This CVPR 2020 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Sparse Layered Graphs for Multi-Object Segmentation

Niels Jeppesen, Anders N. Christensen, Vedrana A. Dahl, Anders B. Dahl
Department of Applied Mathematics and Computer Science, Technical University of Denmark
Richard Petersens Plads, Building 324, 2800 Kgs. Lyngby, Denmark
niejep,anym,vand,abda@dtu.dk

Abstract

We introduce the novel concept of a Sparse Layered Graph (SLG) for s-t graph cut segmentation of image data. The concept is based on the widely used Ishikawa layered technique for multi-object segmentation, which allows explicit object interactions, such as containment and exclusion with margins. However, the spatial complexity of the Ishikawa technique limits its use for many segmentation problems. To solve this issue, we formulate a general method for adding containment and exclusion interaction constraints to layered graphs. Given some prior knowledge, we can create a SLG, which is often orders of magnitude smaller than traditional Ishikawa graphs, with identical segmentation results. This allows us to solve many problems that could previously not be solved using general graph cut algorithms. We then propose three algorithms for further reducing the spatial complexity of SLGs, by using ordered multi-column graphs. In our experiments, we show that SLGs, and in particular ordered multi-column SLGs, can produce high-quality segmentation results using extremely simple data terms. We also show the scalability of ordered multi-column SLGs, by segmenting a high-resolution volume with several hundred interacting objects.

1. Introduction

Most image segmentation research using graph cuts is demonstrated on problems with less than ten labels. This is enough for high-level segmentation tasks like organ, brain, and bone segmentation. However, many segmentation tasks, such as microscopy imaging in medicine and materials science, involve hundreds or more objects. Segmentation tasks with this number of labels have previously been difficult, or even impossible to solve using *s-t* graph cuts. With our method for constructing graphs, well-known graph cut algorithms can efficiently solve segmentation tasks with hundreds of labels.

Computational speed is essential for the practical use of segmentation. Much of the success of graph cuts is owed to the Boykov-Kolmogorov (BK) implementation of the Ford-Fulkerson maxflow/mincut algorithm, which performs well for many image-related optimization problems and gives a globally optimal solution for submodular problems [1]. More recently, the Incremental Breadth-First Search (IBFS) algorithm by Goldberg *et al*. [6] has shown even better performance and run-time guarantees. Another way to speed up the computations is to use a parallel algorithm [4, 18]. However, to our knowledge, these algorithms only work on regular grid-based graphs.

By definition, *s-t* graph cuts provide a binary labeling. For multi-label segmentation with graph cuts, one option is to use the iterative α-expansion method [2]. However, it often gets stuck in weak local minima. Another common approach is to use the Ishikawa layered graph construction [10], where each layer corresponds to one label. Using this technique, it is possible to solve multi-label problems, while enforcing label interaction constraints, such as *containment* and *two-label exclusion* [5]. These interaction constraints are often necessary to ensure that an object is inside another object or that objects do not overlap. However, because the exclusion term is non-submodular, the approach of [5] does not work for more than two exclusive objects.

To enable multi-object exclusion, one approach is to use the QPBO algorithm [11, 16], which can incorporate non-submodular terms, at the cost of completeness. The algorithm guarantees partial optimality, but may not find a complete solution, *i.e*. there may be unlabelled nodes. A higher-level alternative is the Path-Moves algorithm (HINTS) [8], which is also able to incorporate non-submodular terms. Unlike QPBO, it is an iterative algorithm that always provides a complete labeling and has been shown to find good solutions, although they are not guaranteed to be optimal.

The number of objects that can be segmented using the Ishikawa technique is in practice limited by the size of the layered graph. If Ω is a set of nodes, usually corresponding to the pixels of an image, and $\mathcal{L}$ is the set of labels/objects, then the spatial complexity of the nodes in the layered graph is $O(|\Omega||\mathcal{L}|)$. However, the number of objects is not the

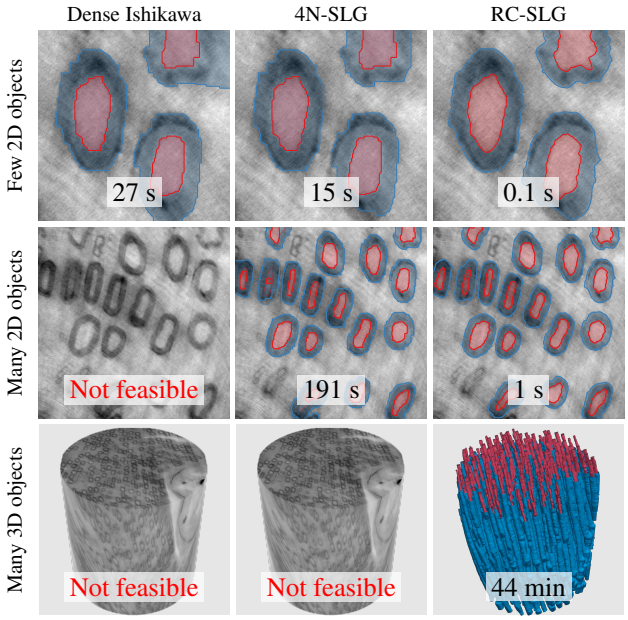


Figure 1. Time spent solving three differently sized problems using *s-t* graph cut with different graph constructions.

only important factor. When applying interactions between the objects, it is possible, and often useful, to specify a minimum margin. For a N-neighborhood regular grid structure [5], the spatial complexity of the interaction terms, and corresponding graph edges, depends polynomially on the size of the interaction margins. In practice, this means that the standard N-neighborhood graph structure is only useful for segmenting a very limited number of objects, with small interaction margins.

