_{st}}{\sigma}, \\ \left(u-\frac{c_{st}}{\sigma}\quad v+\frac{c_{st}}{\sigma}\right), & \text{if}\quad u-v>\frac{2c_{st}}{\sigma}, \\ \left(\frac{u+v}{2}\quad \frac{u+v}{2}\right), & \text{otherwise}, \end{cases} \tag{15}$$

$$\mathrm{prox}_{\frac{1}{n\sigma}g_s}(u)=\begin{cases} 1, & \text{if } s\in\mathscr{S}, \\ 0, & \text{if } s\in\mathscr{T}, \\ \min(1,\max(0,u)), & \text{otherwise}. \end{cases} \tag{16}$$

Here, $|\mathscr{V}|$ denotes the number of pixels and $\mathscr{E}$ the edge set. See Section S.3 in the Supplementary material for the derivations.

5.2. Dual energy of the ADMM formulation

Computing the dual energy for ADMM using Fenchel duality (Rockafellar, 1997) yields

$$E^*(p^s,p^A,\bar{p}^s,\bar{p}^t)=-\sum_s f^*_s(p^s_s)-f^*_A(p^A)-\sum_{(s,t)} f^*_{st}(\bar{p}^s_s,\bar{p}^t_t)+\sum_{s\in\mathscr{S}} Q_s - \sum_{s\notin\mathscr{T}\cup\mathscr{S}}[-Q_s]_+$$

where

$$Q_s=p^s_s+p^A_s+\bar{p}^s_s+\bar{p}^t_s,$$
$$f^*_{st}(p_s,p_t)=\imath\{p_s+p_t=0\wedge|p_s|\leqslant c_{st}\}$$
$$f^*_s(p)=[p-\rho]_+,$$

$$f^*_A(p)=\begin{cases} A_u\max\limits_s\frac{p_s}{A_s}, & \text{if } \exists s:=p_s\geqslant 0, \\ A_l\max\limits_s\frac{p_s}{A_s}, & \text{otherwise}. \end{cases}$$

Here, $(p^s,p^A,\bar{p}^s,\bar{p}^t)=\sigma(z^s,z^A,\bar{z}^s,\bar{z}^t)$, i.e., the dual variables to compute E^* are the scaled dual variables of ADMM; $[x]_+=\max\{0,x\}$ is the ramp function. See Section S.4 in the Supplementary material for the derivations. Note that we need the dual energy of the original relaxed energy (10) and not of the ADMM energy for the branch and bound solution. We also need a feasible energy of the original MINLP (3) and not of the relaxed ADMM energy.[8] Section 6 therefore describes how to compute the appropriate primal and dual energies from the ADMM primal and dual energies.

Algorithm 1. ADMM for the area-constrained segmentation.

```
Data: $\sigma$, $A_l$, $A_u$, $c_{st}$; Initialized variable copies: $u=u^s=u^A=\overline{u}^s=\overline{u}^t$
Result: $u$
repeat
    Update local variables ;
        $(u^A)^{k+1} \leftarrow \mathrm{prox}_{\frac{1}{\sigma}f_A}(u^k+(z^A)^k)$; $(u^s)^{k+1} \leftarrow \mathrm{prox}_{\frac{1}{\sigma}f_s}(u^k+(z^s)^k)$
        $(\overline{u}^s,\overline{u}^t)^{k+1} \leftarrow \mathrm{prox}_{\frac{1}{\sigma}f_{st}}(u^k+(\overline{z}^s)^k, u^k+(\overline{z}^t)^k)$

    Averaging: $u^{k+1}_s \leftarrow \mathrm{avg}^{k+1}_s(u^s,u^A,\overline{u}^s,\overline{u}^t)-\mathrm{avg}^k_s(z^s,z^A,\overline{z}^s,\overline{z}^t)$ ;
    Clamping and enforcing seed points ;
    $u^{k+1}_s \leftarrow 1\ \forall\ s\in\mathcal{S}$; $u^{k+1}_s \leftarrow 0\ \forall\ s\in\mathcal{T}$; $u^{k+1}_s \leftarrow \min(1,\max(0,u^{k+1}_s))$
    Update dual variables ;
        $(z^s)^{k+1} \leftarrow (z^s)^k-(u^s)^{k+1}+u^{k+1}$; $(z^A)^{k+1} \leftarrow (z^A)^k-(u^A)^{k+1}+u^{k+1}$
        $(\overline{z}^s)^{k+1} \leftarrow (\overline{z}^s)^k-(\overline{u}^s)^{k+1}+u^{k+1}$; $(\overline{z}^t)^{k+1} \leftarrow (\overline{z}^t)^k-(\overline{u}^t)^{k+1}+u^{k+1}$
until convergence ;
```

