Graph Cuts-based Reduction For Simultaneous Segmentation And Filtering

Nicolas Lermé, François Malgouyres

Graph Cuts-based Reduction For Simultaneous Segmentation And Filtering

N. Lermé[1,2] F. Malgouyres[3]

(1) LAGA UMR CNRS 7539, (2) LIPN UMR CNRS 7030
Université Paris 13, Avenue J-B. Clément – 93430 Villetaneuse
(3) IMT UMR CNRS 5219
Université Paul Sabatier, 118 route de Narbonne – F-31062 Toulouse Cedex 9

nicolas.lerme@lipn.univ-paris13.fr
francois.malgouyres@math.univ-toulouse.fr

Abstract

Graph cuts have now become a cornerstone in computer vision for efficiently solving a numerous labeling problems. However, the large memory consumption of this method make it hard to solve large-scale problems. Except some exact methods [13, 7, 4], the heuristics present in the literature [9, 10, 12, 5] generally fail to fully capture shape complexities. In this paper, we first review the band-based strategy detailed in [8] for reducing graphs in image segmentation. Unlike [12, 5], the proposed method provide small graphs while accurately preserving thin structures. Next, an extra parameter is introduced for both further reducing graphs and removing small segments in the segmentation. While this parameter is generally less sensitive to variations, it offers a good robustness against noise. Finally, we provide an empirical way to automatically tune it and illustrate its behavior through experiments for segmenting grayscale and color images.

Keywords: graph cuts, reduction, segmentation, filtering.

1. Motivation and scope

Graph cuts have become increasingly popular due to their ability to efficiently compute the Maximum A Posteriori of Markov Random Fields (MRF). This popularity is notably driven by the introduction of a fast maximum-flow (max-flow) algorithm [3] making near real-time performance possible for solving numerous problems.

In parallel, technological advances in image acquisition have both increased the amount and the diversity of data to process. As an illustration, in the satellite SPOT-5 launched by Arianespace in 2002, the high resolution sensors can capture multispectral and panchromatic images with an imaging swath of 60 km $\times$ 60 km. Each image has a size of 12000 $\times$ 12000 which amounts to about 1GB.

Processing this type of data amounts to solve large scale optimization problems. In the image segmentation context, almost all graph cuts methods are impractical to solve such problems due to the memory requirements. To overcome this situation, some amount of work has been recently done in this direction and a number of heuristics [9, 10, 12, 5] and exact methods [7, 4, 13] have been proposed. To our best knowledge, this problem seems to be first addressed in [9] where the underlying graph is built upon a pre-segmentation. Although this approach greatly reduce the computational burden of graph cuts, the results strongly depend on the low-level segmentation algorithm used for computing the pre-segmentation. Also, better results are obtained when over-segmentation occurs, losing in this way the main benefit of such a reduction.

Others have also reported band-based heuristics [10, 12, 5]. The principle is to segment a low-resolution of the image 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. While such an approach drastically reduce time and memory consumption, it is limited to segment roundish objects. The previous problem is notably reduced in [12] but still present for low-contrasted details. In [5], finer bands are obtained using an uncertainty measure associated to each pixel.

Exact methods have been also investigated [7, 4, 13]. In [7], binary energy functions are minimized with graph cuts in a narrow band, while ensuring the optimality on the solution. The principle is to make a band evolve around the object to segment by expanding it when the minimum-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 depends on the initialization and no bound on the band size is given.

A parallel max-flow algorithm yielding a near-linear speedup with the number of processors is also described in [4]. Nevertheless, the algorithm is relatively sensitive to the available amount of physical memory and remains less efficient on small graphs.

In [13], binary energy functions are minimized in a parallelized/distributed fashion using the max-flow algorithm

of [3]. The original problem is decomposed into optimizable sub-problems, solved independently and updated according to the results of the adjacent problems. This process is iterated until convergence and optimality is guaranteed by dual decomposition. Nevertheless, the performance of this approach depends on the object location in the image and remains less effective for dense graphs.

In this paper, we first review the band-based strategy described in [8] for reducing graphs in image segmentation. The graph is built by only adding nodes which locally satisfy a condition. In the manner of [12, 5], the nodes are typically located in a narrow band surrounding the object edges to segment. Empirically, the solutions obtained with reduction are identical to the ones obtained without reduction. Next, an extra parameter is introduced for further reducing the graphs and remove small segments in the segmentation. The key benefit of the proposed method is that it does not require any post-processing steps, unlike traditional filters.