An alternative to the N-neighborhood structure is the ordered multi-column graph [19]. This approach is very useful for surface detection, but may also be used for object segmentation, when combined with resampling [13]. The ordered multi-column structure makes it possible to impose a certain geometry to the solution. Which geometry is imposed, depends on how the data is resampled.

While [13] describes how to handle containment interactions for ordered multi-column graphs, object exclusion is not described. Also, they assume contained objects are sampled identically, which means that objects cannot be sampled at different resolutions or have imposed different geometries. Unlike the approach used by [5], the spatial complexity of the ordered multi-column structure by [13] does not depend on the size of the interaction margins. As a result, appropriate margins can be chosen freely, without worrying about the size of the layered graph.

To overcome the complexity issue of layered Ishikawa graph structure, we introduce the concept of a Sparse Layered Graph (SLG), along with a general method for adding object interactions to layered graphs. Using this method, we construct a N-neighborhood SLG (N-SLG), which is significantly smaller than the corresponding dense Ishiskawa graph. Then, to further reduce the size of the graph, we construct an ordered multi-column SLG (C-SLG), based on the method by Li *et al*. [13]. Like the approach by [13], we need some prior knowledge in the form of approximate size and position of the objects. We propose three algorithms for incorporating interaction constraints in C-SLGs, with very few terms. Experimentally, we show that SLGs can be used to reduce segmentation time and accurately solve segmentation problems with hundreds of objects (see Fig. 1). Such tasks cannot be solved with the traditional dense Ishikawa layered graphs, used by [5, 8], due to the size of the graph.

We compare the segmentation accuracy, time and graph size of different configurations of an N-SLG, C-SLG, and the method by Li *et al*. [13], on an instance segmentation task. Our experiments show the advantages of using SLGs, and in particular C-SLG, over the traditional layered graph. They also show that ordered multi-column graphs can provide accurate segmentations, even with extremely simple models. We then demonstrate the scalability of C-SLGs by segmenting a large volume with several hundred interacting objects using a single graph cut.

Our method uses the QPBO algorithm. This means that we cannot guarantee completeness, but only partial optimality [16]. Thus, we may not be able to label all nodes if the model contains non-submodular terms. Many unlabelled nodes will result in a poor segmentation, so it is critical for the accuracy of our method that unlabelled nodes are rare. To investigate the frequency of unlabelled nodes, we segment a large set of images using SLGs. The results show that accurate segmentation is possible with a simple model, even on a varied data set. Furthermore, unlabelled nodes are rare and have little impact on the segmentation.

Along with this paper, we release an open-source Python package for constructing and solving SLGs (see Section 3).

2. Multi-object segmentation

We consider an image segmentation problem, with several objects, which may be interacting. We use the term *object*, *label* and *layer* interchangeably, depending on the context – whether we refer to the content of an image, outcome of a segmentation, or construction of a graph.

A common way to solve image segmentation problems is by minimizing an energy function of the form

$$E(\mathbf{x}) = \sum_{p \in \mathcal{V}} \theta_p(x_p) + \sum_{p,q \in \mathcal{V}} \theta_{pq}(x_p, x_q) \,. \qquad (1)$$

For images, the node-set $\mathcal{V}$ usually corresponds to image pixels, where the segmentation can be obtained as a pixel labeling with labels $x_p \in \{0, 1\}$. Unary energy terms, θ_p, usually encode a data term, while pairwise energies, θ_{pq},

encode interactions between pixels, as well as interactions between labels.

If the energy function E is submodular, meaning that all pairwise terms, θ_{pq}, between nodes p and q satisfy

$$\theta_{pq}(0,0) + \theta_{pq}(1,1) \leq \theta_{pq}(0,1) + \theta_{pq}(1,0) \qquad (2)$$

then it is possible to use the *s-t* graph cut to find the global energy minimum in polynomial time [12].

In many segmentation models, pairwise interactions are symmetric ($\theta_{pq}(0,1) = \theta_{pq}(1,0)$), finite ($\theta_{pq}(0,1) < \infty$), and submodular. Such interaction will encourage smoothness and will be balanced by the unary terms. On the other hand, asymmetric and infinite terms are useful to impose a certain geometry or object interaction.

2.1. Sparse layered graphs

In the dense Ishikawa layered structure, used by [5, 8, 13], identical graph layers are created for each object, as shown in Fig. 2b. This often results in a large number of irrelevant nodes, as only a fraction of the nodes of a given layer is usually inside or near the object. In an SLG, we remove irrelevant nodes from the layers to reduce the size of the graph. This creates what we call sparse layers, which are layers where not all pixels are represented by nodes. To determine which nodes are relevant for each layer, we use information, which is often available or can be computed in some way, such as approximate position and size of objects.

