Graph reduction and application to lung tumors segmentation

Nicolas Lermé

N° attribué par la bibliothèque

Année 2011

Doctorat de l'Université Paris XIII

Spécialité : Informatique

THÈSE

présentée par

Nicolas Lermé

pour l'obtention du grade de

DOCTEUR DE L'UNIVERSITÉ PARIS XIII

Unités de recherche : LAGA UMR CNRS 7539
LIPN UMR CNRS 7030

Réduction de Graphes et Application à la Segmentation de Tumeurs Pulmonaires

Soutenue publiquement le 7 décembre 2011 devant le jury composé de :

M. Françoise Dibos	Président
M. Michel Couprie	Rapporteur
M. Patrick Pérez	Rapporteur
M. Laurent Cohen	Examinateur
M. François Malgouyres	Directeur de thèse
M. Lucas Létocart	Co-directeur de thèse

À mes parents.

À mes amis.

Remerciements

En premier lieu, je tiens à remercier chaleureusement Mme Françoise Dibos qui m'a fait l'honneur de présider le jury de thèse et sans qui je n'aurai probablement pas pu obtenir de financement pour mener à bien ce travail. Je tiens aussi à remercier Mr. Michel Couprie et Mr. Patrick Pérez qui ont bien voulu accepter la charge de rapporteur.

Je remercie également mes directeurs de thèse Mr. François Malgouyres et Mr. Lucas Létocart pour leur patience et leur soutien constant durant ces trois années. Je tiens à leur exprimer ma gratitude pour m'avoir accueilli avec chaleur et enthousiasme au sein des équipes MTII du LAGA et AOC du LIPN. Leur disponibilité, leurs encouragements permanents et leur ouverture d'esprit ont incontestablement contribué à l'enrichissement de mes connaissances sur des sujets passionnants dont l'intérêt n'est pas démenti.

Je souhaite aussi remercier Mr. Jean-Marie Rocchisani pour m'avoir fournir les données expérimentales dédiées à la validation du prototype de segmentation des tumeurs pulmonaires. Je remercie également Mr. Sébastien Li-Thiao-Té pour sa collaboration sur la partie concernant l'estimation des lois de distributions.

Merci aux autres membres du LAGA et du LIPN pour leur bonne humeur permanente et qui ont directement ou indirectement contribué à ce travail à travers des discussions parfois longues mais instructives. Un grand merci au personnel de l'Université Paris 13 qui ont su remédier à certaines tracasseries administratives et techniques.

Enfin, un grand merci à mes parents pour leur soutien inconditionnel et leurs encouragements permanents. Je remercie plus particulièrement mon père d'avoir éveillé ma curiosité pour l'informatique lors d'une réunion informelle dans les locaux de France Télécom au début des années 90.

Contents

Introduction

Energy minimization is a widespread approach in Computer Vision and Graphics. Usually, vision problems have many solutions due to the uncertainties in the acquisition process and ambiguities in visual interpretation. A classical use of energy minimization is the labeling problem, an abstraction of particular Computer Vision problems such as stereo, motion, restoration or segmentation. The inputs are a set of pixels $\mathcal{P}$ and a set of labels $\mathcal{L}$. The goal is to find a labeling (i.e. a mapping from $\mathcal{P}$ to $\mathcal{L}$) which minimizes some energy function. Usually, the energy function encodes the constraints and prior of the problem and its minimum gives the desired optimal solution. A standard form of such an energy is

$$E(u) = \beta \sum_{p \in \mathcal{P}} E_p(u_p) + \sum_{(p,q) \in \mathcal{N}} E_{p,q}(u_p, u_q), \qquad \beta \in \mathbb{R}^+,$$

among $u \in \mathcal{L}^{\mathcal{P}}$ and where $\mathcal{N} \subset \mathcal{P} \times \mathcal{P}$ is a neighborhood system. The term $E_p(.)$ called data term is a function derived from the observed data. In other words, it measures how much assigning label u_p to pixel p disagrees with initial data. The term $E_{p,q}(.)$ called smoothness term is a function imposing spatial smoothness on the solution by giving penalties to neighboring pixels p and q having different labels. Non-convex $E_{p,q}(.)$ terms are generally preferred over convex ones for discontinuity-preserving since it is important to not over-penalize labelings at borders. In such situations, energy functions have traditionally been minimized with general-purpose optimization techniques (such as simulated annealing). As a consequence of their generality, such techniques usually require an exponential time and are extremely slow to converge in practice. However, efficient techniques such as graph cuts have become increasingly popular.

Graph cuts are a discrete optimization method based on maximum-flow / minimum-cut computations in graphs for minimizing energies frequently arising in Computer Vision and Graphics. Since last decades, this method has become a cornerstone in these communities for solving a wide range of problems such as denoising, segmentation, registration, stereo, scene reconstruction, panorama stitching, etc. We refer the reader to [BK04] for

typical applications of graph cuts. Since seminal work of [GPS89] for denoising binary images, graph cuts have recently known a quick development mainly due to the introduction of a fast maximum-flow algorithm [BK04] and heuristics offering good performance in the multi-labels case [BVZ99].

In parallel, technological advances in image acquisition have exploded both the amount and the diversity of data to process. As an illustration, in the third generation satellite SPOT-5 launched by Arianespace in 2002, the embedded high resolution sensors can capture multispectral and panchromatic 8-bits images with an imaging swath of 60×60 km. In panchromatic modes, each image has a size of 12000×12000 against 6000×6000 in multispectral mode. Notice that the next generation of satellites (namely SPOT-6 and SPOT-7) whose launch is expected for 2012/2013, will form a constellation of earth observation to maintain continuity of high-resolution data collection and distribution provided by the SPOT series. While the imaging swath remains the same as in SPOT-5, a consequence of this decision is an increase of the resolution and hence, the volume of data to process. These new satellites are now able to acquire images of size 40000×40000 in panchromatic mode against 10000×10000 in multispectral mode. The information acquired by these satellites is naturally of great importance with a wide spectrum of applications ranging from defense and cartography to environment and agriculture. Similarly, latest medical imaging systems are now able to acquire 3D and 3D+t volume data with several billion of voxels whereas latest digital cameras embed sensors of 20 million pixels.

Processing such data amounts to solve large-scale optimization problems with a large number of variables. In particular, graph-based methods appear to be totally impractical to solve such problems due to the huge memory requirements. To overcome this situation, some amount of work has been recently done in this direction with some heuristics [LSTS04, CA08, SD07, LSGX05, SG06, KLR10] and exact methods [LB07, SK10]. The rough heuristics appearing in the literature get easily trapped in a local minimum of the energy, losing the main benefit of global optimization. Others have also proposed original ideas where global optimality is guaranteed on the generated solution [LB07, SK10]. While [LB07] is devoted to a particular application, the graph cuts limits are pushed away

in [SK10] by splitting the problem in a parallelized/distributed fashion instead of reducing graphs.

The objective of this thesis is to propose a new band-based strategy (see the work of [LSGX05, SG06, KLR10]) for reducing graphs involved in binary graph cuts segmentation and improve the running time of the max-flow algorithm. First, we do some reminders about graph theory and functionals involved in image segmentation. Then, we present a state of the art of the methods for reducing these graphs. Afterwards, we detail a heuristic and an exact test for reducing them. For each one, massive experiments are provided for evaluating the performance of the proposed algorithms in terms of time and memory. Finally, we present an application of this technique for segmenting lung tumors in an interactive fashion and discuss about the proposed work. We now briefly detail the organization of this document.

Chapter 1

This chapter gives some basic notions about segmentation functionals and graph theory. First, we briefly remind the most popular functionals involved in image segmentation as well as the existing methods (among graph-based methods) for solving this problem using the classification of [Lec09]. Afterwards, some reminders on the graph cuts theory are provided. We first give general definitions about graphs, review the duality between the max-flow and the min-cut problems in a network and the commonly used approaches for efficiently solving them. Then, we remind how this method can be applied in the image segmentation context and describe two widespread energy models: Boykov-Jolly and TV+L^2.

Chapter 2

In this chapter, we first describe the problem of the prohibitive memory consumption of graph cuts and demonstrate that this method is totally impractical for segmenting large volume data. Next, we present a detailed state of the art of methods which overcome this problem, among heuristics or exact methods. We split this state of the art into two parts: sequential algorithms and parallelized / distributed algorithms. For each method, we detail its internal working scheme and discuss about its advantages and its limits.

Chapter 3

This chapter details the strategy adopted for reducing graphs involved in binary image segmentation. First, we introduce a simple test for testing if a node in a graph is really useful to the max-flow computation. The reduced graph is then progressively built by only adding nodes which satisfy this test. This leads to a straightforward algorithm with a worst-case complexity similar to a convolution. The remaining nodes are typically located in a narrow band surrounding the object edges to segment. Numerical results with two different energy models are provided: they clearly show that the solutions obtained on the reduced graphs are identical to the solutions on the whole graphs. Furthermore, when the amount of regularization in the model is of moderate level, the time required by the reduction algorithm is compensated by the decrease of the time for computing the max-flow on the reduced graph. Secondly, we present another test for reducing exactly graphs and establishing a comparison of reduction performance between both tests.

Chapter 4

In this chapter, we address the problem of segmenting accurately lung tumors in an interactive fashion in 3D Computed Tomography (CT) images. We first present a novel energy formulation embedding a prior on the seeds location inhibiting the propagation of object seeds thanks to a Fast Marching algorithm based on the image gradient. The generated graphs are then reduced using the approach detailed in Chapter 3. Afterwards, we evaluate the method against a dataset composed of ground truths provided by an expert. We finally show that consistent and accurate results can be reached, enabling the use of a valid prototype by clinicians.

Conclusion and discussion

We end this document by providing a summary of the developed contributions and a discussion about the possible perspectives at short, middle and long term. We also establish potential connections with other previous works on the same subject.

1 Preliminaries

1.1 Image segmentation: history and methods

1.1.1 What is segmentation?

Originally, segmentation (or more precisely grouping) is a primitive process intervening in the visual perception system of common animal species for partitioning a display into meaningful regions. Since founding paper of Wertheimer in 1923 [Wer23] who crystallized the so-called "Gestalt principles" [1], this process has become a very active field of research in psychology with the Gestalt theory.

At about the same date, Image Processing emerged as a new research field of Signal Processing devoted to the study of digital images. By analogy to visual grouping, the segmentation problem [2] in Image Processing refers to the process of partitioning a digital image into multiple and coherent semantic regions (i.e. sets of pixels, also called superpixels) with a machine. The contours separating two adjacent regions correspond to the boundaries of the segmentation. This problem is closely related to image classification. Image segmentation refers to clustering or grouping pixels into various groups whereas image classification determines to which category belongs an image (or a subset of it). Typical categories can be "landscape", "sea", "person", etc. and are fixed.

In a sense, computer scientists try to mimic the process which naturally arises in the brain of animal species. As grouping, image segmentation is also a primitive process intervening in the elaboration of high-level alrithms such as 3D reconstruction, object recognition, data compression, object tracking [RCD07], etc.

Observe that both disciplines attempt to answer the same question: how to arrive at global percepts from the local information contained into an image? Although obvious

[1] Gestalt is a German word usually meaning "shape" or "form".

[2] Not to be confused with the co-segmentation problem whose objective is to segment a similar object from a pair of images [HS09].

similarities appear in both disciplines, Image Processing did not use at first the Gestalt theory. This lack of initial interaction is not really surprising since Gestalt principles presented in the initial Wertheimer's programme [Wer23] do not translate readily into algorithms. Also, in the case where two (or more) principles apply simultaneously for the same input, a lot of work remains to be done to predict which one will win. While the Gestalt principles are well established among psychologists, they have been criticized for not being able to explain the phenomena they have uncovered. Furthermore, much of the research on this topic was conducted only with two-dimensional drawings.

However, Gestalt specialists commonly admit that grouping appears in the first steps of the visual perception system [Kof35, Köh29, Li00]. The same authors also proved that these first steps are independent of any learning or prior knowledge on the world. Thus, it seems reasonable to think that an algorithm, processing digital images, can reach the same objective. Some amount of work has been recently done to formalize Gestalt principles into a probabilistic setting [DMM04]. The experiments of [DMM04] demonstrate that grouping and image segmentation face exactly the same challenges as in the Gestalt theory. In what follows, we briefly remind the image segmentation techniques and the most popular functionals involved in contemporary work.

1.1.2 History and methods

Image Processing begins to be studied in the 20s for sending images over the Atlantic sea through underwater cables. However, the time for transmitting data is about one week and still needs to be drastically reduced to be popularized. Later, H.G. Bartholomew and M.D. McFarlane reduced this duration to about three hours by compressing data beforehand but do not evolve anymore until the end of the second world war. The true development of Image Processing only begins in the 60s due to technological advances in electronics and computer science. The constant miniaturization of electronic components makes possible for a wide spectrum of applications in various fields such as advertising, entertainment, health, army, industry, security, etc.

The image segmentation problem only begins to be really studied in the 60s/70s lead-

ing to significant results. If human beings naturally know how to distinguish objects in an image, it is mainly due to a global understanding of it. A machine do not possess neither prior knowledge on the image nor a way to check if the obtained result is valid or not. Segmenting an image is also a subjective process. As a consequence, several valid segmentations can exist for the same image. Moreover, the notion of "good" segmentation strongly depends on the image to process and the application. The image segmentation problem early appears to be ill-posed: knowing the observed data, the solution to it does not exist and/or is not unique and/or does not depend continuously on input data. To obtain a well-posed problem, a common and reasonable assumption is to assume that images vary smoothly within regions and discontinuously across boundaries. An important point to note is the duality between regions and contours: a region is defined by its contours while a contour is a boundary between two adjacent regions. Historically, researchers have exploited this duality for segmenting images leading to two main approaches: region-based and contour-based approaches. Contour-based methods search for discontinuities in the image. They are typically divided into two steps: detection of boundaries and thresholding. The first step is based on local properties of boundaries. Earliest methods of finding boundaries used small convolution masks to approximate first [Rob65, Pre70, Sob70] and second derivatives of the image [MH80]. Gaussian averaging improves the detection of boundaries but reduced their localization. From this, Canny [Can86] introduced three criteria that an edge detector must satisfy: reliability of detection, accuracy of localization and unique response per edge. These criteria are then embedded into a cost function to find an optimal edge detector. In the second step, best candidates of boundaries are generally kept by using simple or hysterysis thresholding on edges magnitude.

In the region-based approach, one search for regions in the image that are consistent with respect to a given criterion. More precisely, we want to assign a label to each pixel such that pixels having the same label share common features such as color, direction, texture, motion, proximity, convexity, etc. For instance, when the number of labels is two, the segmentation task consists in dissociating objects on a background and is sometimes refered as binary segmentation. When the number of labels is larger, we will refer to it as multi-labels segmentation in the sequel.

Region-merging algorithms are a typical example of such an approach [BF70, Pav72]. These algorithms generally proceed by aggregating small adjacent regions into larger ones until some criterion is satisfied. As an illustration, Brice and Fenema choose the following criterion [BF70]: two regions are said to be similar if there is low jump along their common border. At the beginning of the algorithm, all pixels sharing the same intensity belong to the same region. A statistical criterion is proposed by Pavlidis [Pav72]: two regions are said to be similar if the variance of the regrouped regions is less than some threshold. The regions initially correspond to a single pixel. Later, Beaulieu and Goldberg improve this criterion using the same initialization [BG89]. At each iteration, the algorithm finds a couple of regions whose the variance of the regrouped regions minus the sum of the variance of each region is minimal and fuse them. The algorithm stops when the desired number of regions is obtained. Horowitz and Pavlidis were the first ones to provide a formal definition of a region-growing algorithm [HP76].

Definition 1. *Let f be a function mapping a pixel $x \in (\Omega \subset \mathbb{Z}^d)$ to a value $f(x)$ and a neighborhood $\mathcal{N} \subset (\Omega \times \Omega)$ defined on Ω. A typical example of neighborhood $\mathcal{N}$ consists of pixel pairs with unit distance. Then, a set of pixels $X \subset \Omega$ is said to be connected if for any pixel pair $(p_1, p_n) \in (X \times X)$, we have $(p_i, p_{i+1}) \in \mathcal{N}$, $\forall i \in \{1, \ldots, n-1\}$.*

Definition 2. *Let $\Omega \subset \mathbb{Z}^d$ be the image domain and $f(x)$ the function mapping each $x \in \Omega$ to a value $f(x)$. If we define a predicate P on the power set of Ω, the segmentation of Ω is defined as a decomposition of Ω in n subsets $\{R_1, \ldots, R_n\}$ such that*

- *$\Omega = \bigcup_{i=1}^{n} R_i$ and $R_i \cap R_j = \emptyset$, $\forall i, j \in \{1, \ldots, n\}$, $i \neq j$, (i.e. $\{R_1, \ldots, R_n\}$ is a partition of Ω),*
- *R_i is connected, $\forall i \in \{1, \ldots, n\}$ (see Definition (1)),*
- *$P(R_i) = true$, $\forall i \in \{1, \ldots, n\}$,*
- *$\forall i, j \in \{1, \ldots, n\}$ s.t. $i \neq j$, R_i is adjacent to $R_j \Rightarrow P(R_i \cup R_j) = false$.*

The predicate P is used for testing the homogeneity of the sets R_i which compose the regions of the input image. Thus, the segmentation is the decomposition of an image into a set of "homogeneous" (in the sense of P) regions. The conditions of Definition 2

can be summarized as follows. The first condition implies that each pixel belongs to one and only one region. In particular, it means that a segmentation algorithm continues until each pixel has been processed. The second and third conditions respectively imply that each region must be connected and homogeneous. Finally, the last condition is a maximality condition denoting that the fusion of two adjacent homogeneous regions must be non-homogeneous. Notice that the number of regions n remains to determine. Notice also that several segmentations can exist for the same predicate P.

Although region-growing algorithms achieve satisfactory results for a large number of images, most of them do not offer any means to regularize boundaries of the segmentation like boundary length and/or curvature [Pav72, BG89]. Furthermore, Definition 2 is somewhat limited to a particular kind of segmentation algorithms.

Variational formulation is an elegant answer to this problem. In the variational framework, problems are characterized by a functional (generally measuring some kind of reconstruction error), and solutions are defined as minimizers of this functional. Modern approaches to image segmentation are mostly based on a variational formulation and date back to the 1980s. The purpose of the following paragraphs is to review the main functionals involved in most contemporary work on image segmentation.

Consider now Ω, an open subset of $\mathbb{R}^d$ ($d > 0$) as the domain of a digital image $g : \Omega \to \mathbb{R}$ where $g(x)$ denotes the value of pixel $x \in \Omega$ in g. A segmentation of g is defined as a pair (u, K) where $u : \Omega \to \mathbb{R}$ is some approximation of g and K denotes the set of boundaries of u. In a founding paper [MS89], Mumford and Shah introduce a suitable functional aiming for reconstruction of the input image by piecewise smooth functions. They propose to minimize among every edge set K and every segmented image u

$$E(u, K) = \underbrace{\alpha_0 \mathcal{L}(K) + \alpha_1 \int_{\Omega \setminus K} |\nabla u|^2 dx}_{\text{Regularity}} + \underbrace{\int_{\Omega} (u - g)^2 dx}_{\text{Data fidelity}}, \tag{1}$$

where $\alpha_0, \alpha_1 \in \mathbb{R}^2$ are free parameters and $\mathcal{L}(K)$ denotes the total length of boundaries K. As usual, the first term in (1) imposes regularity on the boundaries K, the second one also imposes that u varies smoothly within each region except at boundaries, and the last

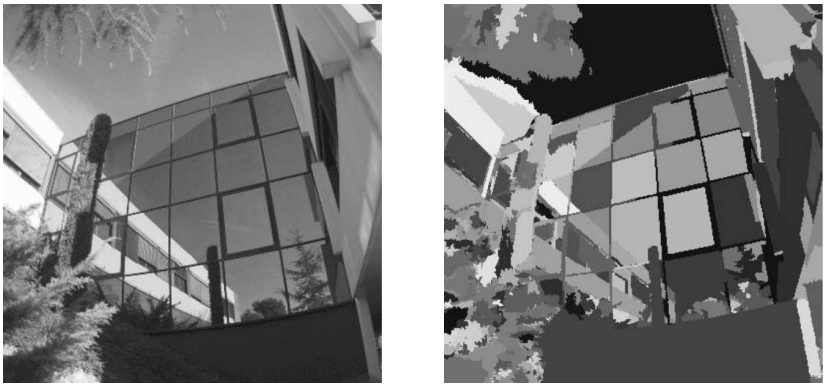


Figure 1: Segmentation (right) of a grayscale image (left) using the Mumford-Shah's functional (2). Observe how regions of approximately constant intensity are assigned to the same label in the segmentation (right).

one ensures proximity between u and observed data g. In the literature, (1) sometimes refers as the "multiphase Mumford-Shah" functional.

While good properties can be exhibited from (1), its optimization remains complex and is not easily accomplished using the standard calculus of variations. For most of the considered applications, it is generally enough to assume that u is a piecewise constant function. In such a situation, the second term in (1) vanishes. Consider now Ω as a partition, i.e. $\Omega = \cup_i \Omega_i$ and $\Omega_i \cap \Omega_j = \emptyset$, $\forall i, j \in \{1, \ldots, n\}$, $i \neq j$. Then, assuming that u is a piecewise constant function and the number of regions n is finite in u, (1) can be rewritten as

$$E(K) = \alpha_0 \mathcal{L}(K) + \sum_{i=1}^{n} \alpha_i \int_{\Omega_i} (u(x) - \bar{u}_i)^2 dx, \tag{2}$$

where $\bar{u}_i = \frac{\int_{\Omega_i} g(x)dx}{\int_{\Omega_i} dx}$ is the mean intensity of region Ω_i and $\alpha_i \in \mathbb{R}^+$ are free parameters, $\forall i \in \{0, \ldots, n\}$. Here, the operator '$\sharp$' stands for the cardinality of a set. An example of segmentation is illustrated in Figure 1. When $n = 2$, Chan and Vese proposed a level-set method (see [OS88]) for numerical realization of the optimization problem (2) [CV01]. With this approach, the unknown boundaries are represented by the zero level-set of a continuous function (e.g. a distance function) $\phi : \Omega \to \mathbb{R}$. The idea is to express the functional (2) in terms of the level-set function $\phi(.)$. They propose to minimize among every level-set function $\phi(.)$ and every mean intensities $\bar{u}_1$ and $\bar{u}_2$ the following binary

problem

$$\int_{\Omega}(|\nabla H(\phi)| + \lambda[H(\phi)(\bar{u}_1 - g(x))^2 + (1 - H(\phi))(\bar{u}_2 - g(x))^2])dx, \tag{3}$$

where $\lambda \in \mathbb{R}^+$ is a free parameter and $H : \mathbb{R} \to \mathbb{R}$ is the Heaviside function defined as $H(x) = 0$ if $x < 0$ and $H(x) = 1$ otherwise. In [CV01], the authors propose to optimize the functional (3) using Partial Differential Equations (PDE) with the following alternating minimization scheme until stability:

1. Obtain an initial guess of mean intensities $\bar{u}_1$ and $\bar{u}_2$.
2. Fix mean intensities $\bar{u}_1$, $\bar{u}_2$ and minimize (3) over ϕ.
3. Fix $\phi(.)$ and update mean intensities $\bar{u}_1$, $\bar{u}_2$ from segmentation u.

In particular, since (3) is non-convex, reaching a global minimum is not guaranteed. Notice that the work of [CV01] can be easily extended to the case where $n > 2$ by introducing more level-set functions to describe a larger number of regions [TC04] or reformulated using only one [BT08]. This algorithm quickly gained popularity mainly due to its ability to segment objects that are not necessarily defined by gradient. Also, in the level-set literature a manually placed contour or circular seeds are often used to obtain a good estimate of $\bar{u}_1$ and $\bar{u}_2$ [CV01] but unsupervised clustering algorithms have also been used in the past [BT08]. Notice that the speed and the reliability of solutions strongly depend on the initialization in [CV01]. Although traditional PDE-based methods remain very popular due to their ability to automatically deal with topology changes, they generally are computationally expensive and suffer from numerical instability and to local minima. Convergence can be sped up using graph cuts in the step (2) of the previous alternating minimization scheme [BT08, ZCP06]. [ZCP06] improve performance by taking advantage of dynamic graph cuts [KT07] between successive iterations. Experiments in [KT07] show that graph cuts outperform level-set methods by several orders of magnitude. Graph cuts also appear to be less sensitive to initialization than level-set methods albeit an initial guess of mean intensities is still necessary.

Finally, we want to mention the work of [RD02] which introduces another Bayesian model for image segmentation when $n = 2$. They pursue the issues adressed by [CV01]

and improve the results obtained by representing each region Ω_i by a Gaussian function instead of a constant intensity $\bar{u}_i$, $\forall i \in \{1, \ldots, 2\}$. They propose to minimize among every edge set K and mean intensities $\bar{u}_1$ and $\bar{u}_2$

$$\alpha_0 \mathcal{L}(K) + \sum_{i=1}^{2} \alpha_i \int_{\Omega_i} e_i(x) dx, \tag{4}$$

with

$$e_i(x) = \frac{(u(x) - \bar{u}_i)}{\sigma_i^2} + log\ \sigma_i^2,$$

where σ_i^2 is the intensity variance of the region Ω_i, $\forall i \in \{1, 2\}$. Again, the functional (4) is optimized using the same alternating minimization scheme as in [CV01]. Previous references are not exhaustive since segmentation methods are relatively abundant in the literature. Differences between these methods generally lie in the functional and the way it is minimized. A good attempt of classification has been recently made in [Lec09] in the medical context and could be extended in various ways:

- **Region-based.** This approach uses localization and identification techniques of connected sets of pixels. Classification methods partition the image into several labels (or classes) and often constitute a first step in image segmentation. In [Lec09], these classification methods are split according to several criteria: probabilistic, deterministic, (non-)parametric and (non-)supervised. For instance, it includes neural networks (deterministic supervised method), k-means and mean shift (deterministic non-supervised methods), Markovian approaches and Support Vector Machines (probabilistic non-parametric methods).
- **Edge-based.** Unlike region-based approach, the primitives to extract are boundaries separating multiple regions. In words, it consists in identifying transition areas and localizing the boundary between regions. In [Lec09], this approch also includes derivative scale-space models and wavelets (and its derivatives such as bandlets or curvelets).
- **Structural.** This approach makes use of set operations to build morphological operators (e.g. erosion, dilatation, morphological gradient, etc.) or higher level algorithms like the Watershed algorithm.

- **Shape.** Methods based on the shape tend to find regions deriving from a shape given as a priori. Such methods include in [Lec09] spherical harmonics, level-sets and active contours.
- **Graph theory.** Graph-based segmentation approaches have recently attracted strong interest from the research community. In these techniques, each pixel is mapped onto a node in a graph. Neighboring nodes are connected by weighted edges (graphs) or hyperedges (hypergraphs [3]). Generally, the weights are fixed in such a way to reflect the similarity or dissimilarity between pixels or regions having the same visual features. Then, the graph is partitioned into multiple subgraphs with an appropriate technique. Once the graph is partitioned, a segmentation can be easily deduced by using the correspondence between nodes in the graph and pixels in the image.

1.2 Graph cuts: principle and algorithms

1.2.1 History and related work

Historically, the theory of graph cuts was first applied in Computer Vision by Greig, Porteous and Seheult in [GPS89]. Their primary interest was in assessing the performance of algorithms used to find the Maximum A Posteriori (MAP), such as Simulated Annealing (SA). In the Bayesian statistical context of image denoising, they showed how the MAP estimate of a binary image can be exactly obtained by maximizing the flow through a capacitated network with two terminals. Thus, their contribution removes any question of convergence when using iterative algorithms such as SA or Iterated Conditional Modes (ICM). The problem was therefore shown to be solved in polynomial time using a maximum-flow/minimum-cut algorithm.

Then, graph cuts stayed behind the scenes during about one decade due to the limited scope of applications of [GPS89]. Although an efficient hierarchical approach was initially

[3] A survey on hypergraphs in Image Processing is available in [BCA]. An application of hypergraphs for segmenting cerebral tissues is notably available in [RCM05b, RCM05a].

proposed in [GPS89] to speed up the maximum-flow computation, the algorithm requires excessive time yet mainly due to computer technology limitations.

In 1998, Roy and Cox relaunched the interest of graph cuts for globally solving the N-camera stereo correspondence problem in $O(n^2d^2log(nd))$, where n is the number of pixels and d is the depth resolution. The latter is transformed into a maximum-flow problem and a graph with two terminals is built. Then, the problem is solved using a standard maximum-flow algorithm [RC98]. Once solved, the minimum-cut associated to the maximum-flow yields a disparity surface for the whole image at once.

The following years yield strong theoretical results [KZ04, IG99, Ish03] in the multi-labels case. Already in 1999, Ishikawa both proved that the energies which can be minimized by graph cuts are convex and did provide a graph construction in such a situation. This result is restated in a more formal manner in [Ish03]. But mostly, graph cuts really gained popularity with the introduction of a fast maximum-flow algorithm [KZ04] and efficient multi-labels heuristics [BVZ99], both making near real-time performance for a wide range of problems in Computer Vision such as image segmentation, restoration, image registration, optical flow, stereovision, multi-view reconstruction, texture synthesis, etc [4]. These heuristics are often applied iteratively to a sequence of binary problems, usually yielding near optimal solutions [BVZ99].

In order to be more rigorous with the current literature, we shall complete the classification of [Lec09] on image segmentation techniques by adding some recent work in the continuous domain such as Normalized Cuts [SM00], Random Walker [GFL04] and Power Watersheds [CGNT09].

In order to avoid small cuts in graph cuts, Shi and Malik propose to optimize a cost function which slightly differs from the traditional minimum-cut [SM00]. First, a grid graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is built as in graph cuts but without embedding any terminals. The problem is formulated as finding a partition of nodes (A, B) of $\mathcal{V}$ in $\mathcal{G}$ by optimizing a quantity (the normalized cut) which both ensures the consistency of nodes in A and B as well as the dissimilarity of A with respect to B. Finding a normalized cut of minimal

[4] We refer the reader to [BK04] for typical applications of graph cuts.

value exactly is an NP-hard problem and the authors choose to relax it in the continuous domain. Then, a pairwise similarity matrix is built and the eigenvectors of this matrix are computed. Then, the graph is partitioned with the second smallest eigenvalue and is possibly subdivided.

As [SM00], the Random Walker algorithm [GFL04] is also formulated on a weighted graph. Assuming that the user provides K seeds, the algorithm determines labels for the unseeded nodes by assigning the pixel to the seed for which it is most likely to send a random walker. This may be also interpreted as an interactive version of the Normalized Cuts [SM00]. The key idea of this algorithm is to compute for each pixel, the probability that it first reaches each of the K seed points. Thus, a vector of size $K-1$ is assigned to each pixel (since the sum of probabilities equals to one). In fact, such an approach amounts to solve a Dirichlet problem. Computing these probabilities amounts to solving a large, sparse and symmetric linear system of equations. Once these probabilities are computed, the most likely label of each pixel p is taken as the maximum of the K probabilities of p.

Finally, the recent work of [CGNT09] unifies the graph cuts framework, the Random Walker and shortest path optimization under a common energy function. They generalize previous links established between graphs cuts and maximum spanning forests by proving that all cuts resulting of the minimization of the terms $E_{p,q}(.)$ (including Random Walker and graph cuts) converge to maximum spanning forest cuts as the edge capacities tend to infinity, under the condition that all maxima of the edge capacity function are seeded. Promising results are obtained by varying the exponent on the difference between neighboring nodes. Moreover, the method can be easily generalized to multi-labels segmentation.

In the subsequent sections, we first review the duality between maximum-flow and minimum-cut problems and describe the most popular algorithms used for solving them. Next, we explain how a pairwise energy functional can be minimized in this framework and detail two models used in image segmentation.

1.2.2 Definitions and notations

We consider a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as a set of directed edges $\mathcal{E} \subset (\mathcal{V} \times \mathcal{V})$ as well as a set of nodes $\mathcal{V} = \mathcal{P} \cup \{s, t\}$ with two terminal nodes s and t respectively named the source and the sink. Notice that the nodes $\mathcal{P} \subset \mathbb{Z}^d$ $(d > 0)$ are disposed on a lattice and usually correspond to pixels/voxels whereas the set of edges $\mathcal{E}$ represents the relations between two adjacent nodes. We also assume that for every node $p \in \mathcal{P}$,

$$(p, s) \notin \mathcal{E} \quad \text{and} \quad (t, p) \notin \mathcal{E}. \tag{5}$$

In words, no edges go back from p to s and from t to p in $\mathcal{G}$. Additionally, we split the set of edges $\mathcal{E}$ into two disjoint sets $\mathcal{E}_n$ and $\mathcal{E}_t$ denoting respectively the n-links (neighborhood links) and t-links (terminal links):

$$\begin{aligned} \mathcal{E}_n &= \{(p, q) \in \mathcal{E} \mid (p, q) \in \mathcal{P}^2\}, \\ \mathcal{E}_t &= \{(s, p) \in \mathcal{E} \mid p \in \mathcal{P}\} \bigcup \{(p, t) \in \mathcal{E} \mid p \in \mathcal{P}\}. \end{aligned} \tag{6}$$

To express a wide variety of energies, we also need to describe how each node interacts with other nodes nearby. We therefore denote the neighbors of any node $p \in \mathcal{V}$ by

$$\sigma_{\mathcal{E}}(p) = \{q \in \mathcal{V} \mid (p, q) \in \mathcal{E} \text{ or } (q, p) \in \mathcal{E}\}, \tag{7}$$

and provide $\mathcal{G}$ with a neighborhood system $\mathcal{N} \subset \mathcal{P} \times \mathcal{P}$ defined as a subset of all pixel pairs $(p, q) \in (\mathcal{P} \times \mathcal{P})$ in $\mathcal{G}$. In this context, the following neighborhoods are used

$$\begin{aligned} \mathcal{N}_0 &= \{(p, q) \in \mathcal{E} \mid \textstyle\sum_{i=1}^{d} |q_i - p_i| = 1\} \quad \text{or,} \\ \mathcal{N}_1 &= \{(p, q) \in \mathcal{E} \mid |q_i - p_i| \leq 1 \ \forall i \{1, \ldots, d\}\}, \end{aligned}$$

where p_i denotes the i^{th} coordinate of the pixel $p \in \mathbb{Z}^d$. As an illustration, each pixel has respectively 4 and 8 neighbors in 2D, 6 and 26 neighbors in 3D and finally 8 and 80 neighbors in 4D, for $\mathcal{N}_0$ and $\mathcal{N}_1$ neighborhoods respectively. In what follows, "connectivity 0" and "connectivity 1" refer to the use of $\mathcal{N}_0$ and $\mathcal{N}_1$ neighborhoods, respectively. Furthermore, we define the edge capacities as a mapping $c : (\mathcal{V} \times \mathcal{V}) \to \mathbb{R}^+$ and denote the capacity of any edge $(p, q) \in (\mathcal{V} \times \mathcal{V})$ by

$$c_{p,q} \geq 0.$$

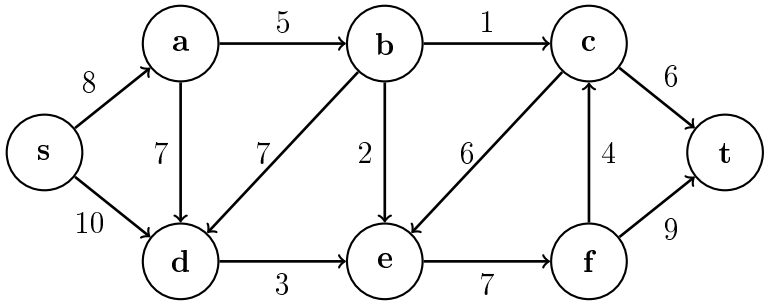


Figure 2: An example of a weighted directed graph.

Although the mapping $c(.)$ is defined for any $(p,q) \in (\mathcal{V} \times \mathcal{V})$, we always set

$$c_{p,q} = 0, \quad \text{when} \quad (p,q) \notin \mathcal{E}. \tag{8}$$

Doing so non-null capacities are only defined on existing edges. Furthermore, we assume, without loss of generality (see [KZ04]) that edge capacities are such that for every grid node $p \in \mathcal{P}$, we have

$$c_{s,p} \neq 0 \quad \Rightarrow \quad c_{p,t} = 0. \tag{9}$$

Therefore, we summarize the edge capacities of t-links and set for all grid node $p \in \mathcal{P}$

$$c_p = c_{s,p} - c_{p,t}. \tag{10}$$

In the sequel, such quantites will be called "contracted capacities" for the sake of clarity. Figure 2 shows an example of a weighted directed graph defined on a 3×2 lattice.

We now describe the notion of graph partitioning which we will always refer to as s-t cut throughout this document. This notion is formally defined in Definitions 3 and 4, and is also illustrated in Figure 3.

Definition 3 (s-t cut). *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted directed graph. An s-t cut $\mathcal{C} = (\mathcal{S}, \mathcal{T})$ in $\mathcal{G}$ is a subset of edges $F \subset \mathcal{E}$ such that, in $\mathcal{G}' = (\mathcal{V}, \mathcal{E} \setminus F)$, there is no path going from s to t and such that no proper subset of F verifies this property. The set $\mathcal{V}$ is then partitioned into two disjoints sets $\mathcal{S}$ and $\mathcal{T}$ (i.e. $\mathcal{S} \cap \mathcal{T} = \emptyset$ and $\mathcal{S} \cup \mathcal{T} = \mathcal{V}$) such that $\forall p \in \mathcal{S}$ (resp. $\mathcal{T}$), there exists a path from s to p (resp. from p to t).*

Definition 4 (s-t cut edges). *Let $\mathcal{C} = (\mathcal{S}, \mathcal{T})$ be an s-t cut of a weighted directed graph $\mathcal{G}$. Then, an edge $(p,q) \in \mathcal{E}$ is said to belong to the s-t cut $\mathcal{C}$ if and only if $p \in \mathcal{S}$ and $q \in \mathcal{T}$.*

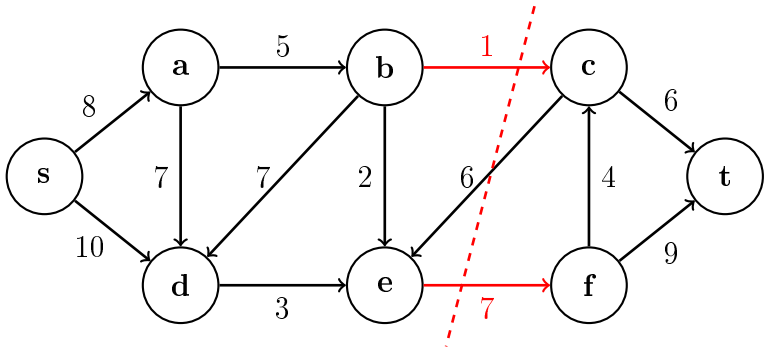


Figure 3: An s-t cut of a weighted directed graph whose weight is equal to 8. The red dashed line denotes the s-t cut $\mathcal{C}$ splitting $\mathcal{V}$ into two disjoint sets of nodes $\mathcal{S} = \{s, a, b, d, e\}$ and $\mathcal{T} = \{c, f, t\}$. Thus, $\mathcal{C}$ consists of the set of edges $\{(b, c), (e, f)\}$.

Observe that an s-t cut contains only edges going from $\mathcal{S}$ to $\mathcal{T}$. As an illustration, the edge (c, e) in Figure 3 does not belong to the s-t cut. Then, we can define the weight of an s-t cut $\mathcal{C}$ as the sum of the capacities of all edges in $\mathcal{C}$.

Definition 5 (s-t cut capacity). *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted directed graph. The capacity of an s-t cut $\mathcal{C}$ in $\mathcal{G}$ is the positive real:*

$$\mathrm{val}_{\mathcal{G}}(\mathcal{C}) = \sum_{(p,q)\in(\mathcal{S}\times\mathcal{T})} c_{p,q} = \sum_{(p,q)\in\mathcal{C}} c_{p,q}.$$

This naturally leads us to the minimum-cut problem which amounts to find an s-t cut $\mathcal{C}^*$ of minimum weight in $\mathcal{G}$ (see Definition (6)).

Definition 6 (minimum s-t cut). *A minimum-cut of a graph $\mathcal{G}$ is an s-t cut $\mathcal{C}$ of minimum weight.*

This notion is illustrated in Figure 4. Notice that the Definition 6 implies that several minimum s-t cuts can co-exist in the same graph $\mathcal{G}$.

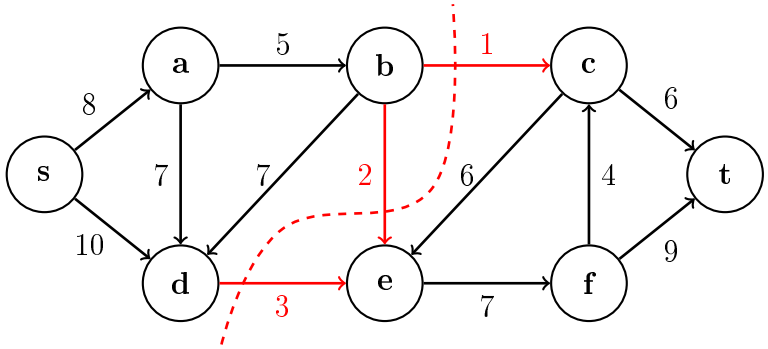


Figure 4: A minimum s-t cut $\mathcal{C}^*$ of a weighted directed graph, whose weight is equal to 6. The minimum s-t cut $\mathcal{C}^*$ consists of the set of edges $\{(b,c),(b,e),(d,e)\}$.

Although the possible number of s-t cuts grows quickly with the number of non-terminal nodes (which is bounded by $2^{\sharp\mathcal{P}}$), it is well known that the minimum s-t cut problem (or min-cut problem for short) is the dual problem of the maximum-flow problem (or max-flow problem for short) and can be solved in polynomial time. By duality, the minimum-cut can be therefore solved with the same complexity [DF56, ECS56]. This is a typical optimization problem often associated to transportation, which was thoroughly studied over the last decades in the domain of operational research.

We now review the max-flow problem and its duality with the min-cut problem. First, we define flows as any mapping $f : (\mathcal{V} \times \mathcal{V}) \to \mathbb{R}^+$. A flow is said to be valid if it fulfills the edge capacity constraint and flow conservation constraint.

Definition 7 (flow). *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph. A flow f is said to be valid if it satisfies the capacity constraint*

$$0 \leq f_{p,q} \leq c_{p,q}, \qquad \forall (p,q) \in (\mathcal{V} \times \mathcal{V}), \tag{11}$$

and if the flow conservation holds for any $p \in \mathcal{V} \setminus \{s,t\}$

$$\sum_{q \in \sigma_{\mathcal{E}}(p)} f_{q,p} = \sum_{q \in \sigma_{\mathcal{E}}(p)} f_{p,q}. \tag{12}$$

Again (8) and (11) guarantee that

$$f_{p,q} = 0, \qquad \forall (p,q) \notin \mathcal{E}. \tag{13}$$

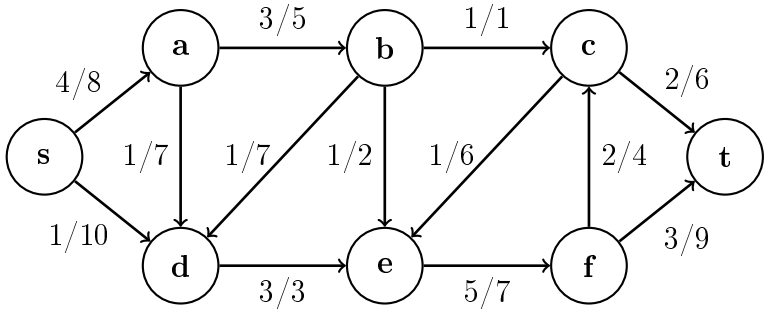


Figure 5: A valid flow on a weighted directed graph. Left numbers denote flow while right number denote edge capacities.

This is the reason why we do not clarify that $(q,p) \in \mathcal{E}$ (resp. $(p,q) \in \mathcal{E}$) in the left (resp. right) hand side sum in (12). We can now define the value of the flow f in $\mathcal{G}$ by [5]

$$\mathrm{val}_{\mathcal{G}}(f) = \sum_{p \in \sigma_{\mathcal{E}}(s)} f_{s,p}. \tag{14}$$

As for edge capacities, we also summarize the flow only passing through t-links for any grid node $p \in \mathcal{P}$

$$f_p = f_{s,p} - f_{p,t}. \tag{15}$$

Furthermore, it is easily seen that for any flow f and any $S \subset \mathcal{V}$, the flow entering S is equal to the flow leaving S:

$$\sum_{\substack{p \in S \\ q \notin S}} f_{q,p} = \sum_{\substack{p \in S \\ q \notin S}} f_{p,q}. \tag{16}$$

Considering (9), (11) and (15), we can now rewrite (16) and obtain that for any $S \subset \mathcal{P}$

$$\sum_{p \in S} f_p + \sum_{\substack{p \in S \\ q \in \mathcal{P} \setminus S}} (f_{q,p} - f_{p,q}) = 0. \tag{17}$$

As the min-cut problem, the max-flow problem consists in finding the maximum amount of flow which can be routed from s to t in $\mathcal{G}$ (see Definition 8).

Definition 8 (Maximum-flow). *A maximum-flow in a graph $\mathcal{G}$ is a valid flow of maximum value.*

[5] Notice that the same notations are used for the flow value and the s-t cut value in $\mathcal{G}$. This abuse of notation will never be ambiguous once in context.

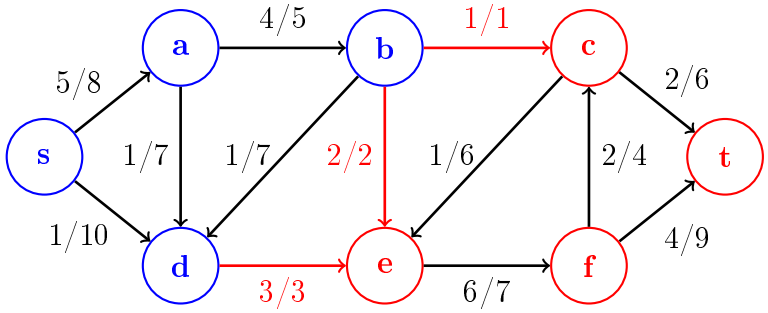


Figure 6: A max-flow in the weighted directed graph of Figure 2. Observe that the value of max-flow reached is equal to the value of the min-cut in Figure 4. Saturated edges belonging the min-cut are shown in red. Red nodes refer to nodes which can be reached from s in the residual graph $\mathcal{G}_f$ whereas blue nodes refer to the remaining nodes in the same graph. Left numbers denote flow while right number denote edge capacities.

An example of a valid flow is illustrated in Figure 5. Notice that it is not a maximum flow since some flow can be routed yet from s to t. Then, the following theorem, independently discovered by Dantzig and Fulkerson [DF56] as well as Elias, Feinstein and Shannon [ECS56] establishes the duality between the max-flow and the min-cut problems [6]. This situation is outlined in Figure 6.

Theorem 1 (min-cut and max-flow duality [DF56, ECS56]). *Let $\mathcal{G}$ be a weighted directed graph. The max-flow value in $\mathcal{G}$ is equal to the smallest value of any s-t cut dividing $\mathcal{V}$ into disjoint sets $\mathcal{S}$ and $\mathcal{T}$. Thus, we have*

$$\max_{f \in \mathcal{F}} \mathrm{val}_{\mathcal{G}}(f) = \min_{\mathcal{C} \in \mathcal{K}} \mathrm{val}_{\mathcal{G}}(\mathcal{C}),$$

where $\mathcal{F}$ denotes the set of all feasible flows in $\mathcal{G}$ and $\mathcal{K}$ the set of all s-t cuts in $\mathcal{G}$.

In the subsequent sections, we detail the most popular approaches designed for solving the max-flow / min-cut problem and explain how this technique can be used to minimize energy functionals frequently arising in Computer Vision problems.

[6] Notice that the edges belonging to $\mathcal{C}^*$ are all saturated but it can exist some saturated edges which do not belong to $\mathcal{C}^*$.

1.2.3 Maximum-flow algorithms

The past five decades have witnessed prolific developments of algorithms for solving the max-flow problem. Currently, these algorithms can be classified in three categories: feasible-flow algorithms, preflow algorithms and pseudoflow algorithms. In the subsequent, we briefly describe all these approaches.

1.2.3.1 Feasible-flow algorithms

First, we denote an s-t path from s to t in $\mathcal{G}$ by $\phi_{\mathcal{G}} = ((s,p_0),(p_0,p_1),\ldots,(p_{l-1},p_l),(p_l,t))$, where $(p_i,p_{i+1}) \in \mathcal{E}$, $\forall i \in \{0,\ldots,l-1\}$ and $(s,p_0),(p_l,t) \in \mathcal{E}^2$. We say that an edge $(p,q) \in (\mathcal{V}\times\mathcal{V})$ is saturated if $f_{p,q} = c_{p,q}$. And by extension, an s-t path $\phi_{\mathcal{G}}$ is said to be an augmenting path if all edges are non-saturated along it in $\mathcal{G}$. Standard feasible-flow algorithms typically work by pushing flow f along augmenting paths in a residual graph $\mathcal{G}_f$ (see Definition 9) until no more augmenting path is found. When such a situation occurs, it can be easily proved that the value of f reaches its maximum (see Theorem 2).