The rest of this paper is organized as follows. First, we review the graph cuts framework in Section 2 and briefly describe our approach for reducing the graphs in Section 3. Finally, the methdology for further reducing graphs and filtering segmentations is explained in Section 4.

2. Notations and preliminaries

Consider an image $I : \mathcal{P} \subset \mathbb{Z}^d \rightarrow [0,1]^c$ $(d > 0, c > 0)$ as a function, mapping each pixel of $p \in \mathcal{P}$ to a pixel $I_p \in [0,1]^c$ [1]. We define a binary segmentation as an application u affecting to each pixel $p \in \mathcal{P}$ either 0 (background) or 1 (object) and we write $u \in \{0,1\}^{\mathcal{P}}$. Then, a popular strategy to segment I is to minimize a MRF of the form [2]:

$$E(u) = \beta \cdot \sum_{p \in \mathcal{P}} E_p(u_p) + \sum_{(p,q) \in \mathcal{N}} E_{p,q}(u_p, u_q), \qquad (1)$$

among $u \in \{0,1\}^{\mathcal{P}}$ and for a fixed $\beta \in \mathbb{R}^+$. The neighborhood system $\mathcal{N} \subset \mathcal{P}^2$ is a subset of all pixel pairs $(p,q) \in \mathcal{P}^2$. In this context, we will use the following standard neighborhoods [2]:

$$\mathcal{N}_0 = \{(p,q) \in \mathcal{P}^2 \ : \ \textstyle\sum_{i=1}^{d} |q_i - p_i| = 1\} \quad \text{or,}$$
$$\mathcal{N}_1 = \{(p,q) \in \mathcal{P}^2 \ : \ |q_i - p_i| \leq 1 \ \forall 1 \leq i \leq d\},$$

where p_i denotes the i^{th} coordinate of p and $|.|$ stands for the absolute value. In (1), the region term $E_p(.)$ is defined as the negative log of the likelihood of a label being assigned to pixel p and is computed from its color and the appearance models of the object and background seeds [3]:

$$\begin{cases} E_p(1) = -log\, \mathbb{P}(I_p | p \in \mathcal{O}) \\ E_p(0) = -log\, \mathbb{P}(I_p | p \in \mathcal{B}) \end{cases}$$

Similarly, the boundary term in (1) is defined as a contrast-sensitive Ising model (see [3] for more details):

$$E_{p,q}(u_p, u_q) = \begin{cases} 0 & \text{if } u_p = u_q, \\ \frac{1}{\|p-q\|_2} exp\Big(- \frac{\|I_p - I_q\|_2^2}{2\sigma^2} \Big) & \text{otherwise,} \end{cases}$$

where $\|.\|$ is the Euclidean norm (either in $\mathbb{R}^d$ or $\mathbb{R}^c$). When the region terms are submodular [6], the minimizer of (1) can be efficiently obtained by computing a minimum-cut in a weighted digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$ with a set of nodes $\mathcal{V} = \mathcal{P} \cup \{s,t\}$, a set of edges $\mathcal{E} \subset \mathcal{V}^2$ and capacities $c : \mathcal{E} \rightarrow \mathbb{R}^+$. The terminal nodes s and t are called the source and the sink, respectively. Moreover, the set of edges $\mathcal{E}$ is split in two disjoint sets $\mathcal{E}_n$ and $\mathcal{E}_t$ denoting respectively n-links (edges linking two nodes of $\mathcal{P}$) and t-links (edges linking a node of $\mathcal{P}$ to s or t). Once the minimum-cut is computed in $\mathcal{G}$, we set $u_p = 1$ if a node p is connected to the source s and $u_p = 0$ if p is connected to the sink t.

3. Reduction

As said earlier, the memory consumption of graph cuts for segmenting high-resolution data is prohibitive. As an illustration, the max-flow algorithm of [3] v2.2 allocates $24\sharp\mathcal{P} + 14\sharp\mathcal{E}_n$ bytes, where the operator '$\sharp$' stands for the cardinality of a set. One can observe that for a fixed amount of RAM, the maximum image size quickly decreases as the dimension d increases. Nevertheless, as shown in [8], most of the nodes in the graph are useless during max-flow computation since they are not traversed by any flow. Ideally, 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 f'^* in $\mathcal{G}'$ identical or very close to the max-flow value f^* in $\mathcal{G}$. In words, we want to minimize the relative size of the reduced graph defined as

$$\rho = 100 \times \frac{\sharp\mathcal{V}'}{\sharp\mathcal{V}}, \qquad (2)$$

under the constraint that $f^* = f'^*$. 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.