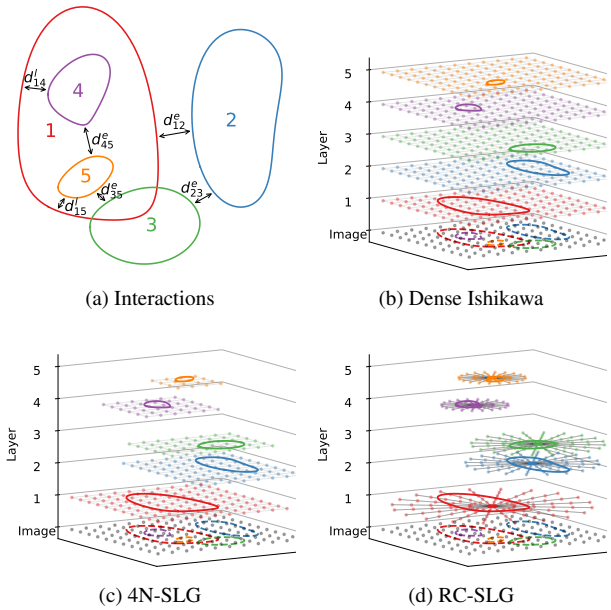
We can create a simple N-SLG, similar to the one in Fig. 2c, by first constructing a dense Ishikawa graph. Then, we crop each layer to remove nodes that are known to be outside the layer object. This way, all nodes still correspond to a single pixel, but not all pixels are represented by nodes. This approach preserves the pixel neighborhood structure between nodes in different layers used by previous methods [5, 8].

A common way to reduce the number of nodes in graph segmentation problems is to downsample data before creating the graph. However, this approach will also reduce the resolution of the segmentation. If we are segmenting interacting objects of varying sizes, it could be favorable to downsample large objects, while keeping small objects at a higher resolution. We could choose to only downsample the data for layers with large objects, but this breaks the inter-layer neighborhood structure.

It turns out that resampling can be used, not just for varying layer resolution, but also to enforce shape priors [13]. However, we need a way of adding object interactions between differently sampled layers.

2.2. Geometric interactions

We focus on two important geometric interactions:



(a) Interactions (b) Dense Ishikawa

(c) 4N-SLG (d) RC-SLG

Figure 2. Five interacting objects, with different minimum containment margins, d^l, and exclusion margins, d^e, are shown in (a). Interactions are defined independently between pairs of objects. With the Ishikawa technique, a layer is created for each object as shown in (b). The gray dots at the bottom indicate nodes that would correspond to pixels in the image. The colored dots are the layer nodes. In a 4N-SLG, shown in (c), we keep the neighborhood graph structure, but sample only a subset of the original pixels. Alternatively, we can sample radially and create a column graph (RC-SLG), as shown in (d).

- **Containment.** One object must be inside another object, with the possibility of specifying a minimum margin between the objects (d^l_{IJ} in Fig. 2a).
- **Exclusion.** Two objects cannot overlap at any point, with the possibility of specifying a minimum distance (d^e_{IJ} in Fig. 2a).

We propose a general way of applying containment and exclusion terms between non-identical graph layers, such as the ones shown in Fig. 2d. All we require, is that we can calculate a distance between nodes in interacting layers.

In the following, we will consider objects $I, J \subset \mathbb{R}^2$ and a set of graph nodes $\mathcal{V} = \mathcal{V}_I \cup \mathcal{V}_J$, where $\mathcal{V}_I$ and $\mathcal{V}_J$ are graph layers for objects I and J. We will write i when we refer to nodes from $\mathcal{V}_I$, and similarly j for $\mathcal{V}_J$. The spatial position of a node is given by a mapping $p : \mathcal{V} \to \mathbb{R}^2$, such that $p(i)$ denotes the position of node i.

Containment, *e.g.* object I contains object J, is simple to enforce by adding an energy term for all pairs of nodes $i \in \mathcal{V}_I$ and $j \in \mathcal{V}_J$

$$\theta_{ij}(0,1) = \infty \ , \ \|p(i) - p(j)\| \leq d^l_{IJ} \ . \qquad (3)$$

Here, d^l_{IJ} is the minimum margin between the outer object, I, and inner object, J. The energy term $\theta_{ij}(0,1) = \infty$ is

submodular and can therefore be translated to a single edge in a graph. Thus, we can solve problems with containment constraints using a standard maxflow solver, such as [1].

Exclusion, *e.g.* objects I and J are exclusive, can be enforced by adding energy terms

$$\theta_{ij}(1,1) = \infty \ , \ \|p(i) - p(j)\| \leq d^e_{IJ}, \tag{4}$$

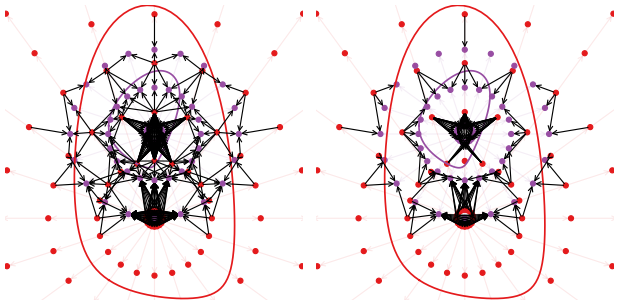
for all pairs of nodes $i \in \mathcal{V}_I$ and $j \in \mathcal{V}_J$, where d^e_{IJ} is the minimum margin between I and J. Because $\theta_{ij}(1,1) = \infty$ is a non-submodular energy term, it cannot be expressed as a single edge in a graph. To overcome this, we use the QPBO algorithm.