[8] The original primal and dual energies and their corresponding ADMM primal and dual energies will be equivalent at convergence. However, for an efficient branch and bound solution we want to be able to test branch and bound termination criteria with respect to the original primal and dual energies *before* convergence.

6. Branch and bound

Building the search tree for a branch and bound solution of (3) requires a method to create subproblems (we use a standard binary division strategy on the b_k), a strategy to select subproblems for evaluation, a strategy to select variables for division, and a way to generate integer-energy estimates. We use a custom implementation of branch and bound where sub-problems are selected based on the lowest current relaxed energies. Branching variables are determined using pseudo-costs (Achterberg et al., 2005), and the lower bounds and integer-energy estimates are computed as described below. See Nemhauser and Wolsey (1988) for an in-depth discussion of branch and bound.

6.1. Obtaining lower and upper bounds

At convergence, the equality of the consensus variables is fulfilled, the bounds are satisfied and the area constraint holds. Therefore primal and dual energies of the relaxed ADMM problem will be finite upon convergence. To terminate solution branches that cannot lead to an optimal solution early, finite-valued dual energy estimates are needed *before* convergence for the dual energy to obtain a lower bound. Further, a feasible integer-valued (or essentially binary) solution is needed to obtain an upper bound. A finite-valued ADMM energy estimate is needed to evaluate the convergence of a current relaxed ADMM solution candidate based on its duality gap (i.e., the difference between primal and dual energy). Section 6.1.1 discusses how to obtain a finite-valued relaxed energy from an ADMM relaxed solution *before or at* convergence. Section 6.1.2 discusses how to obtain a finite-valued dual energy for the relaxed problem from the variables of the relaxed ADMM solution method. Finally, Section 6.1.3 discusses how integer-feasible solutions can be obtained from relaxed solutions by thresholding. Fig. 3 illustrates the connection between the different primal and dual energies.

6.1.1. Estimate of the relaxed energy

A current finite-valued energy estimate, which is an upper bound of the relaxed energy at convergence, can be obtained by projecting the current consensus variable u back onto the constraint set (so that it fulfills the area and bound constraints, $u_s \in [0,1]$). This requires solving the projection

$$u^* = \underset{q}{\operatorname{argmin}} E(q) = \min_q \frac{1}{2}\sum_s (q_s - u_s)^2,$$
$$A_l \leqslant \sum_s q_s \leqslant A_u, \quad q_s \in [0,1],$$

which, in order to project to area A, requires finding a Lagrangian multiplier λ_e s.t.

$$\sum_s u_s^* = \sum_s \min\{1, \lambda_e + u_s\} = A,$$

The optimal λ_e can be found by computing successive relaxed solutions

$$\sum_s \lambda_e^r + u_s = A \to \lambda_e^r = \frac{1}{|\mathcal{V}|}\left(A - \sum_s u_s\right).$$

Since $0 \leqslant \lambda_e^r \leqslant \lambda_e$ the optimization problem can be broken into subpieces and solved efficiently by first sorting the values u (if the current area is smaller than A – a similar reasoning hold in the reverse case).

If there is no relaxed feasible solution, then no integer feasible solution can exist. A feasible relaxed solution can be computed if the area constraint projection steps are feasible, which will be the case if

$$\sum_{s\in\mathcal{T}} A_s \leqslant -A_l + \sum_s A_s, \sum_{s\in\mathcal{S}} A_s \leqslant A_u.$$

Hence, for a given set of foreground/background seedpoints in the branch and bound solver a solution of the relaxed problem only needs to be sought if these conditions hold, otherwise the dual energy is set to $-\infty$ and the energy to ∞.