Definition 9 (Residual graph). *Let f be a valid flow in $\mathcal{G} = (\mathcal{V},\mathcal{E})$. The residual graph $\mathcal{G}_f = (\mathcal{V},\mathcal{E}_f)$ associated to the flow f in $\mathcal{G}$ is built as follows, $\forall (p,q) \in (\mathcal{V}\times\mathcal{V})$:*

- *If $f_{p,q} < c_{p,q}$, $(p,q) \in \mathcal{E}_f$ with the residual capacity $c^f_{p,q} = c_{p,q} - f_{p,q}$.*
- *If $f_{p,q} > 0$, $(q,p) \in \mathcal{E}_f$ with the capacity $c^f_{q,p} = f_{p,q}$.*

Theorem 2. *A valid flow f on a graph $\mathcal{G}$ has a maximum-flow value $\mathrm{val}_{\mathcal{G}}(f)$ if and only if there is no more augmenting paths $\phi_{\mathcal{G}_f}$ in $\mathcal{G}_f$.*

The detailed modus operandi of a feasible-flow algorithm is the following (see Algorithm 1): at the beginning of the algorithm, we set $f(u) = 0\ \forall u \in \mathcal{E}$ and build $\mathcal{G}_f$ according to the Definition 9. As a consequence, $\mathcal{G}_f$ has the same topology as $\mathcal{G}$. Notice that $\mathcal{G}_f$ can have edges that do not belong to $\mathcal{G}$. Then, at each iteration, the algorithm attempts to find a new augmenting path $\phi_{\mathcal{G}_f}$ from s to t in $\mathcal{G}_f$. If so, we compute the maximum amount of flow which can be pushed (denoted by Δf) as the minimum residual capacity along this path. Then, residual capacities on this path are decreased by a quantity Δf on

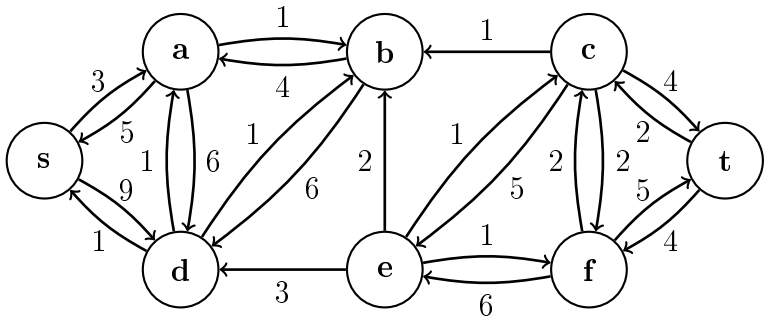


Figure 7: The residual graph of the weighted directed graph of Figure 6.

forward edges (edges oriented from s to t) and increased by Δf on reverse edges (edges oriented from t to s). This situation is illustrated in Figure 7. Thus, this path is no longer an augmenting path since at least one edge becomes saturated along it. Furthermore, the total value of flow f increases by a quantity Δf. The algorithm iterates until no more augmenting path is found in $\mathcal{G}_f$. Thus, a flow f^* of maximum value is reached when all paths $\phi_{\mathcal{G}_f}$ possess at least one saturated edge in $\mathcal{G}_f$. With such representation, $\mathcal{S}$ and $\mathcal{T}$ can be therefore easily deduced from f^*, as $\mathcal{S}$ consists of all nodes which can be reached from terminal s.

Algorithm 1 Feasible-flow algorithm.

INPUTS: A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with two terminals s and t.
OUTPUTS: f^*

1. $f(u) \leftarrow 0, \forall u \in \mathcal{E}$
2. **while** $\exists$ an augmenting path $\phi_{\mathcal{G}_f}$ in $\mathcal{G}_f$ **do**
3. $\quad \Delta f \leftarrow \min\{c^f_{i,j} \mid (i,j) \in \phi_{\mathcal{G}_f}\}$
4. $\quad$ update residual graph $\mathcal{G}_f$
5. $\quad$ increase of Δ_f units the flow on $\mathcal{E}$
6. **endwhile**
7. **return** f

However, Algorithm 1 does not explain how to select an augmenting path in the residual graph $\mathcal{G}_f$. More precisely, this crucial step significantly affects theoretical time complexity and practical efficiency of max-flow/min-cut algorithms. A typical implementation choice is to find paths either with the fewest number of edges (shortest paths) or having the maximum bottleneck capacity (fattest paths). Historically, Ford and Fulkerson were the first to design a generic labeling procedure to select augmenting paths whose

time complexity is $O(\sharp\mathcal{V}\sharp\mathcal{E}U)$, where U is the maximum edge capacity in $\mathcal{G}$ [FF56]. As an illustration, if we have $U = 2^{\sharp\mathcal{V}}$, the complexity becomes $O(\sharp\mathcal{V}\sharp\mathcal{E}2^{\sharp\mathcal{V}})$, i.e. exponential with the number of nodes over the network.

Later, Edmonds and Karp [EK72] propose a (strongly) polynomial algorithm of complexity $O(\sharp\mathcal{V}\sharp\mathcal{E}^2)$. This algorithm performs Breadth-First Search (BFS) for selecting the shortest paths from s to t in the residual graph $\mathcal{G}_f$. Dinic [Din70] lowers this time complexity to $O(\sharp\mathcal{V}^2\sharp\mathcal{E})$ as follows: when all paths of a length k are explored, the algorithm starts again a BFS from scratch by exploring the paths of length $k+1$.

Ideally, one would like to combine the benefits of shortest and fattest paths: while we want a short path to reach the sink quickly, we also want to send as much as possible flow along each path from s to t. However, combining both criteria at the same time is practically impossible. Capacity scaling offers a trade-off by relaxing the "maximum capacity" requirement and setting it for a "sufficiently large capacity" instead. The capacity scaling approach proposed in [EK72, Din70] suggests to select the shortest paths among all paths of capacity higher than some threshold. Thus, initial coarser scales focus on higher capacity paths whereas later finer scales handle remaining lower capacity paths. Selecting a good set of scales is important as it can change the complexity and the running time of the algorithm.

In [JB07], a geometrically decreasing series of thresholds like $A = \{2^{logU}, \ldots, 2^k, 1, 0\}$ is proposed for lowering the complexity of the max-flow algorithm of [BK04]. They obtain a weakly time complexity of $O(\sharp\mathcal{E}^2\sharp\mathcal{V}^2 log(U))$ instead of $O(\sharp\mathcal{E}^2\sharp\mathcal{V}^2\,\mathrm{val}_{\mathcal{G}}(f^*))$ (see below). Experiments suggest that intermediate solutions obtained at first coarse scales are of good quality when running time is a priority. The proposed algorithm also appears robust to noise. However, the role of the regularization parameter probably impacts performance but is surprisingly never discussed. Lastly, capacity scaling generalizes well to more recent methods such as preflow-push algorithms [GT88] or pseudoflow algorithms [Hoc98].

Boykov-Kolmogorov's algorithm

As we have seen before, the augmenting paths of length $k+1$ are explored from scratch as soon as those of length k are all explored in the Dinic's algorithm [Din70]. Nevertheless,

in the context of Computer Vision and Graphics, a BFS implies to scan the major part of the image pixels. This operation naturally turns out to be very costly if it is performed too often.

From here, Boykov and Kolmogorov developed an efficient algorithm [BK04] [7] which maintains two non-overlapping A and B trees respectively rooted at the source and the sink. The nodes in these trees can be either active or passive. Passive nodes represent the leaves of the tree. Other nodes are said to be free. We want also to mention that this algorithm was previously described in [Mur03]. However, since this algorithm gained popularity inside the Computer Vision community, we will refer to the work of [BK04] in the rest of this document.

The algorithm consists of three stages. During the first stage, search trees A and B grow simultaneously by acquiring children along non-saturated edges, until an augmenting path is found (growing stage). Then, the flow is pushed along this path (augmentation stage). After this stage, search trees are broken into forests because some nodes (orphans), are linked to their parent through a saturated edge. The final stage consists in finding a new valid parent for each orphan in the same search tree (adoption stage). The algorithm iterates these steps until search trees cannot grow anymore and are only divided by saturated edges.

The upper bound on the complexity given in [BK04] is $O(\sharp\mathcal{E}\sharp\mathcal{V}^2\,\mathrm{val}_{\mathcal{G}}(f^*))$. While having a worse theoretical time complexity than [Din70], this algorithm outperforms empirically other standard max-flow algorithms (such as the Dinic's [Din70] algorithm and other push-relabel algorithms) on typical vision graphs making possible near real-time performance for a wide range of applications [BK04]. Nevertheless, empirical performance of this algorithm deteriorates on denser (larger neighborhood) grid graphs [BK04] and when moving from 2D to 3D and 3D+t applications.

[7] An implementation is freely available at `http://www.cs.cornell.edu/People/vnk/software.html`.

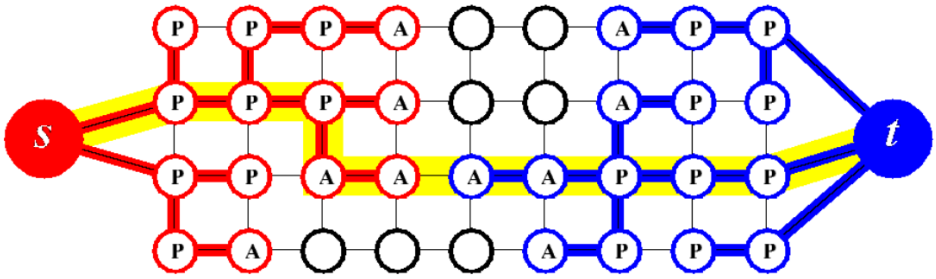


Figure 8: Representation of search trees A (red) and B (blue) at the end of the growing stage when an augmenting path (yellow line) is found. "A" and "B" labels resp. stand for active and passive nodes. Free nodes appear in black. The picture is courtesy of [BK04].

1.2.3.2 Preflow-push algorithms

Initially introduced by Goldberg and Tarjan in [GT88], the approach of the preflow-push (or push-relabel) algorithms differs a lot from the feasible-flow ones. As an illustration, such algorithms maintain a flow (called preflow) that may violate the restriction on the balance of the incoming flow and the outgoing flow into each node other than s and t by permitting excesses (more inflow than outflow). Some nodes are said to be active if the excess of flow is positive.

A labeling of nodes is maintained giving an upper bound on the distance to the sink along non-saturated edges. The algorithm tries to push the excess of flow towards the nodes having a shorter distance to the sink. This operation is typically applied to the nodes having a larger distance to the sink using a FIFO selection strategy. Thus, one tries to push the flow towards the sink at each iteration. If this operation is not possible, the node is relabeled by increasing its distance to the sink. Thus, the node is moved away from the sink and the push operation will be harder in subsequent steps.

A push-relabel algorithm iterates while a push and or a relabel step is possible. The simplest push-relabel implementation has a complexity of $O(\sharp\mathcal{V}^2\sharp\mathcal{E})$ [GT88] but a lot of heuristics have been proposed for the selection strategy in the literature for lowering this complexity.

1.2.3.3 Pseudoflow algorithms

Again, the approach differs from the feasible-flow algorithms and preflow-push ones. Introduced by Hochbaum in [Hoc98], these algorithms further relax the flow-balance constraint on each node other than s and t by permitting both excesses and deficits. Source and sink nodes have no distinguished role and all edges adjacent to the source and the sink are maintained saturated throughout the execution of the algorithm. The method seeks a partition of the set of nodes into subsets, some of which have excesses, and some have deficits, so that all edges going from excess subsets to deficit subsets are saturated. A partition with this property is provably a minimum cut. The first max-flow algorithm using pseudoflows was first proposed by Hochbaum with a complexity of $O(\sharp\mathcal{V}\sharp\mathcal{E}log(\sharp\mathcal{E}))$ [Hoc98].

1.2.4 Energy minimization via graph cuts

In the energy minimization framework, a popular strategy consists in minimizing a pairwise Markov Random Field (MRF) of the form

$$E(u) = \beta \cdot \sum_{p\in\mathcal{P}} E_p(u_p) + \sum_{(p,q)\in\mathcal{N}} E_{p,q}(u_p, u_q), \qquad \beta \in \mathbb{R}^+, \tag{18}$$

among $u \in \mathcal{L}^{\mathcal{P}}$ and where $\mathcal{N} \subset \mathcal{P} \times \mathcal{P}$ is a standard neighborhood system. As usual, in (18), the data term $E_p(.)$ is the cost for assigning the label u_p to the pixel p without regards to its neighbors. In a similar manner, the smoothness term $E_{p,q}(.)$ penalizes pixel pairs $(p, q) \in (\mathcal{P} \times \mathcal{P})$ having different labels.

Let us now describe the principle of graph cuts in the binary case, i.e. when $\sharp\mathcal{L} = 2$. First, for any s-t cut $\mathcal{C} = (\mathcal{S}, \mathcal{T})$ we define $u^{\mathcal{C}} \in \{0, 1\}^{\mathcal{P}}$ by

$$u_p^{\mathcal{C}} = \begin{cases} 0 & \text{if} \quad p \in \mathcal{T} \\ 1 & \text{if} \quad p \in \mathcal{S} \end{cases}, \qquad \forall p \in \mathcal{P}. \tag{19}$$

The founding idea of graph cuts is to build a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that for any s-t cut $\mathcal{C}$, we have

$$\text{val}_{\mathcal{G}}(\mathcal{C}) = E(u^{\mathcal{C}}) + K, \tag{20}$$

for some additional constant $K \in \mathbb{R}$, independent of $\mathcal{C}$. The min-cut in $\mathcal{G}$ corresponds therefore to a minimizer of (18) and can be efficiently computed by using an appropriate max-flow algorithm (see Section 1.2.3). When $E_{p,q}(.)$ is submodular, Kolmogorov and Zabih give a simple construction of a graph $\mathcal{G}$ satisfying (20) in [KZ04]. Therefore, they prove that (18) can be globally minimized (see Theorem 3).

Theorem 3 (binary graph representability [KZ04]). *Let E be an energy function of binary variables. Then, E is graph-representable if and only if each term $E_{p,q}(.)$ is submodular, i.e. satisfies*

$$E_{p,q}(0,0) + E_{p,q}(1,1) \leq E_{p,q}(0,1) + E_{p,q}(1,0).$$

The Theorem 3 is also extended by [KZ04] for energy functions handling pairwise terms and terms composed of three variables. Later, these results were further generalized by Freedman and Drineas in [FD05] for energy functions embedding any number of variables.

When $\sharp\mathcal{L} > 2$, the problem of finding the global minimum of (18) is intractable. Observe first that the space of labelings becomes huge as $\sharp\mathcal{L}$ and $\mathcal{P}$ grow since the number of labelings is equal to $\sharp\mathcal{L}^{\sharp\mathcal{P}}$. Even worse, finding the global minimum of (18) is known to be a NP-hard problem. Hence, any general-purpose energy minimization algorithm will require an exponential time. When $E_{p,q}(.)$ is convex, Ishikawa however proved that (18) can be globally minimized [IG99, Ish03] when $\mathcal{L}$ is ordered (see Theorem 4). Again, a graph construction is also detailed in [IG99, Ish03] for such cases. Furthermore, notice that the topology of $\mathcal{G}$ depends both on the form of $E_{p,q}(.)$ and on the number of labels. Remark also that the space of all labelings $\mathcal{L}^{\mathcal{P}}$ can be relatively large in practice and the difficulty when solving a particular vision problem is generally associated to the underlying number of labels. As an illustration, a typical segmentation problem requires at maximum a dozen of labels whereas optical flow or restoration requires hundreds of them.

Theorem 4 (multi-labels graph representability [Ish03]). *Let E be a general energy function defined on a label set $\mathcal{L}$ ordered. Then, E is graph-representable if and only if $E_{p,q}(.)$ is convex with respect to $\mathcal{L}$, $\forall (p,q) \in \mathcal{E}$.*

Convex smoothness terms $E_{p,q}(.)$ are however usually known to preserve discontinuity

less efficiently than non-convex ones. Optimizing the corresponding energy functional leads to solutions with over-smoothed borders. In the non-convex case, some amount of work has been done in the binary case [RKLS07] and in the multi-labels case [BVZ99]. For instance, authors of [BVZ99] designed heuristics like α-expansion or α-β swap offering good performance for a large number of problems. In the next section, we detail the graph cuts framework applied to image segmentation.

1.2.5 Binary graph cuts-based segmentation

Consider an image $I : \mathcal{P} \subset \mathbb{Z}^d \to [0,1]^c$ ($d > 0$, $c > 0$) as a function, mapping each point (called pixel) $p \in \mathcal{P}$ to a value $I_p \in [0,1]^c$. We define a binary segmentation as a mapping u assigning each element of $\mathcal{P}$ with the value 0 for the background and 1 for the object and we write $u \in \{0,1\}^{\mathcal{P}}$. In the image segmentation context, a popular strategy consists of minimizing an MRF of the form [BJ01]

$$E(u) = \beta \sum_{p \in \mathcal{P}} E_p(u_p) + \sum_{(p,q) \in \mathcal{N}} E_{p,q}(u_p, u_q), \tag{21}$$

among $u \in \{0,1\}^{\mathcal{P}}$ and for a fixed $\beta \in \mathbb{R}^+$. While the data term $E_p(.)$ ensures proximity to initial data, the smoothness term $E_{p,q}(.)$ assumes that the boundaries of the segmentation are smooth. The latter is typically used to better align boundaries of the segmentation on the image edges based on visual features such as intensity, gradient direction or texture information. In this setting, nodes usually correspond to image pixels/voxels while n-links reflects similarity between neighboring nodes. Once a min-cut $\mathcal{C}^*$ is computed, the segmentation readily corresponds to (19). In the next subsequent sections, we briefly review two popular energy functionals used in image segmentation.

1.2.5.1 TV+L^2 energy model

Initially introduced by Rudin, Osher and Fatemi [ROF92], the TV+L^2 (ROF) model and its variants have been a very active research topic in image restoration. This model has also successfully demonstrated its efficiency for segmenting cars in video [RCD07]. It is only defined for grayscale images but can of course be applied to a grayscale image

resulting from a multichannel image. In the image segmentation context, the segmentation is taken as a level-set of the minimizer u^* of

$$\underbrace{\sum_{\mu=0}^{L-2} \sum_{(p,q)\in\mathcal{N}} w_{p,q}|u_p^\mu - u_q^\mu|}_{TV(u)} + \beta\|u - I\|_2^2, \qquad \beta \in \mathbb{R}^+, \tag{22}$$

where L denotes the maximum intensity of I, $\|.\|_2$ denotes the Euclidean distance in $\mathbb{R}^{\sharp\mathcal{P}}$, $I \in \mathbb{R}^{\mathcal{P}}$ is initial data, and $TV(u)$ denotes the Total Variation of $u \in \mathbb{R}^{\mathcal{P}}$. While the second term maintains a proximity to a level-set of I, the solution is regularized by the first one. Expressing the two terms of (22) in terms of level sets, we observe that the μ level set of u^* is a minimizer of the binary energy

$$\underbrace{\sum_{(p,q)\in\mathcal{N}} w_{p,q}|u_p^\mu - u_q^\mu|}_{TV(u^\mu)} + 2\beta \sum_{p\in\mathcal{P}} u_p^\mu[(\mu - \frac{1}{2}) - I_p] + I_p, \tag{23}$$

among $u^\mu \in \{0,1\}^{\mathcal{P}}$ and where the weight $w_{p,q}$ is proportional to the Euclidean distance between p and q (see [DS04]). The latter problem has the form described in (18) and can be minimized by a graph cut. Remind that this formulation cannot handle color images. In practice, color images need to be converted into grayscale images before they are segmented.

1.2.5.2 Boykov-Jolly energy model

In [BJ01], Boykov and Jolly introduced another energy model for segmenting images using graph cuts. Unlike the $TV + L^2$ model, the user must provide object ($\mathcal{O} \subset \mathcal{P}$) and background seeds ($\mathcal{B} \subset \mathcal{P}$) in an interactive fashion (see Figure 9). The role of these seeds is twofold: reducing the cuts space by adding hard constraints and computing probability distributions laws of the object and the background. Formally, we have:

$$\begin{cases} E_p(1) = -log\ \mathbb{P}(I_p \mid p \in \mathcal{O}) \\ E_p(0) = -log\ \mathbb{P}(I_p \mid p \in \mathcal{B}) \end{cases} \quad \text{and} \quad E_{p,q}(u_p, u_q) = B_{p,q}|u_p - u_q|, \tag{24}$$

where $\mathbb{P}(.)$ is a probability density function, $I_p \in [0,1]^c$ denotes the intensity at pixel p and $B_{p,q}$ is a weighting function used to map similarity between pixels to edge capacities.

Figure 9: An example of segmentation (right) of an image (left) using the Boykov-Jolly model [BJ01]. The seeds as well as the segmentation are superimposed on the image by transparency.

The energy (24) encourages coherence in regions of similar intensity where the data term favors the belonging of each pixel to the object or to the background while the smoothness term penalizes neighboring pixels having different labels [8]. In its simplest form, the amount of penalization is determined by the gradient and favors boundaries with a strong gradient. Nevertheless, the weight can also embed more complex features such as gradient direction or texture information via tensors or Gabor filters.

The most common choices for these weighting functions come from the influential work of Perona and Malik on anisotropic diffusion [PM90] and are now used by almost every graph-based segmentation algorithms

$$\text{Gaussian:} \quad B_{p,q} = \frac{1}{\|p-q\|_2} exp\Big(- \frac{\|I_p - I_q\|_2^2}{2\sigma^2} \Big), \tag{25}$$

$$\text{Reciprocal:} \quad B_{p,q} = \frac{1}{\|p-q\|_2} \frac{1}{1+\|I_p - I_q\|_\omega^\omega}, \tag{26}$$

where $\sigma \in \mathbb{R}^+$, $\omega > 1$ represent free parameters, $\|.\|_2$ is the Euclidean norm (either in $\mathbb{R}^d$ or $\mathbb{R}^c$) and $\|.\|_\omega$ is the ℓ^ω norm. Notice that some work has been recently done to measure the impact of these functions on the segmentation results in a medical context [GJ08]. The numerical results highlight that (26) outperforms (25) in terms of both absolute performance achieved on segmentation differences and stability over β values. Since we were aware of the work [GJ08] only recently, all experiments presented in the sequel of this document use the weighted Gaussian function (25).

[8] When the parameter $\beta = 0$, notice that (24) corresponds to the Ising model.

Remark 1. *It is well established that the quality of solutions obtained by graph cuts methods depends both on the graph structure and on the choice of edge capacities. This observation is empirically confirmed in [GJ08]. More precisely, graph cuts may produce noticeable metrication/geometric artifacts [BK05] and blocky structures due to the discrete topology of graphs. A common way for reducing this effect is to increase the size of* $\sigma_{\mathcal{E}}(p)$, $\forall p \in \mathcal{P}$ *(see* (7)*) and choose the edge capacities of* $\sigma_{\mathcal{E}}(p)$ *to better approximate the length of the s-t cut crossing all edges in* $\sigma_{\mathcal{E}}(p)$.

A standard approach to such problem is to use the inverse Euclidean distance between two adjacent nodes as in (25) *and* (26). *Thus, the larger the distance between two nodes is, the smaller the edge capacity connecting them is.*

The choice of these edge capacities is justified in a more formal context in [BK05]. In fact, any s-t cut on a graph embedded in some continuous space can for instance be interpreted as a contour (2D) or a surface (3D). In [BK05], the authors explain how to set the edge capacities in a graph so that the cost of cuts is arbitrarily close to the length of the corresponding contours for any anisotropic Riemannian metric using results from integral geometry.

Also, the distributions of intensities of the object and the background in (24) are generally estimated using either Normalized Histogram (NH) [RKB04]) or Gaussian Mixtures Model (GMM) [RKB04]. We also want to mention the work of [VKR09, RKB04] where authors try to optimize both segmentation and appearance model using either an Expectation-Maximization (EM)-style algorithm [RKB04] or dual decomposition [VKR09]. While the present work can be easily embedded into these approaches, we only deal with a fixed appearance model constructed either from seeds (see the Boykov-Jolly model (23)) or not (see the TV+L^2 model (24)).

Better embedding color information in (24) is a possible way to improve segmentation results. The use of the Red Green Blue (RGB) space for representing image data is very common in Image Processing and Computer Vision, mainly dictated by the availability of such data as produced by digital cameras and camcorder. However, this space is not perceptually uniform in the sense that difference between colors in it does not correspond

to color differences as perceived by humans. Furthermore, the improved performance of non-uniform (CIE Lab and CIE Luv color spaces) or approximately non-uniform spaces over uniform ones (RGB color space) is well established for image segmentation [ZN10] and texture analysis [Pas01].

1.3 Estimating distribution laws

In what follows, we briefly review two standard ways to estimate the distribution laws $\mathbb{P}(I_p \mid p \in \mathcal{O})$ and $\mathbb{P}(I_p \mid p \in \mathcal{B})$ in the Boykov-Jolly model (see Section 1.2.5.2) in the discrete domain (NH) and in the continuous domain (GMM).

1.3.1 Normalized histograms

NH are a popular procedure to approximate a probability density function by a piecewise constant function made of multiple squares. Each of these squares is generally called a bin. Since we use the same strategy for the object and the background, we only describe it for $\mathcal{B}$ for the sake of clarity.

First, let $N_b \in \mathbb{N}^*$ denote the number of bins. Then, we call for $q \in \{0, \dots, N_b - 1\}^c$,

$$H_q = \frac{\sharp\{p \in \mathcal{B} \mid \frac{q_i}{N_b} \leq (I_p)_i < \frac{q_i+1}{N_b}, \forall 1 \leq i \leq c\}}{\sharp\mathcal{B}},$$

where we remind that $I_p \in [0,1]^c$ is the intensity at the pixel p, $(I_p)_i$ and q_i are the i^{th} channel of I_p and q, respectively. Since NH are known to be noise-sensitive, we choose for any $p \in \mathcal{P}$, to estimate $\mathbb{P}(I_p \mid p \in \mathcal{B})$ by

$$(G_{\sigma'} * H)_{I_p}$$

where $G_{\sigma'}$ is a Gaussian kernel of standard deviation σ'. In this document, we will always set $\sigma' = 1$ and use the same number of bins N_b for the object and the background.

Notice that, as it is well known, when the number of bins N_b is too large, H_q is null for most $q \in \{0, \dots, N_b - 1\}^c$. Such observation grows as the number of channels c increases. As a result, $\mathbb{P}(I_p \mid p \in \mathcal{B})$ is not accurately estimated and most contracted capacities of

the graph are set to 0. The learned distribution law overfits the samples. In practice, the model behaves as if we had $\beta = 0$. On the other hand, when N_b is too small, the best possible estimate approximates $\mathbb{P}(I_p \mid p \in \mathcal{B})$ by a piecewise constant function made of large square pieces. This time, H_q is not null for a larger part of $q \in \{0, \ldots, N_b - 1\}^c$ but $\mathbb{P}(I_p \mid p \in \mathcal{B})$ is roughly approximated. Therefore, the number of bins N_b should be a trade-off between these two situations. In practice, we adapt the number of bins to the number of channels. We empirically choose a number of cells $N_b = 256$ and $N_b = 50$ for grayscale and color images, respectively. Smoothing distributions allows to further increase the number of cells where H_q is not null and can further reduce the size of the graph.

1.3.2 Gaussian mixtures model

GMM estimates and approximates a probability density function by a sum of Gaussian functions. Then, the problem of estimating this function amounts to determine the parameters of each Gaussian (i.e. covariance and expectation), the number of Gaussians (called number of components [9]) and the participation of each one in the mixture (called mixture weights). In addition, in a Bayesian setting, notice that the mixture weights and parameters will themselves be random variables and prior distributions will be placed over the variables. Again, since we use the same strategy for the object and the background, we only describe it for $\mathcal{B}$ for the sake of clarity. For any pixel $p \in \mathcal{P}$ and a fixed number of Gaussians N_g in the mixture, the probability density function $\mathbb{P}(I_p \mid p \in \mathcal{B})$ is estimated as

$$\mathbb{P}(I_p \mid p \in \mathcal{B}) = \sum_{i=1}^{N_g} w_i h(I_p \mid \mu_i, \Sigma_i), \tag{27}$$

where we remind that $I_p \in [0,1]^c$ is the intensity of pixel p, w_i, $\forall i = 1, \ldots, N_g$ are the mixture weights, and $h(I_p \mid \mu_i, \Sigma_i)$, $\forall i = 1, \ldots, N_g$ are the Gaussian functions. Each Gaussian function $h(.)$ is of the form

$$h(I_p \mid \mu, \Sigma) = \frac{1}{(2\pi)^{c/2} det(\Sigma)^{1/2}} \cdot exp\Big\{ \frac{-(I_p - \mu)^T \Sigma^{-1} (I_p - \mu)}{2} \Big\},$$

[9] In clustering, this can be also referred as clusters.

with mean vector μ and covariance matrix Σ and where $(.)^T$, $det(.)$ respectively stand for the transpose operator and the determinant. The mixture weights also satisfy the constraint that

$$\sum_{i=1}^{N_g} w_i = 1.$$

Thus, the GMM is fully parameterized by the mean vectors μ_i, the covariance matrices Σ_i and mixture weights, for all Gaussian functions. There exists several variants of this problem where covariance matrices Σ_i can be full rank or diagonal (e.g. for favoring a particular channel), parameters are shared among the Gaussian components with a common covariance matrix or not, etc. In the sequel, we will consider that covariance matrices are not shared and always of full rank. Notice that GMM is useful for a variety of applications including texture and multispectral image segmentation.

Although the problem of estimating Gaussian functions seems particularly hard to solve due to the large number of parameters, one can usually use an EM algorithm to find a good approximation of (27) with a priori given number of components. Initially proposed by Dempster *et al.* in 1977 [DLR77], the EM algorithm alternates an Expectation step (E-step) and a Maximization step (M-step) until some convergence threshold is reached. In the E-step, the parameters are fixed and the expectation of the likelihood is computed by taking into account the latest observed variables. During the M-step, the variables are fixed and the algorithm tries to find the parameters which maximize the likelihood found at the E-step. Then, we use the parameters found during the M-step as a starting point of a new evaluation of the likelihood in the E-step. In its simplest form, some analogy can be established between the EM algorithm with the clustering algorithm K-means.

Furthermore, notice that the problem of estimating the parameters of Gaussians and mixture weights appears more difficult when the number of components N_g also needs to be determined. For the experiments presented in Chapter 3, we use the work of [Bou97] which automatically estimates the number of components N_g with a statistical criterion called Minimum Description Length (MDL). According to [Bou97], this is equivalent to maximum likelihood estimation when the number of components is fixed, but in addition it allows the latter to be more accurately estimated.

2 Reducing graphs in graph cuts optimization: state-of-the-art

2.1 Motivation

While graph cuts behave successfully well on a large number of applications, the memory usage for solving large-scale optimization problems can be prohibitive since underlying graphs can contain billions of nodes and even more edges. As an illustration, the latest version of the max-flow algorithm of [BK04] allocates $25\sharp\mathcal{P} + 16\sharp\mathcal{E}_n$ bytes [1] for segmenting a multi-dimensional image. In Table 1, we observe that for a fixed amount of RAM of 2Gb, the maximum volume size for which the corresponding graph fits in central memory, decreases quickly as the dimension d increases. This experiment gives us a better idea of the limitation of graph cuts in the context of image segmentation. Beyond these limits, other strategies must be considered.

To get round the problem of memory consumption with graph cuts, some amount of work has been recently done in this direction with heuristics [LSTS04, SDB07, CA08, SD07, LSGX05, SG06, KLR10] and exact methods [LB07, SK10].

The purpose of this section is to establish a detailed state-of-the-art on techniques for reducing graphs involved in binary graph cut optimization by putting ahead the advan-

[1] Remind that $\mathcal{P}$ is the set of pixels/voxels and $\mathcal{E}_n$ denotes the set of n-links (see (6)).

	Connectivity 0	**Connectivity 1**
2D	6138	4912
3D	308	209
4D	70	42
5D	28	16

Table 1: Maximum size of a square image on the side for which the corresponding graph fits in 2GB of RAM.

tages and the limitations of each technique. We propose to review these methods in the following way: single-machine algorithms and parallelized/distributed. In what follows, we first review the former and then the latter.

2.2 Sequential strategies

To our best knowledge, Li *et al.* seem to be the first ones to tackle the problem of memory consumption of graph cuts [LSTS04]. Their algorithm works as follows. First, the image is partitioned into small and numerous homogeneous regions thanks to a low-level segmentation algorithm such as watershed [LSTS04, SDB07] or mean shift [CA08]. A region adjacency graph is produced where each region corresponds to a node in the graph (see Figure 1). Then, the max-flow is computed on this graph for getting the segmentation. The underlying assumption is that the final contours are embedded into the pre-segmentation. While this observation is generally not theoretically guaranteed, it is often verified when working on natural images not corrupted by noise. Although this approach drastically reduces the computational burden of graph cuts (about 6x faster according to [LSTS04]), the results strongly depend on the low-level segmentation algorithm used and its noise-sensitivity. Moreover, as fairly observed in [SDB07], this approach generally gives better results when over-segmentation occurs, which looses the main benefit of such a reduction.

Others have also reported band-based heuristics using a multi-resolution scheme [SG06, LSGX05, KLR10]. The principle is to segment a low-resolution image/volume and propagate the solution to the finer level by only building the graph in a narrow band surrounding the interpolated foreground/background interface at that resolution. More specifically, the acceleration strategy consists of three stages (see Figure 2): first, a pyramid of images is built with a coarsening operator (coarsening). Next, the coarsest image is segmented and its contours are extracted (segmentation at coarsest level). Finally, the contours are dilated and interpolated at the next higher resolution for building a new reduced graph (uncoarsening). This process continues until the bottom of the pyramid is reached. Such

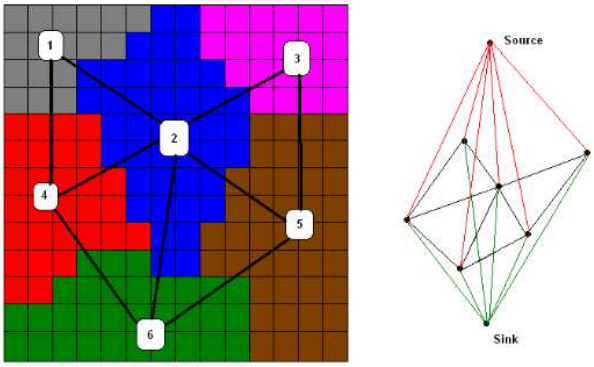


Figure 1: Working scheme of the heuristic using region adjacency graphs. A region adjacency graph (right) is built from a pre-segmentation (left) obtained from a low-level segmentation algorithm. Then, the max-flow is computed inside this graph for getting the final segmentation. The picture is courtesy of [SD07].

an approach greatly reduces time and memory consumption of standard graph cuts (about 8x faster and 4x less memory according to [SG06]). Nevertheless, it generally fails to recover thin structures and is limited to the segmentation of roundish objects. In medical imaging, this is a real drawback since elongated structures like blood vessels are ubiquitous. Moreover, the parameter controlling the band dilation during the projection, plays an important role. Indeed, one usually needs this parameter to be large enough to fully capture details of various shapes complexities. On the other side, wider bands reduce the computational benefits and may also introduce potential outliers far away from the desired object contours.

To avoid the loss of details, Lombaert *et al.* [LSGX05] used the information from a Laplacian pyramid. At each level, the bands are extended by including pixels whose value significantly differs between the image and the "coarsened-uncoarsened image". The idea is to capture thin structures which are not visible in the coarse image. This inclusion is controlled by a thresholding parameter which provides a smooth transition between [LSGX05] and traditional graph cuts. Although the previous problem is notably reduced, it is still present for low-contrasted details.

In [KLR10], Kohli *et al.* proposed recently a finer band-based technique using the multi-resolution scheme proposed in [SG06, LSGX05]. In contrast with [SG06, LSGX05],

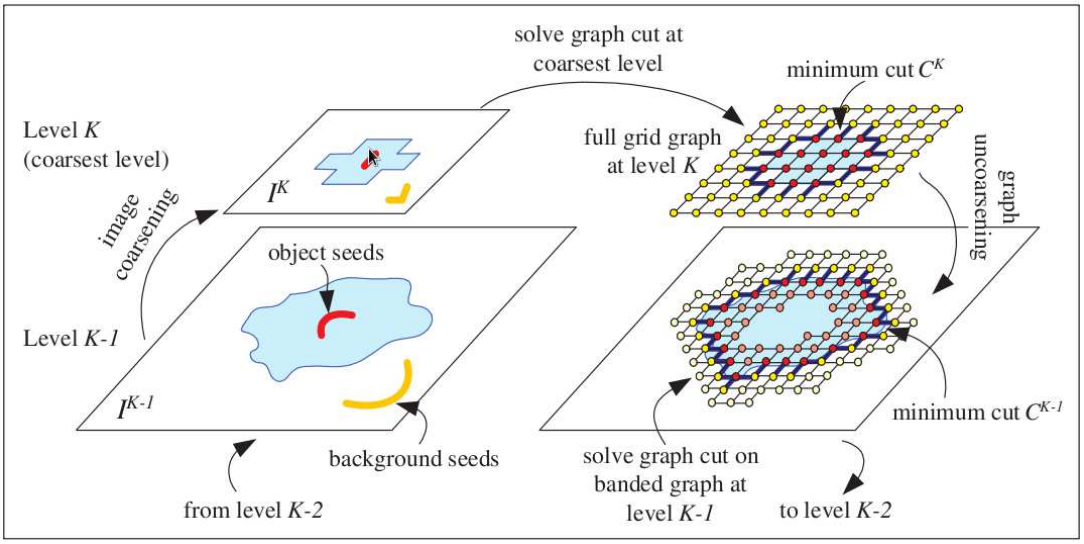


Figure 2: Multi-resolution heuristics principle. A low-resolution image is first segmented and the solution is propagated to the finer level by only building the graph in a narrow band surrounding the interpolated foreground/background interface at that resolution. The picture is courtesy of [LSGX05].

they first define an energy from the full resolution image instead of the low resolution image. Experiments show that this strategy results in significant improvements in both time and segmentation accuracy. But mostly, they compute uncertainty estimates using min-marginals [2] and use them to determine which regions belong to the reduced graph. While their algorithm reaches low pixel errors using only a few variables, this heuristic does not ensure to retrieve thin structures and details as in [SG06, LSGX05].

Finally, Lempitsky and Boykov presented an interesting touch-expand algorithm that is able to minimize binary energy functions with graph cuts in a narrow band, while ensuring the global optimality on the solution [LB07]. The principle is to make a band evolve around the object to segment by expanding the band when the min-cut touches its boundary. This process is iterated until the band no longer evolves. Although the algorithm quickly converges toward the global optimal solution, it strongly depends on

[2] The min-marginal encodes the confidence associated with a variable being assigned to the label in the optimal solution. The min-marginal of a variable x corresponds to the energy obtained by fixing it to a particular label and minimizing over all remaining variables. The exact min-marginals can be determined exactly and efficiently by reusing previous max-flow computations [KT08].

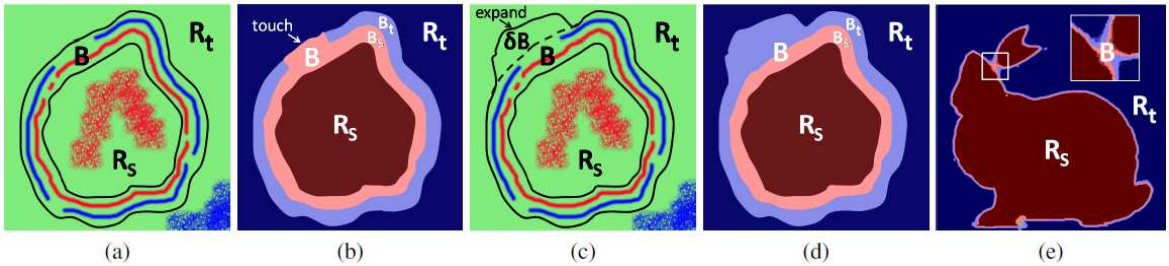


Figure 3: Touch-Expand algorithm maintains a band B and subgraphs R_s and R_t as in (a) where red (resp. blue) pixels/nodes are linked to terminal s (resp. t). If a min-cut in the band B results in a touch as in (b), then B is expanded (c). If not as in (d), the min-cut is guaranteed to be the same than on the whole graph and the algorithm stops. (e) shows a band at convergence for a real problem. The picture is courtesy of [LB07].

the initialization and no bound on the band size is given. Thus, the band can progressively increase to encompass the whole volume in the worst case. However, depending on the initialization, the bands are reasonably small in the context of [LB07] (volume reconstruction). As far as we know, this strategy has not yet been adapted to image segmentation. In particular, the benefit of this strategy strongly depends on the design of an initial band.

2.3 Parallel/distributed strategies

In a recent paper, Delong and Boykov design a method for solving the max-flow problem for graphs which do not fit in memory. They propose a new parallelized max-flow algorithm yielding near-linear speedup with the number of processors [DB08]. As an illustration, on a standard computer, segmenting a volume of size $512 \times 512 \times 256$ takes about 100 secs on a single core against less than 20 secs on eight cores. However, numerical experiments also show that the acceleration of this scheme is very limited since it is sensitive to the amount of physical memory. Furthermore, the proposed algorithm clearly remains less efficient on small graphs than standard graph cuts and can only be applied to grid-like graphs.

More recently, Strandmark and Kahl in [SK10] introduced an original approach for

minimizing binary energy functions in a parallelized/distributed fashion using the max-flow algorithm of [BK04]. The idea is to decompose the original problem into optimizable sub-problems, solve them independently and update them according to the results of the adjacent problems. This process is iterated until convergence. The key point of this method is that optimality on the solution is guaranteed by dual decomposition.

More precisely, the final solutions of the sub-problems are constrained to be equal on an overlap. They solve the original problem by converging Lagrangian multipliers associated to equality constraints on the overlap. This max-min problem is solved by alternating minimization over its primal variables and maximization over its dual variables (the Lagrangian multipliers). The minimizations are done independently of each other on the calculus nodes. The maximization combines the results obtained on the overlapping bands. It consists in an update of the dual variables. To reflect this change, the weights in the graphs corresponding to the sub-problems are modified and the corresponding solutions are recomputed. This scheme is repeated until the solutions of the variables on the overlap are equal. This iterative scheme is efficient since only a few edge costs change between iterations and then search trees can be efficiently reused [KT07]. Moreover, the number of edge costs which change decreases as the number of iterations increases.

Notice that this technique is quite general and can be either solved in parallel on the same computer or across several ones over a network. An example of communication is illustrated in Figure 4 between four computers, each one being assigned with a quarter of the input image.

Experiments in [SK10] for image segmentation and stereo clearly demonstrate both faster processing on multi-core computers and the ability to solve large scale problems over a distributed network. As an illustration, such an approach is able to segment a graph requiring 131GB of memory in 38 secs. To our best knowledge, the proposed work is the first to segment 4D volume data of moderate size using graph cuts while keeping optimality on the solution. Furthermore, in the image segmentation context, the algorithm is stable over a large range of values of the regularization parameter. Nevertheless, the algorithm is slower for solving some instances where the object to segment is not uniformly spread

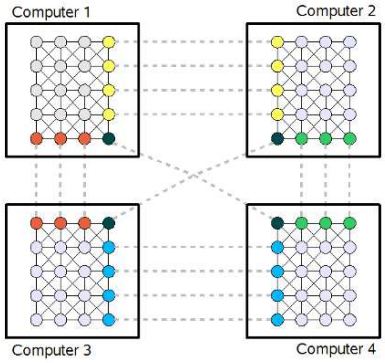


Figure 4: An example of communication between four computers for segmenting a 2D image using [SK10]. Here, the initial graph is split into four parts, each one being assigned to one computer. Color nodes correspond to the overlap where the solutions are constrained to be equal and communicate to their neighbors across the network. The picture is courtesy of [SK10].

over the image. Also, notice that the proposed strategy is only effective for graphs for which the max-flow algorithm of [BK04] is. In particular, the latter becomes less effective than a push-relabel algorithm for dense graphs.

2.4 Conclusion

The current state-of-the-art showed us that interesting heuristics [LSTS04, SDB07, CA08, SD07, LSGX05, SG06, KLR10] have been already designed for reducing the graphs. In words, the work of [KLR10] significantly improves the results of [LSTS04, SDB07, CA08, SD07, LSGX05, SG06] by keeping low pixel error on segmentations. Nevertheless, these methods get easily trapped in a local minimum of the energy losing the main benefit of global optimization.

Recently, more efforts have been concentrated to propose methods guaranteeing the optimality on solutions and showing promising results [LB07, SK10]. While the former is purely dedicated to shape fitting applications, the latter does not look for reducing graphs but instead pushes the graph cuts limits away by splitting the problem into small sub-problems. We now present a new band-based strategy for reducing graphs.

3 Reduced Graph Cuts: a finer approach for reducing graphs

3.1 General principle

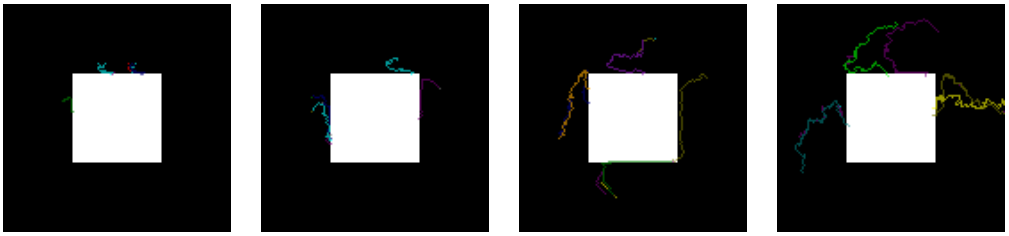
As discussed in Section 2.1, the memory consumption for segmenting high-resolution data using graph cuts can be prohibitive (see [1, 4]). Nevertheless, as explained in [GY09], the running time of the max-flow computation increases with the amount of regularization (i.e. when β is low in our situation). Indeed, a small value of β decreases t-links capacities out of s and into t by a factor proportional to β. Then, reduced t-links capacities alleviate the total amount of flow into the graph. This situation makes n-links to be less likely to be saturated and allows the max-flow algorithm to have more and longer augmenting paths, each carrying less flow on average from s to t.

The effect of varying β on the length of augmenting paths is highlighted in Figure 1 for segmenting a square image with a TV+L^2 model in connectivity 1. In this experiment, the max-flow algorithm of [BK04] is used. On each image, the twenty longest augmenting paths are colored and superimposed on the image. We clearly see that when β decreases, the augmenting paths become longer and more sinuous. Notice also that the generated paths strongly depend on the underlying max-flow algorithm used.

In light to the previous experiment, we can reasonably think that, when the amount of regularization is low, most of the nodes in the graph are useless during the max-flow computation since not traversed by any flow (see Figure 2). With this in mind, one would like to extract the smallest possible graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ from $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ while keeping the max-flow value $\mathrm{val}_{\mathcal{G}'}(f'^*)$ in $\mathcal{G}'$ identical or very close to the max-flow value $\mathrm{val}_{\mathcal{G}}(f^*)$ in $\mathcal{G}$. More formally, we want to minimize the relative size of the reduced graph defined as

$$\rho = \frac{\sharp\mathcal{V}'}{\sharp\mathcal{V}}, \tag{28}$$

under the constraint that $\mathrm{val}_{\mathcal{G}'}(f'^*) \simeq \mathrm{val}_{\mathcal{G}}(f^*)$. Since several segmentations can exist



$\beta = 0.02$ $\beta = 0.01$ $\beta = 0.005$ $\beta = 0.0025$

Figure 1: Influence of β on the length of augmenting paths for segmenting a synthetic image with a TV+L^2 model in connectivity 1. From left to right: the twenty longest augmenting paths are colored and superimposed on the image for a varying value of β.

for the same flow f'^*, we also want the segmentations with and without reduction to be identical or very near. In fact, this is an ideal optimization problem which we will not try to solve since the method for determining $\mathcal{G}'$ also needs to be (very) fast. In order to represent the potential of this idea, we again take up an experiment of [4] where we represent on the middle image of Figure 2, the flow only passing through the t-links when computing the segmentation of the image of Figure 2 with the TV+L^2 model (see Section 1.2.5.1). As with the previous experiment, the max-flow algorithm of [BK04] is again used. In the middle image of Figure 2, light gray pixels (resp. dark gray pixels) indicate that a positive amount of flow passed from the source s to a node p (resp. from a node p to the sink t), for any pixel $p \in \mathcal{P}$. Similarly, we represent on the right image of Figure 2 the outflow only passing through n-links using the same model and parameters. This time, the gray is proportional to the sum of the flow leaving any node p through n-links. For the middle and the right images, gray (resp. black) areas correspond respectively to the nodes not traversed by any flow in the graph. Clearly, only a small part of the nodes is used during the max-flow computation.