Since Eq. (3) and (4) can be applied as long as we can calculate a distance between nodes in interacting layers, we can now apply object interactions between objects sampled in any way. It is possible to sample different object using entirely different sampling schemes and have different graph structures in different layers. However, doing so will of course impact the segmentation results and may introduce a bias. Another important point is that the method is not limited to images in 2D. Interactions can be applied between nodes sampled in any dimension, although increased dimensionality will usually also increase the number of interaction terms significantly. To overcome this issue, we will now look closer at using SLGs with one particular intra-layer structure.

2.3. Ordered multi-column graphs

Li *et al.* [13] describe how to use an ordered multi-column graph to segment multiple interacting surfaces with a star-shaped prior. When an approximate position of an object is known, it has several advantages over a N-neighborhood structure. For instance, the surface smoothness parameter, Δ, makes it very robust, even with noisy data. Also, the number of containment terms remains almost constant for any minimum margin, δ^l. This overcomes a major problem of Eq. (3), namely that the number of terms depends polynomially on the margin size. Furthermore, the column graph allows for the specification of a maximum margin, δ^u, which can be very helpful for many segmentation problems. Such a margin cannot be specified using a N-neighborhood structure. The δ parameters, used by [13], specifically refers to the neighborhood distance on identically sampled layers. Our distance measure, d, used in Eq. (3) and (4), is an arbitrary distance measure. For simplicity, we use Euclidean distance in this paper.

C-SLGs can be seen as a generalization of the resampling-based method used by [13] for object segmentation. Because their method relies on the neighborhood distance, δ, for containment interactions, all layers must be sampled at the same positions. If instead, we use Eq. (3) to add containment terms, based on the Euclidean distance between the sample locations, layers no longer have to be sampled identically. Because of the difference in size and center position of objects, sampling them differently provides a much better basis for our segmentation.



(a) Many redundant terms (b) Few redundant terms

Figure 3. Layered containment terms enforcing a minimum margin between object 1 and 4 in Fig. 2a. (a) No removal of redundant terms. (b) Most redundant terms removed.

Exclusion interactions are not used by [13]. As they use the BK [1] implementation to cut the graph, non-submodular terms cannot be used in their model. Also, δ cannot be used for exclusion margins on radially sampled ordered multi-column graphs, as non-overlapping objects would always be sampled differently.

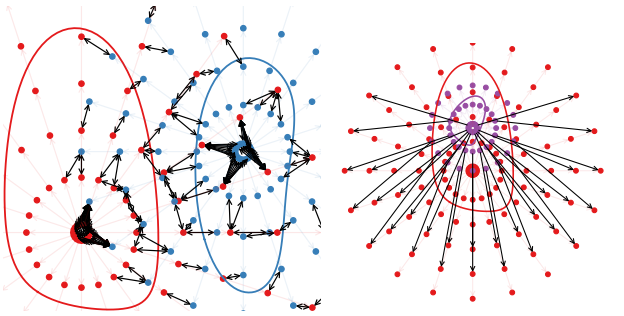
Inspired by the approach of [13], we propose two algorithms for reducing the number of interaction terms in C-SLGs. We also describe an algorithm for enforcing a maximum containment margin in C-SLGs.

To reduce the number of terms, without changing the solution, we rely on the fact that the ordered multi-column graph has infinite cost terms inside each column. As the interaction terms are also infinite cost, many of the terms added by Eq. (3) and (4) are redundant. By not adding the redundant terms to the graph, we can reduce the size of the graph significantly. Because the nodes are ordered, calculating which interaction terms are required and which are redundant, can be done quickly, before constructing the actual graph.

It should be noted that if the sampling resolution is low, meaning that the nodes are far apart, compared to the specified minimum margin, d, margins may not be enforced properly. This is a result of Eq. (3) and (4) only adding terms between nodes within the given margin. It is possible to extend both algorithm 1 and 2 to accommodate for this issue, but for now we will focus on reducing the number of terms and assume that the sampling resolution is sufficiently high compared to the margins.

2.3.1 Algorithm 1: Reducing containment terms

Fig. 3a shows the containment interaction terms for minimum margin, d^l_{14}, created using Eq. (3). Fig. 3b shows the same constraint, but with fewer terms. Our algorithm



(a) Exclusion interaction terms (b) Maximum margin containment interaction terms

Figure 4. (a) Layered exclusion terms with minimum margin between object 1 and 2 in Fig. 2a. (b) Layered containment edges for maximum margin between object 1 and 4 in Fig. 2a

for adding minimum margin interaction with few redundant terms is as follows:

1. For each *inner object* node (purple in Fig. 3), find all *outer object* nodes (red in Fig. 3) within a distance, d^l. Add all these pairs of nodes to the set of candidates, C. This is the approach from Eq. (3).
2. Remove pairs from C, so that only one pair remains for each *outer object* node, *inner object* column combination. The pair kept should be the pair with the innermost *inner object* node.
3. Remove pairs from C, so that only one pair remains for each *inner object* node, *outer object* column combination. The pair kept should be the pair with the outermost *outer object* node.
4. For each pair in C, add a pairwise term $\theta_{oi}(0,1) = \infty$ to E, where o is the *outer object* node and i is the *inner object* node.