6.1.2. Estimate of the relaxed dual energy

A finite lower bound for the relaxed energy can be obtained by adjusting the ADMM dual variables for the terms which would otherwise lead to a $-\infty$ estimate before convergence. We therefore need to find a dual variable pair $(\tilde{p}_s, \tilde{p}_t)$ that is as close as possible to the current estimate (p_s, p_t) while fulfilling the edge variable constraint. Such a pair can by computed by the projection

$$\Pi(p_s, p_t) = \begin{cases} (c, -c), & \text{for } p_s - p_t > 2c_{st}, \\ (-c, c), & \text{for } p_s - p_t < -2c_{st}, \\ \left(\frac{p_s - p_t}{2}, \frac{p_t - p_s}{2}\right), & \text{otherwise}. \end{cases}$$

See Section S.5 in the Supplementary material for the derivation.

6.1.3. Estimate of an integer-feasible solution

To allow termination of suboptimal branches, a good estimate for an integer-feasible (or essentially binary) solution is desirable early during branch and bound. Assume a feasible relaxed solution is given. By thresholding the relaxed u at $\theta \in (0,1)$, we can obtain in a finite number of thresholding steps (determined by bisection) an integer feasible solution, or show that such a thresholded solution does not exist (in which case the estimate is set to ∞). In practice, we terminate the search for a solution candidate after a fixed number of thresholding steps. Terminating the search without finding an integer-feasible solution will not affect the overall branch and bound solution. We will only not be able to produce a good integer-valued energy estimate from this solution, which in turn may effect early termination of search branches and may consequentially result in larger branch and bound search trees.

The relaxed solution candidate may already be essentially binary and fulfill the area constraints given an appropriate selection of seed regions. In general, an optimal essentially binary u is guaranteed to exist if a sufficient number of selection areas $\mathcal{B}_k$ exist, and only an upper or lower bound needs to be enforced. For simultaneous lower and upper bounds the branch and bound algorithm will either find the best integer (and therefore one of the essentially binary) solutions, or will prove that no such solution exists. Non-existence is a pathological case, which is unlikely in practice. We never observed such a case in our experiments, but it is possible to construct toy examples which exhibit this problem. When the area-constrained segmentation formulation requires the solution to be binary for the selection areas, only compliant thresholded solutions will be feasible and hence finite.

7. Results

We tested the area-constrained segmentation method for the segmentation of synaptic vesicles and for double membrane vesicles in epithelial cells infected with SARS-coronavirus (Knoops et al., 2008). All images are slices of electron tomography images. Images for the epithelial cells were obtained from the cell centered database (CCDB) of the National Center for Microscopy and Imaging Research (NCMIR – http://ccdb.ucsd.edu). The images for the synaptic vesicles were approximately at a resolution of (1.0 nm by 1.0 nm)/pixel and for the SARS-coronavirus at (1.2 nm by 1.2 nm)/pixel.

These examples were chosen to demonstrate the properties of our developed area-constrained segmentation method, because segmentations for these electron tomography images are known to be challenging. For example, for the synaptic vesicle segmentation task the vesicle wall is not directly visible in the electron tomography image. Instead it needs to be inferred from the location of proteins (which appear dark) penetrating the vesicle wall which results in a "noisy" appearance of the vesicle wall. Further, a large number of vesicles can be found in one image. Vesicles are closely packed in some areas, which even experts can have difficulty outlining precisely. In our experiments a user was asked to place individual seed points at the center of the objects to be segmented. The selection areas were obtained by eroding a quick-shift segmentation of the complete image. The selection region closest to the user-placed seed point was set as a foreground seed, and the selection areas at the boundaries of a 100 × 100 pixel box centered at the seed point were set as background seeds. This box size was chosen to be sufficiently large to guarantee that the desired objects are contained within it. We used image intensities as edge terms (c_{st}) and set $\rho = 0$. We set $\gamma = 1$ for all ADMM experiments. Setting the selection areas at the boundaries as background seeds is meaningful for our experiment because the object will be, by construction, at the center of the image. However, this is not essential. The boundaries could be included into the optimization, albeit at the price of higher computational cost.