Finally, due to the nature of the problem treated, it can be easily proved that the value of the min-cut in $\mathcal{G}'$ must be less or equal than the value of the min-cut in $\mathcal{G}$ when one (see Proposition 1) or multiple edges (see Proposition 2) are removed from $\mathcal{G}'$. By duality of the max-flow problem, we therefore must have

$$\text{val}_{\mathcal{G}'}(f'^*) \leq \text{val}_{\mathcal{G}}(f^*),$$

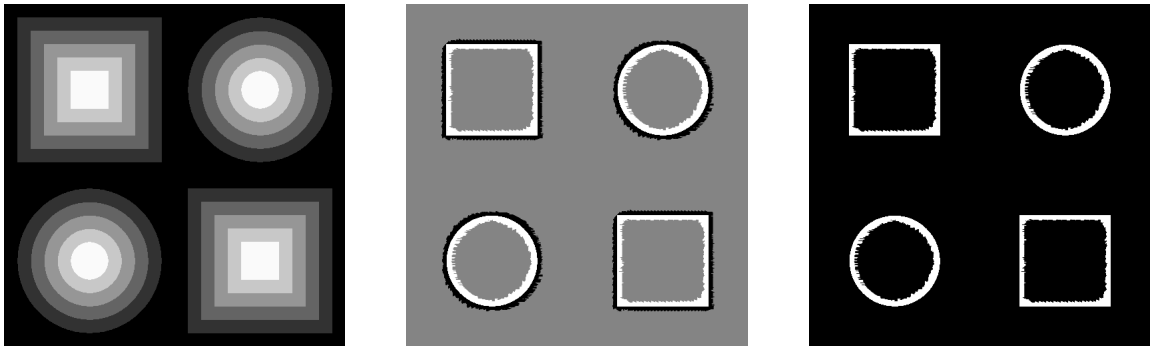


Figure 2: Illustration of the flow passing through t-links (middle) and n-links (right) for segmenting a synthetic 2D image (left) using a TV+L^2 model. On the middle image, light gray pixels (resp. dark gray pixels) indicates that a positive amount of flow passed from s to p (resp. from p to t). On the right image, the gray is proportional to the sum of the flow leaving any node p. On the middle and the right images, gray (resp. black) areas correspond respectively to the nodes not traversed by any flow in the graph.

where we remind that f'^* and f^* denote max-flows in the graph $\mathcal{G}'$ and $\mathcal{G}$, respectively.

Proposition 1. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted directed graph and $\mathcal{C}^*$ a min-cut in $\mathcal{G}$. Let $\mathcal{C}'^*$ be a min-cut of $\mathcal{G}' = (\mathcal{V}, \mathcal{E} \setminus (p,q))$ where $(p,q) \in \mathcal{E}$. Then, we have:* $\text{val}_{\mathcal{G}'}(\mathcal{C}'^*) \leq \text{val}_{\mathcal{G}}(\mathcal{C}^*)$.

Proof.

- Suppose that (p,q) belongs to the s-t cut $\mathcal{C}^*$. Thus, it is straightforward to see that for every min-cut $\mathcal{C}'^*$ in $\mathcal{G}'$, we have

$$\text{val}_{\mathcal{G}'}(\mathcal{C}'^*) \leq \text{val}_{\mathcal{G}'}(\mathcal{C}^*) = \text{val}_{\mathcal{G}}(\mathcal{C}^*) - c_{p,q} \leq \text{val}_{\mathcal{G}}(\mathcal{C}^*).$$

- Suppose now that (p,q) does not belong to $\mathcal{C}^*$. Again, for any min-cut $\mathcal{C}'^*$ in $\mathcal{G}'$, it is easy to see that

$$\text{val}_{\mathcal{G}'}(\mathcal{C}'^*) \leq \text{val}_{\mathcal{G}'}(\mathcal{C}^*) = \text{val}_{\mathcal{G}}(\mathcal{C}^*).$$

Both previous cases are independent of each other and conclude the proof. □

Proposition 2. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted directed graph and $\mathcal{C}^*$ its min-cut. Let $\mathcal{C}'^*$ be a min-cut of $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$ where $\mathcal{E}' \subset \mathcal{E}$. Then, we have*

$$\mathrm{val}_{\mathcal{G}'}(\mathcal{C}'^*) \leq \mathrm{val}_{\mathcal{G}}(\mathcal{C}^*).$$

Proof. Let $\mathcal{E}' = \{e_1, \ldots, e_k\}$ be the set of edges we want to remove from $\mathcal{E}$, where $e_i \in \mathcal{E}$, $\forall 1 \leq i \leq k$. This produces a sequence of graphs where $\mathcal{G}_0 = \mathcal{G}$ and $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}''_{i-1} \setminus e_i)$ ($\mathcal{E}''_i$ corresponds to the set of edges of $\mathcal{G}_i$ at iteration i), $\forall 1 \leq i \leq k$. Then, using proposition (1), we just have

$$\mathrm{val}_{\mathcal{G}_k}(\mathcal{C}^*_k) \leq \ldots \leq \mathrm{val}_{\mathcal{G}_i}(\mathcal{C}^*_i) \leq \ldots \leq \mathrm{val}_{\mathcal{G}_0}(\mathcal{C}^*_0), \quad \forall 0 < i < k.$$

□

Before explaining the general principle of our method for building $\mathcal{G}'$, let us introduce some terminology. Throughout this chapter, we consider a fixed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and its reduced version $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$. Furthermore, we also denote $B \subset \mathbb{Z}^d$ and assume that B and $\mathcal{G}$ are such that

$$\forall p \in \mathcal{P}, \quad (\sigma_{\mathcal{E}}(p) \cap \mathcal{P}) \subset B_p, \tag{29}$$

where $\sigma_{\mathcal{E}}(p)$ is defined in (7) and B_p is the set translation of B at p, i.e.

$$B_p = \{p + q \mid q \in B\}. \tag{30}$$

In practice, we typically think of B as a ball centered at the origin. In such a case, the expression (29) means that the neighbors in the graph $\mathcal{G}$ are close to each other in $\mathbb{Z}^d$. When B is a square of positive radius r, we will denote it as B^r. Moreover, for $Z \subset \mathcal{P}$ and $B \subset \mathbb{Z}^d$, we denote by Z_B the dilation of Z by B as

$$Z_B = \{p + q \mid q \in B, p \in Z\} = \bigcup_{p \in Z} B_p.$$

We also define, for any $Z \subset \mathcal{P}$, the maximal amount of flow that might get in and out through the n-links by

$$P_{in}(Z) = \sum_{\substack{p \notin Z, q \in Z \\ (p,q) \in \mathcal{N}}} c_{p,q}, \quad P_{out}(Z) = \sum_{\substack{p \in Z, q \notin Z \\ (p,q) \in \mathcal{N}}} c_{p,q}.$$

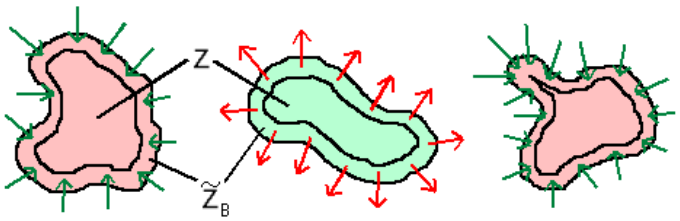


Figure 3: Principle of the reduction. Red area and arrows (resp. green area arrows) denote the flow which get in (resp. out of) Z_B. The nodes from Z are removed since Z satisfies (31). Remaining nodes are typically located in the narrow band $Z_B \setminus Z$.

Finally, we define the maximum amount of flow passing through the t-links and the flow orientation by

$$A(Z) = \sum_{p \in Z} |c_p|, \qquad O(Z) = \sum_{p \in Z} \text{sign}(c_p),$$

where the function $sign(.)$ is defined by

$$\text{sign}(t) = \begin{cases} 1 & \text{if } t > 0, \\ 0 & \text{if } t = 0, \\ -1 & otherwise. \end{cases}$$

The intuitive idea for building $\mathcal{G}'$ is to remove from the nodes of $\mathcal{G}$ any $Z \subset \mathcal{P}$ such that

$$\begin{cases} \text{either} & \Big(O(Z_B) = +\sharp Z_B \text{ and } A(Z_B \setminus Z) \geq P_{out}(Z_B)\Big), \\ \text{or} & \Big(O(Z_B) = -\sharp Z_B \text{ and } A(Z_B \setminus Z) \geq P_{in}(Z_B)\Big). \end{cases} \tag{31}$$

As an illustration of these conditions, the last (resp. first) condition of the test (31) implies that all the flow that might get in (resp. out of) the region Z_B does so by traversing its boundary and can be absorbed (resp. provided) by the band $Z_B \setminus Z$ (see Figure 3).

In the subsequent sections, two conditions for building $\mathcal{G}'$ heuristically or exactly are respectively described in Section 3.2 and in Section 3.3. For each condition, we also provide massive numerical experiments for segmenting 2D, 2D+t and 3D grayscale and color images.

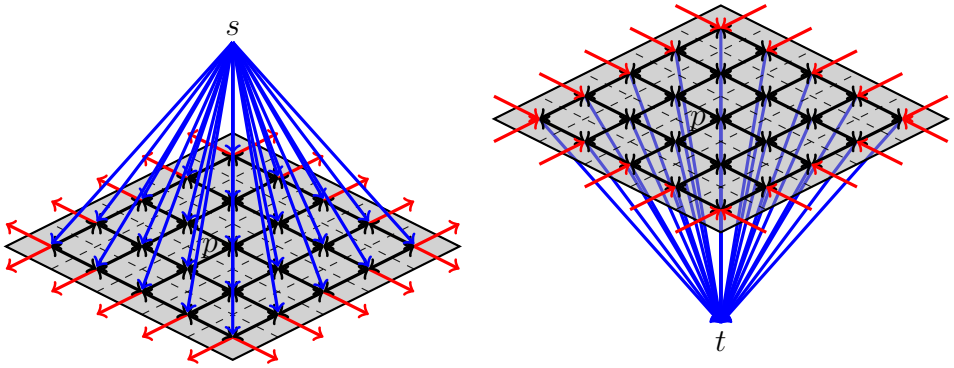


Figure 4: Illustration of the heuristic test (32). In both situations, we remove the central node p from $\mathcal{G}$ if contracted capacities are all greater than $+\delta_r$ (left image) or all less than $-\delta_r$ (right image) inside the square window B^r.

3.2 A heuristic test for reducing graphs

3.2.1 Description

In this section, we propose a condition for building $\mathcal{G}'$ heuristically. First, it is straightforward to see that the sets Z in the test (31) can be easily built by testing each individual pixel $p \in Z$. In order to do so, we know that the conjunction of conditions (31) for every set $\{p\}$, where $p \in Z$, implies (31) for Z. Consider now a graph $\mathcal{G}$ and a square window B^r of size $(2r+1)$ centered at the origin as defined in Section 3.1. We propose an even more conservative condition for $p \in Z$ as

$$\begin{cases} \text{either} & \Big(\forall q \in B_p^r,\ c_q \geq +\delta_r\Big), \\ \text{or} & \Big(\forall q \in B_p^r,\ c_q \leq -\delta_r\Big). \end{cases} \tag{32}$$

where

$$\delta_r = \frac{P(B^r)}{(2r+1)^d - 1}. \tag{33}$$

Here, $P(B^r)$ denotes the perimeter of the square window B, i.e

$$\begin{aligned} P(B^r) = \max(\sharp\{(p,q) \ :\ p \in B^r, q \notin B^r \text{ and } (p,q) \in \mathcal{N}\}, \\ \sharp\{(q,p) \ :\ p \in B^r, q \notin B^r \text{ and } (q,p) \in \mathcal{N}\}). \end{aligned}$$

Although the test (32) is typically stronger than the condition (31), both conditions

can be easily computed (see Figure 4). If moreover

$$c_{p,q} \leq 1, \qquad \forall (p,q) \in \mathcal{E},$$

(which is true for the energies described in Section 1.2.5.1 and 1.2.5.2 [1]) and (32) holds, one can easily check that the condition (31) holds for $Z = \{p\}$. For instance, the first condition of (32) implies

$$\begin{aligned} A(B_p^r \setminus \{p\}) &= \textstyle\sum_{q \in B_p^r \setminus \{p\}} |c_q| \\ &\geq [(2r+1)^d - 1]\delta_r \\ &\geq P(B^r) \\ &\geq P_{out}(B_p^r). \end{aligned}$$

In words, for any node $p \in Z$ satisfying the first (resp. second) condition of (32), all its neighbors $q \in B_p^r$ are only linked to s (resp. t) and the flow that might get in (resp. out) through t-links in $B_p^r \setminus \{p\}$ suffices to saturate the n-links going out of (resp. in) B_p^r. The node p is useless and can be removed from $\mathcal{G}$. Therefore, we consider $\mathcal{G}'$ a subgraph of $\mathcal{G}$ such that $\mathcal{V}' = \mathcal{P}' \cup \{s,t\}$, where

$$\mathcal{P}' = \{p \in \mathcal{P} \mid (32) \text{ does not hold for } p\}.$$

The experiments presented in Section 3.2.3.2 confirm the dependence between the size of the reduced graph and the model parameters (see Figure 5). Indeed, when minimizing (21) via graph cuts as described in Section 1.2.5, the t-links capacities are all multiplied by β. Thus, it is straightforward to observe that the condition (32) is more difficult to satisfy as β decreases. In such a situation, we need a larger window radius for decreasing δ_r in order to reduce the size of the reduced graph. This results in wider bands around the object contours. Notice that when β is small, the role of the regularization term $E_{p,q}(.)$ is increased. Conversely, we can afford large δ_r and therefore small window radius when β is large. Thus, the reduced graph consists of narrow bands around the object edges. Knowing the positive contracted capacity of any node $p \in \mathcal{P}$, it is also trivial to know the minimum radius (denoted r_p^{min}) for which the test (32) holds in p (without regards to the

[1] If the condition (32) does not hold, δ_r can for instance be multiplied by $\max_{(p,q)\in\mathcal{N}} c_{p,q}$.

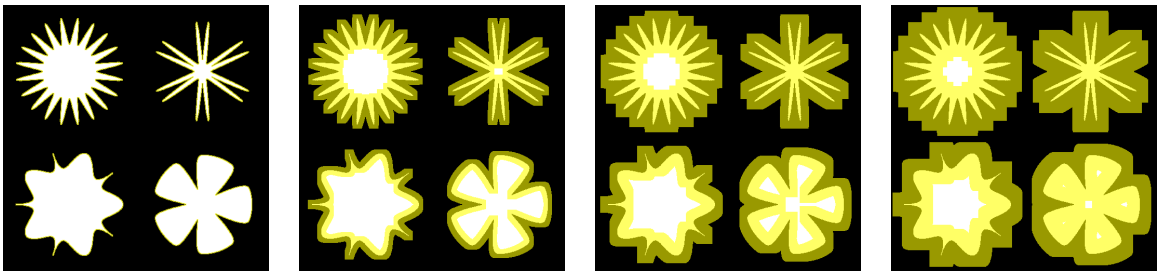
image) when $\beta \neq 0$ by testing with an increasing window radius r from one until

$$|c_p| \geq \frac{P(B^r)}{(2r+1)^d - 1} \tag{34}$$

is satisfied. Using (34), we can compute the minimum radius (denoted r_{min}) for which the test (32) holds for at least one node p (without regards to the image) when $\beta \neq 0$ by

$$r_{min} = \max\{r_p^{min} \mid p \in \mathcal{P}\}. \tag{35}$$

This can be particularly useful to prevent the fact that no memory gain occur when one chooses a window radius $r > r_{min}$. Conversely, choosing a window radius $r \leq r_{min}$ does not imply that some reduction will occur.



$\rho = 5.75\%$ $\rho = 32.58\%$ $\rho = 50.24\%$ $\rho = 64.77\%$

Figure 5: Tuning of the window radius for segmenting a synthetic 2D image with a TV+L^2 model in connectivity 1. From left to right: reduced graphs are superimposed in yellow on the original image for the window radius $r = 1, 8, 15, 22$. The relative size of the reduced graph is indicated below each image. Observe how the reduced graph $\mathcal{G}'$ progressively encompasses the whole image as the window radius r grows.

Additionally, we investigate some ways to relax the condition (32) for further reducing the size of the reduced graph. A simple way is to multiply δ_r by a factor $\gamma \in [0, 1]$. Then, as γ decreases to 0, the condition (32) can be satisfied for a larger number of nodes. Typically, when $\gamma = 0$, we only test the sign of contracted capacities (see (32)). Another way is to allow some nodes in B_p^r to fail complying with the test. The proportion of nodes satisfying the test is controlled by a parameter called $\eta \in [0, 1]$. Then, as η decreases, the condition (32) can be satisfied more easily since a larger proportion of nodes can be

connected to opposite terminals. Embedding these two extra parameters leads to

$$\begin{cases} \text{either} & \Big(\sharp\{q \in B_p^r \mid c_q \geq +\delta_r\gamma\} \geq \eta\sharp B_p^r\Big), \\ \text{or} & \Big(\sharp\{q \in B_p^r \mid c_q \leq -\delta_r\gamma\} \geq \eta\sharp B_p^r\Big). \end{cases} \tag{36}$$

Unlike the window radius parameter, γ and η parameters can further decrease the graph size but do not offer any guarantee on the final segmentation. However, for time-critical applications, this can be particularly useful when optimality does not represent a major constraint. As regard to the parameter η, it can also be used to remove noise in the segmentation.

For this moment, we have not proved that the reduction with the test (32) is exact. By exact, we mean that the max-flow value obtained from the reduced graph $\mathcal{G}'$ is equal to the one obtained in $\mathcal{G}$. Moreover, the experiments presented in Section 3.2.3.2 show low graph sizes while keeping a low pixel error on segmentations. The experiments show that the relative max-flow error between $\text{val}_{\mathcal{G}}(f^*)$ and $\text{val}_{\mathcal{G}'}(f'^*)$ (see Appendix A) is generally equal to zero. In the next section, we detail a fast algorithm for building $\mathcal{G}'$ with the test (36).

3.2.2 Algorithmic considerations

3.2.2.1 Naive algorithm

From the Section 3.2.1, an easy-to-implement algorithm emerges: it consists in checking if the test (36) holds inside the square window B_p^r centered in p, for any node $p \in \mathcal{P}$ of $\mathcal{G}$. If (36) is not satisfied for a node $p \in \mathcal{P}$, we add it to $\mathcal{G}'$ and link it to its neighbors $\sigma_\varepsilon(p)$. Otherwise, p is just removed from $\mathcal{G}'$. Since the square window B^r is visited exactly once for each node $p \in \mathcal{P}$, the algorithm for reducing $\mathcal{G}$ resembles a convolution and has a worst-case complexity of $O(\sharp\mathcal{P}\sharp B^r)$ (see Algorithm 2). Notice that the scalar δ_r is computed at the beginning of the algorithm and is of negligible time. Also, instead of building the set of edges $\mathcal{E}'$ from scratch as mentioned at the end of Algorithm 2, $\mathcal{E}'$ is progressively built by keeping track of neighbors with an array of dimensionality $(d-1)$. As a consequence, the extra memory storage of Algorithm 2 is also negligible with respect

to the size of the input image.

Algorithm 2 Naive reduction algorithm for building the graph $\mathcal{G}'$ using (36)

INPUTS: square window B^r, γ, η, whole graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
OUTPUTS: reduced graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$.

1. $\delta_r \leftarrow$ compute-delta(r) (see (33))
2. $\mathcal{V}' \leftarrow \{s, t\}$
3. **forall** $p \in \mathcal{P}$ **do**
4. $\quad NbLargePositive \leftarrow 0$
5. $\quad NbLargeNegative \leftarrow 0$
6. $\quad$ **forall** $q \in B_p^r$ **do**
7. $\quad\quad$ **if** $c_q \geq +\delta_r\gamma$ **then**
8. $\quad\quad\quad NbLargePositive \leftarrow NbLargePositive + 1$
9. $\quad\quad$ **endif**
10. $\quad\quad$ **if** $c_q \leq -\delta_r\gamma$ **then**
11. $\quad\quad\quad NbLargeNegative \leftarrow NbLargeNegative - 1$
12. $\quad\quad$ **endif**
13. $\quad$ **endfor**
14. $\quad$ **if** $|NbLargePositive| \geq \eta \sharp B_p^r \quad$ or $\quad |NbLargeNegative| \geq \eta \sharp B_p^r$ **then**
15. $\quad\quad \mathcal{V}' \leftarrow \mathcal{V}' \cup \{p\}$
16. $\quad$ **endif**
17. **endfor**
18. $\mathcal{E}' \leftarrow (\mathcal{P}'^2 \cap \mathcal{E}_n) \cup (\{(p, q) \mid p, q \in (\mathcal{V}' \times \mathcal{V}')\} \cap \mathcal{E}_t)$
19. **return** $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$

3.2.2.2 Incremental algorithm

For large window radii, Algorithm 2 becomes inefficient as the image size and the dimensionality d increase. Nevertheless, one can observe that condition (36) can be decomposed as sums along the d dimensions yielding an algorithm with a complexity of $O(\sharp\mathcal{P})$, except for image borders. For the sake of clarity, we only detail this incremental version in the 2D case with a connectivity 0. We consider a square window B^r of size $(2r+1)$, $(r > 0)$. First, for any point $p \in \mathcal{P}$ and $\delta'_r \geq 0$, we define

$$g_{\delta'_r}(p) = \begin{cases} 1 & \text{if} \quad c_p \geq +\delta'_r, \\ 0 & \text{otherwise.} \end{cases} \tag{37}$$

We either denote $g_{\delta_r\gamma}(p)$ or $g_{\delta_r\gamma}(i, j)$ for any pixel $p = (i, j) \in \mathcal{P}$ (it will never be ambiguous once in the context). In what follows, we only describe the computation of $\sharp\{q \in B_p^r \mid c_q \geq +\delta_r\gamma\}$. The other case can easily be deduced by adapting the definition of (37). The key

idea is to decompose $\sharp\{q \in B_p^r \mid c_q \geq +\delta_r\gamma\}$ as two sums where the first one sums over each row in a column while the second one sums over all columns. First, we introduce an array M whose size is the image width, where each element contains the sum of the values of $g_{\delta_r\gamma}(.)$ over a vertical segment of B_p^r. More precisely, if we denote M_{i_0,j_0} the state of array M at the beginning of the computation at the pixel $p=(i_0,j_0)\in\mathcal{P}$, we have

$$M_{i_0,j_0}[i] = \begin{cases} \sum_{l=-r}^{+r} g_{\delta_r\gamma}(i, j_0+l) & \text{if} \quad i \leq i_0+r, \\ \sum_{l=-r}^{+r} g_{\delta_r\gamma}(i, j_0+l-1) & \text{if} \quad i > i_0+r, \end{cases} \tag{38}$$

except for image borders. Additionally, we maintain a variable N_{i_0,j_0} summing the elements of M along an interval of size $2r+1$

$$N_{i_0,j_0} = \sum_{c=-r}^{+r} M_{i_0,j_0}[i_0+c], \qquad \forall (i,j)\in\mathcal{P}.$$

We trivially have $N_{i_0,j_0} = \sharp\{q \in B_p^r \mid c_q \geq \delta_r\gamma\}$, for $p=(i_0,j_0)$. Then, for ensuring the property (38) at the next pixel $p=(i_0+1,j_0)\in\mathcal{P}$, we update M before N with

$$\begin{aligned} M_{i_0+1,j_0}[i_0+r+1] &\leftarrow M_{i_0,j_0}[i_0+r+1] - g_{\delta_r\gamma}(i_0+r+1, j_0-r-1) + g_{\delta_r\gamma}(i_0+r+1, j_0+r) \\ N_{i_0+1,j_0} &\leftarrow N_{i_0,j_0} - M_{i_0+1,j_0}[i_0-r] + M_{i_0+1,j_0}[i_0+r+1] \end{aligned}$$

The contracted capacities are only evaluated once: when shifting from one position to the next one. Therefore, the optimized algorithm builds the reduced graph with a complexity of $O(\sharp\mathcal{P})$, except for image borders. In particular, the complexity becomes independent of the window radius. Also, one can notice that the cost of such an algorithm is directly proportional to the cost for evaluating the contracted capacities. However, for the energy models presented in this document, these capacities can be efficiently pre-computed and stored in lookup tables. The memory storage required by the incremental graph construction algorithm lies in the table M which is of dimensionality $(d-1)$. Thus, the extra memory usage is negligible over the image and the graph size.

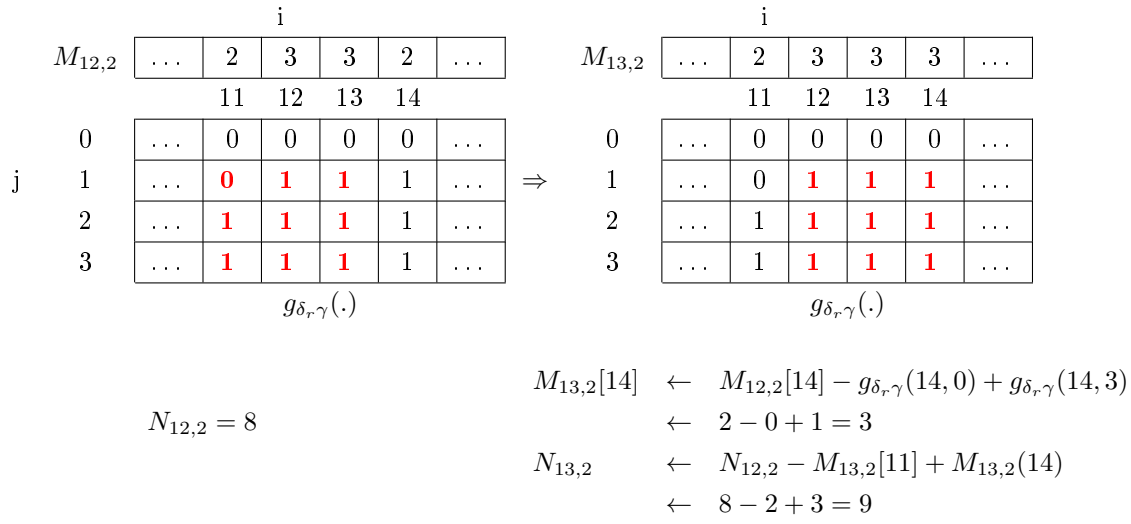


Figure 6: Illustration of the incremental algorithm for building $\mathcal{G}$ on a 2D image with $r = 1$, $\gamma = 1$ and $\eta = 1$. In this example, only the node corresponding to the pixel $p = (13, 2)$ is added to $\mathcal{G}'$ since $|N_{13,2}| = (2r+1)^2 = 9$.

3.2.2.3 Adaptive algorithm

Algorithm 2 remains quite general and can be extended in various ways. Since δ_r diminishes when the window radius r increases, one can easily design an adaptive version of Algorithm 2 where the window radius r varies automatically according to the image content. This implies to compute, for each node $p \in \mathcal{P}$ in the graph $\mathcal{G}$, the optimal window radius r^* for which the test (36) holds for p. This can be done by examining all window radii $r \in \{0, \ldots, r_{max}\}$ ($r_{max} \geq 1$). Notice however that unlike Algorithm 2, this algorithm requires that $\eta = 1$ (see Algorithm 3). In Algorithm 3, observe that the expression (34) permits to discard the current node as early as possible when the minimum radius r_{min} is larger than the maximum radius r_{max} allowed.

Unlike Algorithm 2, the worst-case complexity is $O(\sharp\mathcal{P}\sharp B^{r_{max}})$. Although this approach permits to build smaller graphs (see Figure 7), the computational cost is larger since all window radii must be examined in the worst case. It is also more difficult to avoid repetitive calculations of contracted capacities like in incremental algorithm. The gain brought by this algorithm on the relative reduced graph sizes is thus limited to some particular situations such as noisy or high-contrasted images and when the amount of

regularization is large.

$\rho^* = 35.05\%$ $\rho^* = 16.91\%$

Figure 7: Illustration of memory gain brought by the adaptive algorithm (middle) versus the incremental algorithm (left) for segmenting a 2D image with a TV+L^2 model in connectivity 1. In this experiment we set $r = 13$ and $r_{max} = 13$. On the left and middle images, the reduced graph is superimposed in yellow by transparency. In the right image, any pixel p is assigned with r_p^{min}. Brighter pixels are the nodes which can be removed with a larger window radius r and black pixels belong to $\mathcal{G}'$.

We want also to mention that Algorithms 2 and 3 are both highly parallelizable due to the locality of data and operations. Indeed, the test (36) can be quickly evaluated on each node, independently of the other ones. Furthermore, when the reduced graph $\mathcal{G}'$ contains several connected components, one could launch the max-flow computation on each component independently of the others. In some situations (such as the segmentation of noisy images), this approach could be very efficient since the max-flow computation would become trivial for a large amount of connected components whose nodes are all linked to the same terminal [2]. Due to the lower worst-case complexity of the incremental algorithm compared to the naive one, we always will use the former for the numerical experiments in the rest of this document.

[2]Indeed, the condition (36) does not imply that both terminals are linked to the non-terminals nodes unless we have $\gamma = 0$ and $\eta = 1$.

Algorithm 3 Adaptive reduction algorithm for building the graph $\mathcal{G}'$ using (36)

Inputs: γ, r_{max}, graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Outputs: reduced graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$.

$\mathcal{V}' \leftarrow \{s, t\}$
forall $p \in \mathcal{P}$ **do**
  $NbLargeCapacities \leftarrow 0$
  $r_{min} \leftarrow -\infty$
  forall $r \in \{0, \ldots, r_{max}\}$ **do**
    forall $q \in \partial B_p^r$ **do**
      $r_q^{min} \leftarrow$ compute-rmin(c_q, B^r) (see (34))
      $r_{min} \leftarrow \max\{r_q^{min}, r_{min}\}$
      if $r_q^{min} > r_{max}$ **then**
        goto end
      endif
      $\delta_{r_q^{min}} \leftarrow$ compute-delta(r_q^{min}) (see (33))
      if $c_q \geq +\delta_{r_q^{min}}\gamma$ **then**
        $NbLargeCapacities \leftarrow NbLargeCapacities + 1$
      endif
      if $c_q \leq -\delta_{r_q^{min}}\gamma$ **then**
        $NbLargeCapacities \leftarrow NbLargeCapacities - 1$
      endif
    endfor
    if $|NbLargeCapacities| = \sharp B_p^r$ **then**
      if $r_{min} \leq r$ **then**
        goto end
      endif
    else
      $\mathcal{V}' \leftarrow \mathcal{V}' \cup \{p\}$
      goto end
    endif
  endfor
  end:
endfor
$\mathcal{E}' \leftarrow (\mathcal{P}'^2 \cap \mathcal{E}_n) \cup (\{(p, q) \mid p, q \in (\mathcal{V}' \times \mathcal{V}')\} \cap \mathcal{E}_t)$
return $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$

3.2.3 Numerical experiments

Below, we analyze the role of the window radius parameter and describe massive numerical experiments for segmenting multidimensional grayscale and color images. All experiments are performed on an Athlon Dual Core 6000+ 3GHz with 2GB of RAM using the max-flow algorithm of [BK04] [3]. Running times include the graph construction, the max-flow computation as well as the construction of the solution. Times are averaged over 10 runs.

3.2.3.1 The window radius parameter

The Figure 8 shows measures of the impact of the window radius parameter in terms of speed and memory usage and compares the results to standard graph cuts (bottom row) for segmenting 2D and 2D+t data (top row) in connectivity 1. On the bottom row, the blue curves with squares correspond to time consumption and the red curves with triangles correspond to the memory of the reduced graphs. Standard graph cuts correspond to $r = 0$.

First, the segmentations obtained by standard graph cuts and reduced graph cuts are identical. We also observe that the reduced graph cuts are always faster (except for the image "plane") and requires less memory than the former. One can also observe that both curves are approximately convex and the minimal relative size of the reduced graph (denoted by ρ^*) is reached for some radius $r^* > 0$. Notice that r^* naturally depends both on the image structure and the model parameters. The intuitive reason for both curves to be approximately convex is that each individual test of (36) can be satisfied more easily when r increases, since δ_r decreases with r. Nevertheless, when r is larger, the condition becomes more and more difficult to satisfy because a larger number of individual test must hold. Notice that this experiment is chosen to illustrate the behavior when r changes. However, we generally set $r = 1$ for most of the images (see Tables 2 and 3).

[3] The code is freely available at `http://www.cs.cornell.edu/People/vnk/software.html`

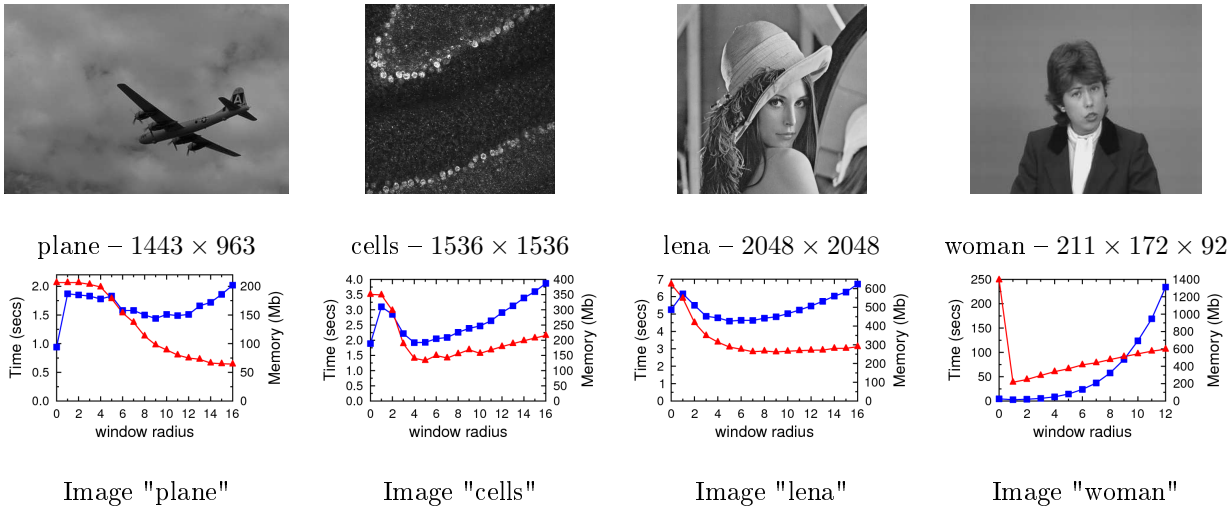


Figure 8: Influence of window radius (bottom row) for segmenting 2D and 2D+t images (top row) with a TV+L^2 model in connectivity 1. On the bottom row, blue curve with squares and red curve with triangles correspond respectively to execution time and to the amount of memory allocated for the graph. Standard graph cuts correspond to $r = 0$.

3.2.3.2 Massive experiments on 2D, 2D+t and 3D images

In this section, we compare the performance of standard graph cuts (SGC) against reduced graph cuts (RGC) in terms of speed and memory consumption for segmenting 2D, 2D+t and 3D grayscale/color images in connectivity 1. We also provide measures to estimate the distance between the segmentations obtained with SGC and RGC as well as the relative max-flow error between $\mathrm{val}_{\mathcal{G}'}(f'^*)$ and $\mathrm{val}_{\mathcal{G}}(f^*)$. These measures will act as a performance indicator for evaluating the efficiency (or not) of the test (36). Also, notice that in the following massive experiments, we always set $\gamma = 1$ and $\eta = 1$.

Let us first describe our experimental setup. For each image, the seeds and the model parameters are manually optimized for getting the best segmentation. Using these seeds and parameters, a reference segmentation is computed with SGC. Then, a second segmentation is computed with RGC using the same seeds and parameters. The differences between both segmentations are then assessed using three evaluation measures (DSC, MSASD and VO) (see Appendix A). Similarly, we also estimate the relative max-flow error between $\mathrm{val}_{\mathcal{G}}(f^*)$ and $\mathrm{val}_{\mathcal{G}'}(f'^*)$ (see Appendix A). The window radius r^* for which

the relative size of the graph ρ^* is minimum, is also provided. For all experiments, notice that some 2D images are extracted from the Berkeley segmentation dataset [4].

The results obtained using a TV+L^2 model (see Section 1.2.5.1) are summarized in Table 2 and illustrated in Figure 9. Similarly, we summarize the results obtained using a Boykov-Jolly model (see Section 1.2.5.2) with NH and GMM in Tables 3 and 4, respectively. Segmentation results are also respectively provided for NH and GMM in Figures 10 and 11.

For both energy models, we observe that RGC globally outperform SGC in terms of memory while the differences between both segmentations as well as the relative max-flow difference are generally null (or remain extremely small). For some of the 2D+t and 3D images, SGC fail to compute the segmentation (due to a large amount of memory needed) while RGC are able to segment them and in a reasonable time. Nevertheless, the minimum relative size of the reduced graph ρ^* of some noisy images remains particularly large (see for instance images "circles", "zen-garden" and "cells") for both energy models. This observation reflects the fact that a lot of neighboring nodes are connected to opposite terminals in $\mathcal{G}'$. The density of nodes connected to the terminals s and t is directly correlated to the amount of noise in the image. An ideal situation therefore consists of large area of nodes linked to the same terminal separated by smooth borders. In the case where these areas contain few nodes connected to wrong terminals, we can obtain better reduction by relaxing the test (32) with the parameter η (see (36) and Section 3.2.3.4).

In some situations, RGC are even faster than SGC. In words, it means that the time required by the reduction is compensated by the time for allocating the useless nodes and the computation of the max-flow on the reduced graph. However, the difference is generally small and becomes smaller as r^* increases. In that case, most of the time of the reduction is indeed spent on the borders. This drawback is strengthened when the number of channels increases. As an illustration, the time spent on the borders for segmenting a color image of size 481×321 can represent more than 50% of the time for reducing

[4] The dataset is freely available at `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/`

the graph with a window radius $r = 5$. This percentage can rise to 80% for a window radius $r = 10$. Although it significantly reduces performances, this also confirms that RGC are fast almost everywhere. Therefore, a better management of borders would lead to a substantial increase of speed of the proposed algorithm. Nevertheless, this situation does not occur often since we generally have $r^* = 1$.

Another key point is that RGC can allocate a larger amount of memory than SGC. This situation typically occurs when β is too small, leading to a very large relative size of the reduced graph (see the image "circles" in Table 2). Since the size of $\mathcal{G}'$ is not known in advance, RGC sometimes need to reallocate an extra memory space for storing the following nodes and edges. In fact, the max-flow algorithm of [BK04] reallocates memory by adding the half of the size of the memory storage taken by nodes and edges. In order to avoid reallocations, we can adapt simple strategies to get an upper bound on the number of nodes and edges belonging to $\mathcal{G}'$. For instance, we can use the test (36) by testing individually each pixel $p \in \mathcal{P}$ with δ_1 or by randomly polling some amount of pixels in the image. Let us now analyze specifically the results obtained in Tables 2, 3 and 4.

For the TV+L^2 model, the average relative size of the reduced graph is 33.51% ($\pm$34.01%) over all images. For 21 images out of 28, RGC allocate less memory than SGC. Similarly, RGC run faster than SGC for 11 images out of 28 for which no memory problem occurs with SGC. For some instances, the optimal window radius is far from being equal to one because the boundary of the segmentation is very rigid in order to avoid undesired parts in the image (see for instance the images "zen-garden" and "sweets" in Figure 9). This leads to a low value of β and therefore a large window radius r^* for lowering δ_r in order to reduce the size of $\mathcal{G}'$.

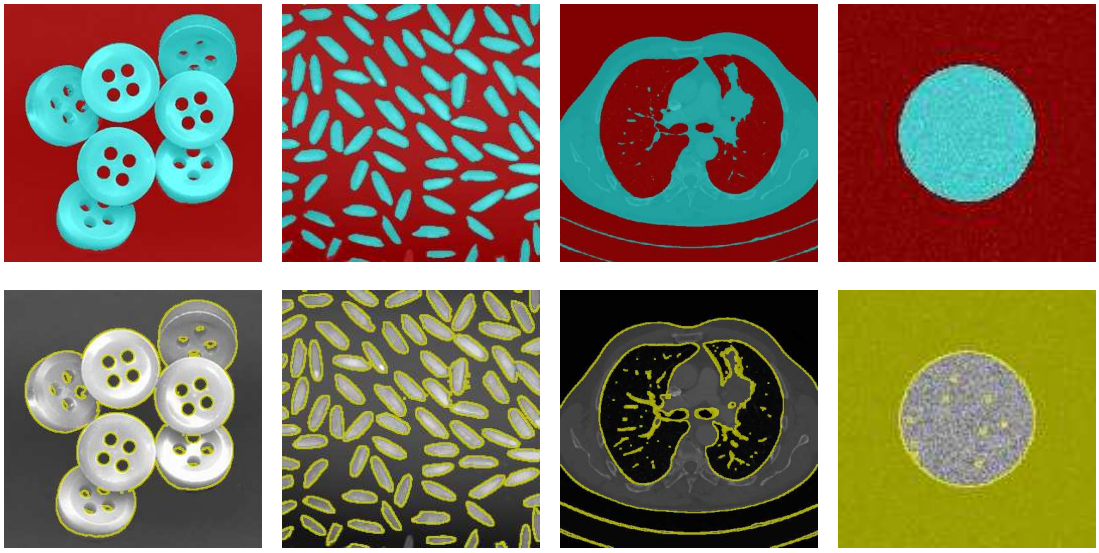
For the Boykov-Jolly model using NH, the average relative size of the reduced graph is 19.24% ($\pm$24.81%) over all images. RGC allocate less memory than SGC for 29 images out of 31. Additionally, RGC run faster for 17 images out of 24 for which no memory problem occurs with SGC. When using GMM, the average relative size of the reduced graph is slightly higher than using NH: 24.99% ($\pm$31.9%). The Table 4 also shows that RGC allocate less memory than SGC for 27 images out of 31. Similarly, for 15 images

out of 24 where no memory problem occurs with SGC, RGC outperform SGC in terms of speed. More generally, we have also compared the distance between segmentations obtained with NH and GMM using the DSC and the ASASD evaluation measures (see Appendix A), when both are available. The outcomes indicate that the segmentations are almost the same with a mean DSC and a mean ASASD respectively equal to 97.88% ($\pm$5.57%) and 4.20 ($\pm$6.97). All these results are also visually confirmed in Figures 10 and 11.

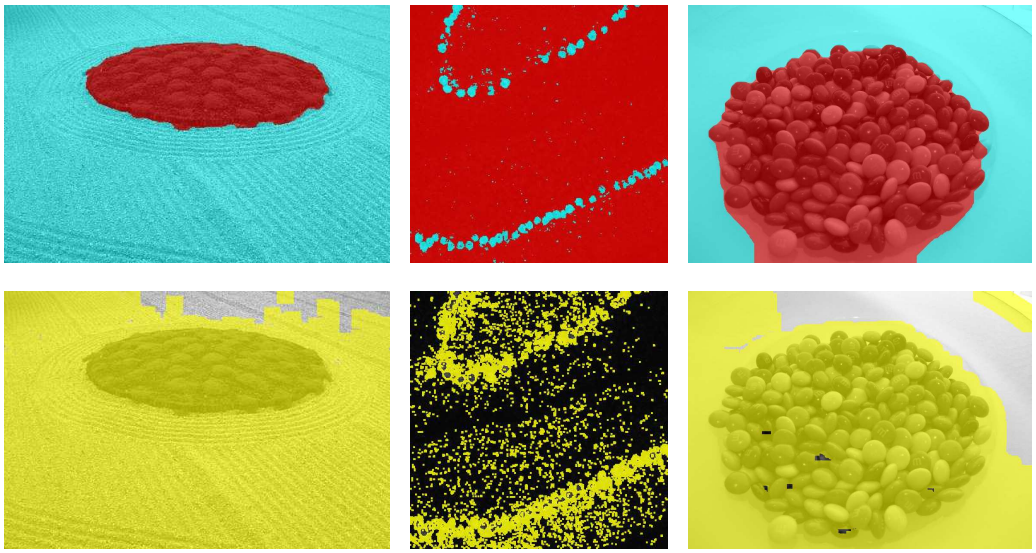
From a memory point of view, the results obtained using GMM are surprisingly slightly worse than using NH. For some images, the relative reduced graph size can be lower when using GMM (see for instance the images "text1" or "red-flowers" in Tables 3 and 4). Unlike NH, GMM acts in the continuous domain and slightly further propagates the distribution laws when they are abrupt. This situation is particularly visible for the image "text1" corresponding to a photo of a book with a raking light. Unlike NH, GMM is able to properly recovers a kind of halo where no pixels belong to the seeds. In this case, the same result could be reached by NH only by using a very large Gaussian kernel. However, this represents a costly operation and is bounded to a particular family of images (see Figure 12).

In other situations, the relative reduced graph size can also be larger when using NH compared to GMM (see images "meadow-and-mountains", "snow-and-clouds" and "birds2" in Tables 3 and 4). For all these color images, the object or background consists of a small cluster inside the RGB space. The EM algorithm approximates too accurately the distribution, leading to an over-estimate of the number of Gaussians in the mixture near the initial cluster. Thus, the algorithm can incorrectly label pixels with a neighboring intensity. This situation is illustrated in Figure 13 where two birds are drawn over a green background. For this image, the number of Gaussian estimated by the MDL criterion is 10 for the birds and 6 for the green background. This is clearly a typical situation where the GMM is unable to correctly label pixels in the background. Since we can not rely on distribution laws, we must set $\beta = 0$ which leads to a very large increase of the reduced graph size. These experiments therefore demonstrate the great importance of estimating as accurately as possible distribution laws for getting a small reduced graph size. Finally,

notice that GMM is able to fully recover thin structures like in the image "ct-thorax" unlike NH where blood vessels are lost in the background. Again, this effect is typically due to the ability of GMM to not being sensitive to discretization problems as NH.



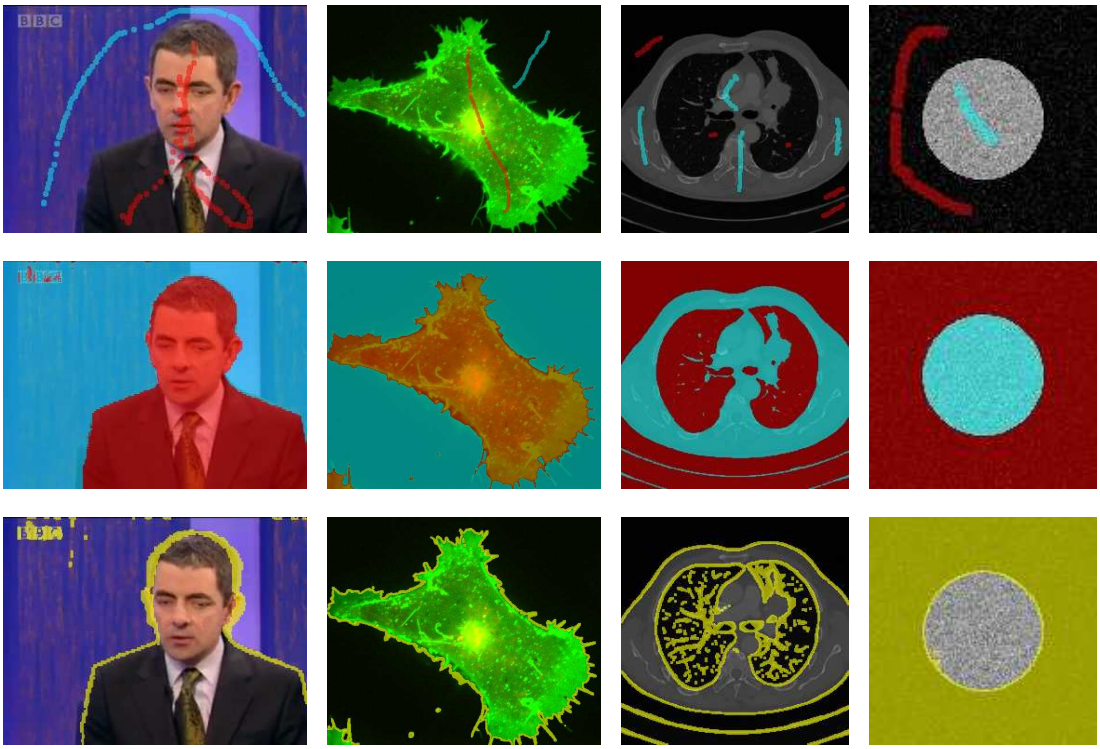
"buttons" (6.91%) "rice" (21.77%) "ct-thorax" (7.87%) "circles" (94.85%)



"zen-garden" (94.72%) "cells" (17.46%) "sweets" (83.97%)

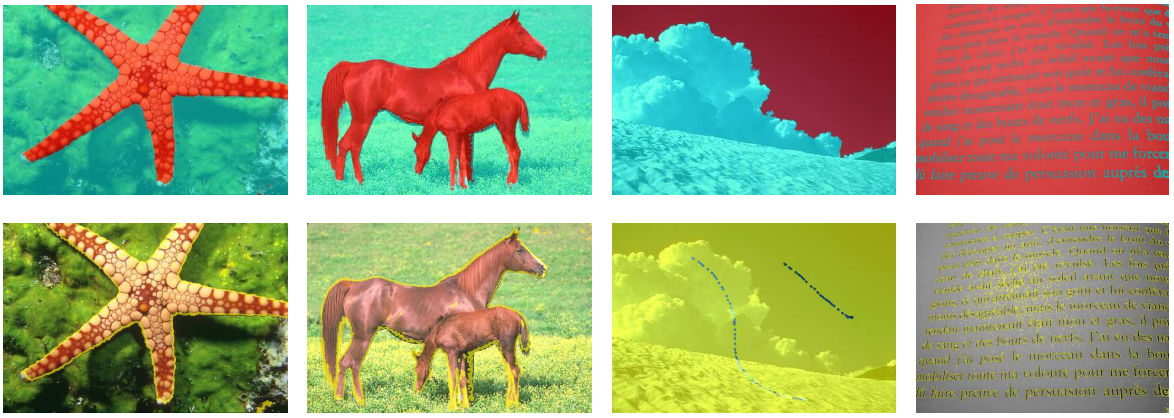
Figure 9: Segmentation results using a TV+L^2 model in connectivity 1. For each image, we represent the segmentation (first and third rows) as well as the reduced graphs (in yellow on the second and fourth rows) superimposed on the original image by transparency. The minimal relative size ρ^* of the reduced graph is also indicated below each sequence of images.