First, let us introduce some terminology before describing our method for building $\mathcal{G}'$. In accordance with the graph construction given in [6], we consider (without loss of generality) that a node is connected to at most one terminal:

$$(s,p) \in \mathcal{E}_t \Rightarrow (p,t) \notin \mathcal{E}_t, \qquad p \in \mathcal{P}.$$

We also summarize the t-links capacities for any node $p \in \mathcal{P}$ by:

$$c(p) = c(s,p) - c(p,t).$$

For any $B \subset \mathbb{Z}^d$ (in practice, B will be a square centered at the origin) and $p \in \mathcal{P}$, we denote by $\widetilde{B}_p$ the set translation of B by the point p:

$$\widetilde{B}_p = \{q + p \mid q \in B\}.$$

[1] Usually, $\mathcal{P}$ corresponds to a rectangle.

[2] In what follows, the terms "connectivity 0" and "connectivity 1" respectively refer to the use of $\mathcal{N}_0$ and $\mathcal{N}_1$, respectively.

[3] In this context, the probabilities of the object and the background are estimated using a Gaussian Mixtures Model. The number of gaussians is automatically computed using a Minimum Description Length criteria [1].

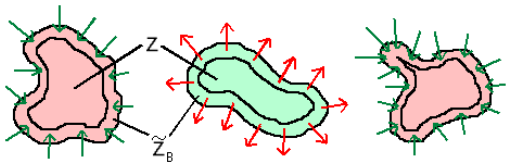


Figure 1: Reduction's principle. Red (resp. green) area and arrows denote the flow that might get in (resp. out of) $\widetilde{Z}_B$. The nodes from Z are removed since (3) holds for any node $p \in Z$. Remaining nodes are located in the band $\widetilde{Z}_B \setminus Z$.

For $Z \subset \mathcal{P}$ and $B \subset \mathbb{Z}^d$, we denote by $\widetilde{Z}_B$ the dilation of Z by the structuring element B as:

$$\widetilde{Z}_B = \{p+q \mid q \in B, p \in Z\} = \bigcup_{p \in Z} \widetilde{B}_p.$$

The intuitive idea for building $\mathcal{G}'$ is to remove from the nodes of $\mathcal{G}$ any $Z \subset \mathcal{P}$ where all nodes are linked to s (resp. to t) and such that all the flow that might get in (resp. out of) the region $\widetilde{Z}_B$ does so by traversing its boundary and can be absorbed (resp. provided) by the band $\widetilde{Z}_B \setminus Z$ (see Figure 1). Building such Z is done by testing each individual pixel $p \in \mathcal{P}$. Thus, in the manner of [12, 5], the remaining nodes are typically located in a narrow band surrounding the object edges to segment. In practice, we proposed in [8] to use an even more conservative test for each node $p \in Z$ in a square window B of size $(2r+1)$ $(r > 0)$ centered in p:

$$\left\{ \begin{array}{ll} \text{either} & \Big(\forall q \in \widetilde{B}_p, c(q) \geq +\delta\Big), \\ \text{or} & \Big(\forall q \in \widetilde{B}_p, c(q) \leq -\delta\Big). \end{array} \right. \quad (3)$$

where $\delta = \frac{P(B)}{(2r+1)^2-1}$. Here, $P(B)$ is the perimeter of B:

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

In words, for any node $p \in Z$ satisfying the first (resp. second) condition of (3), all its neighbors $q \in \widetilde{B}_q$ are only linked to s (resp. t) and the flow that might get in (resp. out) through t-links in $\widetilde{B}_p \setminus \{p\}$ suffices to saturate the n-links going out of (resp. in) $\widetilde{B}_p$. Thus, p becomes useless and can be removed from $\mathcal{G}$. Furthermore, the test (3) can be computed in an incremental way for reaching a complexity of $O(\sharp\mathcal{P})$ (except borders), which is independent of r.