Given the selected segmentation area and the selection regions we compared the following methods for the vesicle datasets:

(1) *UC*: Area-constrained segmentation with a lower bound of 0. This unconstrained case corresponds to a classical graph-cut segmentation with seed points.
(2) *LB/UB/LBUB*: Area-constrained segmentation enforcing upper and lower bounds on the segmentation area separately and jointly.
(3) *BNC*: Biased normalized cut (Maji et al., 2011) using foreground seeds.
(4) *NC*: Normalized cut (Shi and Malik, 2000). No seed regions are supported by this algorithm and hence none were used.
(5) *WS*: Seeded watershed segmentation (Vincent and Soille, 1991) using foreground and background seeds.
(6) *RW*: Random walker segmentation using foreground and background seeds.

To allow comparisons between the algorithms (i) we used the same seed regions for all algorithms, (ii) we used the same edge weights g for all algorithms except for the random walker algorithm, and (iii) determined the best possible thresholds for NC and BNC by searching which threshold results in the best value for the normalized cut. For the random walker segmentation algorithm, we used two sets of edge weights, g, because this is a segmentation model which will not exhibit discontinuities at the putative segmentation boundary and hence treats edge-weights differently than all the other tested models. We report random-walker results using the same edge-weights as the other algorithms (RW) as well as using the more appropriate default settings for random walker segmentation (d-RW).

7.1. Synaptic vesicles

We used a lower area bound of 800 and an upper area bound of 2000 (areas in pixels) with a low number of large selection areas and a larger number of small selection areas. Synaptic vesicles observed in our specimen are estimated to be about 40 nm in diameter. Since we are dealing with slices of a three-dimensional structure, we expect the actual observed diameters to be smaller than this. An area between 800 and 2000 pixels corresponds approximately to diameters between 30 and 50 nm if a perfect circular shape is assumed.

Table 1
Dice similarity coefficients for vesicle segmentation with small (↓) and larger (↑) number of selection areas, $\mathcal{B}_k$. Unconstrained (UC), lower (LB), and lower and upper bound (LBUB) constrained segmentations. Biased normalized cut (BNC), normalized cut (NC), seeded watershed (WS), random walker (RW) and random walker with default settings (d-RW). Bold: best results. Italicized results do not have significantly different mean in comparison to the best method.

	Type	Mean	Median	Std
↑	UC	0.72	0.92	0.32
↓	UC	0.86	0.94	0.22
↑	LB	*0.92*	*0.94*	*0.05*
↓	LB	**0.93**	**0.95**	**0.04**
↑	LBUB	*0.91*	*0.92*	*0.05*
↓	LBUB	*0.91*	*0.91*	*0.04*
↑	BNC	0.79	0.85	0.16
↓	BNC	0.80	0.88	0.17
	NC	0.15	0.18	0.11
↑	WS	0.85	0.91	0.13
↓	WS	0.84	0.90	0.18
↑	RW	0.86	0.86	0.05
↓	RW	0.84	0.84	0.06
↑	d-RW	0.89	0.91	0.04
↓	d-RW	0.86	0.90	0.12