"red-starfish" (13.95%) "horses4" (5.60%) "snow-and-clouds" (15.24%) "text1" (50.18%)

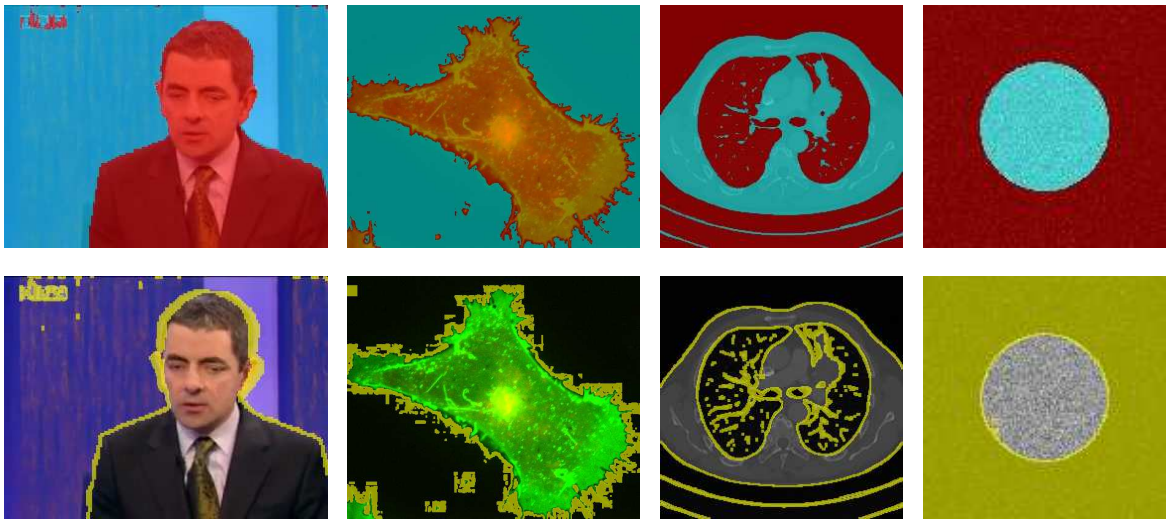


"interview-man1" (6.91%) "fluorescent-cell" (5.88%) "ct-thorax" (17.30%) "circles" (94.57%)

Figure 10: Segmentation results with using a Boykov-Jolly model using NH in connectivity 1. For each image, we represent the seeds (first and fourth rows), the segmentation (second and fifth rows) as well as the reduced graphs (in yellow on third and sixth rows) superimposed on the original image by transparency. The minimum relative reduced graph size ρ^* is also indicated below each sequence of images.

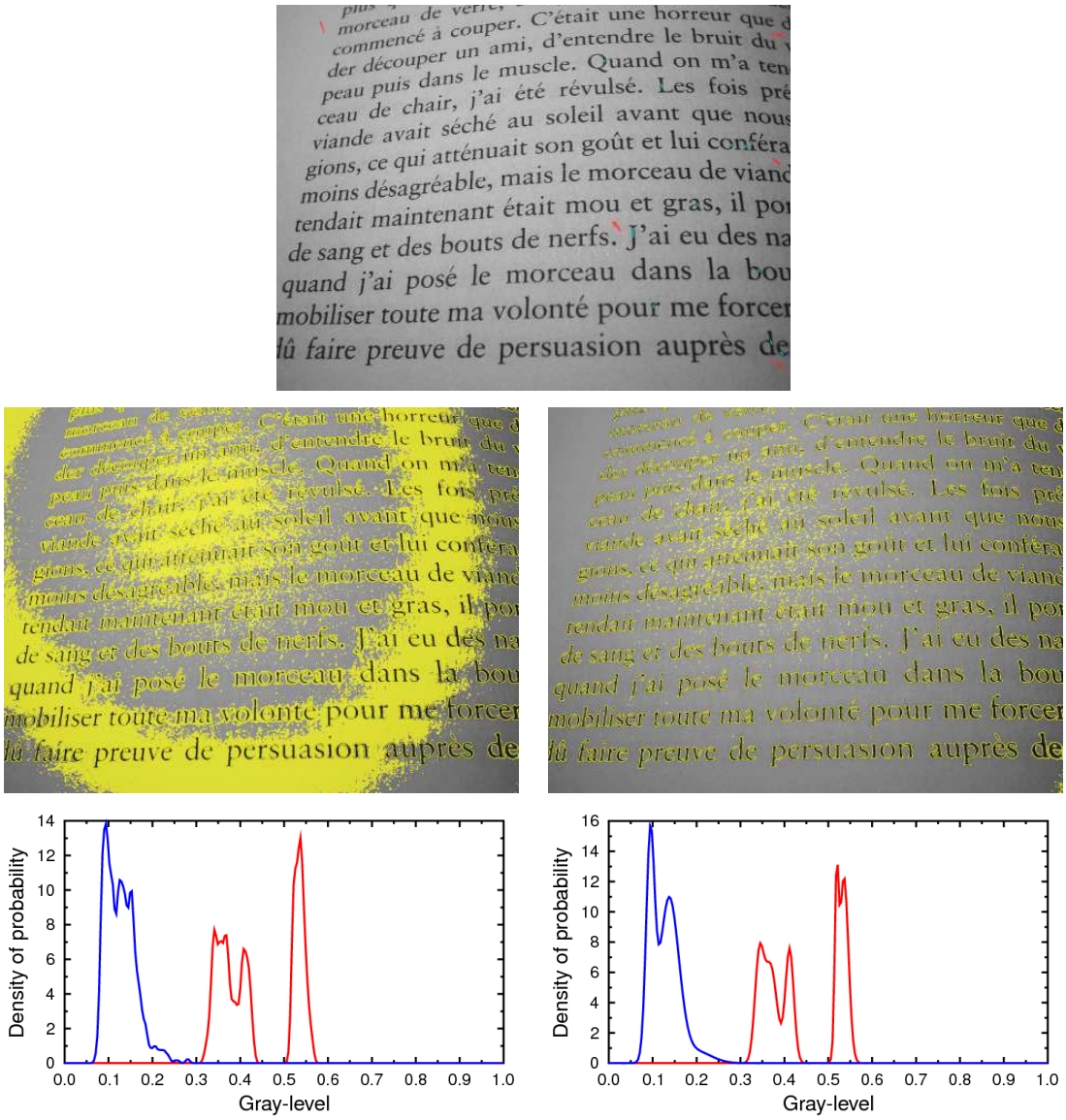


"red-starfish" (4.28%) "horses4" (7.00%) "snow-and-clouds" (99.22%) "text1" (12.55%)



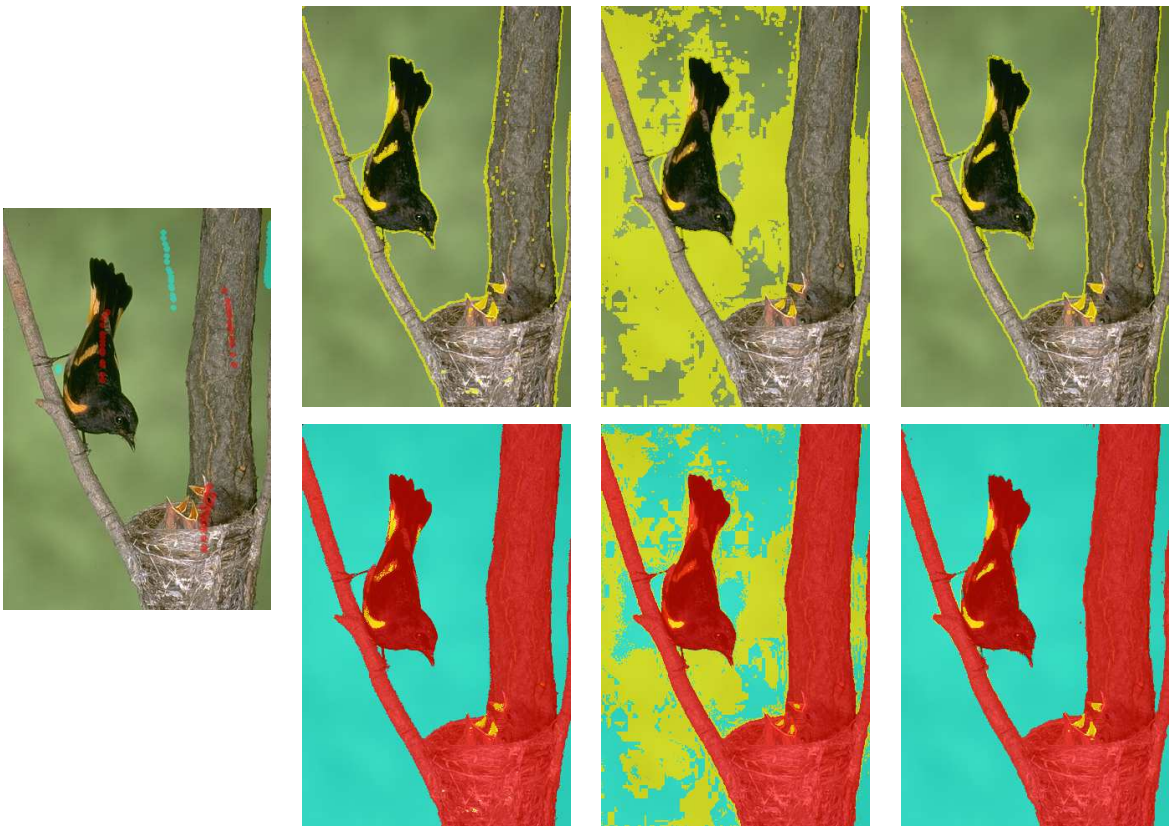
"interview-man1" (7.39%) "fluorescent-cell" (11.37%) "ct-thorax" (13.72%) "circles" (94.52%)

Figure 11: Segmentation results with using a Boykov-Jolly model using GMM in connectivity 1. For each image, we represent the segmentation (first and third rows) as well as the reduced graphs (in yellow on second and fourth rows) superimposed on the original image by transparency. Remind that the same set of seeds are used as in Figure 10. The minimum relative reduced graph size ρ^* is also indicated below each sequence of images.



$\rho^* = 50.18\%$ $\rho^* = 12.55\%$

Figure 12: Positive impact of using GMM (right column) against NH (left column) for segmenting the image "text1" (top row) with a Boykov-Jolly model. The seeds are superimposed by transparency on the image on top row. Similarly, the reduced graphs are superimposed in yellow on the image on middle row whereas distributions of the object (blue curve) and the background (red curve) are shown on the bottom row. Observe how GMM can better label pixels inside the "halo" as background pixels where the mean intensity is about 0.45.



$\rho^* = 6.40\%$ $\rho^* = 39.16\%$ $\rho^* = 5.62\%$

Figure 13: Negative impact of using GMM against NH for segmenting the image "birds2" (left column) with a Boykov-Jolly model. Left column uses NH, middle column uses GMM with MDL criteria and right column uses GMM with only one Gaussian for the object and the background. Reduced graphs are shown on top row whereas the sign of contracted capacities (positive in blue, negative in red and yellow otherwise) is shown on bottom row. Observe how GMM over-estimates the number of Gaussians in the object class while only one larger Gaussian is enough.

	Volume name	Size	SGC		RGC		ρ^* (%)	r^*	RME	DSC (%)	VO (%)	MSASD
			Time	Memory	Time	Memory						
2D	plane	481 ×321	0.12	22.90 Mb	0.42	14.54 Mb	49.89	14	0.0000	100.0000	100.0000	0.0000
	zen-garden	481 ×321	0.12	22.90 Mb	0.27	23.39 Mb	94.72	10	0.0000	100.0000	100.0000	0.0000
	oriental-man	321 ×481	0.26	22.90 Mb	0.48	23.39 Mb	79.85	13	0.0001	100.0000	100.0000	0.0000
	ct-thorax-z	512 ×512	0.18	38.91 Mb	0.11	2.05 Mb	5.74	1	0.0000	100.0000	100.0000	0.0000
	book	3012 ×2048	2.68	917.26 Mb	1.67	78.95 Mb	8.64	1	0.0000	100.0000	100.0000	0.0000
	cells-z	512 ×512	0.20	38.91 Mb	0.33	35.09 Mb	76.39	6	0.0011	100.0000	100.0000	0.0000
	beans	256 ×256	0.07	9.70 Mb	0.13	10.40 Mb	99.83	4	0.0001	100.0000	100.0000	0.0000
	sweets	800 ×600	0.81	71.28 Mb	2.51	78.95 Mb	83.97	22	0.0000	100.0000	100.0000	0.0000
	text1	1600 ×1200	0.82	285.39 Mb	0.57	25.76 Mb	10.56	1	0.0000	100.0000	100.0000	0.0000
	text2	1024 ×768	0.36	116.84 Mb	0.36	35.09 Mb	28.78	1	0.0000	100.0000	100.0000	0.0000
	yeasts1	512 ×512	0.23	38.91 Mb	0.34	49.08 Mb	95.32	3	0.0016	100.0000	100.0000	0.0000
	yeasts2	512 ×512	0.19	38.91 Mb	0.15	6.93 Mb	18.83	1	0.0001	99.9730	99.9460	0.0000
	angiography1	512 ×512	0.18	38.91 Mb	0.11	3.08 Mb	7.97	1	0.0001	100.0000	100.0000	0.0000
	angiography2	350 ×643	0.13	33.39 Mb	0.09	2.26 Mb	7.51	1	0.0000	100.0000	100.0000	0.0000
	f117	588 ×392	0.12	34.20 Mb	0.06	623.05 Kb	1.71	1	0.0000	100.0000	100.0000	0.0000
	black-cat	600 ×400	0.12	35.61 Mb	0.07	415.38 Kb	1.14	1	0.0000	100.0000	100.0000	0.0000
	viking-symbol2	660 ×740	0.27	72.53 Mb	0.28	35.09 Mb	37.06	3	0.0000	100.0000	100.0000	0.0000
	buttons	300 ×300	0.06	13.33 Mb	0.03	934.60 Kb	6.91	1	0.0000	100.0000	100.0000	0.0000
	rice	256 ×256	0.04	9.70 Mb	0.04	2.05 Mb	21.77	1	0.0002	100.0000	100.0000	0.0000
	blood-cells	425 ×280	0.08	17.64 Mb	0.05	3.08 Mb	18.84	1	0.0000	100.0000	100.0000	0.0000
	poppy	409 ×613	0.16	37.21 Mb	0.09	1.01 Mb	2.93	1	0.0002	100.0000	100.0000	0.0000
2D+t	interview-girl	320 ×240 ×150	**OM**	4.72 Gb	8.96	771.00 Mb	15.04	1	**NSR**	**NSR**	**NSR**	**NSR**
	interview-old-man	256 ×256 ×128	**OM**	3.43 Gb	8.50	771.00 Mb	18.83	1	**NSR**	**NSR**	**NSR**	**NSR**
	interview-woman	352 ×288 ×154	**OM**	6.40 Gb	10.42	1.13 Gb	13.36	1	**NSR**	**NSR**	**NSR**	**NSR**
3D	ct-thorax	409 ×409 ×252	**OM**	17.33 Gb	21.43	1.17 Gb	7.87	1	**NSR**	**NSR**	**NSR**	**NSR**
	circles	128 ×128 ×128	5.01	874.57 Mb	7.16	1.10 Gb	94.85	1	0.0138	100.0000	100.0000	0.0000
	cells	409 ×409 ×101	**OM**	6.92 Gb	14.23	1.13 Gb	17.46	1	**NSR**	**NSR**	**NSR**	**NSR**
	brain-p3	181 ×217 ×181	**OM**	2.91 Gb	6.00	342.67 Mb	12.63	1	**NSR**	**NSR**	**NSR**	**NSR**

Table 2: Standard graph cuts (SGC) are compared to Reduced Graph Cuts (RGC) in terms of speed (in secs) and memory for segmenting data using a TV+L^2 model in connectivity 1. Label **OM** stands for "Out of Memory" while label **NSR** stands for "No Segmentation Reference".

	Volume name	Size	SGC		RGC		ρ^* (%)	r^*	RME	DSC (%)	VO (%)	MSASD
			Time	Memory	Time	Memory						
2D	eagle-c	481 ×321	0.20	22.90 Mb	0.15	1.37 Mb	5.54	1	0.0000	100.0000	100.0000	0.0000
	zen-garden-c	481 ×321	0.22	22.90 Mb	0.34	23.39 Mb	90.75	1	0.0000	100.0000	100.0000	0.0000
	columns-c	481 ×321	0.22	22.90 Mb	0.12	276.92 Kb	1.17	1	0.0000	100.0000	100.0000	0.0000
	red-flowers-c	481 ×321	0.19	22.90 Mb	0.21	6.93 Mb	23.30	1	0.0000	100.0000	100.0000	0.0000
	snow-and-clouds-c	481 ×321	0.19	22.90 Mb	0.15	4.62 Mb	15.24	1	0.0000	100.0000	100.0000	0.0000
	marker-c	481 ×321	0.19	22.90 Mb	0.13	623.05 Kb	2.46	1	0.0000	100.0000	100.0000	0.0000
	pyramid-c	481 ×321	0.18	22.90 Mb	0.13	304.97 Kb	1.46	1	0.0000	100.0000	100.0000	0.0000
	red-starfish-c	481 ×321	0.19	22.90 Mb	0.15	3.08 Mb	13.95	1	0.0000	100.0000	100.0000	0.0000
	black-cow-c	481 ×321	0.21	22.90 Mb	0.15	304.97 Kb	1.51	1	0.0000	100.0000	100.0000	0.0000
	church2-c	481 ×321	0.20	22.90 Mb	0.13	1.37 Mb	5.07	1	0.0000	100.0000	100.0000	0.0000
	snake2-c	481 ×321	0.20	22.90 Mb	0.15	1.01 Mb	4.95	1	0.0000	100.0000	100.0000	0.0000
	birds2-c	321 ×481	0.21	22.90 Mb	0.16	1.37 Mb	6.40	1	0.0001	100.0000	100.0000	0.0000
	eagle2-c	481 ×321	0.18	22.90 Mb	0.13	623.05 Kb	2.25	1	0.0000	100.0000	100.0000	0.0000
	greek-temple-c	481 ×321	0.20	22.90 Mb	0.14	2.05 Mb	8.03	1	0.0000	100.0000	100.0000	0.0000
	horses4-c	481 ×321	0.20	22.90 Mb	0.14	1.37 Mb	5.60	1	0.0000	100.0000	100.0000	0.0000
	meadow-and-mountains-c	481 ×321	0.20	22.90 Mb	0.25	15.59 Mb	56.00	1	0.0000	100.0000	100.0000	0.0000
	traditional-houses-c	481 ×321	0.19	22.90 Mb	0.13	934.60 Kb	4.30	1	0.0000	100.0000	100.0000	0.0000
	ct-thorax-z	512 ×512	0.44	38.91 Mb	0.29	4.62 Mb	10.85	1	0.0000	99.7832	99.5674	0.0000
	book	3012 ×2048	7.54	917.26 Mb	5.06	78.95 Mb	8.18	1	0.0000	100.0000	100.0000	0.0000
	cells-z	512 ×512	0.46	38.91 Mb	0.49	23.39 Mb	48.91	2	0.0007	100.0000	100.0000	0.0000
	text1	1600 ×1200	2.15	285.39 Mb	2.49	177.63 Mb	50.18	1	0.0000	100.0000	100.0000	0.0000
	viking-symbol2	660 ×740	0.59	72.53 Mb	0.39	3.39 Mb	5.13	1	0.0023	99.9981	99.9962	0.0000
2D+t	interview-man1-c	320 ×240 ×203	**OM**	6.40 Gb	18.75	514.00 Mb	6.91	1	**NSR**	**NSR**	**NSR**	**NSR**
	interview-man2-c	426 ×240 ×180	**OM**	7.55 Gb	19.86	228.44 Mb	3.21	1	**NSR**	**NSR**	**NSR**	**NSR**
	plane-take-off-c	492 ×276 ×180	**OM**	10.03 Gb	28.91	532.00 Mb	6.20	1	**NSR**	**NSR**	**NSR**	**NSR**
	talk-c	370 ×276 ×190	**OM**	7.96 Gb	32.51	1.13 Gb	15.44	1	**NSR**	**NSR**	**NSR**	**NSR**
	fluorescent-cell-c	478 ×396 ×121	**OM**	9.39 Gb	30.08	514.00 Mb	5.88	1	**NSR**	**NSR**	**NSR**	**NSR**
3D	ct-thorax	245 ×245 ×151	**OM**	3.71 Gb	17.25	771.00 Mb	17.30	1	**NSR**	**NSR**	**NSR**	**NSR**
	circles	128 ×128 ×128	8.13	874.57 Mb	11.41	1.10 Gb	94.57	1	0.0043	100.0000	100.0000	0.0000
	cells	230 ×230 ×57	9.27	1.23 Gb	9.78	771.00 Mb	51.38	1	0.0029	100.0000	100.0000	0.0000
	brain-p3	181 ×217 ×181	**OM**	2.91 Gb	14.45	771.00 Mb	24.38	1	**NSR**	**NSR**	**NSR**	**NSR**

Table 3: Standard graph cuts (SGC) are compared to Reduced Graph Cuts (RGC) in terms of speed (in secs) and memory for segmenting data using a Boykov-Jolly model in connectivity 1. Distributions of probabilities are estimated using NH. Label **OM** stands for "Out of Memory" while label **NSR** stands for "No Segmentation Reference". Color image names are suffixed by "c".

	Volume name	Size	Standard GC		Reduced GC		ρ^* (%)	r^*	RME	DSC (%)	VO (%)	MSASD
			Time	Memory	Time	Memory						
2D	eagle-c	481 ×321	0.72	22.90 Mb	0.72	1.01 Mb	5.27	1	0.0000	100.0000	100.0000	0.0000
	zen-garden-c	481 ×321	1.25	22.90 Mb	1.39	23.39 Mb	98.75	1	0.0000	100.0000	100.0000	0.0000
	columns-c	481 ×321	0.97	22.90 Mb	0.92	623.05 Kb	2.35	1	0.0000	100.0000	100.0000	0.0000
	red-flowers-c	481 ×321	0.94	22.90 Mb	0.94	4.62 Mb	21.80	1	0.0000	100.0000	100.0000	0.0000
	snow-and-clouds-c	481 ×321	1.59	22.90 Mb	1.74	23.39 Mb	99.22	1	0.0000	100.0000	100.0000	0.0000
	marker-c	481 ×321	0.83	22.90 Mb	0.74	203.32 Kb	0.89	1	0.0000	100.0000	100.0000	0.0000
	pyramid-c	481 ×321	0.90	22.90 Mb	0.85	457.46 Kb	2.04	1	0.0000	100.0000	100.0000	0.0000
	red-starfish-c	481 ×321	0.87	22.90 Mb	0.84	934.60 Kb	4.28	1	0.0000	100.0000	100.0000	0.0000
	black-cow-c	481 ×321	0.86	22.90 Mb	0.77	276.92 Kb	1.23	1	0.0000	100.0000	100.0000	0.0000
	church2-c	481 ×321	1.04	22.90 Mb	1.02	1.37 Mb	6.56	1	0.0000	100.0000	100.0000	0.0000
	snake2-c	481 ×321	0.72	22.90 Mb	0.71	2.05 Mb	8.68	1	0.0000	100.0000	100.0000	0.0000
	birds2-c	321 ×481	0.87	22.90 Mb	0.88	10.40 Mb	39.16	1	0.0002	100.0000	100.0000	0.0000
	eagle2-c	481 ×321	0.84	22.90 Mb	0.80	1.37 Mb	5.18	1	0.0000	100.0000	100.0000	0.0000
	greek-temple-c	481 ×321	1.08	22.90 Mb	1.03	6.93 Mb	26.55	1	0.0000	100.0000	100.0000	0.0000
	horses4-c	481 ×321	0.76	22.90 Mb	0.72	1.51 Mb	7.00	1	0.0000	100.0000	100.0000	0.0000
	meadow-and-mountains-c	481 ×321	1.08	22.90 Mb	1.27	23.39 Mb	97.94	1	0.0000	100.0000	100.0000	0.0000
	traditional-houses-c	481 ×321	0.97	22.90 Mb	0.89	1.37 Mb	5.13	1	0.0000	100.0000	100.0000	0.0000
	ct-thorax-z	512 ×512	0.87	38.91 Mb	0.73	3.08 Mb	7.87	1	0.0058	100.0000	100.0000	0.0000
	book	3012 ×2048	24.45	917.26 Mb	21.80	86.94 Mb	9.90	1	0.0000	100.0000	100.0000	0.0000
	cells-z	512 ×512	0.71	38.91 Mb	0.81	23.39 Mb	60.00	2	0.0016	100.0000	100.0000	0.0000
	text1	1600 ×1200	5.66	285.39 Mb	5.37	35.09 Mb	12.55	1	0.0000	100.0000	100.0000	0.0000
	viking-symbol2	660 ×740	1.20	72.53 Mb	1.05	3.08 Mb	4.25	1	0.0000	100.0000	100.0000	0.0000
2D+t	interview-man1-c	320 ×240 ×203	**OM**	6.40 Gb	93.15	514.00 Mb	7.39	1	**NSR**	**NSR**	**NSR**	**NSR**
	interview-man2-c	426 ×240 ×180	**OM**	7.55 Gb	122.26	514.00 Mb	7.02	1	**NSR**	**NSR**	**NSR**	**NSR**
	plane-take-off-c	492 ×276 ×180	**OM**	10.03 Gb	150.08	771.00 Mb	6.36	1	**NSR**	**NSR**	**NSR**	**NSR**
	talk-c	370 ×276 ×190	**OM**	7.96 Gb	122.68	1.13 Gb	14.80	1	**NSR**	**NSR**	**NSR**	**NSR**
	fluorescent-cell-c	478 ×396 ×121	**OM**	9.39 Gb	144.91	988.45 Mb	11.37	1	**NSR**	**NSR**	**NSR**	**NSR**
3D	ct-thorax	245 ×245 ×151	**OM**	3.71 Gb	31.79	514.00 Mb	13.72	1	**NSR**	**NSR**	**NSR**	**NSR**
	circles	128 ×128 ×128	12.81	874.57 Mb	15.99	1.10 Gb	94.52	1	0.0056	100.0000	100.0000	0.0000
	cells	230 ×230 ×57	12.65	1.23 Gb	14.57	798.00 Mb	67.32	1	0.0003	100.0000	100.0000	0.0000
	brain-p3	181 ×217 ×181	**OM**	2.91 Gb	26.30	771.00 Mb	25.82	1	**NSR**	**NSR**	**NSR**	**NSR**

Table 4: Standard graph cuts (SGC) are compared to Reduced Graph Cuts (RGC) in terms of speed (in secs) and memory for segmenting data using a Boykov-Jolly model in connectivity 1. Distributions of probabilities are estimated using GMM. Label **OM** stands for "Out of Memory" while label **NSR** stands for "No Segmentation Reference". Color image names are suffixed by "c".

3.2.3.3 The parameter γ

Figure 14 shows how far the test (36) can be relaxed while nearly having an exact solution. In these experiments, we use the model and parameters as in Figure 8. The window radii are chosen to minimize the memory consumption. Furthermore, the differences between the segmentations computed by SGC and RGC are estimated using two evaluation measures: DSC and MSASD (see Appendix A). Then, we display the DSC (green curve), the MSASD (purple curve) as well as the execution time (blue curve) and the memory consumption (red curve) over a fixed range of γ values ranging from 0 to 1. As γ decreases to 0, we naturally observe that we get a coarser approximation of the solution. In practice, we obtain nearly exact solutions for $\gamma \geq 0.5$ for high-contrasted images. For $\gamma < 0.4$, the solution is slightly different but remains close from the original segmentation.

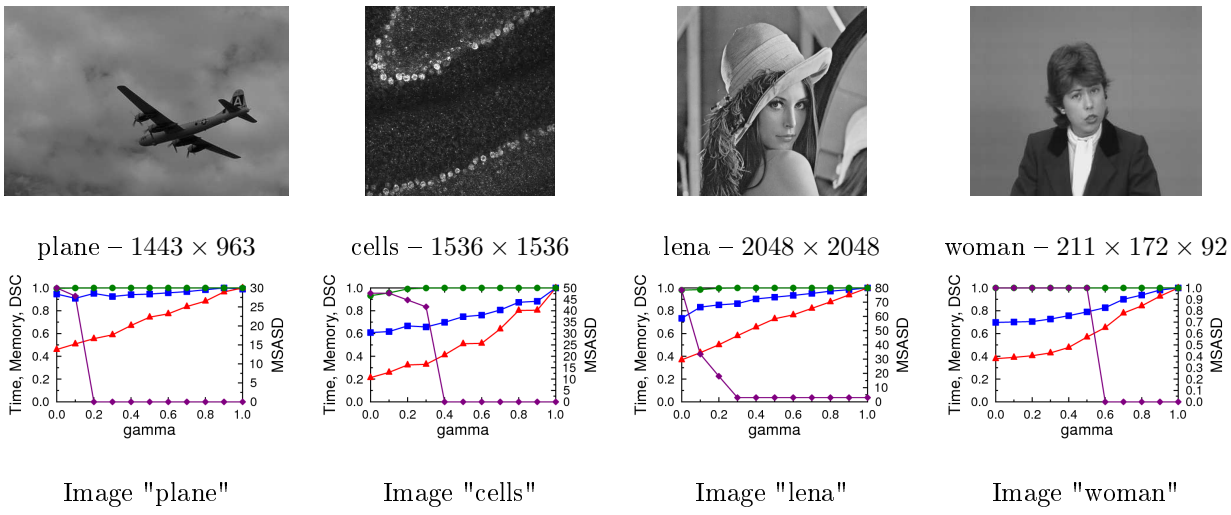


Figure 14: Influence of the parameter γ (bottom row) for segmenting 2D and 2D+t images (top row) with a TV+L^2 model in connectivity 1. On the bottom row, blue curve with squares and red curve with triangles correspond respectively to the gain in time and to the amount of memory allocated for the reduced graph. Green curves with circles and purple curves with diamonds correspond respectively to the DSC and to the MSASD between γ-parameterized segmentations and the segmentations obtained with standard graph cuts.

3.2.3.4 The parameter η

a) Automatic tuning

A lower bound

For a fixed window radius, notice first that the value of η must be sufficiently large for keeping the graph in a whole piece (see Figure 15). Indeed, below some value (denoted by η_{min}), the reduced graph $\mathcal{G}'$ is split into multiple pieces and becomes inconsistent since the min-cut is no longer fully embedded into $\mathcal{G}'$. This implies that some voxels could be wrongly labeled in the segmentation.

Figure 16 illustrates a situation where η_{min} can be easily computed with an image consisting of two highly-contrasted areas. Using (36) with a square window of radius r and $\eta = 1$, the reduced graph $\mathcal{G}'$ corresponds to a thin band of size $2r$. An easy under-estimation of η_{min} is obtained by imposing that η_{min} permits to segment these two contrasted areas. In order to do so, we want the test (36) to be false for any pixel p located at the boundary between these areas. For such a pixel, we have (e.g. if we assume $c_p \geq +\delta_r$)

$$\sharp\{q \in B_p \mid c_q \geq +\delta_r\gamma\} = (r+1)(2r+1)^{d-1}.$$

As a consequence, if

$$\eta \leq \frac{(r+1)(2r+1)^{d-1}}{(2r+1)^d},$$

the pixel p does not belong to the reduced graph $\mathcal{G}'$. Since we want to avoid the situation, we must therefore have

$$\begin{aligned} \eta &> \tfrac{(r+1)(2r+1)^{d-1}}{(2r+1)^d} \\ &= 1 - \tfrac{r}{2r+1} = \eta_{min}. \end{aligned} \tag{39}$$

In particular, (39) does not depend on the dimensionality d of $\mathcal{P}$. By observing (39), it is straightforward to see that, as the window radius r tends to infinity, the proportion of nodes allowed to be connected to opposite terminals tends to $\frac{1}{2}$. However, the lower bound can be too small in areas with high curvature and the reduced graph $\mathcal{G}'$ might be disconnected into multiple pieces (see Figure 15). As a consequence, the min-cut is no longer ensured of being fully embedded into $\mathcal{G}'$. In practice, we also observed that the lower bound (39) is less accurate in connectivity 0 than in connectivity 1 (see Figure 15

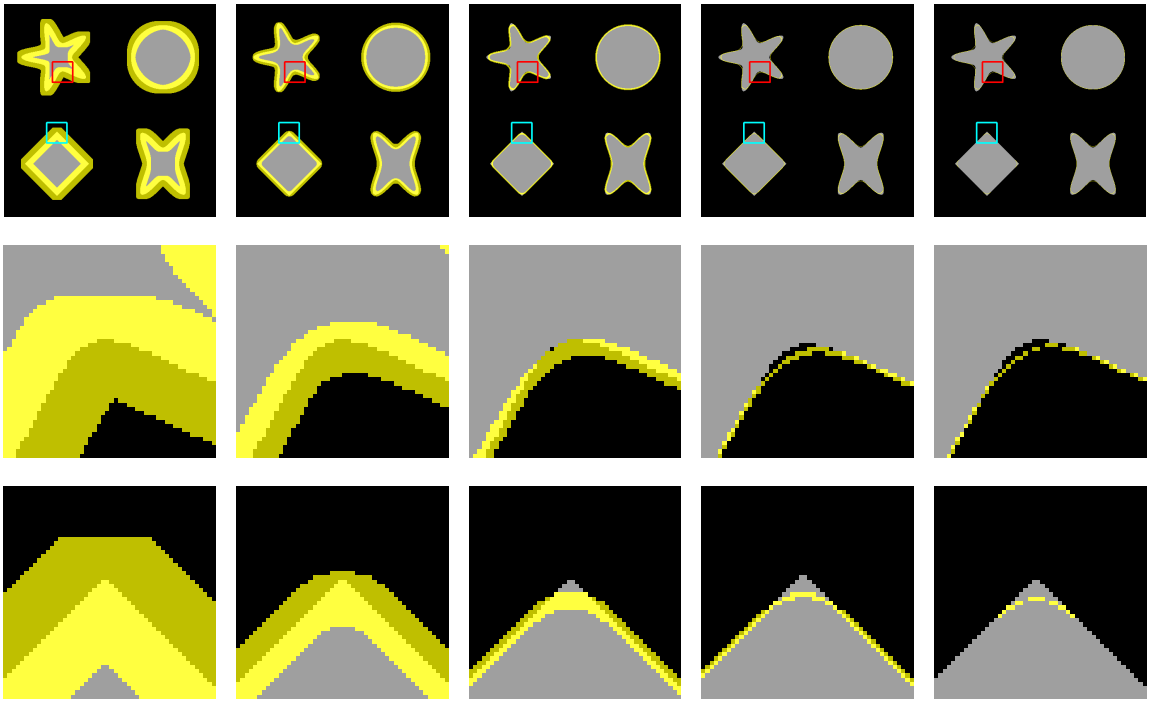


Figure 15: Illustration of the lower bound η_{min} for segmenting a 2D synthetic image using a TV+L^2 model. In this experiment, $\eta_{min} \simeq 0.523$ and we set $r = 10$ using connectivity 1. On all images, the pixels belonging to $\mathcal{G}'$ are superimposed in yellow to the original image by transparency. The middle and the bottom rows correspond respectively to close-ups of the red and cyan areas. Observe how the reduced graph splits into multiple pieces as soon as $\eta \leq \eta_{min}$.

where connectivity 1 is used). For instance, when $\eta = 0.53$, one can observe on the middle row of Figure 15 that the graph $\mathcal{G}'$ is disconnected in connectivity 0 with diagonal segments.

An upper bound

For a fixed window radius r and a positive amount of noise ξ, one can observe that there exists a value of the parameter η for which most of the nodes in noisy regions are removed from $\mathcal{G}$, leading to a diminution of the relative size of the reduced graph ρ. This situation is illustrated in the filtering experiment of Figure 19 in the paragraph b).

The purpose of this paragraph is to identify, from a statistical point of view, a reliable

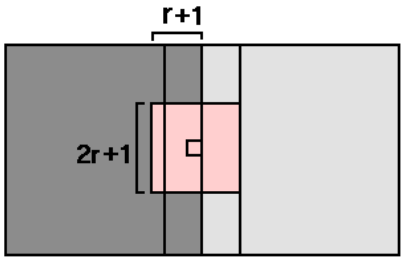


Figure 16: Toy example for computing the lower bound η_{min}.

value of the parameter η for which all nodes of $\mathcal{P}$ are very likely to be removed from $\mathcal{G}$. For a fixed amount of noise ξ in the image I, we therefore want to find an upper bound on η by finding the maximum value of η in such a way that we control the proportion of nodes corresponding to noisy pixels in homogeneous areas.

Consider a noisy constant image I with a noise generated by a Bernoulli distribution of parameter $\xi \in]0,1[$, corresponding to the amount of noise in I [5]. The two cases where $\xi = 0$ and $\xi = 1$ are trivial and are not considered in our analysis. Assume now that the graph $\mathcal{G}$ is defined as in Section 1.2.2 where the nodes corresponding to noise free pixels are connected to the sink t with a capacity $c_q \leq -\gamma\delta_r$ and the nodes corresponding to noisy pixels have capacity $c_q > -\gamma\delta_r$.

First, let X be a discrete random variable counting degraded pixels in a square window B of size $n = (2r+1)^d$ in the image I. Then, the probability that at least k pixels are corrupted in B is

$$\mathbb{P}(X > k) = \sum_{i=k+1}^{n} \binom{n}{i} \xi^i (1-\xi)^{n-i}, \tag{40}$$

where $\binom{n}{i} = \frac{n!}{i!(n-i)!}$. For a fixed window radius r, it is straightforward to see that (40) is decreasing in k and tends to ξ^n if we impose that $\xi \in]0,1[$. Let $k_1, k_2 \in \{1,\ldots,n\}$ be such that $k_2 > k_1$, we have

$$\begin{aligned} \mathbb{P}(X > k_1) - \mathbb{P}(X > k_2) &= \textstyle\sum_{i=k_1+1}^{n} \binom{n}{i} \xi^i (1-\xi)^{n-i} - \sum_{j=k_2+1}^{n} \binom{n}{j} \xi^j (1-\xi)^{n-j} \\ &= \textstyle\sum_{i=k_1+1}^{k_2} \binom{n}{i} \xi^i (1-\xi)^{n-i} \\ &> 0, \end{aligned} \tag{41}$$

since $\binom{n}{i} \geq 1$, $\xi^i > 0$ and $(1-\xi)^{n-i} > 0$ for $\xi \in]0,1[$, $\forall i \in \{k_1,\ldots,k_2-1\}$. According to

[5] Simple histogram-based techniques can be used to estimate the amount of noise ξ in the image.

the test (36) and the hypothesis on $\mathcal{G}$: a node $p \in \mathcal{P}$ can be removed from $\mathcal{G}$ if and only if

$$\sharp\{q \in B_p^r \mid c_q \leq -\gamma\delta_r\} = \sharp\{q \in B_p^r \mid q \text{ is noise free}\} \geq \eta n \tag{42}$$

Moreover, we assumed

$$\sharp\{q \in B_p^r \mid q \text{ is noise free}\} \sim (n - X).$$

Therefore, we have

$$\mathbb{P}(p \text{ is not removed}) = \mathbb{P}((n - X) < \eta n) = \mathbb{P}(X > (1 - \eta)n).$$

Fixing a proportion ε of wrongly constructed nodes, we choose

$$\eta^+ = \max\{\eta \in [0, 1] \mid \mathbb{P}(X > (1 - \eta)n) \geq \varepsilon\}, \tag{43}$$

Considering the lower bound η_{min} defined in (39), we set

$$\eta_{max} = \max\{\eta_{min}, \eta^+\}. \tag{44}$$

Combining the definitions of the lower bound (see (39)) and the upper bound (see (44)), it now becomes easy to get a good estimation of the parameter η^* for a fixed window radius by setting

$$\eta^* = \Big(\frac{\eta_{min} + \eta_{max}}{2}\Big). \tag{45}$$

Let us now analyze the joint behavior of the lower and the upper bounds. When the amount of noise ξ is fixed, one can easily observe that the gap $\Delta\eta = (\eta_{max} - \eta_{min})$ grows as the window radius r increases. Indeed, we have previously seen that the lower bound η_{min} tends to $\frac{1}{2}$ as the window radius r increases (see (39)). The previous observation is also due to the fact that the upper bound η_{max} grows as the window radius r increases.

Similarly, when the window radius r is fixed, remark that $\Delta\eta$ decreases when the amount of noise ξ increases. This situation is consistent because η_{min} remains the same but η_{max} tends to $\frac{1}{2}$ since it is more likely that the number of degraded pixels increase in the same window B. Observe how the noise affects the reliability of the upper bound η_{max}. It is important to notice that increasing the window radius r can compensate the augmentation of the amount of noise only up to $\xi = 0.5$.

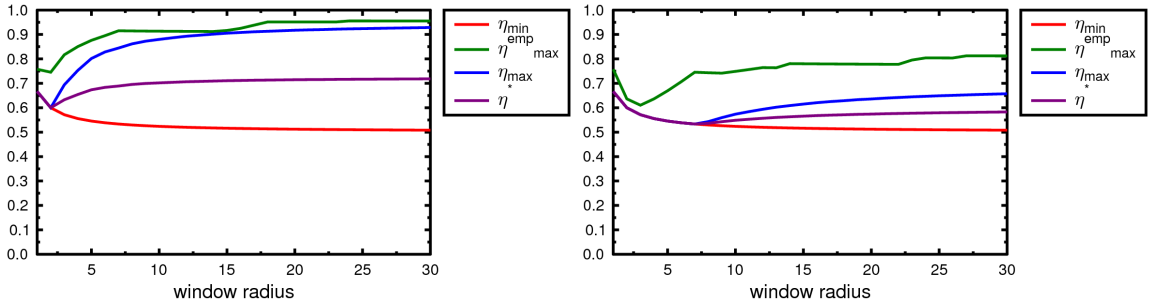


Figure 17: Evolution of the lower bound η_{min}, the statistical upper bound η_{max}, the empirical upper bound η_{max}^{emp} and the optimal value η^*, versus the window radius for a fixed amount of noise $\xi = 0.05$ (left) and $\xi = 0.3$ (right). In these experiments, we set $\varepsilon = 10^{-8}$.

Additionally, we can also derive an empirical upper bound η_{max}^{emp} for evaluating the quality of the statistical upper bound η_{max}. Let us describe how the former is computed. First, we generate a series of images corrupted by the same amount of noise ξ. For each image, we progressively decrease η to zero for finding the one that removes all nodes in $\mathcal{G}'$. Then, the empirical upper bound η_{max}^{emp} corresponds to the average of such η values across all images. These two upper bounds as well as the lower bound η_{min} and η^* are illustrated with a varying window radius in Figure 17 for two different noise levels.

Finally, notice that the thresholding parameter ε much influences the quality of the statistical upper bound η_{max}. One way to obtain a good estimation of the parameter ε is to find the one which minimizes the absolute error between the statistical upper bound η_{max} and the empirical upper bound η_{max}^{emp}. The Figure 18 illustrates this error with a varying ε for several window radii. Observe first that the error depends on the window radius r since it generally cannot be minimized over all window radii when the amount of noise increases. However, the Figure 18 also shows that choosing a value of $\varepsilon < 0.05$ is clearly too restrictive. In light of this experiment, we empirically set $\varepsilon = 0.05$ for this moment.

Automatic tuning of η and r

In the continuity of the previous paragraph, we now explain how we can jointly obtain

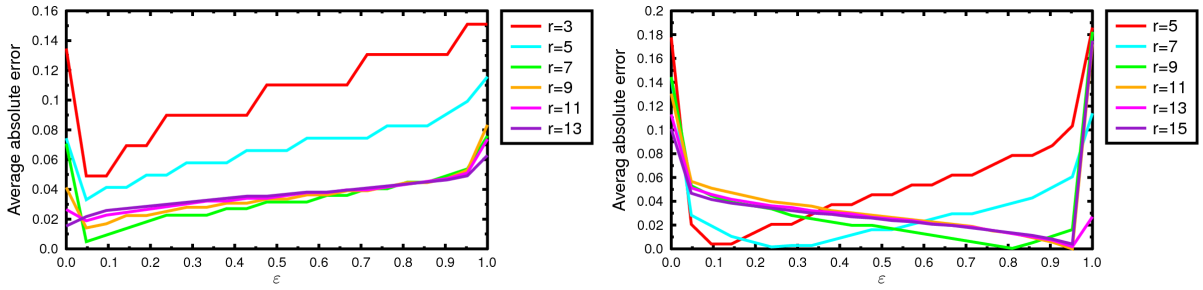


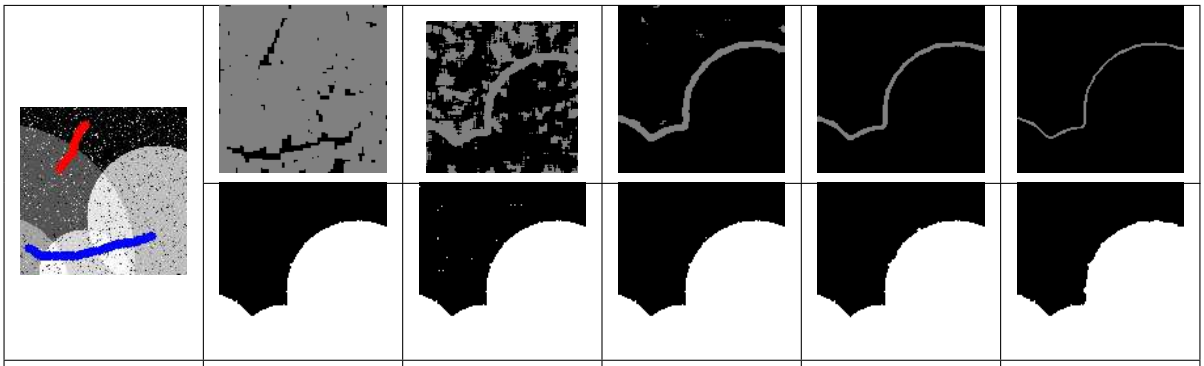
Figure 18: Evolution of the average absolute error between the empirical upper bound η_{max}^{emp} and the statistical upper bound η_{max} versus the parameter ε, for a fixed amount of noise $\xi = 0.05$ (left) and $\xi = 0.2$ (right). Color curves correspond to distinct window radii.

reliable estimates of the parameters r^* and η^* such that all nodes of $\mathcal{P}$ are very likely to be removed from $\mathcal{G}$. Notice first that if $\eta^+ < \eta_{min}$, it means that the test (32) can fail for some nodes in $\mathcal{G}$. However, as discussed before, we can increase the window radius r to increase the gap $\Delta\eta$ between the lower bound η_{min} and the upper bound η_{max}.