The experiments presentend in [8] confirm the intuitive dependence between the size of $\mathcal{G}'$ and model parameters. Indeed, when minimizing (1) by graph cuts, the t-links capacities are all multiplied by β. Thus, it is straightforward to observe that (3) is harder to satisfy as β decreases. In such a situation, we need a larger window radius for decreasing δ in order to reduce the size of $\mathcal{G}'$. This result in wider bands around the object contours. Conversely, when β is large, we can afford a large δ and therefore a small window radius. Thus, $\mathcal{G}'$ consists of narrow bands around the object contours. Finally, we will prove in a forthcoming paper that the reduction is exact but for a slightly stronger test than (3) [4]. Massive experiments for segmenting multi-dimensional grayscale and color images using different energy models exhibit small reduced graphs, while keeping a low pixel error on the segmentations.

4. Simultaneous segmentation and filtering

Additionally, we have also investigated some ways to relax (3). A simple way to do that is to allow some nodes in $\widetilde{B}_p$ to fail complying the test. The proportion of nodes satisfying (3) is controlled by a parameter $\eta \in [0,1]$. As η decreases, the test (3) can be satisfied more easily since a larger ratio of nodes can be connected to opposite terminals. Embedding η in (3) leads to

$$\left\{ \begin{array}{ll} \text{either} & \Big(\sharp\{q \in \widetilde{B}_p \mid c(q) \geq +\delta\} \geq \eta \cdot \sharp\widetilde{B}_p\Big), \\ \text{or} & \Big(\sharp\{q \in \widetilde{B}_p \mid c(q) \leq -\delta\} \geq \eta \cdot \sharp\widetilde{B}_p\Big). \end{array} \right. \quad (4)$$

For instance, the parameter η can be used for reducing the memory consumption. The Figure 4 illustrates how far the test (3) can be relaxed for further reducing graphs while getting nearly the same segmentation. In this experiment, the segmentation as well as the reduced graph are shown for segmenting a 2D synthetic noisy image. Since the test (4) is easier to satisfy as η decreases, the reduced graph around the object contours becomes thicker.

The parameter η can be also used for filtering the segmentation. This behavior is illustrated in Figure 5 for segmenting a 3D noisy image from a confocal microscope. White spots correspond to cell nuclei in a mouse cerebellum. Observe how far the filtering acts for small values of η: small regions in the reduced graph as well as in the segmentation are progressively removed as η decreases. This parameter is typically useful for filtering images corrupted by a noise behaving like an impulsive noise.

Notice that, when $\mathcal{G}'$ consists of a single connected component, η must be large enough for keeping $\mathcal{G}'$ in a whole piece. Indeed, below some value of η (denoted by η_{min}), $\mathcal{G}'$ is split into multiple pieces and becomes inconsistent since the minimum-cut is no longer fully contained in $\mathcal{G}'$. The Figure 3 illustrates a situation where such value can be computed on a image consisting of two high-contrasted area. Using (4) with a square window of radius r and $\eta = 1$, $\mathcal{G}'$ is a thin band of size $2r$. A lower bound on η is to impose that η_{min} permits to segment both areas. Thus, we want (4) to be false for any node p located on the boundary of these areas. For such a pixel p, we have (e.g. if $c(p) \geq +\delta$):

$$\sharp\{q \in \widetilde{B}_p \mid c(q) \geq +\delta\} = (r+1)(2r+1)^{d-1}.$$

As a consequence, if

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

[4] By exact, we mean that the max-flow value in $\mathcal{G}$ is identical to the max-flow value in $\mathcal{G}'$.

the pixel p does not belong to $\mathcal{G}'$. In order to avoid this situation, we set:

$$\begin{aligned} \eta_{min} &= \frac{(r+1)(2r+1)^{d-1}}{(2r+1)^d}, \\ &= 1 - \frac{r}{2r+1}. \end{aligned} \tag{5}$$

Thus, as r increases, the maximum proportion of nodes allowed as being linked to opposite terminals tends to 50%. Notice that this lower bound is not accurate in connectivity 0. Indeed, since diagonal directions are not allowed in $\mathcal{G}'$ with such connectivity type, the reduced graph easily gets disconnected into multiple pieces (see Figure 3). Again, the minimum-cut is no longer ensured to be fully contained in the reduced graph.