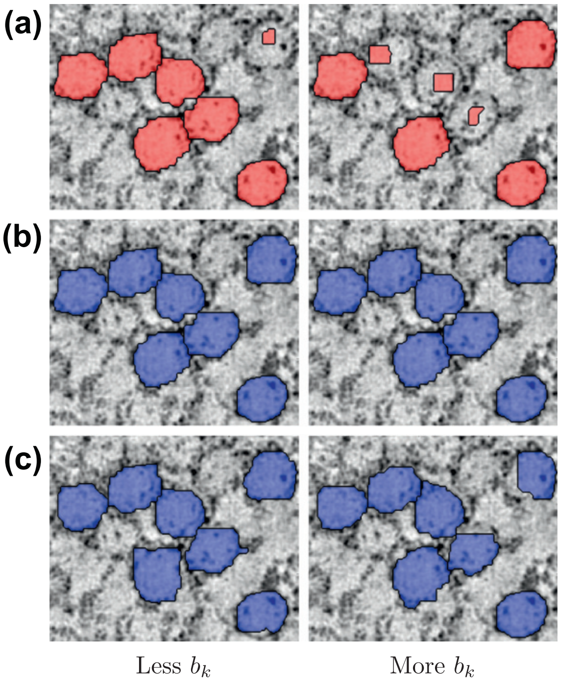

Fig. 4. Subset of vesicle segmentation results; (a) unconstrained, (b) lower bound, and (c) upper and lower bound. The constrained results are better on average, because they avoid mis-segmentation due to shrinking bias. See Figure S2 in the supplementary material for statistical results.

Comparing to an expert segmentation of 38 vesicles many of the vesicles were segmented correctly by both the area-constrained segmentations and by the unconstrained segmentations. However, in the unconstrained case, a substantial number of vesicles was under-segmented (returning only the seed point). In contrast, the area-constrained segmentations successfully segmented these cases and were able to achieve a segmentation result very close to the gold standard regardless of the selection areas. Note that using only a lower bound gives the best results in this example because it retains maximal flexibility for the registration boundary. When upper and lower bounds are enforced, the segmentation needs to conform to the selection areas. Though the area-constrained results are not statistically significantly different with respect to each other, they are statistically significantly better than

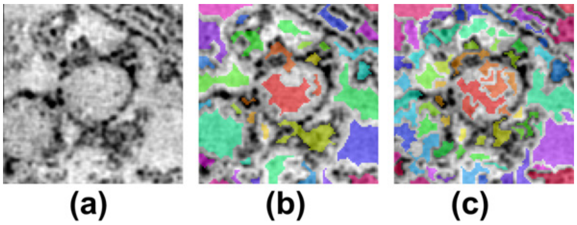

Fig. 5. Example selection areas for a vesicle (a). Few selection areas (less b_k) (b) and many selection areas (more b_k) (c).

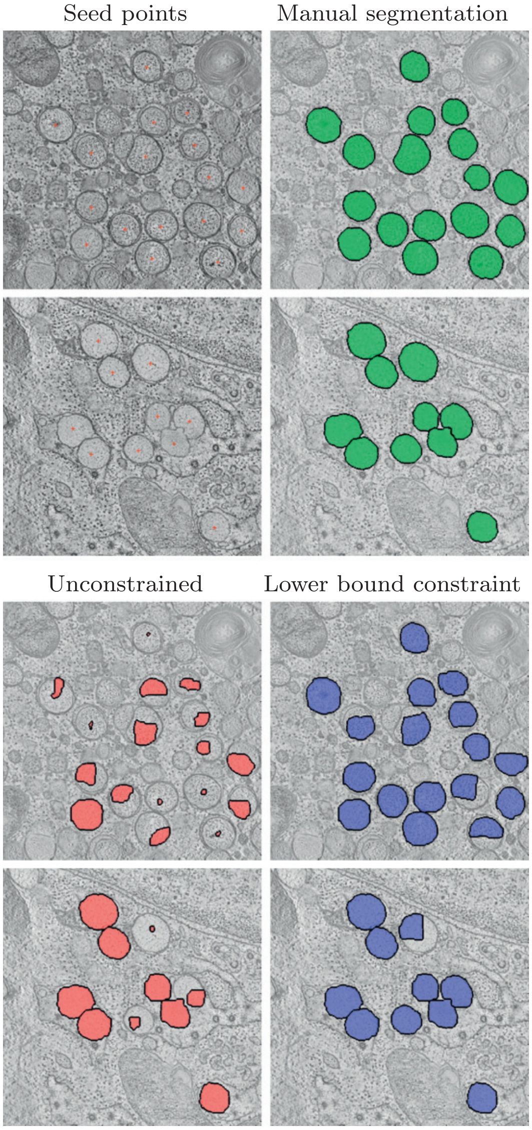

Fig. 6. Segmentation results for a slice of the SARS 6021 (top) and of the SARS 6022 (bottom) electron tomography image. Seed points were placed manually with a single mouse click. Without using an area constraint (red) only few of the vesicles are accurately segmented and in the majority of cases the segmentations are too small indicating that a short boundary length was favored over a segmentation at the desired location of the cell wall. Adding a lower bound on the area (blue) greatly improves the segmentation results. Though a bias for short segmentation boundaries is still present, most of the vesicles are segmented accurately. Since the SARS 6022 image appears less noisy many of the vesicles are also segmented correctly without using a segmentation area constraint. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