A good window radius r^* can be therefore obtained by progressively increasing it from one until $\eta^+ \geq \eta_{min}$. Notice that the number of iterations must be bounded to prevent an infinite loop. Once the optimal radius r^* computed, η^* can for instance be set as using (45). Empirically, we have found that the values of parameters r^* and η^* are generally near to empirical values obtained for segmenting I with a varying amount of noise ξ. Nevertheless, these estimations become less accurate as the amount of noise ξ tends to $\frac{1}{2}$.

b) Further lowering the graph size η

We now detail how the parameter η can be used for reducing the memory usage. The Figure 19 illustrates how far the condition (32) can be relaxed for further reducing graphs while getting nearly the same segmentation. In this experiment, the segmentation and the reduced graph are shown for segmenting a synthetic noisy 2D image with a Boykov-Jolly model using connectivity 1. Since the condition (36) becomes easier to satisfy when η decreases, the graph around the object contours becomes thicker.



η	1.0	0.9	0.8	0.7	0.6
ρ	93.28%	30.99%	5.74%	3.65%	2.00%

Figure 19: Memory gain when segmenting a 2D synthetic image corrupted by 10% of impulsive noise, using a Boykov-Jolly model (left). Top row shows the nodes of the reduced graph in light gray while bottom row shows the corresponding segmentation. In this experiment, we set $r = 3$, $\gamma = 1$ and use connectivity 1. In this experiment, $\eta_{min} \simeq 0.571$.

c) Filtering

The parameter η can also be used for filtering the segmentation. Indeed, this parameter can be tuned to remove small undesired regions in the segmentation due to noise. This behavior is illustrated in Figure 20 for segmenting a 3D noisy image obtained from a confocal microscope using a Boykov-Jolly model. In this picture, white spots correspond to cell nuclei in a mouse cerebellum. Observe how the filtering acts for small values of η: small regions in the graph and in the segmentation are progressively removed as η decreases. This parameter can be typically useful for filtering images corrupted by a noise behaving like an impulsive noise. Notice that unlike traditional filters, RGC do not require any pre or post-processing steps to filter segmentations.

The robustness (see Figures 21 and 22) and the sensitivity to noise of the parameter η (see Figure 23) are now analyzed. Let us describe the experimental setup. The experiment consists in segmenting four grayscale and five color 2D images with an increasing noise level ranging from 4 to 48%. We use the Boykov-Jolly model in connectivity 1 using GMM for estimating the distribution laws of the object and the background. For each image, we compute a reference segmentation on the noise-free image by tuning the seeds and

parameters by hand. Using the approach of [RKB04], the parameter σ is automatically set as

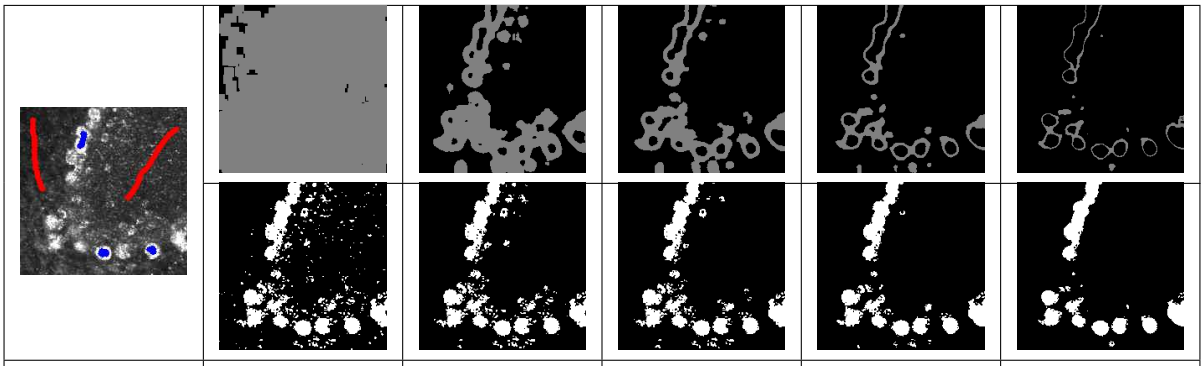
$$\sigma = \sqrt{\frac{1}{\sharp\mathcal{P}} \sum_{(p,q)\in\mathcal{N}} \|I_p - I_q\|_2^2},$$

Then, for each impulsive noise level, we select the segmentation maximizing the DSC (see Appendix A) between the reference image and all segmentations obtained through a fixed range of r and η values. Window radii range from $r = 1$ to $r = 12$ whereas η values range from $\eta = 1$ to η_{min}. Each segmentation is computed using the same seeds and parameters as for the reference segmentation. Again, the σ parameter is automatically set as before. Notice also that the choice of the evaluation measure in this experiment is important. Due to the noise sensitivity of other evaluation measures like MSASD or SRMSSD (see Appendix A), our choice naturally fell on the DSC, less sensitive to small perturbations.

As shown in Figure 21, for an impulsive noise level up to 45%, the parameter η appears to be reasonably robust with a DSC always greater than 94% for all images, except for the image "rice". However, such high and stable noise robustness can be reached at the expense of a more important amount of seeds. The reason why the algorithm behaves poorly on the "rice" image is the following. As said earlier, r must be large enough when the amount of noise increases for removing a maximum number of segments due to noise. This implies wider bands in $\mathcal{G}'$ around the object contours. Nevertheless, the object contours further oscillate as the amount of noise increases since the uncertainty grows inside the band due to noise. Another reason is due to the proximity of the objects to segment. As an illustration, consider two circles over a uniform background, separated by a distance $d_0 > 0$. We clearly see that the test (36) becomes more and more difficult to satisfy when the window radius r increases. When $(2r+1) \geq d_0$, the reduced graphs of both circles fuse into one component. This is typically the case in the image "rice" since this photo consists of small assembled rice grains.

Figure 23 also highlights that the parameter η is not very sensitive to the variations of r and η. The DSC does not vary much, except for the image "rice". This exception can be explained for the same reasons as before.

More generally, the Figures 21, 23 and 22 also show the limits of the parameter η when



η	1.0	0.9	0.8	0.7	0.6
ρ	55.70%	37.15%	18.26%	12.65%	8.87%

Figure 20: Simultaneous segmentation and filtering of a 3D image using a Boykov-Jolly model (left) in connectivity 1. In this picture, white spots correspond to cell nuclei in a mouse cerebellum. Top row shows the nodes of the reduced graph in light gray while bottom row shows the corresponding segmentation. In this experiment, we set $r = 5$ and $\gamma = 1$.

the amount of noise is particularly large. To overcome these problems, the solution probably would be to identify structures belonging to the object in the window surrounding each pixel.

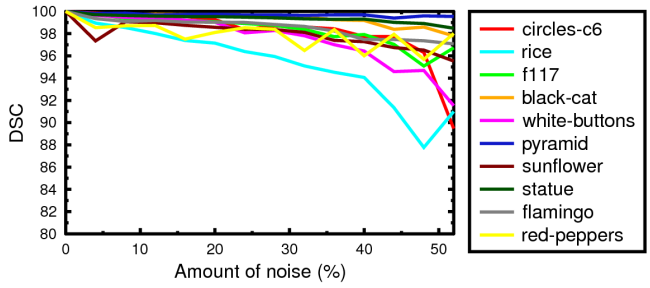


Figure 21: Quantitative analysis of the robustness to noise for segmenting four 2D grayscale images (top-most curves in the list) and five 2D color images with an impulsive noise level ranging from 4 to 48%.

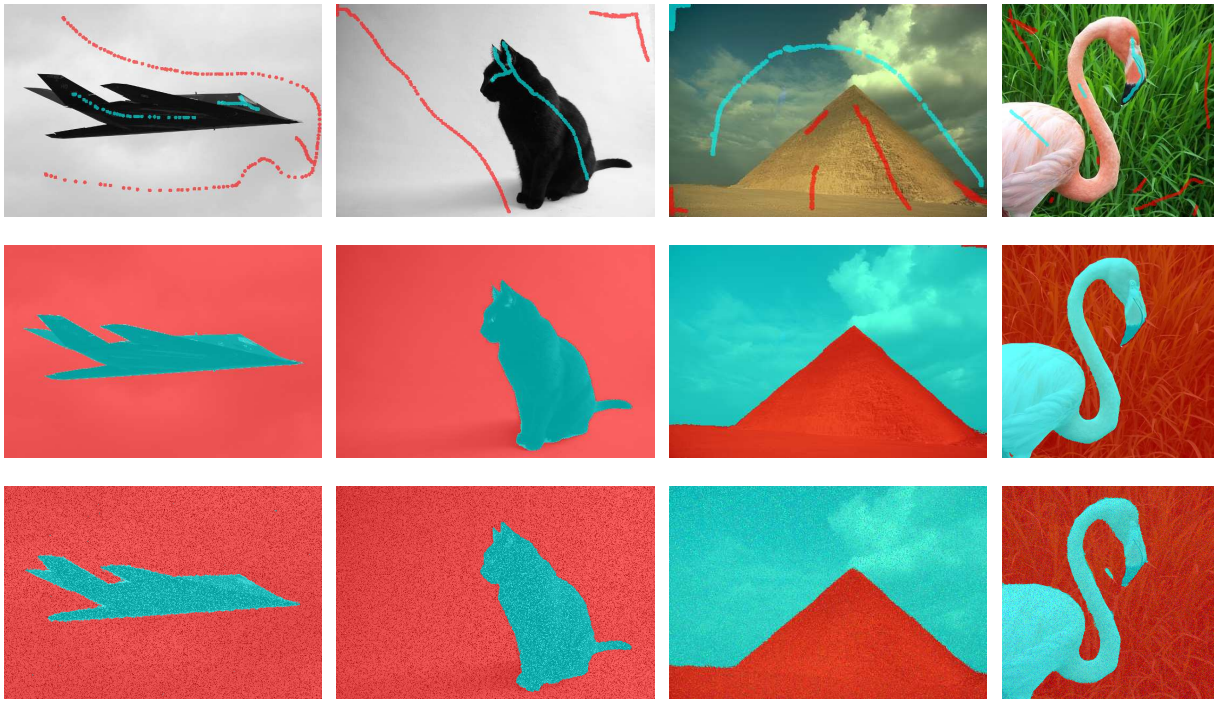


Figure 22: Qualitative analysis of the robustness to noise for segmenting the images "f117" (left-most column), "black-cat" (left column), "pyramid" (right column) and "flamingo" (right-most column) with a fixed impulsive noise level of 36%. The seeds and the model parameters are the same than those used in Figure 21 (top row).

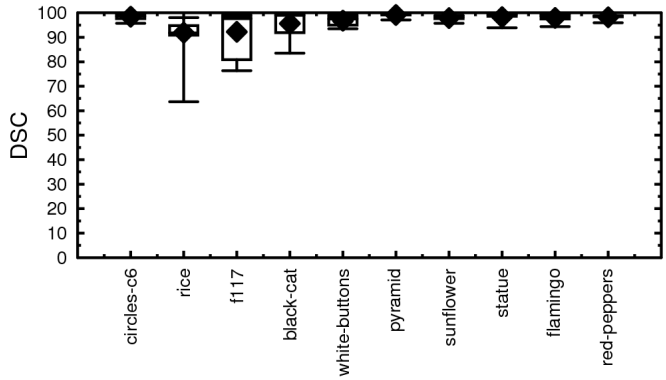


Figure 23: Sensitivity of η for segmenting the images in Figure 21 with an impulsive noise level of 36%. The seeds and model parameters are the same than those used in Figure 21.

3.3 An exact test for reducing graphs

Although the massive numerical experiments presented in Section (3.2) clearly exhibit an overall good performance with low pixel error on segmentations, we do not have demonstrated yet that the test (32) preserves the value of the flow in $\mathcal{G}$. This test therefore remains temporarily a band-based heuristic. This drawback has mainly motivated the proposed work. In this section, we propose an exact test to reduce the graphs. In what follows, we first describe this new test for reducing graphs in Section 3.3.1. We also provide comprehensive numerical experiments in Section 3.3.2 and show that we obtain reduction performance similar to those obtained with the test (32).

3.3.1 Principle

Throughout this section, we consider a fixed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as defined in Section 1.2.2 and a set $B \subset \mathbb{Z}^d$ $(d > 0)$. We also assume that $\mathcal{G}$ and B satisfy (29). We propose to test

$$\begin{cases} \text{either} \quad \forall q \in B_p, \quad c_q \geq 0 \quad \text{and} \quad c_q \geq +\sum_{\substack{q' \in \sigma_{\mathcal{E}}(q) \\ q' \notin B_p}} c_{q,q'}, \\ \text{or} \qquad \forall q \in B_p, \quad c_q \leq 0 \quad \text{and} \quad c_q \leq -\sum_{\substack{q' \in \sigma_{\mathcal{E}}(q) \\ q' \notin B_p}} c_{q,q'}. \end{cases} \tag{46}$$

Theorem 5. *Let $\mathcal{G}$ be the graph defined in Section 1.2.2, let B satisfy* (29) *and let us assume that $p \in \mathcal{P}$ satisfies* (46)*. Then, there exists a max-flow f in $\mathcal{G}$ such that*

$$\forall q \in \sigma_{\mathcal{E}}(p), \quad f_{p,q} = f_{q,p} = 0.$$

As a consequence, removing the node p from the graph $\mathcal{G}$ does not modify its max-flow value.

The proof of the Theorem 5 is contained in the Appendix C. For simplicity, we only prove the Theorem 5 when the node p satisfies the first condition of (46). Algorithmically, the above theorem guarantees that we can test every node, during the graph construction, before it is added to the graph. If the node satisfies (46), it is not useful to the max-flow evaluation and can be removed without alteration of the max-flow value. Since all nodes $p \in \mathcal{P}$ are individually tested using (46), this readily leads to an algorithm with a

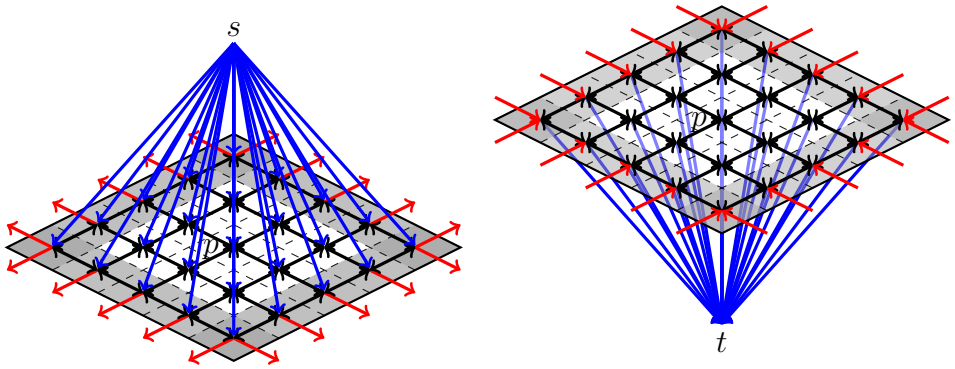


Figure 24: Illustration of the exact test (46). In both situations, we remove the central node p from $\mathcal{G}$ since the contracted capacity of any node $q \in \partial B_p$ is either greater than the sum of the n-links capacities in $\sigma_{\mathcal{E}}(q)$ (left image) or less than the opposite sum of the n-links capacities in $\sigma_{\mathcal{E}}(q)$ (right image).

worst-case complexity of $O(\sharp\mathcal{P}\sharp B\frac{d_{\mathcal{G}}}{2})$, where $d_{\mathcal{G}} = \max\{\sharp\sigma_{\mathcal{E}}(p) \mid p \in \mathcal{P}\}$ is the maximum node degree in $\mathcal{G}$. This algorithm has clearly a superior complexity compared to the one of Algorithm 2 described in Section 3.2.2.2.

It is however straightforward to lower this complexity by finding, for any node $p \in \mathcal{P}$, an upper bound on the sum of n-links capacities for each node $q \in B_p$. On the other hand, such approximation would slightly deteriorate the reduction performance. Compared to Algorithm 2, we want also to point out that the proposed algorithm for the test (46) is fully parallelizable as discussed in Section 3.2.2.3.

Remark that the test (32) introduced in Section 3.2 is less restrictive than the test (46). Indeed, the test (32) is clearly stronger than the test (46) for all nodes in $B_p \setminus \partial B_p$. Notice also that the terms involved in each test in (46) do not depend explicitely on the window radius r as the test (32). The general trend is that the relative reduced graph size ρ is always minimum for $r = 1$. Finally, one could also easily embed the parameters γ and η described in Section 3.2.3.3 and 3.2.3.4 for further reducing the graphs albeit the exactness of the test is not anymore guaranteed.

3.3.2 Massive experiments on 2D, 2D+t and 3D images

In this section, the performance of standard graph cuts (SGC) versus reduced graph cuts (RGC) is compared in terms of speed and memory consumption for reducing graphs. We also estimate the distance between the segmentations obtained with SGC and RGC as well as the relative max-flow error between $\text{val}_{\mathcal{G}'}(f'^*)$ and $\text{val}_{\mathcal{G}}(f^*)$ for evaluating the efficiency (or not) of the test (46). In the following massive experiments, we always set $\gamma = 1$ and $\eta = 1$.

Additionally, we also compare the relative reduced graph sizes obtained with the tests (32) (see Section 3.2.1) and (46) (see Section 3.3.1) when using the same seeds, parameters and images as in Section 3.2.1. Also, remark that the experimental setup adopted is the same as in Section 3.2.3.2. For the sake of clarity, we first remind our procedure for segmenting such data using the Boykov-Jolly and the TV+L^2 models in connectivity 1.

For each image, the best segmentation is achieved by using the same seeds and parameters as in Section 3.2. Using these seeds and parameters, a reference segmentation is computed with SGC. Then, a second segmentation is computed with RGC using the same set of seeds and parameters. The differences between both segmentations are then assessed using DSC and MSASD and the relative max-flow error between $\text{val}_{\mathcal{G}}(f^*)$ and $\text{val}_{\mathcal{G}'}(f'^*)$ is provided (see Appendix A). For each image, we also provide the scalar $\Delta\rho^*$ measuring the difference between the optimal relative reduced graph size ρ^* respectively obtained using the tests (32) and (46). In words, the test (32) is more efficient than the test (46) when $\Delta\rho^* > 0$ and conversely.

On the technical side, all experiments are performed on an Athlon Dual Core 6000+ 3GHz with 2GB of RAM using the max-flow algorithm of [BK04] [6]. Running times include the graph construction, the max-flow computation as well as the construction of the solution. Times are averaged over 10 runs.

The results of the test (46) using a TV+L^2 model (see Section 1.2.5.1) are summarized

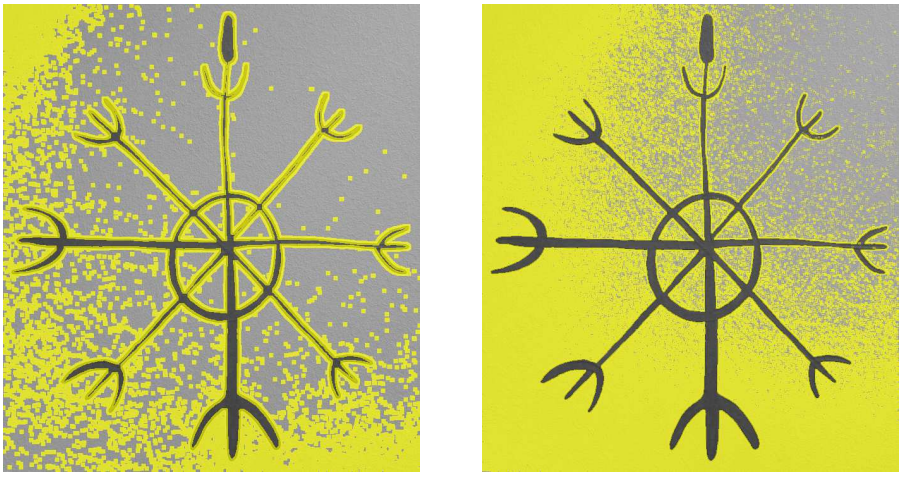
[6] The code is freely available at `http://www.cs.cornell.edu/People/vnk/software.html`

in Table 5. Reduction performance of the test (46) is also compared to the test (32) and illustrated in Figure 25. Similarly, we summarize the outcomes of the test (46) using a Boykov-Jolly model (see Section 1.2.5.2) with NH and GMM respectively in Tables 3 and 4. We also illustrate the reduction performance of the test (46) against (32) in Figure 26. Let us now take an in-depth look at the results obtained.

As the test (32), we observe that the test (46) globally outperforms SGC in terms of memory while the differences between both segmentations and between max-flow values are generally null (or remain extremely small). This situation is reassuring and confirms the consistency between theoretical and experimental results for the test (46). For some images, SGC fail to compute the segmentation (due to the high memory requirements) while RGC demonstrate their ability to segment them in a reasonable time. Furthermore, the outcomes of Tables 5, 6 and 7 and those described in Section 3.2.3.2 also imply that the segmentations obtained using the tests (32) and (46) are nearly the same. The larger theoretical complexity of the test (46) is also confirmed by larger running times than those obtained with SGC. However, notice that this complexity could be improved as with the test (32). Indeed, one can easily find an upper bound on the sum of n-links capacities for each $q \in \partial B_p$.

From a memory point of view, the massive experiments in Tables 5, 6 and 7 also show that the test (46) is globally less efficient than the test (32) with a negative average $\Delta\rho^*$ over all images. This least performance is strengthened for the images where the amount of regularization is very large (see for instance the images "circles", "sweets", "cells", etc.). Indeed, in such a situation, a large window radius can be used in the test (32) for decreasing δ_r and the reduced graph size whereas the test (46) cannot be relaxed. Thus, the relative reduced graph size ρ is minimum for the test (46) when $r = 1$. However, when the amount of regularization is of moderate level, the reduction performance of both tests are almost similar. Let us now describe the results for each model.

For the Boykov-Jolly model, the average relative size of reduced graphs over all images using NH is 20.67% ($\pm$26.59%). For 28 images out of 31, RGC allocate less memory than SGC. Observe that the image "talk" cannot be segmented using the test (46) whereas it



$\rho^* = 37.06\%$ $\rho^* = 64.82\%$

Figure 25: Negative impact of the test (46) (right) against the test (32) (left) for segmenting the image "viking-symbol2" with a TV+L^2 model. On both images, the reduced graph is superimposed in yellow on the image. In these experiments, we take the same set of parameters than those used in Table 5. Observe how the test (32) can remove a large part of nodes due to the raking light, unlike the test (46).

can with the test (32). Additionally, the average $\Delta\rho^*$ over all images is -1.30% ($\pm$4.84%). When using GMM with the same energy model, the average relative size of reduced graphs drops to only 25.74% ($\pm$32.71%). For 27 images out of 31, RGC allocate less memory than SGC. Also, the average $\Delta\rho^*$ over all images is -0.746% ($\pm$3.67%). The difference of reduction performance between the TV+L^2 and Boykov-Jolly models lies in n-links capacities. Indeed, the n-links capacities smoothly vary according to the image content in the latter whereas they remain fixed in the former. Since the n-links capacities are smaller when the gradient is larger (see (25)), the test (46) becomes easier to satisfy than the test (32) around object contours and leads to a smaller reduced graph. Notice that this effect is enforced with GMM due to the better estimate of distribution laws (see Figure 26).

For the TV+L^2 model, the average relative size of reduced graphs over all images is 39.09% ($\pm$39.65%). For 20 images out of 28, RGC allocate less memory than SGC. In the manner of the test (32), the test (46) behaves poorly when the amount of regularization is large (see for instance the images "zen-garden" and "sweets"). As opposite, when the

$\Delta\rho^* = +0.27\%$ $\Delta\rho^* = +1.87\%$

Figure 26: Positive impact of the test (46) against the test (32) for segmenting the image "book" with a Boykov-Jolly model using NH (left) and GMM (right). Notice that we take in these experiments the same set of seeds and parameters than those used in Tables 6 and 7. On both images, we represent the reduced graphs using the test (32) (cyan) and (46) (green) superimposed on a zoomed version of the original image. The intersection is drawn in red. Observe how the test (46) results in thicker bands around the object contours unlike the test (32).

amount of regularization is large enough, the test (46) is sometimes slightly more efficient than the test (32) (see for instance the images "angiography2" and "ct-thorax"). This observation is typically due to the fact that the test (46) can be less conservative than the test (32) since a lower number of contracted capacities must be larger than some threshold. Additionally, the average $\Delta\rho^*$ over all images is -5.57% ($\pm$11.44%) and is slightly larger than using the test (32). This difference of reduction performance (in the particular case of the TV+L^2 model) can be explained as follows. Although the test (46) appears to be less conservative, the threshold in the test (46) is larger than the one involved in the test (32) as the dimensionality d and the window r increase. Let us now demonstrate it in connectivity 0. First, one can easily observe that the test (46) is more restrictive in corners of B_p since the number of neighbors outside B_p is larger. In this test, a node p is therefore removed from $\mathcal{G}$ if $\exists q \in B_p$ such that

$$|c_q| \geq d. \tag{47}$$

where we remind that d is the dimensionality. Given the connectivity type, the exact

value of δ_r in the test (32) can be readily computed as

$$\delta_r = \frac{2d(2r+1)^{d-1}}{(2r+1)^d - 1} \tag{48}$$

By plugging (48) into (32), we have the following statement: in the test (32), a node p is removed from $\mathcal{G}$ if $\exists q \in B_p$ such that

$$\begin{aligned} |c_q| \geq \tfrac{2d(2r+1)^{d-1}}{(2r+1)^d-1} \quad &\Leftrightarrow \quad |c_q| \geq \tfrac{2d(2d+1)^{d-1}}{(2r+1)^d\left(1-\frac{1}{(2r+1)^d}\right)} \\ &\Leftrightarrow \quad |c_q| \geq d \underbrace{\frac{2}{(2r+1)\left(1-\frac{1}{(2r+1)^d}\right)}}_{\Psi(r,d)}. \end{aligned} \tag{49}$$

For any dimensionality $d > 0$ and any window radius $r > 0$, it is therefore straightforward to see that the maximum of $\Psi(.)$ in (49) is reached for $d = 1$ and $r = 1$ and is equal to 1. Since $\Psi(.)$ is decreasing, this gives us a lower bound on $|c_q|$ in (49). One ends up with

$$d \geq |c_q| \geq d\Psi(r, d). \tag{50}$$

Comparing (50) and (47) readily explains why the test (46) is stronger than (32) in such particular case (see Figure 25). This calculus can be generalized to other connectivities.

Finally, as in Section 3.2.3.2, we have also compared the distance between segmentations obtained with NH and GMM using the DSC and the ASASD evaluation measures (see Appendix A), when both are available. The outcomes are almost similar with a mean DSC and a mean ASASD respectively equal to 98.87% ($\pm$1.25%) and 1.88 ($\pm$2.31).

	Volume name	Size	SGC		RGC		ρ^* (%)	$\Delta\rho^*$ (%)	RME	DSC (%)	MSASD
			Time	Memory	Time	Memory					
2D	plane	481 ×321	0.19	22.90 Mb	0.67	32.72 Mb	100.00	-50.11	0.0000	100.0000	0.0000
	zen-garden	481 ×321	0.09	22.90 Mb	0.68	32.72 Mb	100.00	-5.28	0.0000	100.0000	0.0000
	oriental-man	321 ×481	0.24	22.90 Mb	0.76	32.72 Mb	100.00	-20.15	0.0001	100.0000	0.0000
	ct-thorax-z	512 ×512	0.16	38.91 Mb	0.89	2.05 Mb	5.80	-0.06	0.0000	100.0000	0.0000
	book	3012 ×2048	2.82	917.26 Mb	18.64	78.95 Mb	8.59	+0.05	0.0000	100.0000	0.0000
	cells-z	512 ×512	0.19	38.91 Mb	1.12	49.08 Mb	99.97	-23.28	0.0011	100.0000	0.0000
	beans	256 ×256	0.06	9.70 Mb	0.29	10.40 Mb	100.00	-0.17	0.0001	100.0000	0.0000
	sweets	800 ×600	0.80	71.28 Mb	2.40	78.95 Mb	100.00	-16.03	0.0000	100.0000	0.0000
	text1	1600 ×1200	0.82	285.39 Mb	5.84	25.76 Mb	10.53	+0.03	0.0000	100.0000	0.0000
	text2	1024 ×768	0.36	116.84 Mb	2.54	35.09 Mb	29.36	-0.58	0.0000	100.0000	0.0000
	yeasts1	512 ×512	0.22	38.91 Mb	1.09	49.08 Mb	97.73	-2.41	0.0016	100.0000	0.0000
	yeasts2	512 ×512	0.19	38.91 Mb	0.94	6.93 Mb	19.63	-0.8	0.0001	99.9730	0.0000
	angiography1	512 ×512	0.16	38.91 Mb	0.89	3.08 Mb	7.96	+0.01	0.0001	100.0000	0.0000
	angiography2	350 ×643	0.13	33.39 Mb	0.72	2.26 Mb	7.47	+0.04	0.0000	100.0000	0.0000
	f117	588 ×392	0.11	34.20 Mb	0.70	686.19 Kb	2.15	-0.44	0.0000	100.0000	0.0000
	black-cat	600 ×400	0.11	35.61 Mb	0.73	415.38 Kb	1.16	-0.02	0.0000	100.0000	0.0000
	viking-symbol2	660 ×740	0.29	72.53 Mb	1.72	52.63 Mb	64.82	-27.76	0.0000	100.0000	0.0000
	buttons	300 ×300	0.05	13.33 Mb	0.29	934.60 Kb	7.03	-0.12	0.0000	100.0000	0.0000
	rice	256 ×256	0.03	9.70 Mb	0.24	2.05 Mb	21.93	-0.16	0.0002	100.0000	0.0000
	blood-cells	425 ×280	0.08	17.64 Mb	0.42	3.08 Mb	18.95	-0.11	0.0000	100.0000	0.0000
	poppy	409 ×613	0.16	37.21 Mb	0.76	1.01 Mb	2.92	+0.01	0.0002	100.0000	0.0000
2D+t	interview-girl	320 ×240 ×150	**OM**	4.72 Gb	294.60	771.00 Mb	17.41	-2.37	**NSR**	**NSR**	**NSR**
	interview-old-man	256 ×256 ×128	**OM**	3.43 Gb	216.42	771.00 Mb	21.22	-2.39	**NSR**	**NSR**	**NSR**
	interview-woman	352 ×288 ×154	**OM**	6.40 Gb	393.31	1.13 Gb	15.62	-2.26	**NSR**	**NSR**	**NSR**
3D	ct-thorax	409 ×409 ×252	**OM**	17.33 Gb	1084.72	1.17 Gb	7.83	+0.04	**NSR**	**NSR**	**NSR**
	circles	128 ×128 ×128	4.78	874.57 Mb	58.61	1.10 Gb	94.99	-0.14	0.0138	100.0000	0.0000
	cells	409 ×409 ×101	**OM**	6.92 Gb	430.15	1.17 Gb	18.53	-1.07	**NSR**	**NSR**	**NSR**
	brain-p3	181 ×217 ×181	**OM**	2.91 Gb	182.24	354.67 Mb	13.17	-0.54	**NSR**	**NSR**	**NSR**

Table 5: Standard graph cuts (SGC) are compared to Reduced Graph Cuts (RGC) in terms of speed (in secs) and memory for segmenting data using a TV+L^2 model in connectivity 1. Label **OM** stands for "Out of Memory" while label **NSR** stands for "No Segmentation Reference".

	Volume name	Size	SGC		RGC		ρ^* (%)	$\Delta\rho^*$ (%)	RME	DSC (%)	MSASD
			Time	Memory	Time	Memory					
2D	eagle-c	481 ×321	0.20	22.90 Mb	1.05	1.37 Mb	5.49	+0.05	0.0000	100.0000	0.0000
	zen-garden-c	481 ×321	0.22	22.90 Mb	1.23	23.39 Mb	90.24	+0.51	0.0000	100.0000	0.0000
	columns-c	481 ×321	0.22	22.90 Mb	1.32	276.92 Kb	1.17	0.0	0.0000	100.0000	0.0000
	red-flowers-c	481 ×321	0.20	22.90 Mb	1.06	10.40 Mb	46.26	-22.96	0.0000	100.0000	0.0000
	snow-and-clouds-c	481 ×321	0.19	22.90 Mb	1.03	4.62 Mb	15.24	0.0	0.0000	100.0000	0.0000
	marker-c	481 ×321	0.20	22.90 Mb	1.06	457.46 Kb	2.05	+0.41	0.0000	100.0000	0.0000
	pyramid-c	481 ×321	0.18	22.90 Mb	1.01	276.92 Kb	1.29	+0.17	0.0000	100.0000	0.0000
	red-starfish-c	481 ×321	0.18	22.90 Mb	1.01	3.08 Mb	13.68	+0.27	0.0000	100.0000	0.0000
	black-cow-c	481 ×321	0.20	22.90 Mb	1.09	203.32 Kb	1.03	+0.48	0.0000	100.0000	0.0000
	church2-c	481 ×321	0.20	22.90 Mb	1.12	1.37 Mb	5.07	0.0	0.0000	100.0000	0.0000
	snake2-c	481 ×321	0.20	22.90 Mb	1.02	934.60 Kb	4.14	+0.81	0.0000	100.0000	0.0000
	birds2-c	321 ×481	0.21	22.90 Mb	1.03	1.37 Mb	5.98	+0.42	0.0001	100.0000	0.0000
	eagle2-c	481 ×321	0.18	22.90 Mb	0.99	623.05 Kb	2.25	0.0	0.0000	100.0000	0.0000
	greek-temple-c	481 ×321	0.20	22.90 Mb	1.08	2.05 Mb	8.03	0.0	0.0000	100.0000	0.0000
	horses4-c	481 ×321	0.19	22.90 Mb	1.09	1.37 Mb	5.43	+0.17	0.0000	100.0000	0.0000
	meadow-and-mountains-c	481 ×321	0.20	22.90 Mb	1.12	15.59 Mb	55.99	+0.01	0.0000	100.0000	0.0000
	traditional-houses-c	481 ×321	0.20	22.90 Mb	1.06	934.60 Kb	4.13	+0.17	0.0000	100.0000	0.0000
	ct-thorax-z	512 ×512	0.40	38.91 Mb	1.87	4.62 Mb	10.85	0.0	0.0000	99.7832	0.0000
	book	3012 ×2048	7.59	917.26 Mb	42.48	78.95 Mb	7.91	+0.27	0.0000	100.0000	0.0000
	cells-z	512 ×512	0.48	38.91 Mb	2.08	23.39 Mb	61.65	-12.74	0.0007	100.0000	0.0000
	text1	1600 ×1200	2.15	285.39 Mb	12.71	177.63 Mb	50.18	0.0	0.0000	100.0000	0.0000
	viking-symbol2	660 ×740	0.62	72.53 Mb	3.05	3.39 Mb	5.13	0.0	0.0023	99.9981	0.0000
2D+t	interview-man1-c	320 ×240 ×203	**OM**	6.40 Gb	888.53	354.67 Mb	6.49	+0.42	**NSR**	**NSR**	**NSR**
	interview-man2-c	426 ×240 ×180	**OM**	7.55 Gb	1018.84	228.44 Mb	3.21	0.0	**NSR**	**NSR**	**NSR**
	plane-take-off-c	492 ×276 ×180	**OM**	10.03 Gb	1346.40	532.00 Mb	6.09	+0.11	**NSR**	**NSR**	**NSR**
	talk-c	370 ×276 ×190	**OM**	7.96 Gb	**OM**	**OM**	**OM**	**OM**	**NSR**	**NSR**	**NSR**
	fluorescent-cell-c	478 ×396 ×121	**OM**	9.39 Gb	1490.55	514.00 Mb	5.88	0.0	**NSR**	**NSR**	**NSR**
3D	ct-thorax	245 ×245 ×151	**OM**	3.71 Gb	493.23	771.00 Mb	17.30	0.0	**NSR**	**NSR**	**NSR**
	circles	128 ×128 ×128	8.16	874.57 Mb	120.54	1.10 Gb	94.41	+0.16	0.0043	100.0000	0.0000
	cells	230 ×230 ×57	9.40	1.23 Gb	156.79	771.00 Mb	59.38	-8.00	0.0029	100.0000	0.0000
	brain-p3	181 ×217 ×181	**OM**	2.91 Gb	381.72	771.00 Mb	24.22	+0.16	**NSR**	**NSR**	**NSR**

Table 6: Standard graph cuts (SGC) are compared to Reduced Graph Cuts (RGC) in terms of speed (in secs) and memory for segmenting data using a Boykov-Jolly model in connectivity 1 using NH. Label **OM** stands for "Out of Memory" while label **NSR** stands for "No Segmentation Reference". Color images names are suffixed by "c".

	Volume name	Size	SGC		RGC		ρ^* (%)	$\Delta\rho^*$ (%)	RME	DSC (%)	MSASD
			Time	Memory	Time	Memory					
2D	eagle-c	481 ×321	0.74	22.90 Mb	1.59	1.01 Mb	5.19	+0.08	0.0000	100.0000	0.0000
	zen-garden-c	481 ×321	1.28	22.90 Mb	2.23	23.39 Mb	98.75	0.0	0.0000	100.0000	0.0000
	columns-c	481 ×321	1.01	22.90 Mb	1.99	623.05 Kb	2.35	0.0	0.0000	100.0000	0.0000
	red-flowers-c	481 ×321	0.95	22.90 Mb	1.79	6.93 Mb	23.37	-1.57	0.0000	100.0000	0.0000
	snow-and-clouds-c	481 ×321	1.62	22.90 Mb	2.65	23.39 Mb	98.83	+0.39	0.0000	100.0000	0.0000
	marker-c	481 ×321	0.82	22.90 Mb	1.69	184.62 Kb	0.80	+0.09	0.0000	100.0000	0.0000
	pyramid-c	481 ×321	0.91	22.90 Mb	1.70	457.46 Kb	2.02	+0.02	0.0000	100.0000	0.0000
	red-starfish-c	481 ×321	0.87	22.90 Mb	1.68	934.60 Kb	4.14	+0.14	0.0000	100.0000	0.0000
	black-cow-c	481 ×321	0.81	22.90 Mb	1.68	184.62 Kb	0.89	+0.34	0.0000	100.0000	0.0000
	church2-c	481 ×321	1.05	22.90 Mb	1.88	1.37 Mb	6.56	0.0	0.0000	100.0000	0.0000
	snake2-c	481 ×321	0.75	22.90 Mb	1.54	1.51 Mb	8.79	-0.11	0.0000	100.0000	0.0000
	birds2-c	321 ×481	0.89	22.90 Mb	1.68	10.40 Mb	39.16	0.0	0.0002	100.0000	0.0000
	eagle2-c	481 ×321	0.85	22.90 Mb	1.66	1.37 Mb	5.18	0.0	0.0000	100.0000	0.0000
	greek-temple-c	481 ×321	1.11	22.90 Mb	1.99	6.93 Mb	26.55	0.0	0.0000	100.0000	0.0000
	horses4-c	481 ×321	0.77	22.90 Mb	1.63	1.51 Mb	6.80	+0.2	0.0000	100.0000	0.0000
	meadow-and-mountains-c	481 ×321	1.14	22.90 Mb	2.11	23.39 Mb	97.94	0.0	0.0000	100.0000	0.0000
	traditional-houses-c	481 ×321	0.95	22.90 Mb	1.82	1.37 Mb	5.08	+0.05	0.0000	100.0000	0.0000
	ct-thorax-z	512 ×512	0.88	38.91 Mb	2.36	3.08 Mb	7.67	+0.2	0.0058	100.0000	0.0000
	book	3012 ×2048	24.28	917.26 Mb	59.09	78.95 Mb	8.03	+1.87	0.0000	100.0000	0.0000
	cells-z	512 ×512	0.73	38.91 Mb	2.38	35.09 Mb	78.87	-18.87	0.0016	100.0000	0.0000
	text1	1600 ×1200	5.80	285.39 Mb	15.64	35.09 Mb	12.18	+0.37	0.0000	100.0000	0.0000
	viking-symbol2	660 ×740	1.23	72.53 Mb	3.61	3.08 Mb	4.24	+0.01	0.0000	100.0000	0.0000
2D+t	interview-man1-c	320 ×240 ×203	**OM**	6.40 Gb	971.89	354.67 Mb	6.43	+0.96	**NSR**	**NSR**	**NSR**
	interview-man2-c	426 ×240 ×180	**OM**	7.55 Gb	1138.97	514.00 Mb	7.02	0.0	**NSR**	**NSR**	**NSR**
	plane-take-off-c	492 ×276 ×180	**OM**	10.03 Gb	1468.28	771.00 Mb	6.34	+0.02	**NSR**	**NSR**	**NSR**
	talk-c	370 ×276 ×190	**OM**	7.96 Gb	1155.27	1.69 Gb	23.16	-8.36	**NSR**	**NSR**	**NSR**
	fluorescent-cell-c	478 ×396 ×121	**OM**	9.39 Gb	1632.68	1.08 Gb	11.85	-0.48	**NSR**	**NSR**	**NSR**
3D	ct-thorax	245 ×245 ×151	**OM**	3.71 Gb	508.84	514.00 Mb	13.72	0.0	**NSR**	**NSR**	**NSR**
	circles	128 ×128 ×128	12.98	874.57 Mb	125.90	1.10 Gb	94.53	-0.01	0.0056	100.0000	0.0000
	cells	230 ×230 ×57	12.70	1.23 Gb	161.72	798.00 Mb	67.12	+0.2	0.0003	100.0000	0.0000
	brain-p3	181 ×217 ×181	**OM**	2.91 Gb	390.40	771.00 Mb	24.51	+1.31	**NSR**	**NSR**	**NSR**

Table 7: Standard graph cuts (SGC) are compared to Reduced Graph Cuts (RGC) in terms of speed (in secs) and memory for segmenting data using a Boykov-Jolly model in connectivity 1 using GMM. Label **OM** stands for "Out of Memory" while label **NSR** stands for "No Segmentation Reference". Color images names are suffixed by "c".

3.4 Conclusion

In this chapter, we have presented two simple tests for reducing heuristically and exactly graphs involved in graph cuts-based binary image segmentation. For the heuristic test (32), we have proposed an efficient implementation of complexity $O(\sharp\mathcal{P})$ (except for image borders), which is in particular independent of the window radius. The massive numerical experiments presented in Section 3.2.3.2 show that RGC globally outperform SGC in terms of memory with an average relative reduced graph size under 35% while keeping very low pixel error between segmentations obtained with and without RGC. The proposed implementation of the test (36) also runs faster for some images if the amount of regularization is of moderate level. For keeping good reduction performance when the amount of regularization is large, we have introduced two extra parameters for further reducing the graphs and removing small connected components due to noise in the segmentations. Lower and upper bounds have also been provided to automatically tune the parameter η with or without the window size.

More generally, the massive experiments presented in Section 3.2.3.2 and 3.3.2 clearly show that the test (46) offers less performance in terms of memory and speed compared to the test (32). In words, the exactness of the test (46) is at the expense of a larger computational cost. However, we have also seen that the complexity of the test (46) could be easily lowered with a slight deteriation of reduction performance. Furthermore, the solutions provided by the test (32) are near identical to those obtained with the test (46). On the other hand, an exact test has a deeper impact since it guarantees optimality on the solutions. As an illustration, the test (46) can be typically used in critical applications where accuracy is a fundamental requirement.

Finally, we want to mention that the tests described in Section 3.2 and 3.3 are currently protected by a patent [5]. Also, a detailed view of the massive numerical experiments presented in these sections are freely available at `http://lipn.fr/~lerme`.

4 GCSFMP: an application of RGC for segmenting lung tumors

4.1 Introduction

In this chapter, we address the problem of extracting interactively lung tumors in 3D CT images using graph cuts [7]. The originality of this work consists in (1) reducing input graphs using the method described in Section 3.2 to decrease the memory consumption of graph cuts and (2) introducing an original energy formulation with a prior on the seeds location: GCSFMP (Graph Cuts-Based Segmentation with Fast Marching Prior).

This prior is achieved by computing a distance map from object seeds and has a double role: localize the tumor into the CT image and alleviate the propagation of object seeds for reducing graphs. In the subsequent sections, we first detail the problem and its constraints. Afterwards, our strategy for achieving relevant segmentations of lung tumors is presented and evaluated against ground truth provided by an expert. Experiments show how the proposed method yields accurate results in a fast and memory efficient way.

4.1.1 Motivation and scope

The ability to exchange oxygen and carbon dioxide through the lungs is critical to life. In human beings, the lungs are separated by the mediastinum, posed on the diaphragm and protected by the rib cage. The left lung is divided into three lobes while the right lung is divided into two lobes (see Figure 1). Finally, air is conducted through a airway tree of bronchi composed of multiple ramifications. Finally, blood vessels follow bronchi pathways until alveoli where gaz exchanges occur.

In some patients, lungs can contain tumors, which are the most common manifestation of lung cancers. Tumors are classified according to their size as micronodules (less than 6 mm), nodules (from 6 mm to 30 mm) and masses (greater than 30 mm). They include

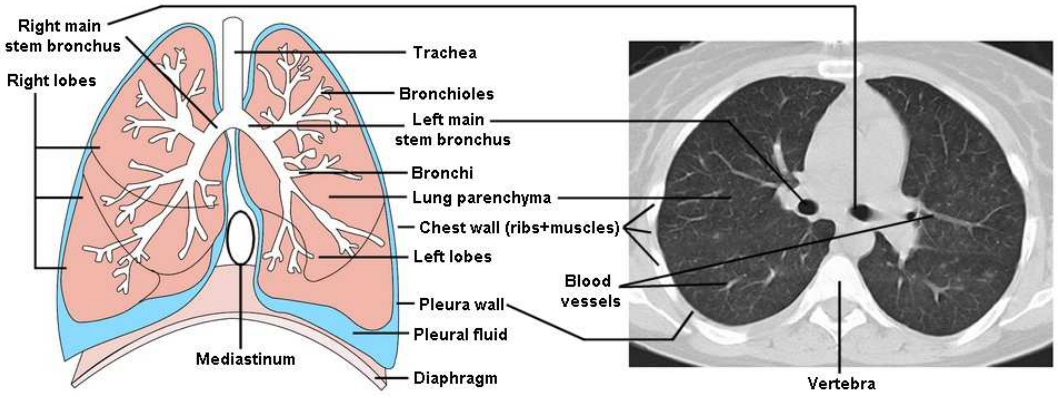


Figure 1: Detailed lung anatomy (left) and surrounding structures (right).

as well areas of ground-glass attenuation. However, all nodules and tumors are not malignant (cancerous). Some of them can also be benign (non-cancerous). Such structures are usually located using CT scans or X-ray radiographs. Their degree of malignancy is generally estimated according to several criteria among shape, size, volume and growing rate in time. The likelihood of malignancy is linked to the history of cancer of the patient and his immediate environment (smoking, pollution, asbestos, etc.). Treatment and prognosis depend on the type of cancer, its degree of dissemination and the health of the patient. Typical treatments include surgery, radiotherapy and chemotherapy. According to the World Health Organization (WHO), lung cancers currently represent the most frequent mortality cause in men and, after the breast cancer, in women. Lung cancer is also the most deadly cancer, responsible of about 1.37 million deaths worldwide in 2008.

Due to the moist and a well oxygenated environment, lungs also represent a perfect breeding ground for bacteria to grow. Infections compromise the ability to move oxygen through the system and can be very serious. Tuberculosis is the most common infection affecting lungs, killing about 1.4 million people worldwide in 2010, according to the WHO. This disease can cause cavities inside the lungs and generally appear as masses in CT scans. Although reduced in the fifties with the introduction of antibiotics, multiresistant strains of tuberculosis have recently emerged. Notice however that masses can be also the manifestation of a lung cancer. In this situation, the surgery can help bring more elements about the nature of these structures.

More generally, nodules are small spherical regions whereas masses have larger sizes

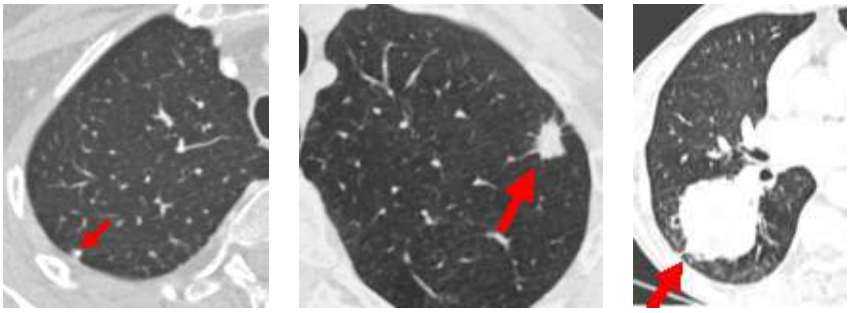


Figure 2: Examples of micronodule (left), nodule (middle) and masse (right) in CT scans.

and present irregular shapes. Nodules and masses may be connected to some extent to vessels, to bronchi, to the pleural wall or to the mediastinum (see Figure 2).

As a consequence, the detection and the measurement of the previous structures are crucial for establishing an early diagnosis of the pathology and improving the patient' survival rate. Accurate measurements of lung tumors sizes has become a challenging task for staging and assessing tumor response to treatments or its progression. While early assessment of the previous structures was performed manually on chest radiographs, the joint use of powerful imaging acquisition systems and computer-aided analysis play now a major role by making the diagnostics of radiologists more and more quick and accurate.

