Reducing Graphs in Graph Cut Segmentation

Nicolas Lermé, François Malgouyres, Lucas Létocart

{nicolas.lerme, lucas.letocart}@lipn.univ-paris13.fr, malgouy@math.univ-paris13.fr

Contribution

Although graph cuts efficiently compute global optimal solutions in segmentation, their huge memory consumption remain a challenging problem since underlying graphs contain billions of nodes and even more edges. To get round the problem of previous heuristics [2, 3], we propose a new strategy for reducing exactly graphs [1]: we progressively build the graph by only adding nodes which satisfy a condition in a small window. Thus, the nodes are typically located in a narrow band around the object edges to segment.

Method

Consider a s-t graph $\mathcal{G} = (\mathcal{P} \cup \{s,t\}, \mathcal{E}, c)$ where the set of grid nodes is $\mathcal{P} \subset \mathbb{Z}^d$ $(d > 0)$ and $\mathcal{N}(p)$ denotes the neighborhood of p, $\forall p \in \mathcal{P}$ (see Figure 1).

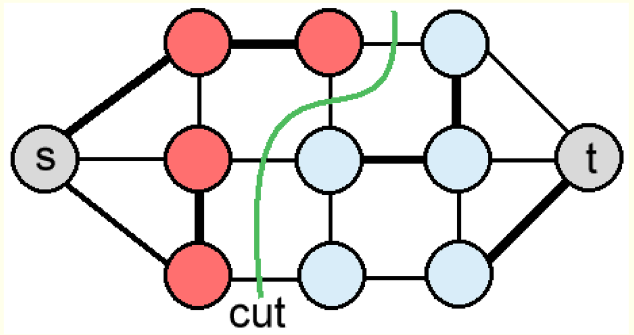


Fig. 1: A simple 3×3 s-t graph cut.

Let $B \subset \mathbb{Z}^d$ and denote by $\widetilde{Z}_B$ the dilation of Z by B. Then, we propose to remove the nodes from $\mathcal{G}$ any $Z \subset \mathcal{P}$ such that either

$$\begin{cases} O(\widetilde{Z}_B) = +|\widetilde{Z}_B| \text{ and } A(\widetilde{Z}_B \setminus Z) \geq P_{out}(\widetilde{Z}_B), \text{ or} \\ O(\widetilde{Z}_B) = -|\widetilde{Z}_B| \text{ and } A(\widetilde{Z}_B \setminus Z) \geq P_{in}(\widetilde{Z}_B), \end{cases} \quad (1)$$

with

$A(.)$	Maximum amount of flow passing Z only through t-links,
$P_{in}(.)$	Maximum amount of flow which come in Z only through n-links,
$P_{out}(.)$	Maximum amount of flow which come out of only Z through n-links,
$O(.)$	Flow orientation (can be positive, negative, or equal to zero).

For instance, the last part of condition (1) implies that all the flow that might come in the region $\widetilde{Z}_B$ comes from its boundary and can be absorbed by the band $\widetilde{Z}_B \setminus Z$ (see Figure 2).

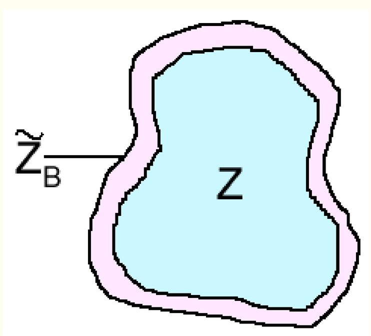


Fig 2.: Reduction's principle. The nodes from Z are removed since the set Z satisfy (1). Remaining nodes are typically located in $\widetilde{Z}_B \setminus Z$.

General algorithm

When B correspond to a square window of size $(2r+1)$ $(r > 0)$ centered at the origin, we adopt in practice a more conservative test for $p \in Z$:

$$\begin{cases} c(q) \geq +\delta \cdot \gamma & \forall q \in \widetilde{B}_p \quad \text{or} \\ c(q) \leq -\delta \cdot \gamma & \forall q \in \widetilde{B}_p, \end{cases} \quad (2)$$

where the contracted capacity is $c(p) = c(s,p) - c(p,t)$, $\gamma \in [0,1]$ and $\delta = \frac{P(B)}{(2r+1)^2 - 1}$, with

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

Both theoretical and empirical evidence suggest that this reduction scheme provides an exact solution when $\gamma = 1$ and becomes heuristic as γ decreases to zero.