all the other tested segmentation methods. Biased normalized cut, watershed segmentation, and the unconstrained segmentation method showed reasonable overall results, but suffered from severe outliers. The standard normalized cut segmentation fails entirely on these datasets because it cannot identify the object of interest since the data is noisy and no seed regions can be used. Random walker segmentation overall performed well, did not show any strong outliers, but performed overall worse than the area-constrained segmentation method. Table 1 shows summary measures for the Dice similarity coefficients for the experiments. Fig. 4 shows the segmentation results for unconstrained and area-constrained segmentation for a subset of an image. Fig. 5 illustrates the different selection areas for a specific example.

An overview of corresponding seed points, the gold standard manual segmentation as well as results for the area-constrained and the unconstrained segmentations is shown in the Supplementary material in Fig. S1. Boxplots for all the segmentation methods and a comparison of their mean performance (as measured by Dice) are shown in Fig. S2.

Adjacent vesicles may overlap because they are treated independently. In practice, overlaps were not observed for vesicle segmentation results of the area-constrained segmentation approach. This is a property of the data combined with the segmentation approach (which encourages short boundaries). In general, overlapping segmentations are possible and present ambiguities in the segmentation. Such ambiguities could be avoided by moving to a multi-label segmentation formulation.

In cases where the unconstrained segmentations resulted in a correct segmentation, the branch and bound search terminated quickly for the area-constrained methods. Most of the computation time was spent to correct the more challenging cases. Between 28 and 90 selection regions were used. Run-times were moderate: on average less than a minute per vesicle with a large number, and four seconds with a small number of selection areas on a single-core CPU implementation. The algorithm can easily be parallelized and implemented on a GPU (with an expected speed-up by at least an order of magnitude).

7.2. SARS: Double membrane vesicles

We used a lower area bound of 2000 (area in pixels) with a low number of large selection areas and repeated a subset of the experiments for the synaptic vesicle segmentation. For a perfect circle, this area would correspond to a diameter of about 60 nm at the

Table 2
Dice similarity coefficients for the SARS 6021 and SARS 6022 images. Unconstrained (UC), lower (LB), and lower and upper bound (LBUB). Biased normalized cut (BNC), normalized cut (NC), seeded watershed (WS), random walker (RW) and random walker with default settings (d-RW). Bold: best results. Italicized results do not have significantly different mean in comparison to the best method. For these images with a clear vesicle boundary watershed segmentation, maxflow with a lower bound, and random walker segmentation work well.

	Type	Mean	Median	Std
SARS 6021	UC	0.41	0.45	0.29
	LB	*0.90*	*0.96*	*0.10*
	BNC	0.72	0.71	0.10
	NC	0.34	0.35	0.12
	WS	**0.96**	**0.97**	**0.02**
	RW	0.68	0.79	0.24
	d-RW	0.72	0.84	0.26
SARS 6022	UC	0.76	0.97	0.36
	LB	*0.94*	*0.97*	*0.11*
	BNC	0.79	0.81	0.11
	NC	0.38	0.35	0.15
	WS	**0.95**	**0.98**	**0.04**
	RW	0.79	0.84	0.18
	d-RW	*0.93*	*0.95*	*0.04*

given resolution. Since the double membrane vesicles have diameters of about 200–300 nm (Knoops et al., 2008), this is a conservative lower bound on the area. Similar conclusions as for the synaptic vesicle experiment apply. However, since the images for the double membrane vesicles are significantly less noisy than the images for the synaptic vesicles watershed segmentation, random walker segmentation, as well as the area-constrained segmentation method, work well. The area-constrained segmentation method matches the performance of the best segmentation method (seeded watershed) for both SARS images. Generally, the segmentation using a lower bound on the segmentation area performed better than the unconstrained segmentation. Fig. 6 shows overviews of the resulting segmentations for SARS 6021 and 6022, respectively, for the unconstrained and the area-constrained segmentations. Table 2 gives an overview of the obtained Dice similarity coefficients. Fig. S3 in the Supplementary material shows boxplots for the segmentation results for all the tested methods and statistical significance levels between the methods with respect to mean Dice performance.