Given the amount of noise in the image, one can also derive an upper bound on η by finding the maximum value of η for which (2) is minimum, without regards to the image and the model. Consider that the image is corrupted by a salt-and-pepper noise generated by a Bernoulli distribution of parameter ξ. The amount of noise corresponds to the probability that a pixel p is corrupted, denoted by ξ. Let X be a discrete random variable counting such pixels in a square window B of size $n = (2r+1)^d$. Then, the probability that at least k pixels are corrupted in B is

$$\mathbb{P}(X \geq k) = \sum_{i=k}^{n} \binom{n}{i} \xi^i (1-\xi)^{n-i}. \tag{6}$$

If we impose that $\xi \in]0,1[$, for a fixed window radius r and η, it is straightforward to see that (6) is decreasing and tends to ξ^n. Assuming that all the nodes corresponding to noise-free pixels are linked to the same terminals, for a fixed $\kappa \in [0,1]$ and ξ in (6), (4) holds for any node $p \in \mathcal{P}$ with

$$\eta \leq 1 - \kappa. \tag{7}$$

Although κ must be minimized for getting a larger value of η, we also need to minimize $\mathbb{P}(X \geq k)$. Otherwise, (4) can fail for some nodes when the value of η is too large. Therefore, there is a trade-off between the two situations and the optimal value of κ can be computed as

$$\kappa^* = \min\{\kappa, \mathbb{P}(X \geq \lceil \kappa \sharp B \rceil) < \epsilon\}, \tag{8}$$

for some $\epsilon \simeq 0$. From (8), η_{max} is set as

$$\eta_{max} = \max\{\eta_{min}, 1 - \kappa^*\}. \tag{9}$$

Since (6) is decreasing for a fixed r and η, the upper bound η_{max} can be quickly computed using a dichotomic search of complexity $O(d\, log_2(2r+1))$. When ξ is fixed, one can observe that the gap $\Delta\eta = (\eta_{max} - \eta_{min})$ grows as r increases. Indeed, η_{min} tends to 0.5 as r increases (see (5)). As opposite, η_{max} grows as r increases since it is more likely to .... Similarly, when r is fixed, remark that $\Delta\eta$ decreases when ξ increases. Indeed, η_{min} remains the same but η_{max} tends to 0.5 since it is more likely that the number of noisy pixels increase in the same window.

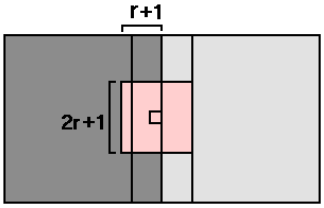


Figure 2: Minimalist example for computing η_{min}.

Additionally, one can also jointly estimate a couple of values r^* and η^* for minimizing (2). If the number of pixels which can be corrupted by ξ is less than η_{min}, it means that we have $\kappa^* \leq \eta_{min}$. Therefore, we should increase r for both decreasing κ^* and η_{min}. Then, it suffices to progressively increase r from 1, while $\kappa^* > \eta_{min}$. Once r^* is computed, η^* can be for instance set as $(\eta_{min} + \eta_{max})/2$. With the previous algorithm, r^* is very near to the experimental value obtained for segmenting a synthetic noisy image. However, one can clearly observe that r^* increases quickly as p tends to 0.5. Therefore, the previous algorithm should be stopped once a maximum number of iterations is reached for instance.

The robustness to noise (see Figure 7 and 6) and the sensitivity of the parameter η (see Figure 8) have been also analyzed. Let us describe the experimental procedure for segmenting four grayscale and five color 2D images with an increasing noise level ranging from 4 to 48%. For each image, we compute a reference segmentation on the noise-free image by tuning the seeds and parameters by hand. The σ parameter is automatically estimated as in [11]. Then, for each impulsive noise level, we select the segmentation maximizing the Dice Similarity Coefficient (DSC) between the reference image and all segmentations obtained through a fixed range of r and η values. Each segmentation is computed using the same seeds and parameters as for the reference segmentation. Again, the σ parameter is automatically estimated using the same method.

As shown in Figure 7, 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 only be reached by increasing the amount of seeds (see Figure 7). The reason why the algorithm behaves poorly on the "rice" image is the following. As said earlier, r must be large enough when ξ increases for removing a maximum number of segments to noise. This implies wider bands in $\mathcal{G}'$ around the object contours. However, the object contours further oscillate as ξ increases since the uncertainty grows inside the band due to noise. Finally, the Figure 8 also illustrates 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". The latter problem can be explained for the same reason as before.