While a large number of measures have been proposed to assess the likelihood of malignancy, measuring size and more recently volumes of tumors/nodules remains a standard choice among radiologists. Revised RECIST criteria, largely used by radiologists, are based on the measurement of one diameter on a few number of lesions [vPvMGB10], and suffer from a lack of reproducibility [STJ+10]. Alternatively, tumor volumetry has been proposed to overcome those difficulties in order to improve the staging of nodules [GKMP09], the evaluation of tumor aggressiveness [QCS+08], tumor response to chemotherapy [BMG+10, ZSM+06] or to radiotherapy [MGBA+09] and the progression rate of tumors [QCS+08] or metastases [MAS+07]. Moreover, it becomes a necessary tool for the automatic screening of lung nodules on CT scans, and is currently on evaluation on ongoing trials [vKOP+09].

Originally, nodules, masses and tumors were manually detected and measured in CT images. The purpose of this work is therefore to provide tools to radiologists and

radiotherapists for making measurements easily. Our tool permits to draw interactively in 3D the shape of the lesion.

4.1.2 Constraints

Beside the main objective described in Section 4.1.1, the proposed method must faces several constraints. In what follows, we propose to detail them in an increasing order of importance and provide for each constraint, key references about the subject.

A major constraint of the method is to efficiently deal with regular and irregular shapes in the most transparent way. As a consequence, a shape prior cannot be embedded into the algorithm to increase the accuracy of the method. This main constraint naturally makes the initial problem more challenging. To our best knowledge, no work has been previously done to address this issue using graph cuts.

A second constraint is due to the fact that tumoral tissues typically appear in the same range of intensities than healthy tissues. This difficulty is also strengthened in situations where the tumors are directly connected to the pleural surface. Furthermore, no delineation is present between both tissues and is a problem for establishing accurate diagnostics even for clinicians. This requires a segmentation to rely on another criteria for correctly distinguishing tumoral tissues from healthy tissues. To tackle the issue of connection to the pleural surface, methods often make use of basic morphological operators [KRYH03, KDB+06, MLJM+09], clipping planes [RCY+06] or volumetric shape indexes [YBS09]. A classification of those methods can be found for instance in [SSPvG06] and [GKMP09].

In [YBS09], authors provide a robust method for automatically segmenting nodules. To our best knowledge, this technique is the only one to address this problem using graph cuts. The method works as follows. First, a volumetric shape index is computed for each voxel, based on local Gaussian and mean curvatures. This shape index represents the local shape feature at each voxel and assigns a unique value for every distinct shape. Afterwards, at each voxel, the shape index along with the image intensity and the spatial position are concatenated into a five-dimensional geometric feature vector. Mean-shift

clustering is applied on these vectors producing intensity and shape mode maps. The solution is regularized using graph cuts upon these maps. While the method gives astonishing results by detaching correctly nodules from surrounding undesired structures, the approach remains limited to a particular kind of structure and cannot be used to segment masses and tumors.

In [RCY+06], nodules are segmented by approximating their interface with the pleural surface as a plane. The algorithm iteratively pushes the plane toward the pleural surface and reorients it to minimize the volume error of the nodule. While such an algorithm seems to converge in a large number of cases (about 88.7% of success on a large dataset [RCY+06]), better results occur when the interface between the nodule and the pleural surface can be approximated by a plane. When this interface is convex, the algorithm is no longer guaranteed to converge. Furthermore, this technique can only be used for small nodules. Indeed, the interface between the pleural surface and the nodule becomes more and more curved as the nodule size grows due the rib cage curvature.

A last but not least constraint lies in the imaging acquisition system itself. Although these systems became more and more sophisticated and accurate over the last decades, the CT images produced by them can contain a large number of artifacts. These artifacts are almost represented by the partial volume effect but also by noise, ring, streak and motion artifacts. Partial volume effect occurs in medical imaging and more generally in biological imaging where a single voxel contains a mixture of multiple tissue values. This can be observed at the interface between different kinds of tissues (e.g. between gray and white matter in the brain) or when thin structures are smaller than the resolution of the acquisition system (e.g. blood vessels in the lungs). This effect is alleviated as the resolution increases but is strengthened by the Point Spread Function (PSF) of the acquisition system. The noise is caused by a low signal to noise ratio and occurs more commonly when a thin slice thickness is used. But it can also occur when the power supplied to the X-ray tube is insufficient to penetrate the anatomy. The ring artifact is a mechanical artifact which appears as small oscillations in the image. Streak artifacts can be observed around materials that block most x-rays such as metal or bones. They appear as radial rays emitting from the source in the images. Finally, motion artifacts

can be seen as blurring or streaking and are mainly caused by voluntary or involuntary movements of the patient inside the acquisition system during examination.

4.2 Proposed method

4.2.1 Overview

The underlying common assumption of graph cuts-based segmentation models like the Boykov-Jolly (see Section 1.2.5.2) and TV+L^2 (see Section 1.2.5.1) models, is that object and background parts appear in different range of intensities. For example, in the Boykov-Jolly, when this is not the case, we must reduce the importance of the data term by setting $\beta = 0$ to get consistent segmentation results. As a consequence, the result is only determined by the contrast-preserving function $B_{p,q}$ in (24).

However, setting the parameter $\beta = 0$ cannot be considered in this context since tumoral and healthy tissues appear in the same range of intensities in CT images. Therefore, setting $\beta > 0$ appears as the unique solution and relaxes the tendency to smoothness in regions of high contrast, as shown in [BJ01, RKB04]. Furthermore, RGC typically exclude camouflage situations (i.e. when a significant overlap exist between the distributions of the object and the background) and would lead in no improvement of the relative size of the reduced graph. Furthermore, due to the previous constraints, remind that in many situations (and in the experiments presented in Section 4.3) the tumors can be connected to the pleural surface and no delineation exists at the interface between both tissues.

To solve these issues, we propose to add in the Boykov-Jolly energy model a prior on the location of the tumor, obtained from the location of the object seeds. This is achieved by computing a distance map from object seeds. The role of this prior is twofold. First, it is used to locate and detach tumors from pleural surface. Second, the prior is also used to progressively lower the propagation of the object seeds as the distance increases from them. This allows us to keep a large value of β and categorize background voxels easily, i.e. when the distance from object seeds is too large.

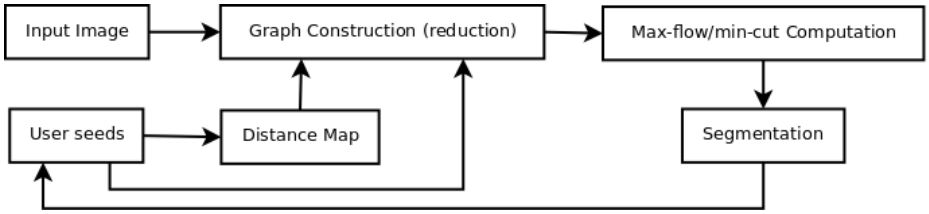


Figure 3: Flow diagram of our approach.

In words, the user puts object seeds inside the tumor and labels undesired parts with background seeds. Afterwards, a distance map is computed from the object seeds and a Region Of Interest (ROI) is extracted by thresholding the map. This threshold is set small enough for keeping voxels which appear sufficiently near from the tumor. Then, the graph is reduced in the ROI using the method described in Section 3.2. Finally, the max-flow is computed inside this graph using [BK04] and the segmentation is returned. The overview of the approach is outlined in Figure 3. We now detail our novel energy formulation.

4.2.2 Energy function

As discussed earlier, we propose to add in the Boykov-Jolly model a prior on the location of the tumor, obtained from the location of the object seeds. Embedding this prior naturally leads to a modification of (24) where the regularity criterion is taken as the same with a Gaussian weighting function (25), i.e.

$$E_{p,q}(u_p,u_q) = B_{p,q}\cdot|u_p-u_q| \qquad \text{and} \qquad B_{p,q} = \frac{1}{\|p-q\|_2}\cdot exp\Big(-\frac{(I_p-I_q)^2}{2\sigma^2}\Big), \quad (51)$$

where I is the input image of a single channel and σ is a contrast parameter. The parameter σ is automatically computed as described in Section 3.2.3.4. Also, the data term is modified in such a way to lower the importance of object seeds as the distance from them increases. The data term is defined in Table 8.

In Table 8, the distributions laws $\mathbb{P}(I_p \mid p\in\mathcal{O})$ and $\mathbb{P}(I_p \mid p\in\mathcal{B})$ are estimated using NH, $\sigma_a \in \mathbb{R}_*^+$ is a free parameter and $d(p,\mathcal{O})$ is a distance function between the point $p\in\mathcal{P}$ and object seeds $\mathcal{O}\subset\mathcal{P}$. The parameter σ_a controls how far object seeds propagate

	$p \in A_{\sigma_a}$	$p \notin A_{\sigma_a}$
$E_p(0)$	$-log\ \Big[\mathbb{P}(I_p \mid p \in \mathcal{O}) \times exp\Big(-\Big(\frac{d(p,\mathcal{O})}{\sigma_a}\Big)^2\Big)\Big]$	$+\infty$
$E_p(1)$	$-log\ \mathbb{P}(I_p \mid p \in \mathcal{B})$	0

Table 8: Definition of the data term.

from their location and then corresponds to an area of influence A_{σ_a} defined by

$$A_{\sigma_a} = \{p \in \mathcal{P} \mid d(p,\mathcal{O}) \leq 1 + \sigma_a\sqrt{-log(\varepsilon)}\} \quad \text{with} \quad \varepsilon \simeq 0.$$

Beyond this area, the nodes are only linked to the background terminal with a large capacity. A large capacity ensures that the corresponding node is labeled as background voxel. As a consequence, background voxels satisfy the test (32) and be removed from the graph. Although the parameter σ_a plays an important role that impacts the way of positioning the seeds in the image, we always set $\sigma_a = 10$ in our experiments.

In words, the main difference between the proposed energy and (24) lies in the distance term. The function $d(.)$ is defined as

$$d(p,\mathcal{O}) = \min\{dist(p,q) \mid q \in \mathcal{O}\}, \quad \forall p \in \mathcal{P}, \tag{52}$$

where $dist(.)$ denotes a notion of distance between two points. However, the results naturally depend on the choice of $dist(.)$. We have made two attempts for this function:

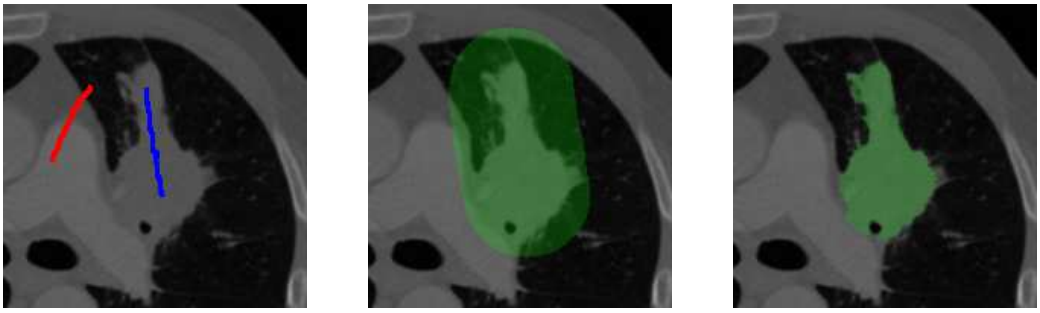
- **The Euclidean distance**. In this case, we only take into account spatial information. The distance between two nodes $p, q \in \mathcal{P}^2$ in $\mathcal{G}$ is defined as

$$dist(p,q) = \|p - q\|_2. \tag{53}$$

  Thus, embedding (53) in (52) amounts to find the shortest path from any node $p \in \mathcal{P}$ to the set $\mathcal{O}$. Notice that (53) is defined for any couple of nodes. This distance can be efficiently computed with the algorithm described in [FH04]. However, we mostly use it for the purpose of illustration.

- **The geodesic distance**. In this case, the distance between two adjacent nodes $(p,q) \in \mathcal{N}$ in $\mathcal{G}$ mixes spatial and intensity information and is defined as

$$dist(p,q) = \sqrt{(I_p - I_q)^2 + \alpha\|p - q\|_2}, \tag{54}$$



(a) Image and seeds (b) Euclidean distance (c) Geodesic distance

Figure 4: Area of influence for an Euclidean and a geodesic distance using $\sigma_a = 40$.

where $\alpha \in \mathbb{R}^+$ is a parameter. This time, plugging (54) in (52) amounts to minimize both the variations of intensities and the proximity of any node $p \in \mathcal{P}$ to the set $\mathcal{O}$. The parameter α acts as a trade-off between the Euclidean distance and the gradient norm. For instance, increasing α can help to further propagate distances within regions of low gradient and can be used to segment elongated and/or curved objects. The geodesic distance can be efficiently computed with a Fast Marching algorithm [Iko05]. This algorithm is similar to Dijkstra's algorithm: distances are progressively propagated from the object toward the boundaries of the image (see Appendix B). Notice that [Iko05] require that $dist(.)$ must be a metric. Also, unlike [FH04], the distance map computed in [Iko05] is not exact due to the discretization of $\mathcal{P}$. Using larger neighborhoods in $\mathcal{G}$ yield better approximations of the true geodesic distance but is however more costly on the computational side.

In Figure 4, a level-set of the area of influence for the above two metrics is shown in green color, for some $\varepsilon \simeq 0$. Remind that the area of influence corresponds to the voxels which are sufficiently near $\mathcal{O}$, with respect to the metric $dist(.)$. Observe how the level-set of geodesic distance better fits to the tumor than the level-set of the Euclidean distance. The choice of a geodesic distance allows to take into account areas with strong gradient and therefore, avoids to include areas which would be too distant from the object seeds. There might nonetheless be leakage through "gaps" in contours.

4.2.3 Segmenting lung tumors: a practical case

VLVDP (Very Large Volume Data Processing) is a project aiming to build prototypes and software solutions for handling very large volume data. VLVDP is relatively recent but tends to unit more and more people through interdisciplinary projects (Cancéropôle of Toulouse) and a large field of applications including combinatorial optimization (segmentation, restoration, stereovision), tomographic reconstruction and compression [1]. The algorithmic solutions for addressing those issues are based on graph cuts as well as wavelets and wavelet packets. I/O are managed with ImageMagick, supporting a large number of image formats. The core of VLVDP is written in C and C++ whereas the graphical user interface is written in C++ with Qt and VTK. Combinatorial optimization tools (including RGC) are developped in C++ and place especially emphasis on speed and low memory usage as well as clean and extensible object-oriented design. VLVDP also offers a user-friendly interface to visualize segmentations and compare them with provided ground truths (see below). VLVDP also provides an access to CUDA to benefit from GPGPU processing. Finally, additional features can be added by plugins.

We now briefly detail a typical scenario of how a segmentation of a lung tumor can be obtained in the VLVDP interface. After having selected the volume, we load it, we adapt the visualization and localize the lung tumor using sagittal, coronal and transverse planes (see Figure 5). Once localized, the centers of mass of the tumor are evenly selected with object seeds along the z-axis using the mouse. Undesired parts like healthy tissues are also marked with background seeds (see Figure 6). Then, the energy model is chosen with adequate values of parameters (see Figure 7) and Reduced Graph Cuts compute the desired segmentation. Finally, the segmentation obtained and the ground truth can be compared through 2D or 3D views (see Figure 8).

[1] Please consult `http://www.math.univ-toulouse.fr/~fmalgouy/software/VLVDP.html` for more information.

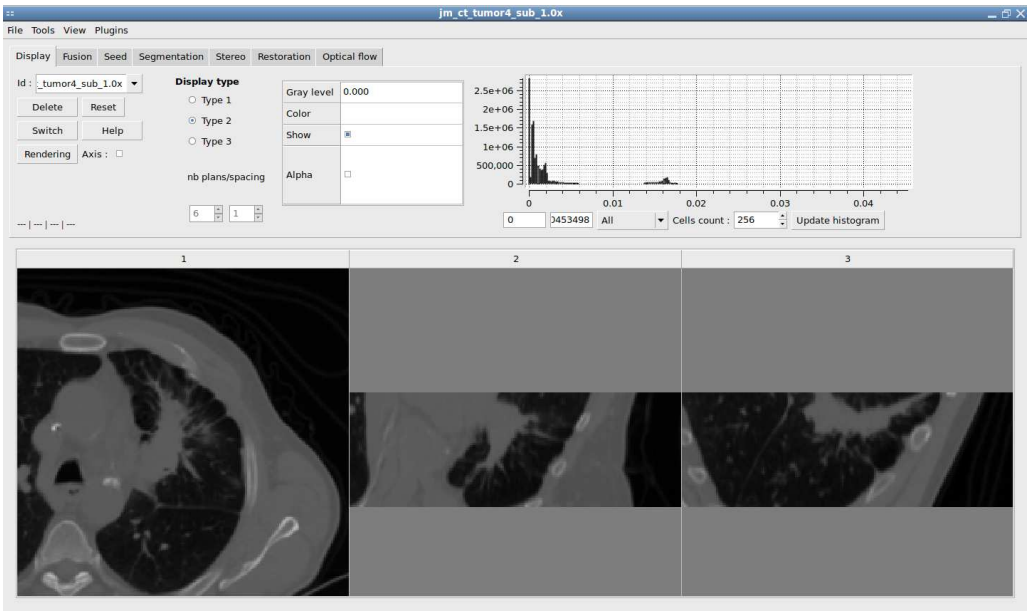


Figure 5: Localization and visualization of the lung tumor using sagittal, coronal and transversal planes.

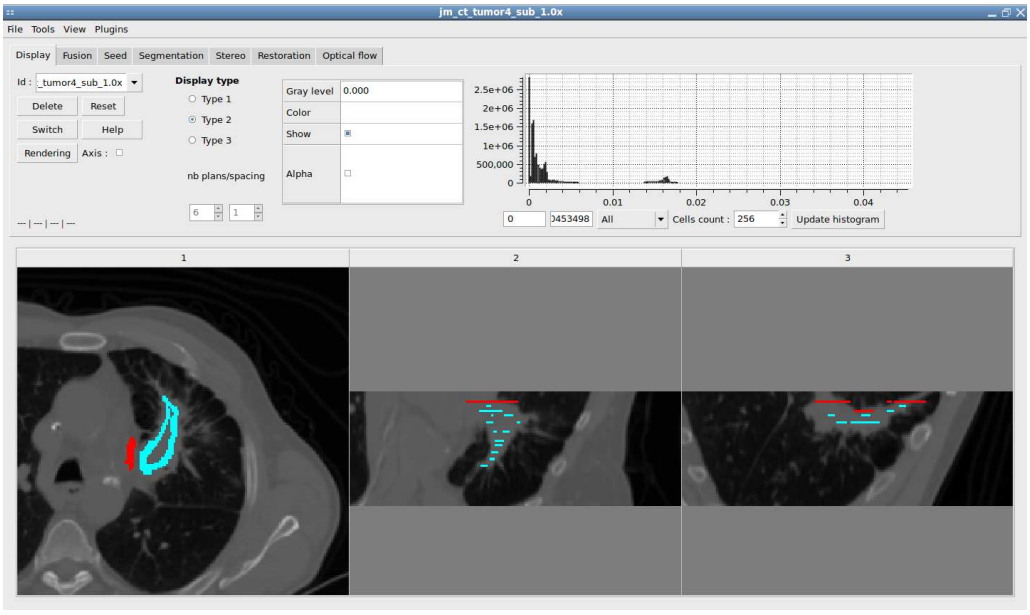


Figure 6: Drawing object and background seeds using sagittal, coronal and transversal planes.

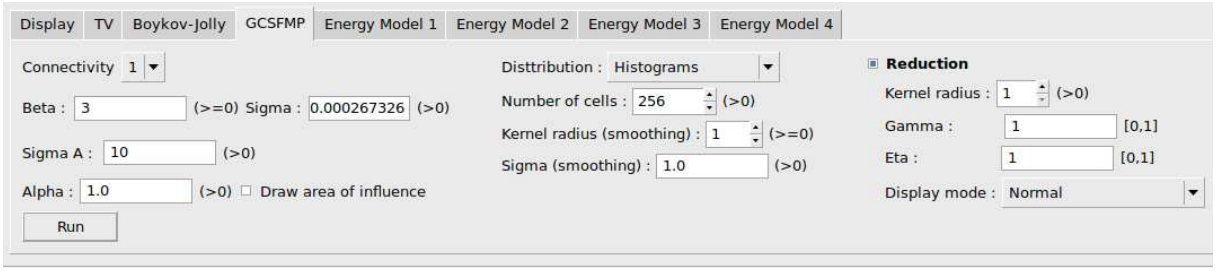


Figure 7: Entering model parameters before running RGC.

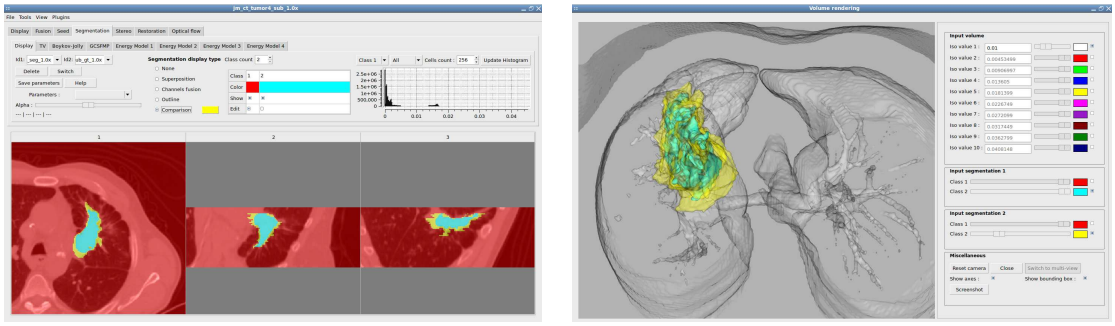


Figure 8: Visualization and comparison of the segmentation and the ground truth in 2D (left) and 3D (right). Ground truth and segmentation are resp. in yellow and cyan.

4.3 Evaluation

In this section, we present experiments for segmenting ten 3D CT images consisting of nodules, masses and tumors. A summary of these materials is available in Table 9. Notice that all volumes have the same size on x ad y axes. All tumors are composed of 50 images on the z-axis except for the tumor T8 (316) and the tumor T10 (70). The resolution on xy-plane varies from 0.63 to 1.17 mm. Similarly, the slice thickness on the z-axis ranges from 1 to 3 mm.

All experiments are performed using connectivity 1. Remind that distributions laws are estimated using NH, the σ parameter is computed as explained in Section 4.2.2 and we set $\sigma_a = 10$. Finally, seeds are placed manually for each CT image several times until satisfaction (about 2-3 times maximum).

Tumor	Type	Sizes (x,y,z)	Description
T1	Mass	$512 \times 512 \times 50$	Mass of the upper right lobe (CT)
T2	Nodule	$512 \times 512 \times 50$	Nodule of the right apex (CT)
T3	Nodule	$512 \times 512 \times 50$	Nodule of the lower right lobe (CT)
T4	Tumor	$512 \times 512 \times 50$	Marge left hilar tumor inducing a peripheral atelectasia (CT)
T5	Tumor	$512 \times 512 \times 50$	Same as T4 (dosimetric CT scanner)
T6	Mass	$512 \times 512 \times 50$	Mass of the lower left lobe appended to the pleura (CT)
T7	Mass	$512 \times 512 \times 50$	Same as T6, after four months of treatment (CT)
T8	Tumor	$512 \times 512 \times 316$	Large left hilar tumor and peripheral atelectasia, before treatment (contrast enhanced CT)
T9	Tumor	$512 \times 512 \times 50$	Same as T8, after chemo-radiotherapy (CE-CT)
T10	Mass	$512 \times 512 \times 70$	Right hilar lymph node mass

Table 9: Description of lung tumors.

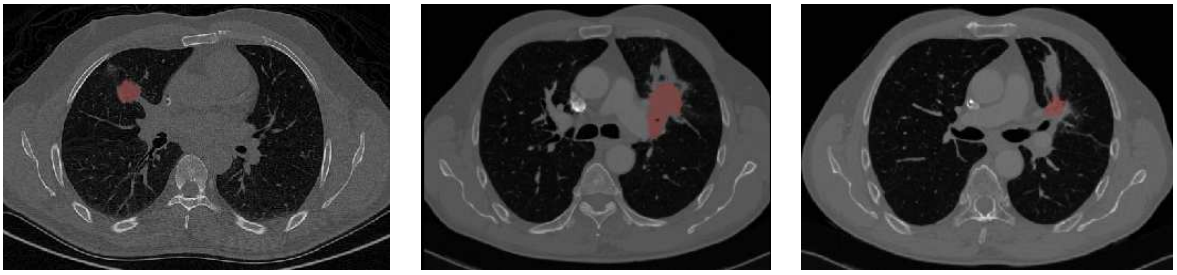


Figure 9: Overall context of lung tumors T1 (left), T8 (middle) and T9 (right). Ground truth are superimposed in red on CT images.

4.3.1 Qualitative and quantitative results

In this section, we evaluate the relevance and the accuracy of our method for segmenting the CT images shown in Table 9. This is achieved by comparing them against ground truth provided by an expert, using several evaluation measures (see Appendix A). Quantitative results are summarized in Table 10. For all tumors, we observe a Dice Coefficient always greater than 70%, which is according to radiologists, sufficient for validating the method. The Average Surface Distance between both segmentations remains generally very low but the Volume Overlap may be sometimes quite low too.

We also evaluate our method in a qualitative manner. The Figure 11 shows the

segmentations and the ground truth obtained at equally spaced values on the z-axis for tumors which obtained the better statistics in Table 10. The segmentations and the ground truth are superimposed by transparency on the original image. The segmentations were obtained using the seeds illustrated in Figure 10. For illustrating their propagation, we show in Figure 10 the seeds for equally spaced on the z-axis but for different values than previously. One can clearly observe how the seeds propagate around object seeds without effort, avoiding radiologists to mark each slice in CT images. For instance, the segmentation of the tumor T1 is very close from the ground truth, while the segmentations obtained for T8 and T9 differ slightly on the borders. This observation is also confirmed in Figure 12 where segmentations are ground truth are visualized in 3D from top, bottom and side views.

Globally, our method depicts quite mitigated results and the poor quality of them can be explained as follows. First, the ground truths provided by the expert are sometimes wrongly labeled. Some voxels are categorized as belonging to the tumor instead of the background. However, due the geodesic distance transform embedded to our method, the algorithm cannot label correctly these voxels due to the high gradient between tumoral and healthy tissues. But mostly, the ground truth shape can change a lot between two successive slices in the z-axis. Again, this problem is inherent to the geodesic distance which propagates uniformly distances between two successive slices. One way to solve this issue would be to use background seeds for further constraining the propagation of distances. However, this situation is not necessarily desirable from a user point of view.

4.3.2 Performance

In this section, we compare the performance of standard graph cuts (SGC) against reduced graph cuts (RGC) both in terms of speed and memory consumption (see Table 11) for segmenting the CT images shown in Table 9. In these experiments, the same seeds and parameters are used as in Section 4.3.1. Experiments were performed on an Athlon Dual Core 6000+ 3GHz with 2GB RAM. Times include graph construction/reduction, distance map and max-flow computation; they are averaged over ten runs. The numerical results

Tumor	Dice Coefficient (%)	Volume Overlap (%)	Volume Difference (%)	Average Surface Distance	RMS Surface Distance	Maximum Surface Distance
T1	90.97	83.45	7.39	0.86	0.92	4.42
T2	80.95	67.99	4.98	1.25	1.54	6.63
T3	72.95	57.42	15.76	1.26	1.50	6.87
T4	71.33	55.44	42.31	3.30	4.01	14.34
T5	80.53	67.41	29.22	3.63	4.55	16.56
T6	86.63	76.42	18.02	1.30	1.49	5.90
T7	82.49	70.21	22.28	1.34	1.56	5.16
T8	89.25	80.59	9.59	1.20	1.47	9.32
T9	72.66	57.07	34.17	1.75	2.09	7.36
T10	74.04	58.79	41.09	4.97	5.55	15.99
Average	80.18	67.47	22.48	2.08	2.46	9.25

Table 10: Comparison between our method and the segmentations provided by the expert.

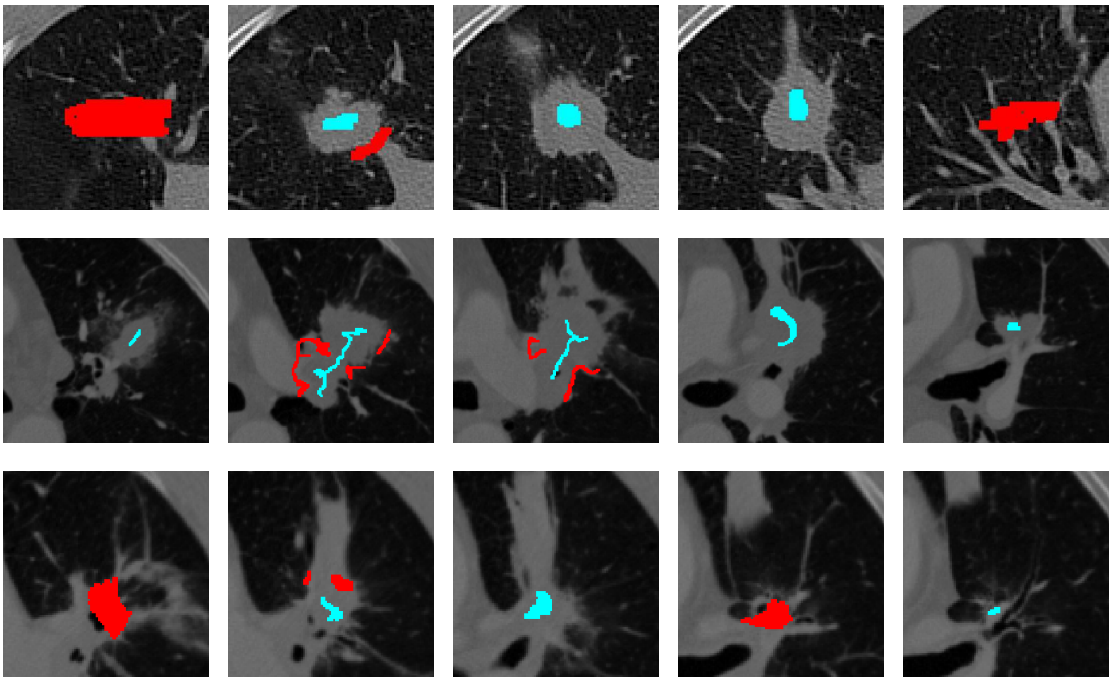


Figure 10: Seeds location for segmenting lung tumors T1 (top row), T8 (middle row) and T9 (bottom row). Object seeds (cyan) and background seeds (red) are superimposed on successive slices of the original image.

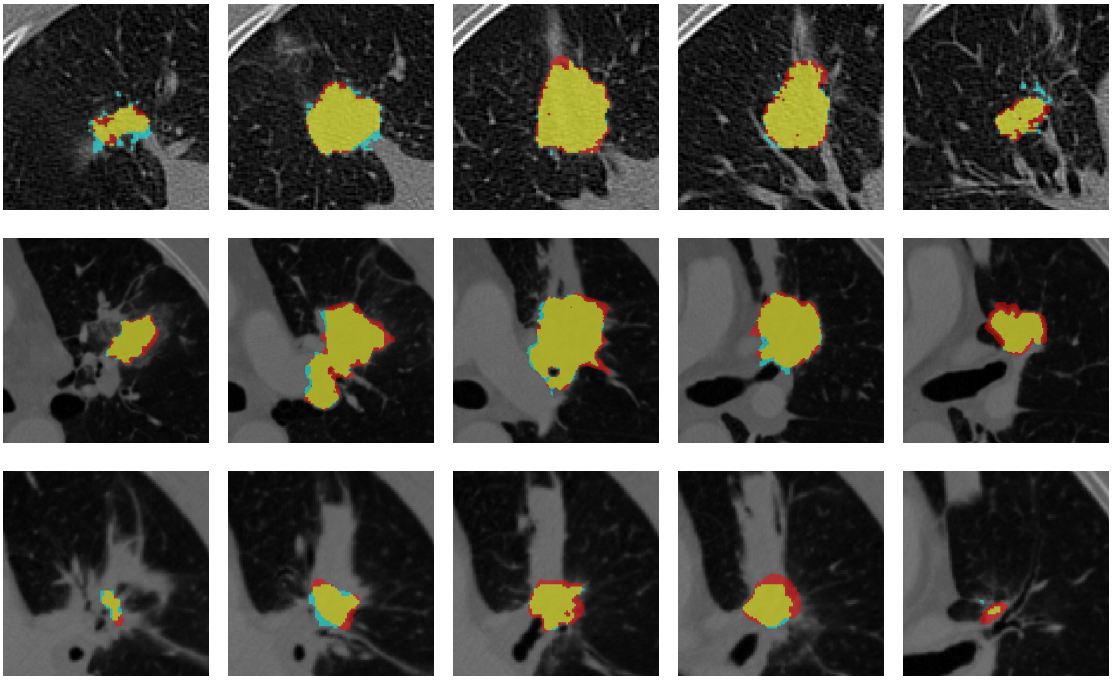


Figure 11: Segmentation of lung tumors T1 (top row), T8 (middle row) and T9 (bottom row). Ground truth (red) and segmentation (cyan) are superimposed on the original image. Yellow color corresponds to the intersection of our solution and the ground truth.

are summarized in Table 11. In this table, we also provide the amount of object seeds over the tumor volume in the ground truth. This gives an objective measure of the amount of interaction. by estimating the effort required by the user for positioning object seeds.

Table 11 show that RGC outperforms SGC for all segmented tumors. While running time is only improved by a factor of low magnitude, the amount of memory allocated by RGC ranges from 7.70 to 41.38x less than SGC while getting exactly the same solution. We also want to emphasize that RGC are able to segment all tumors only in a couple of seconds and are therefore compatible for clinical routine.

Finally, the amount of object seeds entered by the user remains generally negligible with respect to the tumor volume, meaning that a low level of interaction is required for segmenting tumors. This quantity increases a little bit for tumors T4 and T5 due to the difficulty of segmenting correctly them. While such an approach strengthens the role of object seeds, it is also relatively sensitive to the seeds location almost when the tumor is linked to healthy tissues and requires some care when positioning them.

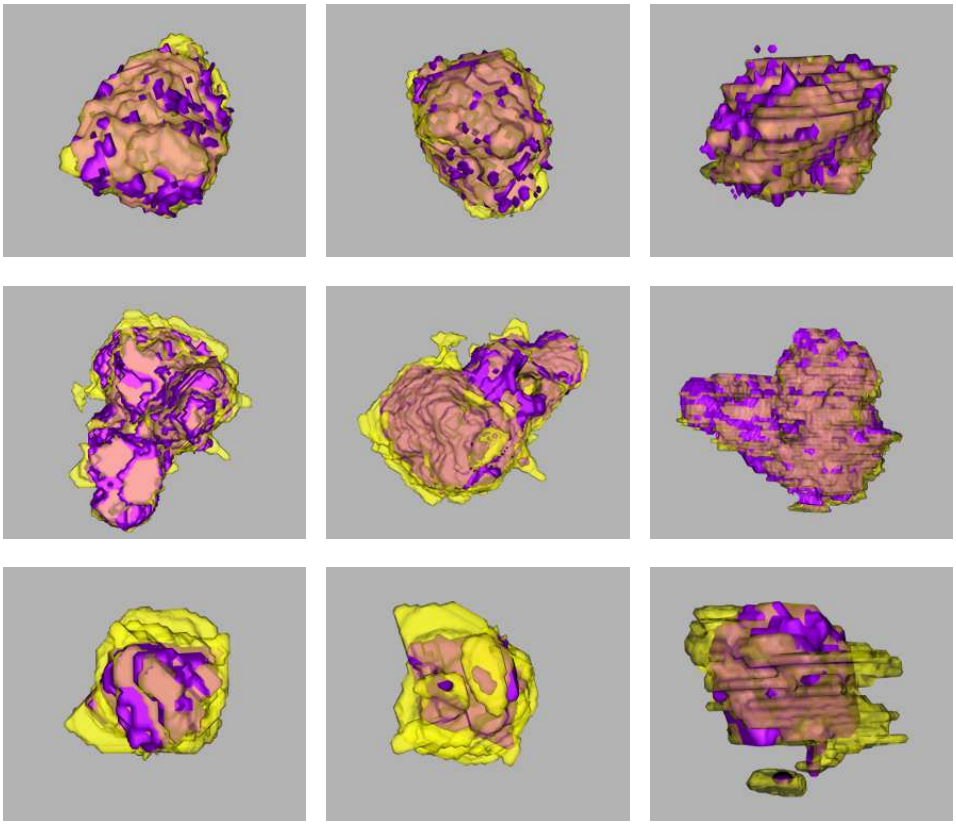


Figure 12: Segmentation of lung tumors T1 (top row), T8 (middle row) and T9 (bottom row) with top (left column), bottom (middle column) and side (right column) views. Yellow corresponds to ground truth (purple) whereas segmentation is shown in purple. The intersection of ground truth and segmentation is shown in pink.

Tumor	SGC		RGC		Amount of object seeds (%)
	Time	Memory	Time	Memory	
T1	2.34	231.85	2.05	8.72	2.56
T2	2.84	271.84	2.50	15.00	2.59
T3	3.21	310.34	2.87	19.27	2.00
T4	2.82	261.06	2.50	18.92	10.99
T5	3.00	284.47	2.72	16.41	8.75
T6	8.87	889.11	7.10	14.96	3.78
T7	4.42	386.27	3.49	23.29	11.18
T8	4.65	419.54	3.82	41.38	2.89
T9	3.62	344.68	3.10	13.65	5.55
T10	6.13	461.76	5.45	59.93	9.48

Table 11: Speed (secs) an memory usage (Mb) for our method and the graph cuts without reduction. The label `MP` means that there is not enough memory for allocating the graph.

4.4 Conclusion

In this chapter, we have presented a new strategy for segmenting lung tumors in an interactive fashion thanks to an original energy formulation embedding a prior on the location of object seeds. The propagation of seeds is also controlled by a Fast Marching algorithm which takes into account the gradient of the CT image. In the experiments, qualitative and quantitative comparisons both exhibit satisfying outcomes with a Dice Coefficient always greater than 70%. Additionally, the computation of the distance map, the construction of the graph, its reduction as well as the max-flow/min-cut computation generally take only a few seconds. Thus, our method demonstrates its ability to segment lung tumors quickly without requiring much effort if it is supported by a well designed graphical user interface.

Nevertheless, the time required for the segmentation depends both on the image size and the skill of the user for positioning the seeds not too far from the contours of the object to segment and the ergonomics of the graphical user interface. Moreover, the segmentation accuracy also depends directly on the seeds location. Additional corrections can be however done quickly if some parts of the structures are incorrectly delineated. While the numerical results appear satisfying from a medical point of view, they also attest of the difficulty of the task.

Conclusion and discussion

In Chapter 1, we have done some general reminders about graph theory and explained how graph cuts can be used in the energy minimization framework. Afterwards, we have described the problem of the prohibitive memory consumption of graph cuts in Chapter 2 and demonstrated that this method is totally unsuitable for solving large-scale optimization problems. Finally, we have proposed a new band-based strategy for reducing graphs in Chapter 3 as well as an application using it for segmenting lung tumors in Chapter 4. We now summarize the contributions of Chapter 3 and 4 and discuss about potential perspectives for each one.

In Chapter 3, two simple tests have been presented for reducing either heuristically or exactly graphs involved in binary graph cuts segmentation. First, an heuristic test (32) has been proposed where the reduced graph size can be adjusted by tuning the window size according of the amount of regularization asked. Then, we have shown that this test can achieve near linear-time performance with an efficient implementation of complexity $O(\sharp\mathcal{P})$ (except for image borders). In particular, this complexity is independent of the window size. Also, the reduction performance has also been evaluated against standard graph cuts for segmenting 2D, 2D+t and 3D grayscale and color images using a TV+L^2 model and a Boykov-Jolly model with NH (Normalized Histograms) and GMM (Gaussian Mixtures Model). The massive numerical experiments presented in Section 3.2.3.2 show that this algorithm globally outperforms standard graph cuts in terms of memory with an average relative reduced graph size under 35% for all energy models tested, while keeping a very low pixel error on segmentations. The proposed implementation of the test (32) also runs faster for some images and if the amount of regularization is of moderate level. For keeping good reduction performance in situations where the amount of regularization is large, two extra parameters have been introduced for further reducing the graphs and removing small connected components due to noise in the segmentations. In case of noisy images, lower and upper bounds on the parameter η have also been provided to automatically set it with or without the window size, in order to suppress all undesired

segments due to noise in the segmentation.

In a second time, we proposed another test (see (46)) of complexity $O(\sharp\mathcal{P}\sharp B^{\frac{d_\mathcal{G}}{2}})$ [2] and proved its exactness in Appendix C. In the manner of the test (32), massive numerical experiments for evaluating the test (46) have also been accomplished in Section 3.2.3.2. These experiments exhibit similar reduction performance to the test (32) with an average relative reduced graph size under 40%, while keeping a low distance between the segmentations. The solutions provided by the tests (32) and (46) are empirically almost the same. Also, the overall reduction performance using (46) is slightly lower than with (32) except for some instances. In words, the exactness of the test (46) as well as better reduction performance for a few instances is at the expense of a larger computational cost. However, one could easily lower the complexity for evaluating (46) for any node $p \in \mathcal{P}$ by finding an upper bound on the sum of n-links capacities for each $q \in \partial B_p$. Furthermore, the exactness of the test (46) has a deeper impact than (32) since the optimality on the solutions is preserved.

More generally, the proposed approach for reducing graphs remains general, non-invasive and can be built on top of other techniques, enabling attractive perspectives. Although the running time of our algorithms depends on the underlying max-flow algorithm used, both are dissociated from each other. Also, since the reduced graphs result in narrow bands around the object edges, metrication and geometric artifacts mentioned in Section 1.2.5.2 could be further reduced by increasing the neighborhood size at the expense of a slightly larger memory consumption. This pushes away the trade-off between memory usage and segmentation quality of graph cuts. As mentioned in Section 3.3.1 and 3.2.2.3, all described reduction algorithms for the tests (32) and (46) are favorable to parallelization and could probably be sped up by several orders of magnitude. Other interesting perspectives include the work of [SK10] and [GY09]. For example, one could apply the reduction using (32) on each sub-optimizable problem involved in the dual decomposition [SK10]. This task would be relatively straightforward to set since the procedure is independent for each sub-problem. This would further push away the limits of [SK10] and let us use fewer machines for segmenting the same amount of data. The theoretical result

[2] We remind that $d_\mathcal{G}$ corresponds to the maximum node degree of $\mathcal{G}$.

of Appendix C also implicitly enables us to restore larger images with a TV+L^2 model using the dyadic scheme of [GY09] while keeping optimality on the solutions. Indeed, we only exploit information of a single level-set of the solution and generalizing to more level-sets is straightforward. However, we want to emphasize that this

Also, even though we do not study the more general TV+L^α model for $1 \leq \alpha < \infty$, we want to emphasize that a method similar to the one described in [GY09] for the TV+L^2 model can solve it. Similarly, the proposed work could also be embedded into alternating minimization schemes for solving the Rousson-Deriche model [RD02] or the Mumford-Shah model [BT08, ZCP06]. A last but not least perspective of the proof in Appendix C would be to embed the reduction into local search algorithms for solving multi-labels problems. α-expansion and α-β swap heuristics represent perfect candidates for this task. Although the optimality is naturally not anymore ensured due to the NP-hardness of the problem, this would at least guarantee that each move inside the labels space (corresponding to a single graph cut computation) is optimal.

Although the proposed strategy for reducing graphs is more efficient for dense graphs with a large proportion of nodes linked to terminals, we want to emphasize that it is not bounded to the particular case of binary segmentation and could probably be applied to other problems such as multi-view stereo [VTC05]. Remind that multi-view stereo consists in finding a three-dimensional reconstruction of an object from a sequence of two-dimensional images taken by different cameras. A functional solving this problem is defined and discretized on a grid in [VTC05]. First, each image of the sequence is segmented using the method of [BK05] and the silhouette [3] is extracted. Knowing the parameters of each camera, each silhouette defines a back-projected cone containing the object. The visual hull corresponding to the intersection of the cones is determined and allows one to discard voxels outside but also inside using a distance map. A weighted directed graph is then built on this reduced grid. As usual, the degree of regularization is controlled by a free parameter which multiplies t-links capacities. The optimal surface under this discretized functional is then obtained as the min-cut solution of this graph. Nevertheless, since the distance from the visual hull to the inner border is fixed, it does

[3] A silhouette is a 2D projection of the corresponding 3D object in the scene.

not ensure that all information is captured. When the amount of regularization remains low, one could probably obtain thinner bands in the graph while avoiding to fix a distance parameter.

A natural extension of this work is to propose similar ideas of the tests (32) and (46) in the multi-labels case (i.e. when $\sharp\mathcal{L} > 2$) in order to discard areas where the optimization occurs. Surprisingly, this problem does not seem to be importantly tackled and is generally bounded to particular vision problems such as stereo. For instance in the stereo problem, one usually needs to define an appropriate disparity search range for looking to the corresponding pixels in the second image. Depending on the image, this range can be sometimes more or less large (e.g. 20 to 80 labels). The technique proposed in [Vek06] is slightly different: instead of discarding areas where the optimization occurs, Veksler tries to reduce the disparity search range for any node $p \in \mathcal{P}$ in $\mathcal{G}$ in the case of the α-expansion. Since the running time of stereo with α-expansion depends linearly on the number of disparity labels searched, reducing the number of disparities explored per node by half divides the running time by half. Reducing appropriately the disparity range for all grid nodes without significant degradations in the results would therefore increase performance. Three different strategies for reducing this range are evaluated: best-candidate based, hierarchical and local-method based. Best-candidate based strategy amounts to select the "best" disparities by thresholding the $E_p(.)$ term or keeping the k best disparities, $\forall p \in \mathcal{P}$. In both cases, the parameters must generally be large enough for including the correct disparity label leading in very little computational savings. The hierarchical approach uses disparity results at coarser levels to restrict the disparity range at higher levels. However, if a detail is mistaken at a coarser level, the mistake gets propagated to higher levels. Among local methods proposed, the window matching globally achieves the best results. The window matching algorithm assigns to any $p \in \mathcal{P}$ the disparity

$$d_p = \underset{l \in \mathcal{L}}{\operatorname{argmin}} \sum_{q \in B_p} E_q(l).$$

where we remind that B_p is a window of size $(2r+1)$ centered in p. Then, the idea is the following: a label l should be in the set of candidate disparities for pixel p if there is a pixel q within some fixed Manhattan distance h which got assigned disparity l by the

above window matching algorithm. This strategy was evaluated on a dataset of 32 real stereo pairs from the Middlebury database against ground truths. Outcomes exhibit good performance with an average speed-up of 2.8 while keeping a low error percentage with respect to ground truths.

As an early attempt, we have also tried to restrict the areas where the multi-labels optimization occurs. The Figure 13 show some examples for segmenting objects in grayscale and color images with a generalized Boykov-Jolly model where each variable u_p takes now values in $\mathcal{L}$, $\forall p \in \mathcal{P}$. In this experiment, multi-labels optimization is accomplished through α-expansion. Consider a square window $B \subset \mathbb{Z}^d$ of size $(2r+1)^d$. Then, a first approach would be to remove from $\mathcal{P}$ any node $p \in \mathcal{P}$ if

$$\exists l_0 \in \mathcal{L} \quad \text{such that} \quad \forall l_1 \in \mathcal{L}, \quad [E_q(l_1) - E_q(l_0)] \geq \delta_r, \quad \forall q \in B_p, \tag{55}$$

where δ_r is set as in the test (32). If (55) holds for any $p \in \mathcal{P}$, we set $u_p = l_0$. Remaining nodes (denoted by ρ) therefore correspond to the areas where the optimization does not occur. The test (55) exploits the same ideas as developped before for binary problems: better reduction will be achieved when the image consists of large uniform areas (see Table 12). The test (55) can be computed in $O(\sharp B_p \sharp \mathcal{L} log(\sharp \mathcal{L}))$ by sorting all energy costs for each $q \in B_p$. Such an approach can easily leverage from parallelization to obtain near-linear time with the number of processors. Although the test (55) remains perfectible and would require further investigations to improve it, this potentially opens new perspectives.