As we will show experimentally, this algorithm can reduce the number of interaction terms by orders of magnitude. Theoretically, if N_I is the number of nodes in the inner object layer, N_O is the number of nodes in the outer object layer, and N_{IC} and N_{OC} are the number of columns in the inner and outer layers respectively, the worst-case number of containment terms is reduced from $N_I \cdot N_O$ to approx. $N_{IC} \cdot N_O + N_{OC} \cdot N_I$. In the case of radial resampling, N_{IC} and N_{OC} are the number of sample angles.

2.3.2 Algorithm 2: Reducing exclusion terms

As with containment, when we create a C-SLG, we can reduce the number of exclusion interaction terms compared to the general approach from Eq. (4). The algorithm is as follows:

1. For each *object 1* node (red in Fig. 4a), find all nodes in *object 2* (blue in Fig. 4a) within a distance, d^e. Add all these pairs of nodes to the set candidates, C. This is the general approach from Eq. (4).
2. Remove pairs from C, so that only one pair remains for each *object 1* node, *object 2* column combination. The pair kept should be the pair with the innermost *object 2* node.
3. Remove pairs from C, so that only one pair remains for each *object 2* node, *object 1* column combination. The pair kept should be the pair with the innermost *object 1* node.
4. For each pair in C, add a pairwise term $\theta_{o_1 o_2}(1,1) = \infty$ to E, where o_1 is the *object 1* node and o_2 is the *object 2* node.

The result of adding exclusion between object 1 and 2 in Fig. 2a can be seen in Fig. 4a.

2.3.3 Algorithm 3: Maximum containment margin

Because maximum containment margins cannot be enforced on N-neighborhood structured graphs, we have no general method for adding this type of interaction. Nevertheless, as shown by [13], it is possible to add this type of interaction when using an ordered multi-column graph with identical layers. However, for non-identical layers, there is no simple way of determining which columns and nodes in the two objects should interact.

We propose an algorithm for adding maximum containment interactions to C-SLGs with non-identical layers. It is designed to add intuitive maximum margin constraints using very few terms. The algorithm is as follows:

1. Calculate the node position gradient for both the *outer object* (red in Fig. 4b) and *inner object* (purple in Fig. 4b) along the columns. This indicates the direction of the column in the sample space. For columns where node positions form a straight line, such as radially sampled columns, the gradient is the same for all nodes in a column.
2. Move the *inner object* nodes in the direction of their gradient with the distance d^u.
3. For each node in the *inner object*, find the four nearest nodes in the *outer object* and add these pairs to the set of candidates, C.
4. For each pair in C, calculate the original distance between the two nodes, as it was before the *inner object* nodes were moved. Remove any pairs from C, where this distance is less than d^u.
5. For each pair in C, calculate the angle between the two nodes using the gradient from before. If the angle between the gradient vectors is more than 90 degrees, remove the pair from C.
6. Remove pairs from C, so that only one pair remains for each *outer object* node, *inner object* column combination. The pair kept should be the pair with the outermost *inner object node*.
7. Remove pairs from C so that only one pair remains for

each *inner object* node. The pair kept should be the pair with the smallest angle between the node gradients.

8. For each pair in C, add a pairwise term $\theta_{io}(0,1) = \infty$ to E, where i is the *inner object* node and o is the *outer object* node.

The result of using this algorithm is shown in Fig. 4b. We see that the algorithm effectively applies the interaction terms between nodes in columns pointing in the same direction, which is what we want.

2.4. Solving the graph

To handle non-submodular terms we use QPBO, because it is a general algorithm for solving problems of the form shown in Eq. (1). In general, unlabelled nodes may occur when using the QPBO algorithm to solve problems with non-submodular energy terms, such as exclusion. However, when using QPBO with SLGs, our experience shows that the number of unlabelled nodes is negligible.

3. Experiments

We have tested our method on two different datasets. A high-resolution 3D image of nerves, collected using μCT, and the nuclei image set BBBC038v1, available from the Broad Bioimage Benchmark Collection [14]. The primary goal of the experiments is to show the scalability of SLGs for both 2D and 3D multi-object segmentation. Secondly, we want to highlight the benefits of using C-SLGs, compared to neighborhood-based graph structures.

All experiments were run on an Azure H8m virtual machine running an Intel Xeon E5-2667 v3 CPU with 8 virtual processors and 112 GB memory.

Data, code and Jupyter notebooks for all experiments can be found at DOI 10.11583/DTU.12016941. A Python package for building and cutting sparse layered graphs has also been published to GitHub[1].

3.1. Nerve segmentation

The μCT volume is $2048 \times 2048 \times 2048$ voxels and contains several hundred nerves. Each nerve consists of a dark outer ring (myelin) and a bright core (axon). Because the axon and background have the same intensity, and because the intensities vary a lot between the nerves, accurately segmenting all nerves using the same parameters is difficult.