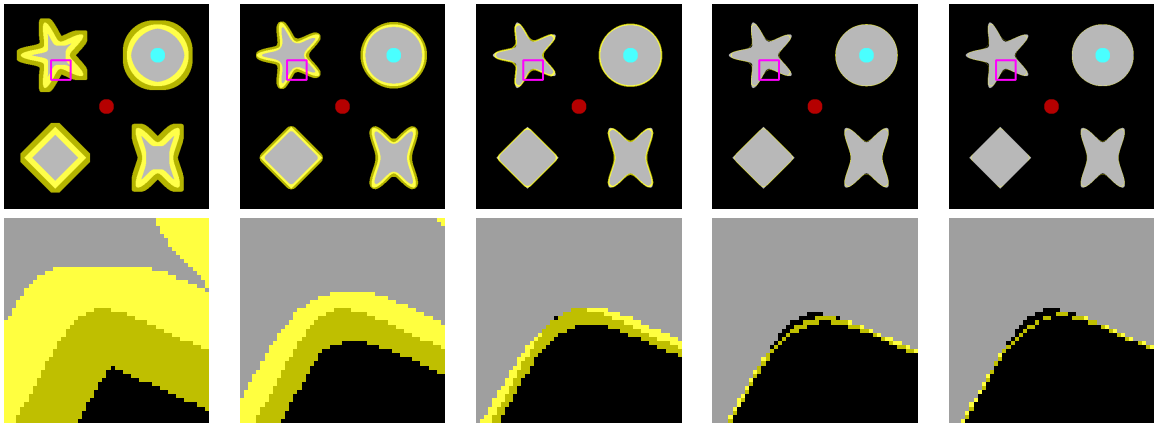
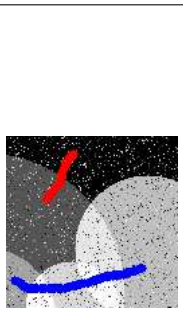
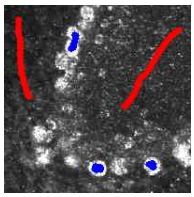


Figure 3: Illustration of the lower bound η_{min} for segmenting a 2D synthetic image. In this experiment, $\eta_{min} \simeq 0.523$ and we set $r = 10$ using connectivity 1. On all images, the pixels belonging to $\mathcal{G}'$ are superimposed in yellow to the original image by transparency. The bottom row correspond to a close-up of the box in purple color. Observe how the reduced graph split into multiple pieces as soon as $\eta \leq \eta_{min}$.



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

Figure 4: Memory gain when segmenting a 2D synthetic image corrupted by 10% of impulsive noise (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 $r = 3$ and use connectivity 1.



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

Figure 5: Simultaneous segmentation and filtering of a 3D noisy image (left). In this picture, the 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 use connectivity 1.