8. Conclusion and future work

We developed a new method for image segmentation with area constraints. The method readily extends to higher dimensions using higher-dimensional generalizations of the selection regions. The proposed method relies on the solution of a mixed integer nonlinear program, which is solved using branch and bound. To reduce computational effort in solving the area-constrained segmentation, we proposed to use selection variables based on eroded super-pixels. This allows computation of the segmentations for practical problems. The behavior of the method was demonstrated for segmentations of vesicles from slices of electron tomography images. When area-constraints were available, statistically significant increases in segmentation quality were obtainable even in challenging cases. In particular, due to the global optimality properties of the algorithm, it performs well for noisy data.

Future directions include improvements on the optimization method: e.g., should computations be performed directly on super-pixels? Further, the sensitivity of the obtained results on the type and size of the superpixels should be explored. Another interesting direction would be to vary the area constraints to define an area-based scale space, which would allow us to automatically extract coherent sub-structures at different size levels from the images. Extensions to multiple objects or the inclusion of topological constraints (e.g., to enforce one connected component) are other possible research directions.

Acknowledgements

This publication was made possible by Grant No. 5P41EB002025-28 from NIBIB/NIH. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NIBIB or NIH.

Appendix A. Supplementary material

Supplementary material associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.media.2012.09.002.

References

Achterberg, T., Koch, T., Martin, A., 2005. Branching rules revisited. Operations Research Letters 33, 42–54.

Appleton, B., Talbot, H., 2006. Globally minimal surfaces by continuous maximal flows. IEEE Transactions on Pattern Analysis and Machine Intelligence, 106–118.

Ayed, I., Li, S., Islam, A., Garvin, G., Chhem, R., 2008. Area prior constrained level set evolution for medical image segmentation. In: Proceedings of SPIE, pp. 691402.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., 2010. Distributed optimization and statistical learning via the alternating direction method of multipliers. Machine Learning 3, 1–123.

Boykov, Y., Funka-Lea, G., 2006. Graph cuts and efficient nd image segmentation. International Journal of Computer Vision 70, 109–131.

Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J.P., Osher, S., 2007. Fast global minimization of the active contour/snake model. Journal of Mathematical Imaging and Vision 28, 151–167.

Chan, T., Vese, L., 2001. Active contours without edges. IEEE Transactions on Image Processing 10, 266–277.

Comaniciu, D., Meer, P., 2002. Mean shift: a robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 603–619.

Combettes, P., Pesquet, J., 2010. Proximal splitting methods in signal processing. Fixed-Point Algorithms for Inverse Problems in Science and Engineering.

Cootes, T., Edwards, G., Taylor, C., 2001. Active appearance models. IEEE Transactions on Pattern Analysis and Machine Intelligence 23, 681–685.

Dahl, G., Flatberg, T., 2007. An integer programming approach to image segmentation and reconstruction problems. Geometric Modelling, Numerical Simulation, and Optimization, 475–496.

Eriksson, A., Olsson, C., Kahl, F., 2011. Normalized cuts revisited: a reformulation for segmentation with linear grouping constraints. Journal of Mathematical Imaging and Vision 39, 45–61.

Falkner, J., Rendl, F., Wolkowicz, H., 1994. A computational study of graph partitioning. Mathematical Programming 66, 211–239.

Ford, L.R., Fulkerson, D.R., 1956. Maximal flow through a network. Canadian Journal of Mathematics 8, 399–404.

Grady, L., Schwartz, E., 2006. Isoperimetric graph partitioning for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 28, 469–475.

Hager, W., Phan, D., Zhang, H., 2009. An exact algorithm for graph partitioning. Arxiv preprint arXiv:0912.1664.