Before we start our experiments, center lines have been created for 216 of the nerves. Also, a single slice has been taken out of the volume and cropped to 512×512 pixels. The slice contains 17 nerves and has been segmented manually to obtain a ground truth segmentation shown in Fig. 5a.

[1] https://github.com/Skielex/slgbuilder

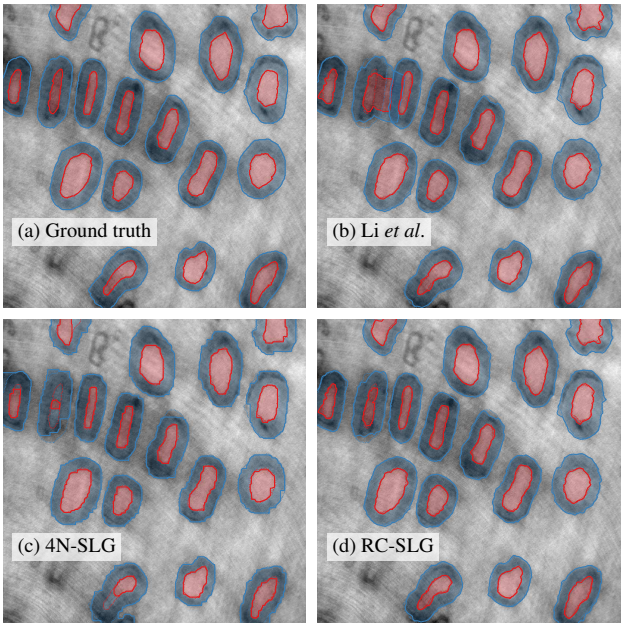


Figure 5. The most accurate nerve slice segmentation for each method and ground truth. The corresponding accuracy for each method is shown in Table 1.

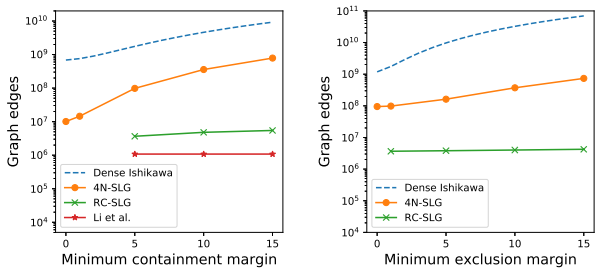
	4N-SLG		RC-SLG		Li *et al.*	
Nodes (mil.)	2.46		0.58		0.28	
	Min	Max	Min	Max	Min	Max
Edges (mil.)	6.8	1425	3.6	6.1	1.1	1.1
Time (s)	1.95	545	0.48	2.70	0.19	0.74
F1	0.91		**0.94**		0.92	
Precision	**0.95**		0.93		0.90	
Recall	0.90		**0.96**		0.95	

Table 1. Results for nerve slice segmentation using a 250 different configurations. The number of configurations was 25, 180 and 45 for the 4N-SLG, RC-SLG, and Li *et al.*, respectively. The parameters varied were the three interaction margins and the surface smoothness. For each method, the number of nodes is the same for all configurations, while the number of edges and solve time change. The accuracy is calculated as the mean score of all masks for a given configuration.

3.1.1 Single volume slice

We use the 512×512 image to compare the accuracy and graph size of the original method by Li *et al.* to a 4-neighborhood SLG (4N-SLG) and a radially resampled column SLG (RC-SLG). For each method, we evaluate several different configurations for margin sizes and surface smoothness.

Fig. 5 shows the most accurate segmentation for each method. As shown in Table 1, the RC-SLG provides the most accurate segmentation. The 4N-SLG struggles when the contrast between the myelin and the axon are low, while the method by [13] does not support exclusion and thus in-



(a) (b)

Figure 6. The two plots show the increase in the number of graph edges as the minimum interaction margins are increased for the 2D nerve segmentation problem shown in Fig. 5. The numbers for the Dense Ishikawa graph are theoretical, while the numbers for the other three methods are based on our experiments. In (a) we see how the graph size increase for the different methods as the minimum containment margin is increased. Other parameters are kept constant and as similar as possible. Li *et al*. does not have exclusion terms, which is is the main reason it has fewer edges than the RC-SLG. In (b) we similarly increase the exclusion margin. Here Li *et al*. is omitted since it does not have exclusion.

correctly overlap some segments. We also see that the number of edges and solve time vary significantly for the 4N-SLG, depending on the configuration. Fig. 6 shows how the number of edges changes depending on the margins. It is clear that the N-neighborhood-based methods do not scale well, as we increase the interaction margin. In fact, the Ishikawa graph quickly grows so big we cannot create the graph due to lack of memory. The two ordered multi-column-based methods do not have this problem, allowing us to set appropriate margins with little to no cost.

3.1.2 Full volume

To show the scalability of C-SLGs we segment all 216 nerves (432 objects) in the 2048-cubed volume with a single graph cut. For this, we used the same radial approach as for the slice, just in 3D and with a lower resolution. Along the annotated centerline, we sample points radially on planes orthogonal to the centerline. In total, we sample 182 million different positions in the volume, with which we construct the RC-SLG. It takes 44 minutes to solve the problem using the QPBO algorithm. The complete graph contains a total of 363 million nodes and 2.1 billion edges. The result contains no unlabelled nodes, which means we found the globally optimal solution to the problem. Solving this problem with a dense Ishikawa structured graph is not possible due to hardware limitations.