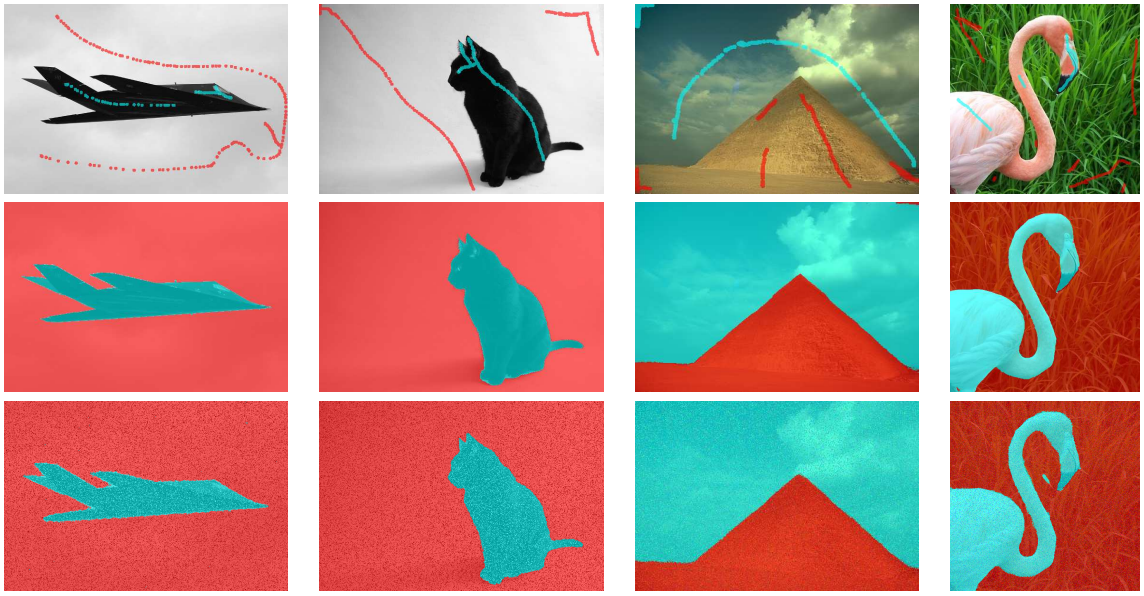


Figure 6: Qualitative analysis of the robustness to noise for segmenting the images "f117" (left-most column), "black-cat" (left column), "pyramid" (right column) and "flamingo" (right-most column) with a fixed impulsive noise level of 36%. The seeds and the model parameters are the same than those used in Figure 7 (top row).

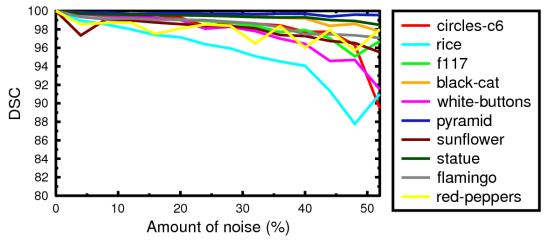


Figure 7: Quantitative analysis of the robustness to noise for segmenting four 2D grayscale images (top-most curves in the list) and five 2D color images with an impulsive noise level ranging from 4 to 48%.

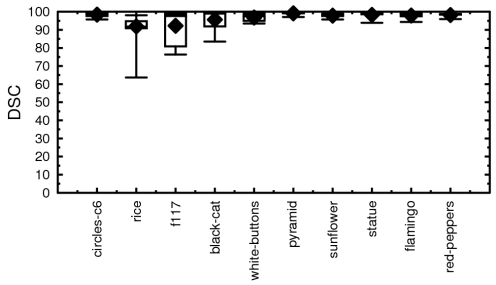


Figure 8: Sensitivity of η for segmenting the images in Figure 7 with an impulsive noise level of 36%. The seeds and model parameters are the same than those used in Figure 7.

References

[1] C. A. Bouman, Cluster: An unsupervised algorithm for modeling Gaussian mixtures, April 1997.

[2] 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, pp. 105–112, 2001.

[3] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision", *IEEE Transactions on PAMI*, 26(9), 2004, pp. 1124–1137.

[4] A. Delong and Y. Boykov, "A scalable graph-cut algorithm for N-D grids", In *CVPR*, pp. 1–8, 2008.

[5] P. Kohli, V. Lempitsky, and C. Rother, "Uncertainty driven multi-scale energy optimization", In *DAGM*, pp. 242–251, 2010.

[6] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?", *IEEE Transactions on PAMI*, 26(2), 2004, pp. 147–159.

[7] V. Lempitsky and Y. Boykov, "Global optimization for shape fitting", In *CVPR*, pp. 1–8, 2007.

[8] N. Lermé, F. Malgouyres, and L. Létocart, "Reducing graphs in graph cut segmentation", In *ICIP*, pp. 3045–3048, 2010.

[9] Y. Li, J. Sun, C. Tang, and H. Shum, "Lazy Snapping", *ACM Transactions on Graphics*, 23(3), 2004, pp. 303–308.

[10] H. Lombaert, Y. Sun, L. Grady, and C. Xu, "A multilevel banded graph cuts method for fast image segmentation", In *ICCV*, volume 1, pp. 259–265, 2005.

[11] C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut": Interactive foreground extraction using iterated graph cuts", In *SIGGRAPH*, pp. 309–314, 2004.

[12] A. Sinop and L. Grady, "Accurate banded graph cut segmentation of thin structures using laplacian pyramids", In *MICCAI*, volume 9, pp. 896–903, 2006.

[13] P. Strandmark and F. Kahl, "Parallel and distributed graph cuts by dual decomposition", In *CVPR*, pp. 2085–2092, 2010.