Hijazi, H., Bonami, P., Cornuéjols, G., Ouorou, A., 2009. Mixed integer nonlinear programs featuring on/off constraints: convex analysis and applications. Working paper.

Jain, A., Murty, M., Flynn, P., 1999. Data clustering: a review. ACM Computing Surveys (CSUR) 31, 264–323.

Ji, X., 2004. Graph Partition Problems with Minimum Size Constraints. Ph.D. thesis. Rensselaer Polytechnic Institute.

Kernighan, B., Lin, S., 1970. An efficient heuristic procedure for partitioning graphs. Bell System Technical Journal 49, 291–307.

Keuchel, J., Schnörr, C., Schellewald, C., Cremers, D., 2003. Binary partitioning, perceptual grouping, and restoration with semidefinite programming. IEEE Transactions on Pattern Analysis and Machine Intelligence 25, 1364–1379.

Knoops, K., Kikkert, M., Worm, S., Zevenhoven-Dobbe, J., van der Meer, Y., Koster, A.J., Mommaas, A.M., Snijder, E.J., 2008. SARS-coronavirus replication is supported by a reticulovesicular network of modified endoplasmic reticulum. PLoS Biology 6, e226.

Kolmogorov, V., Zabin, R., 2004. What energy functions can be minimized via graph cuts? IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 147–159.

Kropotov, D., Laptev, D., Osokin, A., Vetrov, D., 2010. Variational segmentation algorithms with label frequency constraints. Pattern Recognition and Image Analysis 20, 324–334.

Li, S.Z., 2009. Markov Random Field Modeling in Image Analysis. Springer.

Lim, Y., Jung, K., Kohli, P., 2010. Energy minimization under constraints on label counts. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 535–551.

Lisser, A., Rendl, F., 2003. Graph partitioning using linear and semidefinite programming. Mathematical Programming 95, 91–101.

Maji, S., Vishnoi, N., Malik, J., 2011. Biased normalized cuts. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE. pp. 2057–2064.

Nemhauser, G.L., Wolsey, L.A., 1988. Integer and Combinatorial Optimization. Wiley.

Nocedal, J., Wright, S.J., 2006. Numerical Optimization. Springer.

Olsson, C., Eriksson, A.P., Kahl, F., 2008. Improved spectral relaxation methods for binary quadratic optimization problems. Computer Vision and Image Understanding, 3–13.

Pham, D., Xu, C., Prince, J., 2000. Current methods in medical image segmentation. Annual review of biomedical engineering 2, 315–337.

Pizer, S., Fletcher, P., Joshi, S., Thall, A., Chen, J., Fridman, Y., Fritsch, D., Gash, A., Glotzer, J., Jiroutek, M., et al., 2003. Deformable m-reps for 3d medical image segmentation. International Journal of Computer Vision 55, 85–106.

Reinbacher, C., Pock, T., Bauer, C., Bischof, H., 2010. Variational segmentation of elongated volumetric structures. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pp. 3177–3184.

Rockafellar, R., 1997. Convex Analysis. Princeton Univ Pr..

Rohlfing, T., Brandt, R., Menzel, R., Russakoff, D., Maurer, C., 2005. Quo vadis, atlas-based segmentation? Handbook of Biomedical Image Analysis, 435–486.

Sapiro, G., 2001. Geometric Partial Differential Equations and Image Analysis. Cambridge.

Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 888–905.

Sonka, M., Hlavac, V., Boyle, R., 2008. Image Processing, Analysis, and Machine Vision. Thomson.

Vedaldi, A., Soatto, S., 2008. Quick shift and kernel methods for mode seeking. ECCV, 705–718.

Vincent, L., Soille, P., 1991. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Transactions on Pattern Analysis and Machine Intelligence 13, 583–598.

Zach, C., Niethammer, M., Frahm, J.M., 2009a. Continuous maximal flows and Wulff shapes: application to MRFs. In: Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1911–1918.

Zach, C., Shan, L., Frahm, J.M., Niethammer, M., 2009b. Globally optimal Finsler active contours. In: DAGM-Symposium, pp. 552–561.