3.2. Nuclei segmentation

The purpose of this experiment is to compare the 4N- and RC-SLG on a large number of different images with ground truth segmentation masks. We compare both complexity and accuracy and investigate the frequency of unlabelled nodes.

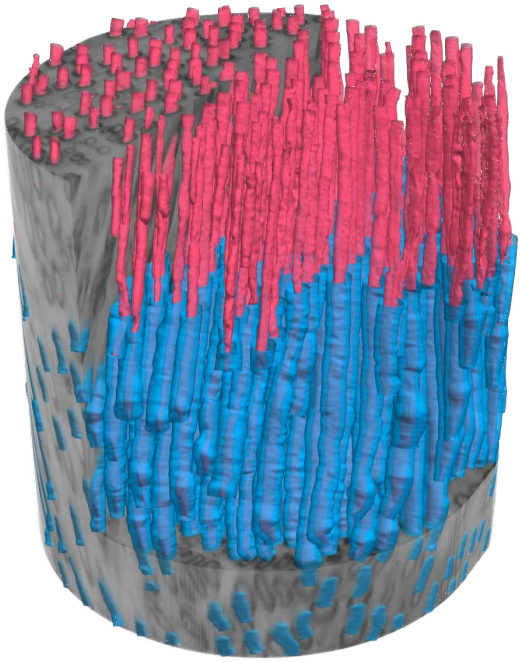


Figure 7. Nerve volume segmentation of the myelin and axon of 216 nerves created using a RC-SLG.

The BBBC038v1 `strage1_train` image dataset contains 670 images with a total of 29,461 segmented nuclei. The images were acquired using different imaging modalities and vary in size. The type of cells and their size vary between images and the number of nuclei per image ranges from a few to several hundred.

As a part of the experiment, we use the exact same configuration for all images. This allows us to test how sensitive/robust the methods are. Ideally, it should not be necessary to configure parameters for each individual image. For this reason, we also use the same simple gradient-based data terms for all images. One of the segmentation results is shown in Fig. 8.

In Table 2 we see that the RC-SLG significantly outperforms the 4N-SLG, both in terms of accuracy and speed. The low recall and high precision score of the 4N-SLG, indicate that it tends to underestimate the size of the nuclei. The RC-SLG does not have this issue and accurately segments nuclei of all sizes, even though they vary from a few to over a hundred pixels in diameter. Furthermore, segmentations by the RC-SLG have fewer unlabelled nodes than those by the 4N-SLG, although they hardly impact accuracy in either case. In fact, for RC-SLG, 99.8% of the masks and 97% of the images are completely labeled.

Fig. 9a further highlights the scalability of SLGs, and

	4N-SLG		RC-SLG	
Per image	Mean	Max	Mean	Max
Nodes (mil.)	2.25	19.2	0.71	6.08
Edges (mil.)	14.6	442	3.12	60.3
Time (s)	2.55	69.5	1.02	26.1
Per mask	Mean	Max	Mean	Max
Unlabelled	3.2	428	0.48	876
Per mask	Mean	Std.	Mean	Std.
F1	0.48	0.32	**0.85**	0.12
Precision	**0.97**	0.09	0.85	0.13
Recall	0.40	0.34	**0.89**	0.16

Table 2. Results for nuclei segmentation on 670 image with 29,461 segmented nuclei. The RC-SLGs are generally smaller than the 4N-SLGs, which also makes them faster to solve. In terms of accuracy, the RC-SLGs perform significantly better than the 4N-SLGs. In terms of unlabelled nodes, the RC-SLGs also perform best, leaving only 0.006% of nodes unlabelled, compared to 0.04% by the 4N-SLGs.

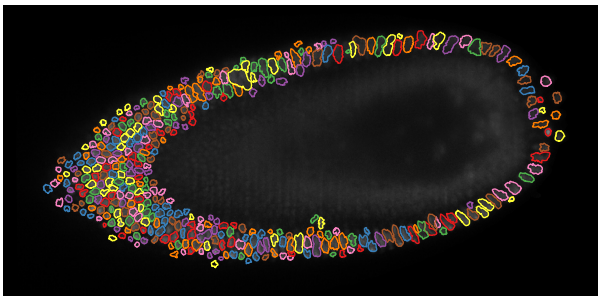
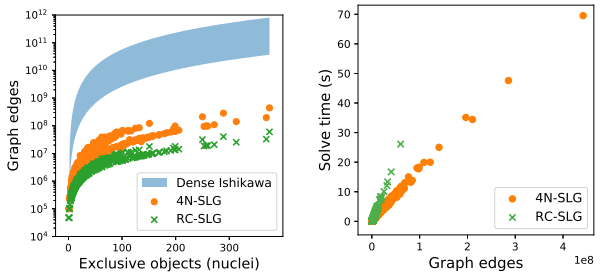


Figure 8. Largest nuclei image segmented using the RC-SLG. The image is 1272×603 pixels and has 375 segmented nuclei. The RC-SLG had approx. six million nodes and 60 million edges. A dense Ishikawa graph would have had over one billion nodes and need over 100 billion edges to enforce exclusion between all nuclei. With an exclusion margin of five, this number increases to over 10 trillion edges.

in particular C-SLGs. Fig. 9b is interesting as it shows a linear correlation between the number of graph edges and the solve time for both methods in our experiment. This means that as long as we can keep the number of edges low, we should be able to find a solution fast.