Nevertheless, the results obtained for reducing graphs with the tests (32) and (46) should be put in perspective. Indeed, the reduction performance is both dependent on the image and the model parameters used. Indeed, the massive numerical experiments presented in Section 3.2.3.2 and Section 3.3.2 clearly show that better performance is achieved when the amount of regularization remains low. This implicitly restricts the method to a specific subset of high-contrasted images and is not fully satisfying. To overcome this difficulty, we think that the work of [SK10] is probably the most serious and promising way to follow. We would like to generalize (if possible) their approach to solve multi-labels problems and eventually reduce each sub-problem as discussed earlier. But we have also observed that the reduction is highly problem-dependent and only gives

Name	Size	SGC time	RGC time	ρ (%)
circles	512×200	11.67	1.91	4.31
sunflower-c	460×600 (*)	17.52	4.28	9.46
zebra-c	800×600 (*)	30.53	8.60	13.84
rubikscube-c	350×359 (*)	12.83	4.50	20.76
red-flowers-c	481×321 (*)	9.35	2.00	6.49
brain	$181 \times 217 \times 181$	MP	633.19	24.97

Table 12: Comparison of standard graph cuts and the sequential test (55) in terms of speed (secs) and memory for segmenting 2D and 3D images with $r = 1$. Color images are suffixed by "c" in their names. Label MP stands for memory problem.

satisfactory results when input data are well separated.

Another point concerns the estimation of distribution laws of the Boykov-Jolly model. Although this problem was not the main purpose addressed in Section 3.2.3.2, we have observed that GMM behaves poorly over NH when the seeds are limited to a small homogeneous cluster in the feature space. The EM algorithm approximates too accurately the distribution laws, leading to an over-estimation of the number of Gaussians in the mixture near the initial cluster. Since distribution laws are wrongly estimated, we cannot rely on them and we must set $\beta = 0$, leading to a large increase of the reduced graph size. Accurately estimating distribution laws is therefore a crucial step for keeping good performance of our reduction algorithms. To overcome these difficulties, one could slightly modify the EM algorithm to add simple geometric constraints like shape, orientation or volume on the covariance matrix Σ_k, for any Gaussian k in a GMM. Another possibility to constraint the shape of Gaussians would be to turn to kernel methods.

Finally, we have presented in Chapter 4 an application of the reduction for interactively segmenting lung tumors in 3D CT images. Although the structures to segment appear to be well localized in the images, their can potentially be in contact with similar intensities between healthy and tumoral tissues. This makes the initial problem more challenging since we cannot only rely on local features. This difficulty is also strengthened with the

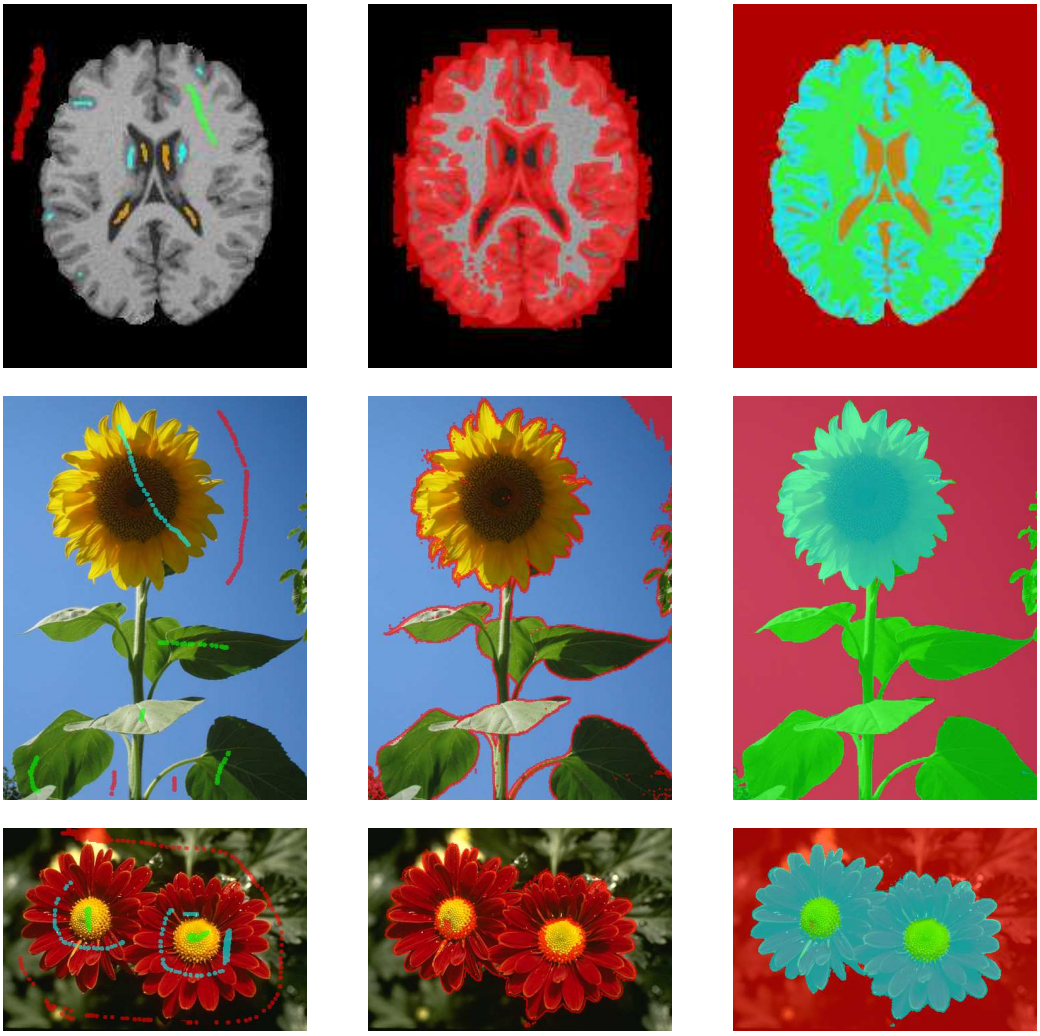


Figure 13: Ilustration of the test (55) for segmenting grayscale and color images with a generalized Boykov-Jolly model in connectivity 1 with α-expansion. In this experiment, we set the number of labels as $\sharp\mathcal{L} = 3$ for middle and bottom images and $\sharp\mathcal{L} = 4$ for top image. The seeds (left column) and the segmentation (right column) are superimposed on the image. On middle column, red pixels correspond to the areas where multi-labels optimization is performed. In these experiments, we set the window radius as $r = 1$.

high variability in shape and color of these structures. To overcome these difficulties, we have proposed to add a prior on the location of object seeds to reduce their importance inside the graph. The propagation of these seeds is also controlled by a Fast Marching algorithm which takes into account gradient information and leads to better tumor delineation. Experimental results against a dataset exhibit a DSC always greater than 70% for all images. The results also show that the use of the reduction requires less time than standard graph cuts and uses 7 to 500 times less memory while keeping very low pixel error between segmentations. Nevertheless, the time required for segmenting such structures depends both on the image size and the skill of the user for positioning the seeds and the ergonomics of the interface. The segmentations can be however quickly corrected if some particular structures have been mislabeled. While these outcomes appear satisfying from a medical point of view, they also attest of the difficulty of the problem. Moreover, better targeting these structures inside the body is essential for enabling precise delivery of high radiation doses.

Embedding more information seems to be a natural extension of this work to better delineate tumors. However, tumor segmentation using both Positron Emission Tomography (PET) and CT modalities is notoriously challenging with a low contrast in CT images and a low spatial resolution/blur in PET due to a long time exposure. Currently, dual-modality PET-CT imaging is widely spread in clinical therapy and increasingly in the treatment process planning. Because the information presented in these modalities is complementary, dual-modality PET-CT images have been empirically proven to bring superior segmentation accuracy. Unlike other works, the strategy adopted in [HBS+11] leverages from the strength of both modalities by penalizing the difference between the PET and CT segmentations. This is achieved by building a large MRF consisting of both images as well as additional nodes encoding this difference. The results are very promising but the segmentation is limited to a small size of volume data due to the huge memory storage required.

Glossary

Breadth-First Search (BFS) is a simple graph search algorithm that begins at the root node and progressively explores all the neighboring nodes. Then, for each of those nearest nodes, it explores their unvisited neighbor nodes, and so on, until it finds the desired goal, 34

Computed Tomography (CT) is a medical imaging method employing tomography created by computer processing. Digital geometry processing is used to generate a 3D volume of an object from a large series of 2D X-ray images taken around a single axis of rotation. Since its introduction in the 1970s, CT has become an important tool in medical imaging to supplement X-rays and medical ultrasonography., 12, 105–107, 109, 110, 116–118, 120, 130, 132

Expectation-Maximization (EM) is an algorithm able to find the maximum likelihood probabilistic model parameters when the model depends on non observable latent variables, 42, 45, 73, 130

Gaussian Mixtures Model (GMM) usually serves to estimate parametrically the distribution of random variables by modeling them as a sum of several Gaussian (called kernels), 42–45, 71–74, 79, 89, 95–97, 99, 100, 125, 130

Iterated Conditional Modes (ICM) is a deterministic algorithm for obtaining the configuration of an MRF that maximizes the joint probability. This is done by iteratively maximizing the probability of each variable on the rest, 23

Maximum A Posteriori (MAP) is a mode of the posterior distribution in Bayesian statistics. The MAP can be used to obtain a point estimate of an unobserved quantity on the basis of empirical data, 23

Minimum Description Length (MDL) is a concept in which the best hypothesis for a given set of data is the one that leads to the best compression of the data. MDL was introduced in 1978 and remains an important concept in information and learning

theory, 45, 73, 74

Markov Random Field (MRF) is a graphical model in which a set of random variables have a Markov property described by an undirected graph. Then, the Maximum A Posteriori can be computed inside this graph to minimize an appropriate energy function to solve low to mid level tasks in Image Processing and Computer Vision, 37, 39, 132

Normalized Histogram (NH) is a discrete estimate of a probability distribution and was first introduced by K. Pearson, 42, 43, 71–74, 79, 95–97, 99, 100, 111, 116, 125, 130

Partial Differential Equations (PDE) are a type of differential equation, i.e., a relation involving unknown function(s) of several independent variables and their partial derivatives with respect to those variables, 21

Positron Emission Tomography (PET) is a nuclear medicine imaging technique that produces a 3D image or picture of functional processes in the body. The system detects pairs of gamma rays indirectly emitted by a positron-emitting radionuclide (called tracer), introduced into the body of the patient on a biologically active molecule, 132

Point Spread Function (PSF) describes the response of an imaging system to a point source or point object. It is a recurrent concept arising in astronomical imaging and other microscopy materials. In words, an intensity in the output image results of a convolution with a mathematical function and images generally appear blurred, 109

Red Green Blue (RGB) is an additive color model in which red, green and blue light are added together to reproduce a wide range of colors, 42

Region Of Interest (ROI) is a selected subset of samples within a dataset identified for a particular purpose, 110

Simulated Annealing (SA) is a probabilistic metaheuristic for obtaining an approximation of a given function in a large search-space, discovered by S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi in 1983. Roughly speaking, each step of the algorithm

replaces the current solution by a random nearby solution, chosen with a probability depending both on the difference between function values and a decreasing parameter mimicking the role of temperature in metallurgy, 23

Watershed is a popular segmentation technique introduced by S. Beucher and C. Lantuéjoul in 1979. Since a grey-level image can be seen as a topographic relief, this method consists in placing a water source in each regional minimum, to flood the relief from sources, and build barriers when different sources are meeting. The resulting set of barriers constitutes a watershed by flooding, 22

Appendix

A Evaluation measures

This appendix describes the evaluation measures used in this document. Let $SG, GT \subset \{0,1\}^N$ $(N > 0)$ denote respectively a segmentation and the ground truth. We also denote by $d : (\mathbb{Z}^d \times \mathbb{Z}^d) \to \mathbb{R}^+$ a metric.

Dice Similarity Coefficient (DSC) (%):

Dice Similarity Coefficient is a similarity measure related to the Jaccard Index and introduced in 1945 by Dice [Dic45]. This coefficient is defined as twice the shared information (intersection) over the combined set. Its value is 1 for a perfect segmentation and 0 in the worst case. We have

$$DSC(SG, GT) = 2 \cdot \frac{\sharp(SG \cap GT)}{\sharp SG + \sharp GT} \times 100$$

Volumetric Overlap (VO) (%):

This is the number of voxels in the intersection of segmentation and ground truth, divided by the number of voxels in the union of segmentation and ground truth. Its value is 1 for a perfect segmentation and is bounded from below by 0, when there is no overlap at all between the segmentation and the ground truth. We have

$$VO(SG, GT) = \frac{\sharp(SG \cap GT)}{\sharp(SG \cup GT)} \times 100$$

Its value is 100 for a perfect segmentation and is bounded from below by 0, when there is no overlap at all between the segmentation and the ground truth.

Relative Absolute Volume Difference (%) (RAVD):

The total volume of the segmentation is divided by the total volume of the ground truth. From this number 1 is subtracted, the absolute value is taken and the result is multiplied by 100. This value is 0 for a perfect segmentation and larger than zero otherwise. Note that the perfect value of 0 can also be obtained for a non-perfect segmentation, as

long as the volume of that segmentation is equal to the volume of the ground truth. We have

$$RAVD(SG, GT) = \left| \frac{\sharp SG}{\sharp GT} - 1 \right| \times 100$$

Average Symmetric Absolute Surface Distance (ASASD):

The border voxels of the segmentation and the ground truth are determined. For each voxel in these sets, the closest voxel in the other set is determined (e.g. using the Euclidean distance). All these distances are stored, for border voxels from both the ground truth and the segmentation. The average of all these distances gives the Average Symmetric Absolute Surface Distance. This value is 0 for a perfect segmentation. We have

$$ASASD(SG, GT) = \left(\frac{\sum_{p \in \partial SG} \min_{q \in \partial GT} d(p, q) + \sum_{q \in \partial GT} \min_{p \in \partial SG} d(p, q)}{\sharp \partial SG + \sharp \partial GT} \right)$$

Symmetric RMS Surface Distance (SRMSSD):

This measures is similar to the previous measure, but stores the squared distances between the two sets of border voxels. We take the square root of the squared distances of the average determined for ASASD. The final value gives the symmetric RMS surface distance and is 0 for a perfect segmentation. We have

$$SRMSSD(SG, GT) = \sqrt{\left(\frac{\sum_{p \in \partial SG} \min_{q \in \partial GT} d(p, q)^2 + \sum_{q \in \partial GT} \min_{p \in \partial SG} d(p, q)^2}{\sharp \partial SG + \sharp \partial GT} \right)}$$

Maximum Symmetric Absolute Surface Distance (Hausdorff distance) (MSASD):

This measure is similar to the previous two, but only the maximum of all voxel distances is taken instead of the average. This value is 0 for a perfect segmentation. We have

$$MSASD(SG, GT) = \max\{ \max_{p \in \partial SG} \min_{q \in \partial GT} d(p, q), \max_{q \in \partial GT} \min_{p \in \partial SG} d(p, q) \}$$

Relative Max-flow Error (RME) (%):

The Relative Max-flow Error corresponds to the percentage of the relative error between the max-flow values $\text{val}_{\mathcal{G}}(f^*)$ and $\text{val}_{\mathcal{G}'}(f'^*)$ of graphs $\mathcal{G}$ and $\mathcal{G}'$, respectively. Notice that this quantity is always non-negative since a larger amount of flow cannot be routed

from the source to the sink in $\mathcal{G}'$ because it contains less edges than in $\mathcal{G}$. This value is 0 when the max-flow is the same. The RME is defined as

$$RME(\mathcal{G},\mathcal{G}') = \frac{(\text{val}_{\mathcal{G}}(f^*) - \text{val}_{\mathcal{G}'}(f'^*))}{\text{val}_{\mathcal{G}}(f^*)} \times 100$$

B Dijkstra algorithm for computing distance maps

Below, we review the algorithm of [Iko05] for computing the distance map $\mathcal{D} : \Omega \to \mathbb{R}^N$ of a binary image $\mathcal{H} : \Omega \to \{0,1\}^N$. In this algorithm, we denote Q as a standard priority queue where elements are pulled with highest-priority-first criteria. We also denote $d : (E \times E) \to \mathbb{R}^+$ a metric defined in some metric space E. For instance, $d(.)$ can be set as the Euclidean distance between points of Ω, a metric based on the intensity difference or a mixture of both.

In Algorithm 4, reference pixels are enqueued to the minimum heap and then dequeued for processing them in priority order [4]. Distance values are propagated from the dequeued pixel to its neighbors. Then, neighbors which obtain a new distance value are enqueued to the heap. The priority order ensures that only final distance values are propagated further. If a shortest path is found to a pixel, which has already been enqueued, the distance value is replaced. Previous instances of the pixel in the queue become obsolete and can be discarded. When the queue becomes empty, all distances values are definitive. Unlike recursive propagation with Chamfer masks, no local distance is computed more than once in Algorithm 4.

C Exactness of the reduction test (46)

This appendix is devoted to the proof for the exactness of the test (46) described in Section 3.3. We also want to highlight that the proposed work has been realized in collaboration with F. Malgouyres.

[4] Pixels to enqueue can be limited to the border of reference pixels/calculation area for lowering the heap size.

Algorithm 4 Dijkstra's algorithm.

INPUTS: Binary image $\mathcal{H}$, metric $d(.)$
OUTPUTS: Distance map $\mathcal{D}$

1. Initialize $\mathcal{D}$ such that $\mathcal{D}(x) = \begin{cases} 0 & \text{if } \mathcal{H}(x) = 0 \text{ (reference pixels)} \\ +\infty & \text{if } \mathcal{H}(x) = 1 \text{ (calculation area)} \end{cases}, \quad \forall x \in \Omega$
2. Put pixels with $\mathcal{D}(x) = 0$ to priority queue Q, $\forall x \in \Omega$.
3. **while** Q is not empty **do**
4. $p = dequeue(Q)$ ($\mathcal{D}_q(p)$ was the smallest distance in Q).
5. **if** $\mathcal{D}_q(p) > \mathcal{D}(p)$ (obsolete distance value) **then**
6. Continue from step 3.
7. **endif**
8. % $\mathcal{D}(p)$ becomes $\mathcal{D}^*(p)$ (distance value is definitive)
9. **forall** neighbors x of p with $\mathcal{D}(x) > \mathcal{D}^*(p)$ **do**
10. **if** $\mathcal{D}^*(p) + d(p,x) < \mathcal{D}(x)$ **then**
11. $\mathcal{D}(x) \leftarrow \mathcal{D}^*(p) + d(p,x)$
12. **endif**
13. **endfor**
14. **endwhile**

Notations

Throughout this appendix, we consider a fixed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$, with $\mathcal{V}$, $\mathcal{E}$ and c and a structuring element $B \subset \mathbb{Z}^d$ ($d > 0$) as defined in Section 1.2.2. Our purpose is to exhibit a max-flow satisfying some condition for this graph.

We also denote a walk of positive length $l \in \mathbb{N}^*$ by $p_0 - p_1 - \ldots - p_l$, where $p_i \in \mathcal{V}$, for all $i \in \{0, \ldots, l\}$, and $(p_i, p_{i+1}) \in \mathcal{E}$, for all $i \in \{0, \ldots, l-1\}$. We also remind that a closed walk is such that $p_0 = p_l$. We denote by $W_a(p,q)$ the set containing all the walks starting at $p \in \mathcal{V}$ and ending at $q \in \mathcal{V}$.

For any $S \subset \mathcal{P}$, we denote the value of the s-t cut $(S \cup \{s\}, (\mathcal{P} \setminus S) \cup \{t\})$ in $\mathcal{G}$ by $\text{val}_{\mathcal{G}}(S)$ instead of the one used in Section 1.2.2. We remind that

$$\text{val}_{\mathcal{G}}(S) = \sum_{\substack{p \in S \cup \{s\} \\ q \notin S \cup \{s\}}} c_{p,q}.$$

Notice that, we have not clarified that $(p,q) \in \mathcal{E}$ in the above summation thanks to (8). Notice that we use the again same notation for the value of a flow and the value of a s-t cut in $\mathcal{G}$. This abuse of notation will never be ambiguous once in context.

Avoiding useless flow on closed walks

In this section we remind a known result. We also prove it so-that the appendix is self contained.

Proposition 3. *Let $\mathcal{G}$ be the graph defined in Section C. There exists a max-flow f in $\mathcal{G}$ satisfying*

$$\begin{cases} \text{for any length } l \text{ and any closed walk } p_0 - p_1 - \ldots - p_l \text{ of length } l \text{ in } \mathcal{G}, \\ \text{there exists } i \in \{0, \ldots, l\} \text{ such that } f_{p_i,p_{i+1}} \leq f_{p_{i+1},p_i} \text{ where we denote } p_{l+1} = p_0. \end{cases} \tag{56}$$

Proof. Let f be a max-flow in $\mathcal{G}$. For any l and any closed walk $w = p_0 - p_1 - \ldots - p_l$ of length l in $\mathcal{G}$, we set $p_{l+1} = p_0$ and denote $(P_{f,w})$ the statement:

$$(P_{f,w}) : \forall i \in \{0, \ldots, l\}, f_{p_i,p_{i+1}} > f_{p_{i+1},p_i}.$$

In particular, a closed walk w satisfying the previous statement is not allowed to take reverse edges. We also denote

$$W(f) \stackrel{\text{def}}{=} \{w, w \text{ is a closed walk satisfying } (P_{f,w})\}.$$

Notice first that if

$$\#W(f) = 0, \tag{57}$$

where $\#$ denotes the cardinality of a set, the flow f necessarily satisfies (56).

We show, in the remaining of the proof, that if f is such that $\#W(f) > 0$, there exist f' such that

$$\#W(f') < \#W(f),$$

where $\#$ denotes the cardinality of a set. Since for any max-flow f the set $W(f)$ is finite, any initial max-flow lead to a max-flow satisfying (57) (and therefore (56)) after a finite number of such recursion.

Let us now assume that f is such that $\#W(f) > 0$. Let us also consider a closed walk $w = p_0 - p_1 - \ldots - p_l \in W(f)$.

We denote $p_{l+1} = p_0$ and

$$\delta \stackrel{\text{def}}{=} \min_{i\in\{0,\dots,l\}} (f_{p_i,p_{i+1}} - f_{p_{i+1},p_i}).$$

Since w satisfies $(P_{f,w})$, we have $\delta > 0$.

We define the mapping $f' : (\mathcal{V}\times\mathcal{V}) \to \mathbb{R}^+$ such that for all $(p,q) \in (\mathcal{V}\times\mathcal{V})$:

$$f'_{p,q} = \begin{cases} f_{p,q} - f_{q,p} - \delta & \text{, if } (p,q) = (p_i, p_{i+1}), \text{ for some } 0 \le i \le l \\ 0 & \text{, if } (p,q) = (p_{i+1}, p_i), \text{ for some } 0 \le i \le l \\ f_{p,q} & \text{, otherwise.} \end{cases} \tag{58}$$

Notice that this definition is not ambiguous. Indeed, we cannot simultaneously have $(p,q) = (p_i, p_{i+1})$ and $(p,q) = (p_{j+1}, p_j)$ for some $i \neq j$ since w satisfies $(P_{f,w})$.

Also, since f is a flow in $\mathcal{G}$, we clearly have for all $(p,q) \in (\mathcal{V}\times\mathcal{V})$

$$0 \le f'_{p,q} \le c_{p,q}.$$

In order to prove the flow conservation, we consider $p \in \mathcal{V}$. Let us first assume that $p \neq p_i$, for all $i \in \{0,\dots,l\}$. Then (58) guarantees that $f'_{p,q} = f_{p,q}$ for all $q \in \sigma_{\mathcal{E}}(p)$ and we trivially get

$$\sum_{q\in\sigma_{\mathcal{E}}(p)} f'_{q,p} = \sum_{q\in\sigma_{\mathcal{E}}(p)} f'_{p,q}.$$

Let us now assume that there exists $i \in \{0,\dots,l\}$ such that $p = p_i$. We denote

$$I = \{j \in \{0,\dots,l\}, p = p_j\}$$

and $p_{-1} = l$.

We have

$$\sum_{q\in\sigma_{\mathcal{E}}(p)} (f'_{q,p} - f'_{p,q}) = \sum_{\substack{q\in\sigma_{\mathcal{E}}(p)\\ q\neq p_{j+1},\forall j\in I\\ q\neq p_{j-1},\forall j\in I}} (f'_{q,p} - f'_{p,q}) + \sum_{j\in I}(f'_{p_{j+1},p_j} - f'_{p_j,p_{j+1}}) + \sum_{j\in I}(f'_{p_{j-1},p_j} - f'_{p_j,p_{j-1}})$$

Using (58), we obtain for each term

$$\sum_{\substack{q\in\sigma_{\mathcal{E}}(p)\\ q\neq p_{j+1},\forall j\in I\\ q\neq p_{j-1},\forall j\in I}} (f'_{q,p} - f'_{p,q}) = \sum_{\substack{q\in\sigma_{\mathcal{E}}(p)\\ q\neq p_{j+1},\forall j\in I\\ q\neq p_{j-1},\forall j\in I}} (f_{q,p} - f_{p,q}), \tag{59}$$

$$\sum_{j\in I}(f'_{p_{j+1},p_j} - f'_{p_j,p_{j+1}}) = \sum_{j\in I} -(f_{p_j,p_{j+1}} - f_{p_{j+1},p_j} - \delta), \tag{60}$$

and

$$\sum_{j\in I}(f'_{p_{j-1},p_j} - f'_{p_j,p_{j-1}}) = \sum_{j\in I}(f_{p_{j-1},p_j} - f_{p_j,p_{j-1}} - \delta). \tag{61}$$

Summing (59), (60), (61) and simplifying, we finally get

$$\begin{aligned}\sum_{q\in\sigma_{\mathcal{E}}(p)}(f'_{q,p} - f'_{p,q}) &= \sum_{\substack{q\in\sigma_{\mathcal{E}}(p)\\ q\neq p_{j+1},\forall j\in I\\ q\neq p_{j-1},\forall j\in I}}(f_{q,p} - f_{p,q}) + \sum_{j\in I}\left((f_{p_{j+1},p_j} - f_{p_j,p_{j+1}}) + (f_{p_{j-1},p_j} - f_{p_j,p_{j-1}})\right)\\ &= \sum_{q\in\sigma_{\mathcal{E}}(p)}(f_{q,p} - f_{p,q})\\ &= 0\end{aligned}$$

As a conclusion, f' is a flow. It is of course a max-flow. Indeed, (5) and (13) guarantee that $f_{p,s} = 0$, for all $p \in \mathcal{P}$. Since w satisfies $(P_{f,w})$, this ensures that

$$s \neq p_i, \forall i \in \{0,\dots,l\}.$$

Using (14) and (58), we finally get

$$\mathrm{val}_{\mathcal{G}}(f') = \mathrm{val}_{\mathcal{G}}(f).$$

We still need to show that

$$\#W(f') < \#W(f).$$

With that in mind, we consider $w' = p'_0 - p'_1 - \ldots - p'_{l'} \in W(f')$. Denoting $p'_{l'+1} \overset{\text{def}}{=} p'_0$, we know that for any $j \in \{0,\dots,l'\}$

$$0 < (f'_{p'_j,p'_{j+1}} - f'_{p'_{j+1},p'_j}).$$

Together with (58), this guarantees that for any $j \in \{0,\dots,l'\}$

$$(p'_j, p'_{j+1}) \neq (p_{i+1}, p_i), \text{ for all } i \in \{0,\dots,l\}.$$

Such a situation is illustrated in Figure 14. Using (58) again, we therefore necessarily have

$$0 < (f'_{p'_j,p'_{j+1}} - f'_{p'_{j+1},p'_j}) \leq (f_{p'_j,p'_{j+1}} - f_{p'_{j+1},p'_j}).$$

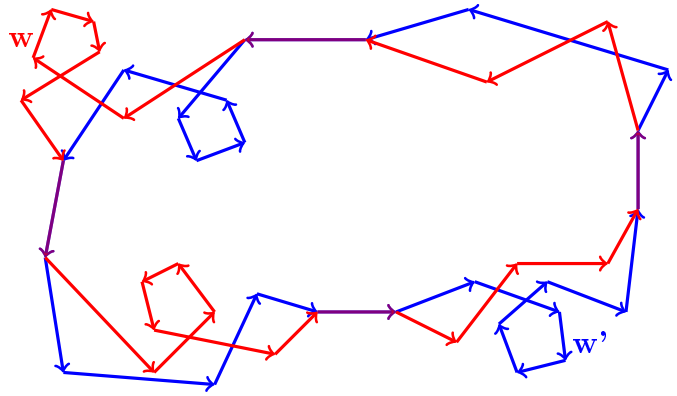


Figure 14: A situation where the closed walks $w \in W(f)$ (red) and $w' \in W(f')$ (blue) in $\mathcal{G}$ both satisfy (58). The edges in purple correspond to the intersection of w and w'. Observe that w cannot follows reverse edges in w' and conversely since (58) must be satisfied.

This means that $w' \in W(f)$ and, as a result,

$$W(f') \subset W(f).$$

In order to show that this inclusion is strict, we denote

$$i_0 \in \underset{i\in\{0,\dots l\}}{\operatorname{argmin}}(f_{p_i,p_{i+1}} - f_{p_{i+1},p_i}).$$

Using (58), we trivially obtain that

$$f'_{p_{i_0},p_{i_0+1}} = f'_{p_{i_0+1},p_{i_0}} = 0,$$

and therefore $w \notin W(f')$.

This concludes the proof. □

Avoiding useless traversing flow

Throughout this section, we consider a graph $\mathcal{G}$ as constructed in Section C and a max-flow f in $\mathcal{G}$ satisfying (56). We also consider $p \in \mathcal{P}$ such that

$$\forall q \in B_p, f_q \geq 0,$$

where B_p is defined in (30). [5]

[5] Notice that all the content of this section could be adapted to a situation where $f_q \leq 0$, for all $q \in B_p$.

The purpose of this section is to establish a sufficient condition so-that f can be modified in such a way that

$$f_{p,q} \geq f_{q,p}, \text{ for all } q \in \sigma_{\mathcal{E}}(p).$$

In words, the node p globally sends more flow to its neighbors than it can receive from them.

In order to do so, we consider

$$\Sigma_i(p) = \{q \in \mathcal{P}, \exists p_0 - \ldots - p_l \in W_a(q,p) \text{ such that } \forall i \in \{0,\ldots,l-1\}, f_{p_i,p_{i+1}} > f_{p_{i+1},p_i}\},$$

and

$$\Sigma_o(p) = \{q \in B_p, \exists p_0 - \ldots - p_l \in W_a(p,q) \text{ such that } \forall i \in \{0,\ldots,l-1\}, f_{p_i,p_{i+1}} > f_{p_{i+1},p_i}\},$$

where we remind that $W_a(q,p)$ (resp. $W_a(p,q)$) contains all the walks starting at q (resp. p) and ending at p (resp. q).

Let us first notice that, since f satisfies (56),

$$\forall q \in (\Sigma_i(p) \cap \sigma_{\mathcal{E}}(p)), \quad f_{q,p} \geq f_{p,q} \tag{62}$$

and

$$\forall q \in (\Sigma_o(p) \cap \sigma_{\mathcal{E}}(p)), \quad f_{p,q} \geq f_{q,p}. \tag{63}$$

Similarly, since f satisfies (56), we have

$$\Sigma_i(p) \cap \Sigma_o(p) = \emptyset. \tag{64}$$

Moreover,

$$p \notin \Sigma_i(p) \text{ and } p \notin \Sigma_o(p).$$

For simplicity, we denote

$$\Sigma^- = \Sigma_i(p) \text{ and } \Sigma^+ = \Sigma_o(p) \cup \{p\}.$$

Also, since f satisfies (56), we have

$$\forall q \in \Sigma^-, \forall q' \in \Sigma^+, \quad f_{q,q'} \geq f_{q',q}. \tag{65}$$

Otherwise, we could easily build a closed walk contradicting (56).

We also denote

$$\mathcal{P}' = \Sigma^- \cup \Sigma^+ \quad , \quad \mathcal{V}' = \mathcal{P}' \cup \{s,t\} \tag{66}$$

and construct the graph

$$\mathcal{G}' = (\mathcal{V}', \mathcal{E}', c')\,,$$

where $\mathcal{E}'$ and c' are defined below. We set

$$\mathcal{E}' = \mathcal{E}'_t \cup (\mathcal{E}'_n \cap \mathcal{E}^T), \tag{67}$$

where $\mathcal{E}^T = \{(q,q'), (q',q) \in \mathcal{E}\}$ and with

$$\mathcal{E}'_t = \{(q,t), \text{ with } q \in \Sigma^- \text{ such that } f_q \geq 0\} \bigcup \left(\{s\} \times \Sigma^+\right) \tag{68}$$

and

$$\mathcal{E}'_n = \left(\Sigma^+ \times \Sigma^+\right) \bigcup \left((\Sigma^- \cup \{p\}) \times (\Sigma^- \cup \{p\})\right). \tag{69}$$

The capacities c' are defined by

$$c'_{q,t} = f_q \qquad \text{, for } q \in \Sigma^- \text{ such that } f_q \geq 0, \tag{70}$$

$$c'_{s,q} = c_q - f_q \qquad \text{, for } q \in \Sigma^+, \tag{71}$$

and

$$c'_{q,q'} = \begin{cases} f_{q',q} - f_{q,q'} & \text{, if } f_{q',q} > f_{q,q'} \\ 0 & \text{, otherwise} \end{cases} \qquad \text{, for } (q,q') \in (\mathcal{E}'_n \cap \mathcal{E}^T). \tag{72}$$

Notice that there exist some nodes in Σ^- which are linked to no terminals. An example of configuration with B_p and the graph $\mathcal{G}'$ is outlined in Figure 15.

As in Section C, we artificially extend all the capacities c' and set

$$c'_{q,q'} = 0, \text{ for all } (q,q') \in ((\mathcal{V}' \times \mathcal{V}') \setminus \mathcal{E}')\,.$$

Notice that, in the graph $\mathcal{G}'$ all the flow sent by s goes in Σ^+ and all the flow arriving at t comes from Σ^-. Moreover, all the edges between Σ^+ and Σ^- contain p.

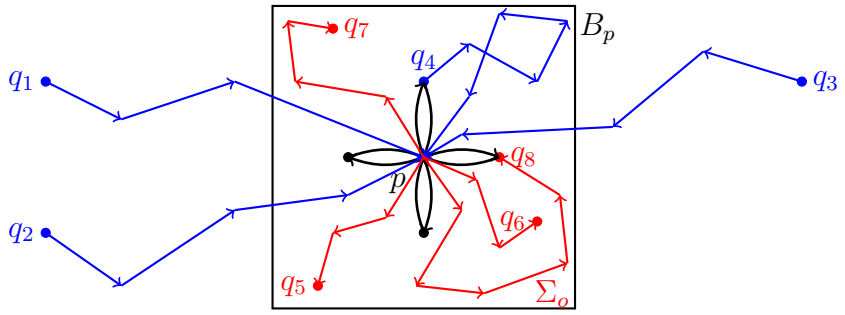


Figure 15: An example of graph $\mathcal{G}'$ where the flow sent by s goes in the nodes $q_1, q_2, q_3, q_4 \in \Sigma^-$ (blue) and all the flow arriving at t comes from the nodes $q_5, q_6, q_7, q_8 \in \Sigma^+$ (red). Notice that the nodes of Σ^+ are bounded to B_p whereas the nodes of Σ^- are a subset of $\mathcal{P}$.

Also, for any $S \subset \mathcal{P}'$, we denote the value of the s-t cut $(S \cup \{s\}, (\mathcal{P}' \setminus S) \cup \{t\})$ in $\mathcal{G}'$ by

$$\mathrm{val}_{\mathcal{G}'}(S) = \sum_{\substack{q \in (S \cup \{s\}) \\ q' \notin (S \cup \{s\})}} c'_{q,q'}.$$

Using (70), (71) and (72), we find

$$\mathrm{val}_{\mathcal{G}'}(S) = E_1 + E_2 + E_3,$$

where we write

$$E_1 = \sum_{q \in (\Sigma^+ \setminus S)} c'_{s,q} \quad , \quad E_2 = \sum_{q \in (\Sigma^- \cap S)} c'_{q,t} \quad \text{and} \quad E_3 = \sum_{\substack{q \in S \\ q' \in (\mathcal{P}' \setminus S)}} c'_{q,q'}. \tag{73}$$

In particular, using (64) and (66), we have

$$\mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) = \sum_{\substack{q \in \Sigma^+ \\ q' \in \Sigma^-}} c'_{q,q'},$$

which, using (69), (64) becomes

$$\mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) = \sum_{q \in \Sigma^-} c'_{p,q}.$$

Finally, we obtain using (72) and (62)

$$\mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) = \sum_{q \in \Sigma^-} (f_{q,p} - f_{p,q}). \tag{74}$$

The following proposition holds.

Proposition 4. *Let $\mathcal{G}'$ be the graph constructed in Section C. For any $S \subset \mathcal{P}'$,*

$$\mathrm{val}_{\mathcal{G}'}(S) \geq \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) + \sum_{q \in \Sigma^+ \setminus (S \cup \{p\})} \left[c_q + \sum_{q' \notin \Sigma^+} (f_{q',q} - f_{q,q'}) \right]. \tag{75}$$

Proof. Let us first decompose E_3 according to

$$E_3 = E'_1 + E'_2 + E'_3 + E'_4,$$

with

$$E'_1 = \sum_{\substack{q \in (S \cap \Sigma^+) \\ q' \in (\Sigma^+ \setminus S)}} c'_{q,q'} \qquad , E'_2 = \sum_{\substack{q \in (S \cap \Sigma^+) \\ q' \in (\Sigma^- \setminus S)}} c'_{q,q'}$$
$$E'_3 = \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \in (\Sigma^+ \setminus S)}} c'_{q,q'} \qquad , E'_4 = \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \in (\Sigma^- \setminus S)}} c'_{q,q'}$$

We rewrite, using (72),

$$E'_1 = \sum_{\substack{q \in (S \cap \Sigma^+) \\ q' \in (\Sigma^+ \setminus S) \\ f_{q',q} > f_{q,q'}}} (f_{q',q} - f_{q,q'}) \qquad , E'_2 = \sum_{\substack{q \in (S \cap \Sigma^+) \\ q' \in (\Sigma^- \setminus S) \\ (q,q') \in \mathcal{E}', f_{q',q} > f_{q,q'}}} (f_{q',q} - f_{q,q'}) \tag{76}$$

$$E'_3 = \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \in (\Sigma^+ \setminus S) \\ (q,q') \in \mathcal{E}', f_{q',q} > f_{q,q'}}} c'_{q,q'} \qquad , E'_4 = \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \in (\Sigma^- \setminus S) \\ f_{q',q} > f_{q,q'}}} (f_{q',q} - f_{q,q'}) \tag{77}$$

Using (69) and (64), then (72) and (62), we immediately find that

$$E'_2 = \begin{cases} \sum_{q \in (\Sigma^- \setminus S)} (f_{q,p} - f_{p,q}) & \text{, if } p \in S \\ 0 & \text{, otherwise,} \end{cases} \qquad \text{and} \qquad E'_3 = 0. \tag{78}$$

Moreover, since the total amount of flow entering and exiting $(S \cap \Sigma^-)$ are equal, we have (see (17))

$$\sum_{\substack{q \in (S \cap \Sigma^-) \\ f_q \geq 0}} f_q + \sum_{\substack{q \in (S \cap \Sigma^-) \\ f_q < 0}} f_q + \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \notin (S \cap \Sigma^-)}} (f_{q',q} - f_{q,q'}) = 0$$

Moreover, if we decompose the last term and reorganize the equation we obtain

$$\sum_{\substack{q \in (S \cap \Sigma^-) \\ f_q \geq 0}} f_q + \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \in (\Sigma^- \setminus S)}} (f_{q',q} - f_{q,q'}) = - \sum_{\substack{q \in (S \cap \Sigma^-) \\ f_q < 0}} f_q - \sum_{\substack{q \in (S \cap \Sigma^-) \\ q' \in \Sigma^+}} (f_{q',q} - f_{q,q'})$$

Together with the definition of E_2 in (73), the definition of E'_4 in (77) and (70) this leads to

$$\begin{aligned} E_2 + E'_4 &\geq \sum_{\substack{q\in(S\cap\Sigma^-)\\ f_q\geq 0}} f_q + \sum_{\substack{q\in(S\cap\Sigma^-)\\ q'\in(\Sigma^-\setminus S)}} (f_{q',q} - f_{q,q'}) \\ &\geq -\sum_{\substack{q\in(S\cap\Sigma^-)\\ f_q< 0}} f_q - \sum_{\substack{q\in(S\cap\Sigma^-)\\ q'\in\Sigma^+}} (f_{q',q} - f_{q,q'}) \\ &\geq \sum_{q\in(S\cap\Sigma^-)} (f_{q,p} - f_{p,q}) + \sum_{\substack{q\in(S\cap\Sigma^-)\\ q'\in(\Sigma^+\setminus\{p\})}} (f_{q,q'} - f_{q',q}). \end{aligned}$$

Then, using (65), we immediately obtain

$$E_2 + E'_4 \geq \sum_{q\in(S\cap\Sigma^-)} (f_{q,p} - f_{p,q}).$$

Together with (78) and (74), this leads to the following intermediate result:

$$E_2 + E'_2 + E'_3 + E'_4 \geq \begin{cases} \mathrm{val}_{\mathcal{G}'}(\Sigma^+) & \text{, if } p \in S \\ \sum_{q\in(S\cap\Sigma^-)} (f_{q,p} - f_{p,q}) & \text{, otherwise.} \end{cases} \tag{79}$$

In order to finish the proof, let us first notice that using the definition of E_1 in (73), (71) and the definition of E'_1 in (76)

$$E_1 + E'_1 \geq \sum_{q\in(\Sigma^+\setminus S)} (c_q - f_q) + \sum_{\substack{q\in(S\cap\Sigma^+)\\ q'\in(\Sigma^+\setminus S)}} (f_{q',q} - f_{q,q'}) \tag{80}$$

Expressing that the total amount of flow entering and exiting $(\Sigma^+ \setminus S)$ are equal, we have (see (17))

$$\sum_{q\in(\Sigma^+\setminus S)} f_q + \sum_{\substack{q\in(\Sigma^+\setminus S)\\ q'\in(\Sigma^+\cap S)}} (f_{q',q} - f_{q,q'}) + \sum_{\substack{q\in(\Sigma^+\setminus S)\\ q'\notin\Sigma^+}} (f_{q',q} - f_{q,q'}) = 0.$$

Together with (80), this guarantees that

$$\begin{aligned} E_1 + E'_1 &\geq \sum_{q\in(\Sigma^+\setminus S)} c_q + \sum_{\substack{q\in(\Sigma^+\setminus S)\\ q'\notin\Sigma^+}} (f_{q',q} - f_{q,q'}), \\ &\geq \sum_{q\in(\Sigma^+\setminus S)} \left[c_q + \sum_{q'\notin\Sigma^+} (f_{q',q} - f_{q,q'}) \right] \end{aligned} \tag{81}$$

When $p \in S$, by combining the latter result with (79), we immediately get (75). If $p \notin S$, (81) can be rewritten using (74)

$$E_1 + E_1' \geq \sum_{q \in (\Sigma^+ \setminus (S \cup \{p\}))} \left[c_q + \sum_{q' \notin \Sigma^+} (f_{q',q} - f_{q,q'}) \right] + c_p + \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right).$$

Since $c_p \geq 0$, and (79) and (62) guarantee that $E_2 + E_2' + E_3' + E_4' \geq 0$, this ensures that (75) holds even when $p \notin S$ and concludes the proof. □

All along the remaining of this Section, we consider a max-flow f' in $\mathcal{G}'$. Notice also that $\mathcal{G}'$ satisfies (5), (9). Therefore, as in Section C, we denote

$$f_q' = f_{s,q}' - f_{q,t}',$$

for all $q \in \mathcal{P}'$. We also artificially extend the flow f' and set

$$f_{q,q'}' = 0, \text{ for all } (q,q') \in ((\mathcal{V}' \times \mathcal{V}') \setminus \mathcal{E}').$$

We are now going to combine f and f' in order to build a mapping $f'' : \mathcal{E} \to \mathbb{R}$ which will turn out to be a max-flow in $\mathcal{G}$ such that

$$f_{p,q}'' \geq f_{q,p}'' = 0 \quad , \forall q \in \sigma_{\mathcal{E}}(p).$$

Let us begin with the definition of f''. We set

$$f_{q,q'}'' = f_{q,q'} \qquad \text{, for } (q,q') \notin \mathcal{E}', q \neq s, q' \neq t, \tag{82}$$

$$\begin{cases} f_{s,q}'' = 0 & \text{and } f_{q,t}'' = -f_q \quad \text{, for } q \in \mathcal{P}' \text{ such that } f_q < 0 \\ f_{s,q}'' = f_q + f_q' & \text{and } f_{q,t}'' = 0 \quad \text{, for } q \in \mathcal{P}' \text{ such that } f_q \geq 0 \end{cases} \tag{83}$$

$$f_{q',q}'' = \begin{cases} \underbrace{f_{q',q} - f_{q,q'}}_{c'_{q,q'}} - f_{q,q'}' & \text{, if } f_{q',q} > f_{q,q'}, \\ & \text{, for } (q',q) \in \mathcal{P}'^2 \text{ such that } (q,q') \in \mathcal{E}'. \\ 0 & \text{, otherwise} \end{cases} \tag{84}$$

Notice that the equations (82), (83) and (84) permit to define $f''_{q,q'}$ for all $(q,q') \in \mathcal{E}$. Once again, we extend f'' outside $\mathcal{E}$ and set

$$f''_{q,q'} = 0, \text{ for all } (q,q') \in ((\mathcal{V} \times \mathcal{V}) \setminus \mathcal{E}).$$

We also denote

$$f''_q = f''_{s,q} - f''_{q,t} \quad , \forall q \in \mathcal{P}.$$

Notice that, since $f'_q = 0$ for all $q \notin \mathcal{P}'$ as well as for $q \in \mathcal{P}'$ such that $f_q < 0$ (see (70) and (71)), we always have, according to (82) and (83),

$$f''_q = f_q + f'_q \quad , \forall q \in \mathcal{P}. \tag{85}$$

Proposition 5. *The mapping $f'' : (\mathcal{V} \times \mathcal{V}) \to \mathbb{R}$ is a max-flow in $\mathcal{G}$.*

Proof. Let us first show that f'' satisfies the capacity constraints. Let $(q',q) \in \mathcal{E}$.

- If q or $q' \notin \mathcal{P}'$, $q \neq s$, $q' \neq s$: then $(q',q) \notin \mathcal{E}'$ and using (82) we have

$$0 \leq f''_{q',q} = f_{q',q} \leq c_{q',q}.$$

- If $q' \in \Sigma^-$ and $q = s$ or t:
  - If moreover $f_{q'} < 0$, then using (83), $0 \leq f''_{s,q'} = 0 \leq c_{s,q'}$ and $0 \leq f''_{q',t} = f_{q',t} \leq c_{q',t}$.
  - If $f_{q'} \geq 0$, then using (83) and (71), we find that $0 \leq f''_{s,q'} = f_{s,q'} - f'_{q',t} \leq c_{s,q'}$ and $0 \leq f''_{q',t} = 0 \leq c_{q',t}$.
- If $q' \in \Sigma^+$ and $q = s$ or t: since $q' \in B_p$, we necessarily have $f_{q'} \geq 0$, then using (83) and (71), we have $0 \leq f''_{s,q'} = f_{s,q'} + f'_{s,q'} \leq c_{s,q'}$ and $0 \leq f''_{q',t} = 0 \leq c_{q',t}$.
- If $(q',q) \in (\mathcal{P}' \times \mathcal{P}')$:
  - If moreover $f_{q',q} \leq f_{q,q'}$, then (84) guarantees $0 \leq f''_{q',q} = 0 \leq c_{q',q}$.
  - If $f_{q',q} > f_{q,q'}$, using (72), we have

$$0 \leq f'_{q,q'} \leq c'_{q,q'} = f_{q',q} - f_{q,q'},$$

    and finally (84) guarantees that

$$0 \leq f''_{q',q} = f_{q',q} - f_{q,q'} - f'_{q,q'} \leq c_{q',q}.$$

Let us now prove the flow conservation. Let $q \in \mathcal{P}$.