Moreover, the condition (2) leads to the algorithm described in Figure 3. It has a straightforward implementation with a worst-case complexity of $O(|B|)$ but remains computationnally expensive for large window radii. Decomposing the condition (2) along the volume's dimensions d yields an algorithm of complexity $O(1)$ (except for borders).

```
δ ← compute_delta()
for p ∈ P do
    sum ← 0
    for q ∈ (B̃_p ∩ P) do
        if c(q) ≥ +δ·γ then
            sum ← sum + 1
        if c(q) ≤ −δ·γ then
            sum ← sum − 1
    if |sum| ≠ card(B̃_p ∩ P) then
        1 - We add p to the graph G.
        2 - We link p to its neighbors in G.
```

Fig. 3: Reduction algorithm.

Influence of parameters

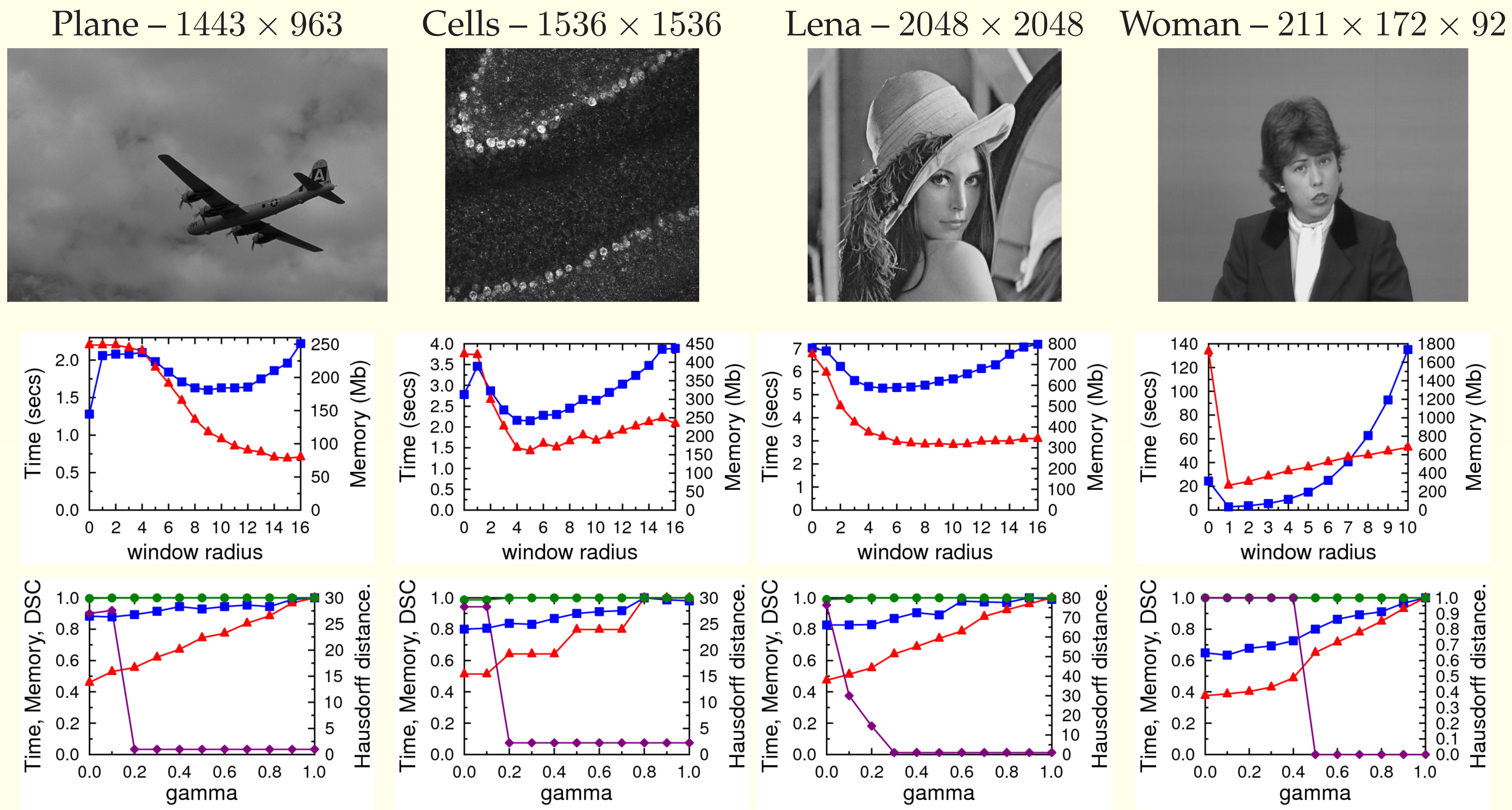
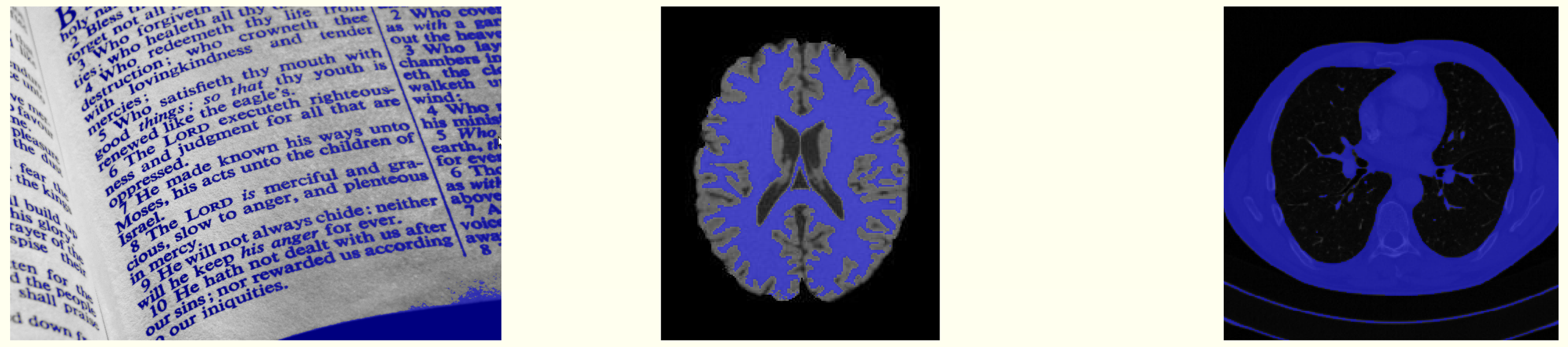