4. Discussion and conclusion

A limitation of our method is that it requires some prior knowledge about the number of objects, and where they are. However, most graph cut-based segmentation methods require this kind of prior knowledge. Another challenge is that segmentations may be incomplete, as we rely on the QPBO algorithm for solving non-submodular problems. Nevertheless, our experiments show unlabelled nodes are rare, and thus not a problem for the accuracy of the segmentation. One reason for this could be that we only



(a) (b)

Figure 9. Two plots based on our results from segmenting 670 images of nuclei. In (a), we see how the number of objects (nuclei) in the images affect the number of edges. We assume that the reason the points for the 4N-SLG and RC-SLG are not on a smooth line is that the number of interaction terms also depends on the relative positions and sizes of the objects. The numbers for the dense Ishikawa graph are theoretical. The area indicates the number of edges for the smallest and largest image in the image set. Plot (b) shows the correlation between the time it took to solve the graph cut and the number of graph edges for each image. For both methods, the Pearson correlation coefficient is over 0.99, which indicate a linear correlation between the number of edges and solve time.

use non-submodular terms for exclusion. It also appears that the RC-SLG is better for avoiding unlabelled nodes than the 4N-SLG. This is interesting, as using QPBO with N-neighborhood-based geometric priors [9] has previously been shown to result in many unlabelled nodes [8]. To label unlabelled nodes, extensions to QPBO, such as QPBO-I and QPBO-P have been proposed [16]. Although we do not use these extensions in our experiments, they could be used to further reduce the small number of unlabelled nodes in our segmentations.

Overall, our experiments show that SLGs, and in particular C-SLGs, can be used to segment very large images (2D and 3D) accurately, even with simple gradient-based data terms. These tasks are unsolvable using traditional dense graph structures. Although we have focused on images, SLGs, as well as Eq. (3) and Eq. (4) are general and can be used for 4D or point cloud data as well.

It is clear that the C-SLGs, created using our three algorithms, provide a particularly effective way of solving large segmentation tasks. In this paper, we enforced a star-like prior for the C-SLGs by using radial resampling. However, we believe there is a large potential in using C-SLGs with sampling schemes based on other priors [7, 9, 17], or by sampling based on surfaces in 3D [3, 15].

Acknowledgments This work is supported by FORCE Technology and The Center for Quantification of Imaging Data from MAX IV (QIM).

References

[1] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, Sept 2004.

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, Nov 2001.

[3] V. A. Dahl, A. B. Dahl, and R. Larsen. Surface detection using round cut. In *2014 2nd International Conference on 3D Vision*, volume 2, pages 82–89, Dec 2014.

[4] A. Delong and Y. Boykov. A scalable graph-cut algorithm for n-d grids. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

[5] A. Delong and Y. Boykov. Globally optimal segmentation of multi-region objects. In *2009 IEEE 12th International Conference on Computer Vision*, pages 285–292, Sept 2009.

[6] A. V. Goldberg, S. Hed, H. Kaplan, P. Kohli, R. E. Tarjan, and R. F. Werneck. Faster and more dynamic maximum flow by incremental breadth-first search. In N. Bansal and I. Finocchi, editors, *Algorithms - ESA 2015*, pages 619–630, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[7] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman. Geodesic star convexity for interactive image segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3129–3136, June 2010.

[8] H. Isack, O. Veksler, I. Oguz, M. Sonka, and Y. Boykov. Efficient optimization for hierarchically-structured interacting segments (HINTS). In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4981–4989, July 2017.

[9] H. Isack, O. Veksler, M. Sonka, and Y. Boykov. Hedgehog shape priors for multi-object segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2434–2442, June 2016.

[10] H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, Oct 2003.

[11] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts – a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1274–1279, July 2007.

[12] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, Feb 2004.

[13] K. Li, X. Wu, D. Z. Chen, and M. Sonka. Optimal surface segmentation in volumetric images – a graph-theoretic approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):119–134, Jan 2006.

[14] V. Ljosa, K. L. Sokolnicki, and A. E. Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature Methods*, 9(7):637–637, 2012.

[15] I. . Oguz and M. Sonka. Logismos-b: Layered optimal graph image segmentation of multiple objects and surfaces for the brain. *IEEE Transactions on Medical Imaging*, 33(6):1220–1235, June 2014.

[16] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[17] O. Veksler. Star shape prior for graph-cut image segmentation. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision – ECCV 2008*, pages 454–467, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[18] V. Vineet and P. J. Narayanan. Cuda cuts: Fast graph cuts on the gpu. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, June 2008.

[19] X. Wu and D. Z. Chen. Optimal net surface problems with applications. In P. Widmayer, S. Eidenbenz, F. Triguero, R. Morales, R. Conejo, and M. Hennessy, editors, *Automata, Languages and Programming*, pages 1029–1042, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.