- If $q \notin \mathcal{P}'$ and $q \neq s$, then for any $q' \in \sigma_{\mathcal{E}}(q)$ the definition of $\mathcal{E}'$ given in (67) guarantees that both (q,q') and $(q',q) \notin \mathcal{E}'$. Using (82), we obtain $f''_{q,q'} = f_{q,q'}$ and $f''_{q',q} = f_{q',q}$, for all $q' \in \sigma_{\mathcal{E}}(q)$, and therefore
$$\sum_{q' \in \sigma_{\mathcal{E}}(q)} f''_{q',q} = \sum_{q' \in \sigma_{\mathcal{E}}(q)} f_{q',q} = \sum_{q' \in \sigma_{\mathcal{E}}(q)} f_{q,q'} = \sum_{q' \in \sigma_{\mathcal{E}}(q)} f''_{q,q'}.$$
- If $q \in \mathcal{P}'$, the flow conservation constraint given by (17) for f and f' at q can be decomposed to provide
$$f_q + \sum_{\substack{q' \in \sigma_{\mathcal{E}}(q) \\ q' \notin \sigma_{\mathcal{E}'}(q)}} (f_{q',q} - f_{q,q'}) + \sum_{\substack{q' \in \sigma_{\mathcal{E}'}(q) \\ f_{q',q} > f_{q,q'}}} (f_{q',q} - f_{q,q'}) + \sum_{\substack{q' \in \sigma_{\mathcal{E}'}(q) \\ f_{q',q} \leq f_{q,q'}}} (f_{q',q} - f_{q,q'}) = 0$$
and
$$f'_q + \sum_{\substack{q' \in \sigma_{\mathcal{E}'}(q) \\ f_{q',q} > f_{q,q'}}} (0 - f'_{q,q'}) + \sum_{\substack{q' \in \sigma_{\mathcal{E}'}(q) \\ f_{q',q} \leq f_{q,q'}}} (f'_{q',q} - 0) = 0.$$
Summing these equalities and using (85), (82) and (84), we obtain
$$f''_q + \sum_{\substack{q' \in \sigma_{\mathcal{E}}(q) \\ q' \notin \sigma_{\mathcal{E}'}(q)}} (f''_{q',q} - f''_{q,q'}) + \sum_{\substack{q' \in \sigma_{\mathcal{E}'}(q) \\ f_{q',q} > f_{q,q'}}} (f''_{q',q} - f''_{q,q'}) + \sum_{\substack{q' \in \sigma_{\mathcal{E}'}(q) \\ f_{q',q} \leq f_{q,q'}}} (f''_{q',q} - f''_{q,q'}) = 0.$$
The latter corresponds to flow conservation constraint (17) at the node q for f''.

Altogether, we now know that f'' is a flow. We still need to show that it is a max-flow. The latter property is in fact trivially obtained since (83) and (82) guarantee that $f''_{q,t} = f_{q,t}$, for all $q \in \mathcal{P}$. Therefore, the value of f'' is equal to the value of f. Since f is a max-flow, this value is maximal and f'' is a max-flow. □

Proposition 6. *If Σ^+ is a minimum s-t cut in the graph $\mathcal{G}'$ defined in Section C, then the max-flow f'' is such that*
$$\forall q \in \sigma_{\mathcal{E}}(p), \qquad f''_{q,p} = 0.$$
As a consequence,
$$\forall q \in \sigma_{\mathcal{E}}(p), \qquad f''_{p,q} \geq f''_{q,p}.$$

Proof. Since f' is a max-flow in $\mathcal{G}'$ and Σ^+ is a min s-t cut in $\mathcal{G}'$, Ford-Fulkerson theorem guarantees that they have the same value. We therefore have

$$\begin{aligned} \mathrm{val}_{\mathcal{G}'}(f') = \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) &= \sum_{\substack{q'\in\Sigma^+\\ q\notin\Sigma^+\\ (q',q)\in\mathcal{E}'}} c'_{q',q} \\ &= \sum_{q\in\Sigma^-} c'_{p,q} \end{aligned} \tag{86}$$

Moreover, since f' is a flow, the total amount of flow entering and exiting Σ^+ are equal. Therefore, we have (see (17))

$$\sum_{q\in\Sigma^+} f'_q + \sum_{\substack{q'\in\Sigma^+\\ q\notin\Sigma^+\\ q\in\sigma_{\mathcal{E}'}(q')}} (f'_{q,q'} - f'_{q',q}) = 0.$$

Together with (14) and (68), this guarantees that

$$\mathrm{val}_{\mathcal{G}'}(f') = \sum_{q\in\Sigma^+} f'_q = \sum_{\substack{q'\in\Sigma^+\\ q\notin\Sigma^+\\ q\in\sigma_{\mathcal{E}'}(q')}} (f'_{q',q} - f'_{q,q'}) = \sum_{q\in\Sigma^-} (f'_{p,q} - f'_{q,p}).$$

Combined with (86), this provides

$$\sum_{q\in\Sigma^-} c'_{p,q} = \sum_{q\in\Sigma^-} f'_{p,q} - \sum_{q\in\Sigma^-} f'_{q,p}. \tag{87}$$

As a consequence,

$$\sum_{q\in\Sigma^-} f'_{q,p} = \sum_{q\in\Sigma^-} (f'_{p,q} - c'_{p,q}) \leq 0.$$

However, since for all $q\in\Sigma^-$, $f'_{q,p}\geq 0$, we finally obtain that

$$\forall q\in\Sigma^-, f'_{q,p} = 0.$$

Using (87) again, (62) and (72), this provides

$$\forall q\in\Sigma^-, f'_{p,q} = c'_{p,q} = f_{q,p} - f_{p,q}.$$

Therefore, using (62) and (84),

$$\forall q\in\Sigma^-, f''_{q,p} = 0. \tag{88}$$

Moreover, using (63) and (84), we also have

$$\forall q \in (\Sigma^+ \cap \sigma_{\mathcal{E}}(p)), f''_{q,p} = 0. \tag{89}$$

Combining (88) and (63), we finally obtain

$$\forall q \in \sigma_{\mathcal{E}}(p), f''_{q,p} = 0,$$

which concludes the proof. □

Proposition 7. *Let $\mathcal{G}$ be the graph defined in Section C, let B satisfy* (29) *and let us assume that $p \in \mathcal{P}$ is such that*

$$\forall q \in B_p, \quad c_q \geq 0 \quad \text{and} \quad c_q \geq \sum_{\substack{q' \in \sigma_{\mathcal{E}}(q) \\ q' \notin B_p}} c_{q,q'}, \tag{90}$$

then, there exists a max-flow f in $\mathcal{G}$ such that

$$\forall q \in \sigma_{\mathcal{E}}(p), f_{p,q} \geq f_{q,p} = 0. \tag{91}$$

Proof. This is a straightforward consequence of Proposition 5, Proposition 4 and Proposition 6.

Indeed, if (90) holds, we know that for any max-flow f in $\mathcal{G}$ and any $S \subset \mathcal{P}'$

$$\sum_{q \in \Sigma^+ \setminus (S \cup \{p\})} \left[c_q + \sum_{q' \notin \Sigma^+} (f_{q',q} - f_{q,q'}) \right] \geq 0$$

and therefore, Proposition 4 guarantees that Σ^+ is a min s-t cut in $\mathcal{G}'$. Then, Proposition 5 guarantees that f'' is a max-flow in $\mathcal{G}$ and Proposition 6 guarantees that f'' satisfies (91). □

A useless node

Throughout this section, we consider a graph $\mathcal{G}$ as constructed in Section C, a set B satisfying (29), a pixel $p \in \mathcal{P}$ satisfying (90) and a max-flow f in $\mathcal{G}$ satisfying (91).

The purpose of this section is to modify f so-that it remains a max-flow in $\mathcal{G}$ and satisfies

$$\forall q \in \sigma_{\mathcal{E}}(p), \quad f_{p,q} = f_{q,p} = 0.$$

The latter obviously implies that the node p is useless when computing the max-flow in $\mathcal{G}$.

Since the method for modifying f is analogous to the one used in Section C, we chose to use the same notations for the objects playing the same role. Beware not to confuse their definition.

First, we denote

$$\mathcal{P}' = B_p \quad , \Sigma^+ = B_p \setminus \{p\} \quad \text{and} \quad \Sigma^- = \{p\}. \tag{92}$$

In order to modify f, we build a graph $\mathcal{G}' = (\mathcal{P}', \mathcal{E}', c')$ where $\mathcal{E}'$and c' are defined below. We consider

$$\mathcal{E}' = \big(\mathcal{E} \cap \big(\Sigma^+ \times \Sigma^+\big)\big) \ \bigcup \ \big((\sigma_{\mathcal{E}}(p) \cap \Sigma^+) \times \Sigma^-\big) \ \bigcup \ \big(\{s\} \times \Sigma^+\big) \ \bigcup \ \{(p,t)\}. \tag{93}$$

We define the capacities c' by

$$c'_{q,q'} = c_{q,q'} - f_{q,q'} + f_{q',q} \qquad , \forall (q,q') \in \big(\mathcal{E} \cap \big(\Sigma^+ \times \Sigma^+\big)\big) \tag{94}$$

$$c'_{q,p} = f_{p,q} \qquad , \forall q \in (\sigma_{\mathcal{E}}(p) \cap \Sigma^+) \tag{95}$$

$$c'_{s,q} = c_q - f_q \qquad , \forall q \in \Sigma^+ \tag{96}$$

$$c'_{p,t} = f_p \tag{97}$$

As usual, in order to simplify the notations, we artificially set

$$c'_{q,q'} = 0 \quad , \forall (q,q') \in (\mathcal{P}' \times \mathcal{P}') \setminus \mathcal{E}', \tag{98}$$

and we write

$$c'_q = c'_{s,q} - c'_{q,t} \quad , \forall q \in \mathcal{P}'. \tag{99}$$

Notice first that, for any $S \subset \mathcal{P}'$, the value of the s-t cut $((S \cup \{s\}), (\mathcal{P}' \setminus S) \cup \{t\})$ in $\mathcal{G}'$ depends on whether $p \in S$ or $p \notin S$. If $p \in S$, we have

$$\mathrm{val}_{\mathcal{G}'}(S) = c'_{p,t} + \sum_{q \in (\Sigma^+ \setminus S)} c'_q + \sum_{\substack{q \in S \\ q' \in (\mathcal{P}' \setminus S)}} c'_{q,q'}.$$

Therefore, we trivially have using (94)-(99)

$$\mathrm{val}_{\mathcal{G}'}(S) \geq c'_{p,t} = f_p \quad , \text{if } p \in S. \tag{100}$$

Moreover, for any $S \subset \mathcal{P}'$, the value of the s-t cut $((S \cup \{s\}), (\mathcal{P}' \setminus S) \cup \{t\})$ in $\mathcal{G}'$ is given by

$$\mathrm{val}_{\mathcal{G}'}(S) = \sum_{q \in (\Sigma^+ \setminus S)} c'_q + \sum_{\substack{q \in S \\ q' \in (\mathcal{P}' \setminus S)}} c'_{q,q'} \quad , \text{if } p \notin S. \tag{101}$$

In particular, if $S = \Sigma^+$, we obtain using (95), the conservation of the flow f at p and (91) that

$$\begin{aligned} \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) &= \sum_{q \in (\Sigma^+ \cap \sigma_{\mathcal{E}'}(p))} c'_{q,p}, \\ &= \sum_{q \in (\Sigma^+ \cap \sigma_{\mathcal{E}}(p))} f_{p,q}, \\ &= f_p \ . \end{aligned} \tag{102}$$

The following proposition holds.

Proposition 8. *Let $\mathcal{G}'$ be the graph constructed in Section C. For any $S \subset \mathcal{P}'$,*

- *if $p \notin S$*

$$\mathrm{val}_{\mathcal{G}'}(S) = \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) + \sum_{\substack{q \in S \\ q' \in (\Sigma^+ \setminus S)}} c_{q,q'} + \sum_{q \in (\Sigma^+ \setminus S)} \left[c_q + \sum_{q' \notin \mathcal{P}'} (f_{q',q} - f_{q,q'}) \right], \tag{103}$$

- *if $p \in S$*

$$\mathrm{val}_{\mathcal{G}'}(S) \geq \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right). \tag{104}$$

Proof. Notice first that, if $p \in S$, (104) is a straightforward consequence of (100) and (102). Let us assume from now on that $p \notin S$.

Since f is a flow, the total amount of flow entering and exiting $(\mathcal{P}' \setminus S)$ are equal (see (17)) and therefore, using (92)

$$f_p + \sum_{q \in (\Sigma^+ \setminus S)} f_q + \sum_{\substack{q \in (\mathcal{P}' \setminus S) \\ q' \notin (\mathcal{P}' \setminus S)}} (f_{q',q} - f_{q,q'}) = 0.$$

Using (102), (96) and (99), we obtain

$$\mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) + \sum_{q \in (\Sigma^+ \setminus S)} (c_q - c'_q) + \sum_{\substack{q \in (\mathcal{P}' \setminus S) \\ q' \notin (\mathcal{P}' \setminus S)}} (f_{q',q} - f_{q,q'}) = 0.$$

Combined with (101), this becomes

$$\mathrm{val}_{\mathcal{G}'}(S) = \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) + \sum_{q \in (\Sigma^+ \setminus S)} c_q + \sum_{\substack{q \in (\mathcal{P}' \setminus S) \\ q' \notin (\mathcal{P}' \setminus S)}} (f_{q',q} - f_{q,q'}) + \sum_{\substack{q \in S \\ q' \in (\mathcal{P}' \setminus S)}} c'_{q,q'}. \tag{105}$$

We now decompose the last term of the above equation using (94), (95) and (91) and write

$$\begin{aligned}
\sum_{\substack{q \in S \\ q' \in (\mathcal{P}' \setminus S)}} c'_{q,q'} &= \sum_{\substack{q \in S \\ q' \in (\Sigma^+ \setminus S)}} (c_{q,q'} - f_{q,q'} + f_{q',q}) + \sum_{q \in S} f_{p,q} \\
&= \sum_{\substack{q \in S \\ q' \in (\Sigma^+ \setminus S)}} c_{q,q'} - \sum_{\substack{q' \in S \\ q \in (\Sigma^+ \setminus S)}} (f_{q',q} - f_{q,q'}) + \sum_{q' \in S} (f_{p,q'} - f_{q',p}) \\
&= \sum_{\substack{q \in S \\ q' \in (\Sigma^+ \setminus S)}} c_{q,q'} - \sum_{\substack{q \in (\mathcal{P}' \setminus S) \\ q' \in S}} (f_{q',q} - f_{q,q'})
\end{aligned}$$

Combining the latter with (105), we finally obtain

$$\mathrm{val}_{\mathcal{G}'}(S) = \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) + \sum_{q \in (\Sigma^+ \setminus S)} c_q + \sum_{\substack{q \in S \\ q' \in (\Sigma^+ \setminus S)}} c_{q,q'} + \sum_{\substack{q \in (\mathcal{P}' \setminus S) \\ q' \notin \mathcal{P}'}} (f_{q',q} - f_{q,q'}).$$

Using (29), we remark that for any $q' \notin \mathcal{P}'$, $q' \notin \sigma_{\mathcal{E}}(p)$ and we can finally deduce that (103) holds for all $S \subset \mathcal{P}'$ such that $p \notin S$. □

As in Section C, we will from now on consider a max flow f' in the graph $\mathcal{G}'$ built in the current section. We also artificially extend the flow f' and set

$$f'_{q,q'} = 0, \text{ for all } (q,q') \in ((\mathcal{V}' \times \mathcal{V}') \setminus \mathcal{E}'). \tag{106}$$

Once again, the graph $\mathcal{G}'$ satisfies (5) and (9), therefore, as usual, we denote for simplicity

$$f'_q = f'_{s,q} - f'_{q,t} \quad , \forall q \in \mathcal{P}'. \tag{107}$$

We are now going to combine f and f' in order to build a mapping $f'' : \mathcal{E} \to \mathbb{R}$ which will turn out to be a max-flow in $\mathcal{G}$ such that

$$f''_{p,q} = f''_{q,p} = 0 \quad , \forall q \in \sigma_{\mathcal{E}}(p).$$

As for $\mathcal{G}'$ and f', beware that the mapping f' is different in Section C and in the current section.

Let us begin with the definition of f''. We set

$$f''_q = f_q \qquad \forall q \notin \mathcal{P}' \tag{108}$$

$$f''_{q,q'} = f_{q,q'} \qquad \forall (q,q') \in \mathcal{E}, \text{ with } q \notin \mathcal{P}' \text{ or } q' \notin \mathcal{P}' \tag{109}$$

$$f''_q = f_q + f'_q \qquad \forall q \in \mathcal{P}' \tag{110}$$

$$f''_{q,q'} = (f_{q,q'} + f'_{q,q'}) - (f_{q',q} + f'_{q',q}) \qquad \forall (q,q') \in \left(\mathcal{E} \cap (\Sigma^+)^2\right) \text{ and } f_{q,q'} + f'_{q,q'} \geq f_{q',q} + f'_{q',q} \tag{111}$$

$$f''_{q,q'} = 0 \qquad \forall (q,q') \in \left(\mathcal{E} \cap (\Sigma^+)^2\right) \text{ and } f_{q,q'} + f'_{q,q'} < f_{q',q} + f'_{q',q} \tag{112}$$

$$f''_{p,q} = f_{p,q} - f'_{q,p} \qquad \forall q \in (\mathcal{P}' \cap \sigma_{\mathcal{E}}(p)) \tag{113}$$

$$f''_{q,p} = 0 \qquad \forall q \in (\mathcal{P}' \cap \sigma_{\mathcal{E}}(p)) \tag{114}$$

We also define

$$f''_{s,q} = \max(f''_q, 0) \quad \text{ and } \quad f''_{q,t} = \max(-f''_q, 0) \quad , \forall q \in \mathcal{P}. \tag{115}$$

Notice that the equation (108)-(115) permit to define $f''_{q,q'}$ for all $(q,q') \in \mathcal{E}$. Once again, we extend f'' outside $\mathcal{E}$ and set

$$f''_{q,q'} = 0, \text{ for all } (q,q') \in \left((\mathcal{V} \times \mathcal{V}) \setminus \mathcal{E}\right).$$

The following proposition holds.

Proposition 9. *The mapping $f'' : (\mathcal{V} \times \mathcal{V}) \to \mathbb{R}$ is max-flow in $\mathcal{G}$.*

Proof. Notice first that, if f'' is a flow in $\mathcal{G}$ it is necessarily a max flow since, according to (90), $(\sigma_{\mathcal{E}}(t) \cap \mathcal{P}') = \emptyset$ and therefore, using (108), we always have $f''_{q,t} = f_{q,t}$, for all $q \in \sigma_{\mathcal{E}}(t)$. Therefore, the $\mathrm{val}_{\mathcal{G}}(f'') = \mathrm{val}_{\mathcal{G}}(f)$ and the latter is maximal in $\mathcal{G}$.

In order to show that f'' is a flow we first show that it satisfies the capacity constraints. Let $(q, q') \in \mathcal{E}$.

- If $q = s$ and $q' \notin B_p$ or if $q \notin B_p$ and $q' = t$, using (108) and (115), we know that

$$0 \leq f''_{q,q'} = f_{q,q'} \leq c_{q,q'} \quad \text{and} \quad 0 \leq f''_{q',q} = f_{q',q} \leq c_{q',q}.$$

- If $q \notin B_p$ or $q' \notin B_p$, using (109), we obtain again

$$0 \leq f''_{q,q'} = f_{q,q'} \leq c_{q,q'}.$$

- If $q = s$ and $q' \in \Sigma^+$, using (110) and (96), we get

$$0 \leq f''_{q,q'} = f_{s,q'} + f'_{s,q'} \leq c_{q,q'}.$$

- If $q = s$ and $q' = p$, using (110) and (97), we get

$$0 \leq f''_{q,q'} = f_{s,p} - f'_{p,t} \leq c_{q,q'}.$$

- If $(q, q') \in (\Sigma^+)^2$ and $f_{q,q'} + f'_{q,q'} \geq f_{q',q} + f'_{q',q}$, using (111) and (94), we obtain

$$0 \leq f''_{q,q'} = f_{q,q'} + f'_{q,q'} - f_{q',q} - f'_{q',q} \leq c_{q,q'} - f'_{q',q} \leq c_{q,q'}.$$

- If $(q, q') \in (\Sigma^+)^2$ and $f_{q,q'} + f'_{q,q'} < f_{q',q} + f'_{q',q}$, using (111), we trivially have

$$0 \leq f''_{q,q'} = 0 \leq c_{q,q'}.$$

- If $q = p$ and $q' \in (B_p \cap \sigma_{\mathcal{E}}(p))$, using (113) and (95), we get

$$0 \leq f''_{q,q'} = f_{p,q'} - f'_{q',p} \leq c_{q,q'}.$$

- If $q \in (B_p \cap \sigma_{\mathcal{E}}(p))$ and $q' = p$, then (114) trivially guarantees that

$$0 \leq f''_{q,q'} = 0 \leq c_{q,q'}.$$

In order to show the flow conservation constraints, we consider, from now on, $q \in \mathcal{P}$.

- If $q \notin \mathcal{P}'$, we have, using (108) and (109), we have $f''_{q,q'} = f_{q,q'}$ and $f''_{q',q} = f_{q',q}$, for all $q' \in \sigma_{\mathcal{E}}(q)$. Therefore,

$$\sum_{q\in\sigma_{\mathcal{E}}(q)} f''_{q',q} = \sum_{q\in\sigma_{\mathcal{E}}(q)} f_{q',q} = \sum_{q\in\sigma_{\mathcal{E}}(q)} f_{q,q'} = \sum_{q\in\sigma_{\mathcal{E}}(q)} f''_{q,q'}.$$

- If $q \in \Sigma^+$, expressing that the two flows f and f' are conserved at q, we obtain using (13) and (91)

$$f_q + \sum_{\substack{q'\in\sigma_{\mathcal{E}}(q)\\ q'\notin\mathcal{P}'}} (f_{q',q} - f_{q,q'}) + \sum_{\substack{q'\in\sigma_{\mathcal{E}}(q)\\ q'\in\Sigma^+}} (f_{q',q} - f_{q,q'}) + f_{p,q} = 0$$

and

$$f'_q + \sum_{\substack{q'\in\sigma_{\mathcal{E}}(q)\\ q'\in\Sigma^+}} (f'_{q',q} - f'_{q,q'}) \quad - f'_{q,p} = 0.$$

Summing those inequalities and using (109)-(114), we obtain

$$f''_q + \sum_{\substack{q'\in\sigma_{\mathcal{E}}(q)\\ q'\notin\mathcal{P}'}} (f''_{q',q} - f''_{q,q'}) + \sum_{\substack{q'\in\sigma_{\mathcal{E}}(q)\\ q'\in\Sigma^+}} (f''_{q',q} - f''_{q,q'}) + (f''_{p,q} - f''_{q,p}) = 0.$$

The latter expresses that f'' is conserved at the node q.

- If $q = p$, then using (110), (113) and (114) as well as (29) and (97), we obtain

$$\sum_{q\in\sigma_{\mathcal{E}}(p)} (f''_{q,p} - f''_{p,q}) = f_{s,p} - f'_{p,t} - \sum_{q\in(\mathcal{P}'\cap\sigma_{\mathcal{E}}(p))} (f_{p,q} - f'_{q,p}).$$

Using that $f_{p,t} = 0$ (see (90), (10) and (9)), $f'_{s,p} = 0$ (see (93) and (98)), $f_{q,p} = 0$ (see (91)) and $f'_{p,q} = 0$ (see (93) and (98)), we obtain

$$\sum_{q\in\sigma_{\mathcal{E}}(p)} (f''_{q,p} - f''_{p,q}) = (f_{s,p} - f_{p,t}) + (f'_{s,p} - f'_{p,t}) - \sum_{q\in(\mathcal{P}'\cap\sigma_{\mathcal{E}}(p))} \left[(f_{p,q} - f_{q,p}) + (f'_{p,q} - f'_{q,p})\right].$$

Simplifying, we finally obtain

$$\begin{aligned}\sum_{q\in\sigma_{\mathcal{E}}(p)} (f''_{q,p} - f''_{p,q}) &= \sum_{q\in\sigma_{\mathcal{E}}(p)} (f_{q,p} - f_{p,q}) + \sum_{q\in\sigma_{\mathcal{E}}(p)} (f'_{q,p} - f'_{p,q}),\\ &= 0,\end{aligned}$$

since the two flows f and f' are conserved at p.

This concludes the proof. □

Proposition 10. *If Σ^+ is a minimum s-t cut in the graph $\mathcal{G}'$ defined in Section C, then the max-flow f'' is such that*

$$\forall q \in \sigma_{\mathcal{E}}(p), \qquad f''_{q,p} = f''_{p,q} = 0.$$

As a consequence, removing the node p from the graph $\mathcal{G}$ does not modify its maximal flow value.

Proof. If Σ^+ is a minimum s-t cut in the graph $\mathcal{G}'$ defined in Section C, then Ford-Fulkerson theorem, (102) and (93) guarantee that

$$f_p = \mathrm{val}_{\mathcal{G}'}\left(\Sigma^+\right) = \mathrm{val}_{\mathcal{G}'}\left(f'\right) = \sum_{q \in \Sigma^+} f'_q.$$

Now, since the total amount of flow f' entering and exiting Σ^+ are equal, we obtain, using (93), that

$$f_p = \sum_{q \in (\sigma_{\mathcal{E}}(p) \cap \Sigma^+)} f'_{q,p}.$$

Using that the flow f' is preserved at p and (93), we finally get

$$f_p = f'_{p,t}.$$

Using (110), (107) and (106) this yields

$$f''_p = f_p - f'_{p,t} = 0,$$

which, using (115), provides

$$f''_{s,p} = f''_{p,t} = 0.$$

Together with (114), this guarantees that

$$\text{for all } q \in \sigma_{\mathcal{E}}(p), \quad f''_{q,p} = 0. \tag{116}$$

Expressing the flow conservation constraint at p for f'', we deduce from (116) that

$$\sum_{q \in \sigma_{\mathcal{E}}(p)} f''_{p,q} = \sum_{q \in \sigma_{\mathcal{E}}(p)} f''_{q,p} = 0,$$

which guarantees that

$$\text{for all } q \in \sigma_{\mathcal{E}}(p), \quad f''_{p,q} = 0,$$

since $f''_{p,q} \geq 0$, for all $q \in \sigma_{\mathcal{E}}(p)$.

Together with (116), this concludes the proof. □

We can now conclude with the following proposition.

Proposition 11. *Let $\mathcal{G}$ be the graph defined in Section C, let B satisfy* (29) *and let us assume that $p \in \mathcal{P}$ satisfies* (90)*. Then, there exists a max-flow f in $\mathcal{G}$ such that*

$$\forall q \in \sigma_{\mathcal{E}}(p), \qquad f_{p,q} = f_{q,p} = 0. \tag{117}$$

As a consequence, removing the node p from the graph $\mathcal{G}$ does not modify its maximal flow value.

Proof. This is a straightforward consequence of Proposition 7, Proposition 8, Proposition 9 and Proposition 10.

Indeed, if (90) holds, we know that there is max-flow f in $\mathcal{G}$ satisfying (91). Therefore, using the notations of Section C, we know that for any $S \subset \mathcal{P}'$ such that $p \notin S$

$$\sum_{q \in \Sigma^+ \setminus S} \left[c_q + \sum_{q' \notin \mathcal{P}'} (f_{q',q} - f_{q,q'}) \right] \geq 0.$$

Therefore, for $\mathcal{G}'$ as defined in Section C, Proposition 8 guarantees that for any $S \subset \mathcal{P}'$

$$\text{val}_{\mathcal{G}'}(S) \geq \text{val}_{\mathcal{G}'}\left(\Sigma^+\right),$$

and therefore Σ^+ is a min s-t cut in $\mathcal{G}'$. Then, Proposition 9 guarantees that f'' is a max-flow in $\mathcal{G}$ and Proposition 10 guarantees that f'' satisfies (117). □

D Computation details of the upper bound η_{max}

Computing the upper bound η_{max} amounts to loop over all possible proportions of degraded pixels k from n to 1 and find the one such that $\mathbb{P}(X > k) \geq \varepsilon$. For each proportion k, the distribution function $\mathbb{P}(X > k)$ is recursively computed and a single iteration corresponds to the computation of $\mathbb{P}(X = k)$. Let us denote $T(n)$ the cost for evaluating (40) at k. The complexity of the algorithm for estimating η_{max} is therefore $O(nT(n))$. Since (40) is decreasing (see (41)), the convergence of this algorithm can be improved using a dichotomic search, leading to a complexity of $O(T(n)log_2(n))$.

When estimating the upper bound η_{max}, some technical difficulties can appear with (40). Indeed, for large window radii, the combinations in (40) can include a dozen of digits and even more. Due to the limited precision of machines, the traditional approach for computing these combinations quickly appears to be a dead-end when $r > 15$ in practice. To overcome this difficulty, a common procedure consists in taking the logarithm of $\mathbb{P}(X = k)$.

For a fixed amount of noise $\xi \in]0, 1[$, one ends up with

$$\begin{aligned} log(\mathbb{P}(X = k)) &= log(\tbinom{n}{k}\xi^k(1-\xi)^{n-k}) \\ &= log(\tbinom{n}{k}) + log(\xi^k) + log((1-\xi)^{n-k}) \\ &= log\left(\tfrac{n}{k!(n-k)!}\right) + klog(\xi) + (n-k)log(1-\xi) \\ &= \underbrace{log(n!) - log(k!) - log((n-k)!)}_{(*)} + klog(\xi) + (n-k)log(1-\xi). \end{aligned} \tag{118}$$

From (118), we have chosen to get an approximation of (*) using the Ramanujan's formula because the complexity is only $O(1)$.

The approximation of $log(x!)$ proposed by Ramanujan in [Ram88] is [6]

$$log(x!) \simeq xlog(x) - x + \frac{log(x(1+4x(1+2x)))}{6} + \frac{log(\pi)}{2}. \tag{119}$$

[6] The absolute error between $log(x!)$ and (119) is indeed maximum for $x = 1$ and progressively decreases as x tends to infinity. For $x = 1$, this error is about 10^{-4} and therefore is negligible.

References

[BCA] A. Bretto, H. Cherifi, and D. Aboutajdine. Hypergraph imaging: An overview. *Pattern Recognition*, 35(3):651–658.

[BF70] C. Brice and C. Fenema. Scene analysis using regions. *Artificial Intelligence*, 1(3):205–226, 1970.

[BG89] J. Beaulieu and M. Goldberg. Hierarchy in picture segmentation: A stepwise optimization approach. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 11(2):150–163, 1989.

[BJ01] Y. Boykov and M-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, volume 1, pages 105–112, 2001.

[BK04] 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, 2004.

[BK05] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *ICCV*, pages 26–33, October 2005.

[BMG+10] A. J. Buckler, J. L. Mulshine, R. Gottlieb, B. Zhao, P. D. Mozley, and L. Schwartz. The use of volumetric CT as an imaging biomarker in lung cancer. *Academic Radiology*, 17(1):100–106, January 2010.

[Bou97] C. A. Bouman. Cluster: An unsupervised algorithm for modeling Gaussian mixtures. April 1997.

[BT08] E. Bae and X-C. Tai. Graph cuts for the multiphase mumford-shah model using piecewise constant level set methods. Technical report, UCLA CAM, 2008.

[BVZ99] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV*, pages 377–384, 1999.

[CA08] C. Cigla and A.A. Alatan. Region-based image segmentation via graph cuts. In *ICIP*, pages 2272–2275, 2008.

[Can86] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence*, 8(6):679–714, 1986.

[CGNT09] C. Couprie, L. Grady, L. Najman, and H. Talbot. Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest. In *ICCV*, pages 731–738, 2009.

[CV01] T.F. Chan and L. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[DB08] A. Delong and Y. Boykov. A scalable graph-cut algorithm for N-D grids. In *CVPR*, pages 1–8, 2008.

[DF56] G.B. Dantzig and D.R. Fulkerson. On the max-flow min-cut theorems of networks. *Annals of Mathematics Study*, 38:215–221, 1956.

[Dic45] L. Dice. Measure of the amount of ecological association between species. *Ecology*, 26(3):297–302, 1945.

[Din70] E.A. Dinic. Algorithm for solution of a problem of maximum flow in a network with power estimation. *Soviet Mathematics Doklady*, 11:1277–1280, 1970.

[DLR77] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[DMM04] A. Desolneux, L. Moisan, and J-M. Morel. *Seeing, Thinking and Knowing*, chapter Gestalt Theory and Computer Vision, pages 71–101. Kluwer Academic Publishers, 2004.

[DS04] J. Darbon and M. Sigelle. Exact optimization of discrete constrained total variation minimization problems. In *IWCIA*, volume 3322, pages 548–557, 2004.

[ECS56] P. Elias, D.L. Collins, and C. Shannon. A note on the maximum flow through a network. *IRE Transactions on Information Theory*, 2(4):117–119, 1956.

[EK72] J. Edmonds and R.M. Karp. Theoretical improvements in algorithmic efficiency for network flows problems. *Journal of the ACM*, 19(2):248–264, 1972.

[FD05] D. Freedman and P. Drineas. Energy minimization via graph cuts: Settling what is possible. In *CVPR*, volume 2, pages 939–946, 2005.

[FF56] L.R. Ford and D.R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[FH04] P. F. Felzenszwalb and D. P. Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell Computing and Information Science, 2004.

[GFL04] L. Grady and G. Funka-Lea. Multi-label image segmentation for medical applications based on graph-theoretic eletrical potentials. In *ECCV*, pages 230–245, 2004.

[GJ08] L. Grady and M-P. Jolly. Weights and topology: A study of the effects of graph construction on 3D image segmentation. In *Proceedings of MICCAI*, volume 1, pages 153–161, 2008.

[GKMP09] M. A. Gavrielides, L. M. Kinnard, K. J. Myers, and N. Petrick. Noncalcified lung nodules: Volumetric assessment with thoracic CT. *Radiology*, 251(1):26–37, April 2009.

[GPS89] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989.

[GT88] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum-flow problem. *Journal of the ACM*, 35(4):921–940, 1988.

[GY09] D. Goldfarb and W. Yin. Parametric maximum flow algorithms for fast total variation minimization. *SIAM Journal on Scientific Computing*, 31:5, October 2009.

[HBS+11] D. Han, J.E. Bayouth, Q. Song, A. Taurani, M. Sonka, J.M. Buatti, and X. Wu. Globally optimal tumor segmentation in PET-CT images: A graph-based co-segmentation method. In *IPMI*, volume 22, pages 245–256, 2011.

[Hoc98] D. S. Hochbaum. The pseudoflow algorithm and the pseudoflow-based simplex for the maximum flow problem. In *IPCO*, pages 325–337, 1998.

[HP76] S. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *Journal of The Association for Computing Machinery*, 23(3):368–388, April 1976.

[HS09] D.S. Hochbaum and V. Singh. An efficient algorithm for co-segmentation. In *ICCV*, pages 269–276, 2009.

[IG99] H. Ishikawa and D. Geiger. Mapping image restoration to a graph problem. In *EURASIP*, pages 189–193, 1999.

[Iko05] L. Ikonen. Pixel queue algorithm for geodesic distance transforms. In *Discrete Geometry for Computer Imagery*, volume 3429, pages 228–239, April 2005.

[Ish03] H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003.

[JB07] O. Juan and Y. Boykov. Capacity scaling for graph cuts in vision. In *ICCV*, pages 1–8, 2007.

[KDB+06] J-M. Kuhnigk, V. Dicken, L. Bornemann, A. Bakai, D. Wormanns, S. Krass, and H-O. Peitgen. Morphological segmentation and partial volume analysis for volumetry of solid pulmonary lesions in thoracic CT scans. *IEEE Transactions on Medical Imaging*, 25(4):417–434, April 2006.

[KLR10] P. Kohli, V. Lempitsky, and C. Rother. Uncertainty driven multi-scale energy optimization. pages 242–251, 2010.

[Kof35] K. Koffka. *Principles of Gestalt Psychology.* Routledge & Kergan Paul, 1935.

[KRYH03] W. J. Kostis, A. P. Reeves, D. F. Yankelevitz, and C. I. Henschke. Three-dimensional segmentation and growth-rate estimation of small pulmonary nodules in helical CT images. *IEEE Transactions on Medical Imaging*, 22(10):1259–1274, October 2003.

[KT07] P. Kohli and P. H. S. Torr. Dynamic graph cuts for efficient inference in markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2079–2088, 2007.

[KT08] P. Kohli and P. H. S. Torr. Measuring uncertainty in graph cut solutions. *Computer Vision and Image Understanding*, 112(1):30–38, 2008.

[KZ04] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[Köh29] W. Köhler. *Psychologie de la forme.* Gallimard, 1929.

[LB07] V. Lempitsky and Y. Boykov. Global optimization for shape fitting. In *CVPR*, pages 1–8, 2007.

[Lec09] J. Lecoeur. *Segmentation d'IRM cérébrales multidimensionnelles par coupe de graphe*. PhD thesis, Université de Rennes I, 2009.

[Li00] Z. Li. Pre-attentive segmentation in the primary visual cortex. *Spatial Vision*, 13(1):25–50, 2000.

[LSGX05] H. Lombaert, Y.Y. Sun, L. Grady, and C.Y. Xu. A multilevel banded graph cuts method for fast image segmentation. In *ICCV*, volume 1, pages 259–265, 2005.

[LSTS04] Y. Li, J. Sun, CK. Tang, and HY. Shum. Lazy Snapping. *ACM Transactions on Graphics*, 23(3):303–308, 2004.

[MAS+07] K. Marten, F. Auer, S. Schmidt, E. J. Rummeny, and C. Engelke. Automated CT volumetry of pulmonary metastases: The effect of a reduced growth threshold and target lesion number on the reliability of therapy response assessment using RECIST criteria. *European Radiology*, 17(10):2561–2571, October 2007.

[MGBA+09] M. F. McNitt-Gray, L. M. Bidaut, S. G. Armato, C. R. Meyer, M. A. Gavrielides, C. Fenimore, G. McLennan, N. Petrick, B. Zhao, A. P. Reeves, R. Beichel, H-J. Kim, and L. Kinnard. Computed tomography assessment of response to therapy: Tumor volume change measurement, truth data, and error. *Translational Oncology*, 2(4):216–222, December 2009.

[MH80] D. Marr and E. Hildreth. Theory of edge detection. In *Proceedings of the Royal Society of London B*, volume 207, pages 187–217, 1980.

[MLJM+09] J. H. Moltz, Bornemann L., Kuhnigk J-M., Dicken V., Peitgen E., S. Meier, H. Bolte, M. Fabel, H. C. Bauknecht, M. Hittinger, A. Kiessling, M. Pusken, and H. O. Peitgen. Advanced segmentation techniques for lung nodules, liver metastases, and enlarged lymph nodes in CT scans. *Selected Topics in Signal Processing*, Volume: 3 , Issue: 1:122–134, 2009.

[MS89] D. Mumford and J. Shah. Optimal approximations of piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42(5):577–685, 1989.

[Mur03] K. Murota. *Discrete convex analysis*. Society for Industrial Mathematics, 2003.

[OS88] S. Osher and J.A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulation. *Journal of Computational Physics*, 79(1):12–49, November 1988.

[Pas01] G. Paschos. Perceptually uniform color spaces for color texture analysis: An empirical evaluation. *IEEE Transactions on Image Processing*, 10(6), June 2001.

[Pav72] T. Pavlidis. Segmentation of pictures and maps through functional approximation. *Computer Graphics and Image Processing*, 1(4):360–372, 1972.

[PM90] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[Pre70] J.M.S. Prewitt. Object enhancement and extraction. In *Picture Processing and Psychopictorics*, pages 75–149, 1970.

[QCS+08] L. E. Quint, J. Cheng, M. Schipper, A. C. Chang, and G. Kalemkerian. Lung lesion doubling times: Values and variability based on method of volume determination. *Clinical Radiology*, 63(1):41–48, January 2008.

[Ram88] S. Ramanujan. *The Lost Notebook And Other Unpublished Papers*. Springer Berlin, 1988.

[RC98] S. Roy and I.J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *ICCV*, pages 492–499, 1998.

[RCD07] F. Ranchin, A. Chambolle, and F. Dibos. Total variation and graph cuts approaches for variational segmentation. In *Proceedings of SSVM*, pages 743–753, June 2007.

[RCM05a] S. Rital, H. Cherifi, and S. Miguet. A segmentation algorithm for noisy images. In *International Conference on Computer Analysis of Images and Patterns*, pages 205–212, September 2005.

[RCM05b] S. Rital, H. Cherifi, and S. Miguet. Weighted adaptive neighborhood hypergraph partitioning for image segmentation. In *International Conference on Advances in Pattern Recognition*, pages 522–531, August 2005.

[RCY+06] A.P. Reeves, A.B. Chan, D.F. Yankelevitz, C.I. Henschke, B. Kressler, and W.J. Kostis. On measuring the change in size of pulmonary nodules. *IEEE Transactions on Medical Imaging*, 25(4), 2006.

[RD02] M. Rousson and R. Deriche. A variational framework for active and adaptative segmentation of vector valued images. In *Proceedings of the Workshop on Motion and Video Computing*, MOTION'02, pages 56–61, 2002.

[RKB04] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[RKLS07] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary mrfs via extended roof duality. In *CVPR*, 2007.

[Rob65] L.G. Roberts. *Machine Perception of Three Dimensional Solids*, pages 159–197. MIT Press, 1965.

[ROF92] L.I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[SD07] J. Stawiaski and E. Decencière. Region merging via graph-cuts. In *ICISS*, 2007.

[SDB07] J. Stawiaski, E. Decencière, and F. Bidault. Computing approximate geodesics and minimal surfaces using watershed and graph-cuts. In *ISMM*, pages 349–360, 2007.

[SG06] A.K. Sinop and L. Grady. Accurate banded graph cut segmentation of thin structures using laplacian pyramids. In *MICCAI*, volume 9, pages 896–903, 2006.

[SK10] P. Strandmark and F. Kahl. Parallel and distributed graph cuts by dual decomposition. In *CVPR*, pages 2085–2092, 2010.

[SM00] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.

[Sob70] I. Sobel. 1970.

[SSPvG06] I. Sluimer, A. Schilham, M. Prokop, and B. van Ginneken. Computer analysis of computed tomography scans of the lung: a survey. *IEEE Transactions on Medical Imaging*, 25(4):385–405, April 2006.

[STJ+10] C. Suzuki, M. R. Torkzad, H. Jacobsson, G. Aström, A. Sundin, T. Hatschek, H. Fujii, and L. Blomqvist. Interobserver and intraobserver variability in the response evaluation of cancer therapy according to RECIST and WHO-criteria. *Acta Oncologica*, 49(4):509–514, May 2010.

[TC04] X-C. Tai and T.F. Chan. A survey on multiple level set methods with applications for identifying piecewise constant functions. *International Journal of Numerical Analysis and Modeling*, 1(1):25–47, 2004.

[Vek06] O. Veksler. Reducing search space for stereo correspondence with graph cuts. In *BMVC*, volume 2, pages 709–718, 2006.

[vKOP+09] R. J. van Klaveren, M. Oudkerk, M. Prokop, E. T. Scholten, K. Nackaerts, R. Vernhout, C. A. van Iersel, K. A. van den Bergh, S. van 't Westeinde, C. van der Aalst, E. Thunnissen, D. M. Xu, Y. Wang, Zhao Y.,

H. A. Gietema, B. J. de Hoop, H. J. Groen, G. H. de Bock, P. van Ooijen, C. Weenink, J. Verschakelen, J. W. Lammers, W. Timens, D. Willebrand, A. Vink, W. Mali, and H. J. de Konin. Management of lung nodules detected by volume CT scanning. *The New England Journal of Medicine*, 361(23):2221–2229, December 2009.

[VKR09] S. Vicente, V. Kolmogorov, and C. Rother. Joint optimization of segmentation and appearance models. In *ICCV*, pages 755–762, 2009.

[vPvMGB10] E. L. van Persijn van Meerten, H. Gelderblom, and J. L. Bloem. RECIST revised: Implications for the radiologist. a review article on the modified RECIST guideline. *European Radiology*, 20(6):1456–1467, June 2010.

[VTC05] G. Vogiatzis, P.H.S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *CVPR*, volume 2, pages 391–398, 2005.

[Wer23] M. Wertheimer. Untersuchungen zur lehre von der gestalt, ii. *Psychological Research*, 4(1):301–350, 1923. Translation published as Laws of Organization in Perceptual Forms, in Ellis, W. (1938).

[YBS09] X. Ye, G. Beddoe, and G. Slabaugh. Graph cut-based automatic segmentation of lung nodules using shape, intensity, and spatial features. In *Workshop on Pulmonary Image Analysis, MICCAI*, 2009.

[ZCP06] Y. Zeng, W. Chen, and Q. Peng. Efficiently solving the piecewise constant mumford-shah model using graph cuts. Technical report, Department of Computer Science, Zhejiang University, P.R. China, 2006.

[ZN10] Q. Zhang and K.N. Ngan. Multi-view video based multiple objects segmentation using graph cut and spatiotemporal projections. *Journal of Visual Communication and Image Representation*, 21(5–6), July 2010.

[ZSM+06] B. Zhao, L. H. Schwartz, C. S. Moskowitz, M. S. Ginsberg, N. A. Rizvi, and M. G. Kris. Lung cancer: Computerized quantification of tumor response–initial results. *Radiology*, 241(3):892–898, December 2006.

Publications

[1] N. Lermé, L. Létocart, and F. Malgouyres. Reduced graphs for min-cut/max-flow approaches in image segmentation. *Electronic Notes in Discrete Mathematics*, 37:63–68, August 2011.

[2] N. Lermé and F. Malgouyres. A reduction method for graph cut optimization. *preprint Hal*, 2011. (submitted).

[3] N. Lermé and F. Malgouyres. Graph cuts-based reduction for simultaneous segmentation and filtering. In *hal-00624093*, 2012. (submitted to CVPR).

[4] N. Lermé, F. Malgouyres, and L. Létocart. Reducing graphs in graph cut segmentation. In *ICIP*, pages 3045–3048, 2010.

[5] N. Lermé, F. Malgouyres, and L. Létocart. Reduction of "vision graphs". Patent No. FR2955408 (A1), January 2010.

[6] N. Lermé, F. Malgouyres, L. Létocart, and J-M. Rocchisani. Une méthode de réduction exacte pour la segmentation par graph cuts. In *ORASIS*, number inria-00596724, 2011.

[7] N. Lermé, F. Malgouyres, and J-M. Rocchisani. Fast and memory efficient segmentation of lung tumors using graph cuts. In *MICCAI – Workshop on Pulmonary Image Analysis*, pages 9–20, 2010.

Abstract

In this thesis, we first present a new band-based strategy for reducing the graphs involved in binary graph cut segmentation. This is done by locally testing if a node is really useful to the maximum flow computation in these graphs. Like previous band-based methods, the remaining nodes are typically located in narrow bands surrounding the object edges to segment. In a first time, we propose an heuristic condition to decide if a node can be added to the reduced graph which can be computed in constant time (except for image borders). When the amount of regularization is large, extra parameters are embedded into this test for both further reducing the graphs and removing segments due to noise in the segmentations. When the amount of regularization is of moderate level, the time required by this algorithm is even compensated by the maximum flow time on the reduced graph. In this situation, we experimentally show that this algorithm drastically reduce the memory usage of standard graph cuts while keeping a low pixel error on segmentations. In a second time, we describe another test with a slightly higher computational cost. We prove that each node satisfying this test can be safely removed without modifying the maximum flow value. Numerical experiments exhibit similar performance than the heuristic test. In a second part, we present an application of this reduction technique devoted to the semi-interactive segmentation of lung tumors in 3D CT images. The novelty of this work is to embed a prior on the object seeds location and control their propagation thanks to a Fast Marching algorithm based on the image gradient. Qualitative and quantitative results against provided ground truths exhibit an accurate delineation of tumors with a Dice coefficient greater than 80% in average.

Keywords: graph cut, segmentation, lung tumor, discrete optimization, reduction.

Résumé

Dans cette thèse, nous présentons d'abord une nouvelle stratégie à base de bandes pour réduire les graphes impliqués dans la segmentation binaire par graph cuts. Ceci est effectué en testant localement si un noeud est réellement utile au calcul du flot maximum dans ces graphes. À l'instar des méthodes antérieures à base de bandes, les noeuds restants sont typiquement localisés dans des bandes étroites autour des contours de l'objet à segmenter. Dans un premier temps, nous proposons un test heuristique pour décider si un noeud peut être ajouté au graphe réduit qui peut être calculée en temps constant (excepté pour les bords de l'image). Lorsque le degré de régularisation est élevé, des paramètres supplémentaires sont intégrés à ce test pour à la fois réduire davantage les graphes et supprimer les zones dues au bruit dans les segmentations. Lorsque le degré de régularisation est moindre, le temps requis par cet algorithme est même compensé par le temps de calcul du flot maximum sur le graphe réduit. Dans cette situation, nous montrons expérimentalement que cet algorithme réduit significativement la consommation mémoire des graph cuts standard tout en conservant une erreur quasi nulle sur les segmentations. Dans un second temps, nous décrivons un autre test avec un coût computationnel légèrement supérieur. Nous démontrons que chaque noeud vérifiant ce test peut être retiré sans altérer la valeur du flot maximum. Des expériences numériques permettent d'exhiber des performances équivalentes au test heuristique. Dans une seconde partie, nous présentons une application de cette technique de réduction à la segmentation semi-interactive de tumeurs pulmonaires dans des images CT 3D. L'originalité de ce travail consiste à intégrer un a priori sur la localisation des graines objet et contrôler leur propagation grâce à un algorithme de Fast Marching basé sur le gradient de l'image. Les résultats quantitatifs et qualitatifs comparés aux vérités terrains fournies montrent une délimitation précise des tumeurs avec un coefficient de Dice supérieur à 80% en moyenne.

Mots-clés : coupe de graphe, segmentation, tumeur pulmonaire, optimisation discrète, réduction.

Discipline : informatique.

Institut Galilée – Université Paris-Nord
99, avenue Jean-Baptiste Clément
93430 Villetaneuse