Fig. 4: influence of window radius (middle row) and γ (bottom row) for segmenting 2D and 3D images (top tow) with a $TV + L^2$ model. Standard graph cuts correspond to a window radius equal to zero.

Legend: **Blue:** time execution **Red:** memory consumption.
Green: Dice coefficient between normal and γ-parametrized segmentations.
Purple: Hausdorff distance between normal and γ-parametrized segmentations.

Numerical experiments



Book – 3012×2048 Brain – $181 \times 217 \times 181$ CT-thorax – $245 \times 245 \times 151$

	Standard graph cuts		Our method	
Image	Time	Memory	Time	Memory
Book	5.58	1.08 Gb	3.25	231.25 Mb
Brain	/	3.59 Gb	9.02	734.64 Mb
CT-thorax	/	4.58 Gb	8.25	606.27 Mb

Fig. 5: speed (secs) and memory usage for segmenting 2D/3D images (top) with a Boykov/Jolly's energy model. In this experiment, we set $r = 1$. Segmentations are superimposed in light blue.

References

[1] N. Lermé, F. Malgouyres, and L. Létocart. Reducing graphs in graph cut segmentation. In *CANUM, Carcans-Maubuisson*, 2010.

[2] Y. Li, J. Sun, CK. Tang, and HY. Shum. Lazy Snapping. *ACM Transactions on Graphics*, 23(3):303–308, 2004.

[3] 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.