

# Reducing graphs for graph cut segmentation

Nicolas Lermé<sup>1,2</sup>, François Malgouyres<sup>1</sup>, Lucas Létocart<sup>2</sup>

<sup>1</sup>LAGA, <sup>2</sup>LIPN – Université Paris 13

CANUM 2010, Carcans-Maubuisson, France

June 1, 2010

# Summary

- 1 Introduction
  - Notations
  - General problem
- 2 Background
  - Graph cuts framework
  - Energy models for segmentation
  - Conclusion on graph cuts
- 3 Reducing graphs
  - State of the art
  - Proposed method
  - Numerical results
- 4 Conclusion

# Notations

## Images

An N-D image is defined by a pair  $(\mathcal{P}, I)$  consisting of a finite discrete set  $\mathcal{P} \subset \mathbb{Z}^d$  ( $d > 0$ ) of N-D points (pixels in  $\mathbb{Z}^2$ , voxels in  $\mathbb{Z}^3$ , etc.) and a function  $I$ :

$$I: \begin{array}{l|l} \mathcal{P} & \longrightarrow \mathcal{L} \\ \hline p & \longmapsto u_p, \end{array}$$

where  $\mathcal{L} = \{l_1, \dots, l_k\}$  is a finite and ordered set of labels.

## Neighborhoods

$\mathcal{N}(p)$  will denote the neighborhood of any point  $p \in \mathcal{P}$ .

$$\begin{aligned} \mathcal{N}_0(p) &= \{q : \sum_{i=1}^d |q_i - p_i| = 1\} & \forall p \in \mathcal{P}, \\ \mathcal{N}_1(p) &= \{q : |q_i - p_i| \leq 1 \forall 1 \leq i \leq d\} & \forall p \in \mathcal{P}, \end{aligned}$$

where  $p_i$  denote the  $i^{\text{th}}$  coordinate of the point  $p$ .

## Level sets

$$u^\mu = \{u_p^\mu \mid p \in \mathcal{P}, u_p^\mu = 1\}, \quad u_p^\mu = \mathbf{1}_{\{u_p \geq \mu\}}.$$

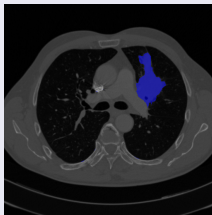
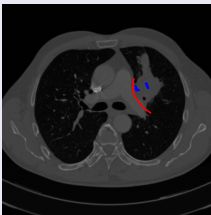
# General problem I

## Objective

Partition an image in disjoint homogeneous regions according to some criteria.

## Applications

### Medical diagnosis



### Photo edition



# General problem II

## Approach by energy minimization

A segmentation is represented by a binary image  $u \in \{0, 1\}^{\mathcal{P}}$ . Then, we select  $u^* \in \operatorname{argmin}_u E(u)$ , for

$$E(u) = \beta \sum_{p \in \mathcal{P}} E_p(u_p) + \sum_{\substack{p, q \in \mathcal{P}^2 \\ q \in \mathcal{N}(p)}} E_{p,q}(u_p, u_q), \quad (1)$$

where

- $E_p$  and  $E_{p,q}$  depend on inputs (image and user interaction),
- $\beta \in \mathbb{R}^+$  is a parameter.

## Remarks

- Minimizing (1) is NP-hard.
- Resolution by max-flow/min-cut in polynomial time, for some energy classes.

# Graph cuts framework I

## Description and history

- Global optimization method based on max-flow/min-cut computations in graphs.
- Initially introduced by Greig *et al.* for binary image restoration [GPS89].
- Recently rediscovered with the arrival of a new max-flow algorithm [BK04].
- Good heuristics for solving multi-labels problems.

## Graph cuts framework II

## Image graphs

Given an image, we build a directed weighted graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$  :

- **Nodes** :  $\mathcal{V} = \mathcal{P} \cup \{s, t\}$  ( $s$ =source,  $t$ =sink),
- **Edges** :  $\mathcal{E} = \mathcal{E}_n \cup \mathcal{E}_t \subset \mathcal{V}^2$  (n-links and t-links),
- **Edge capacities** :  $c : \mathcal{E} \rightarrow \mathbb{R}^+$ ,
- **Flow** :  $f : \mathcal{V}^2 \rightarrow \mathbb{R}^+$ .

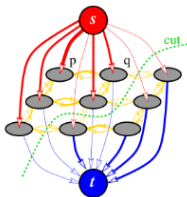


Figure 1: A simple  $3 \times 3$  grid graph.

## Graph cuts framework III

## Remark 2.1 (s-t cut = segmentation)

A cut  $\mathcal{C} = (\mathcal{S}, \mathcal{T})$  is a partition of  $\mathcal{V}$  (i.e.  $\mathcal{S} \cup \mathcal{T} = \mathcal{V}$  and  $\mathcal{S} \cap \mathcal{T} = \emptyset$ ) such that:

$$s \in \mathcal{S}, t \in \mathcal{T}$$

The mapping  $\mathcal{C} \mapsto u^{\mathcal{C}} \in \{0, 1\}^{\mathcal{P}}$ , such that  $u_p^{\mathcal{C}} = \begin{cases} 1 & \text{if } p \in \mathcal{S} \\ 0 & \text{if } p \in \mathcal{T} \end{cases}$  makes a **one to one correspondance** between cuts and segmentations.

## Definition 2.2 (capacity of a cut)

The **capacity of a cut**  $\mathcal{C}$  is defined by  $v(\mathcal{C}) = \sum_{\substack{i \in \mathcal{S}, j \in \mathcal{T} \\ (i,j) \in \mathcal{E}}} c(i,j)$ .

## Idea

Build capacities  $c(\cdot)$  such that  $v(\mathcal{C}) = E(u^{\mathcal{C}})$ .

Hence, the minimum cut in  $\mathcal{G}$  is a minimizer of  $E$  (see [KZ04]).

Use a max-flow algorithm to compute a minimizer of  $E$ .



## Graph cuts framework IV

## Definition 2.3 (Maximum-flow problem)

A flow  $f$  in  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$  is valid iff:

$$\text{Flow bounds :} \quad 0 \leq f(i, j) \leq c(i, j) \quad \forall (i, j) \in \mathcal{E} \quad (2)$$

$$\text{Flow conservation :} \quad \underbrace{\sum_{(j,i) \in \mathcal{E}} f(j, i)}_{f_{in}(i)} = \underbrace{\sum_{(i,j) \in \mathcal{E}} f(i, j)}_{f_{out}(i)} \quad \forall i \in \mathcal{P} \setminus \{s, t\} \quad (3)$$

Then, the maximum-flow problem can be formulated as:

$$\max_f \sum_{(s,i) \in \mathcal{E}} f(s, i), \text{ subject to (2) and (3).}$$

## Theorem 2.4 (Ford-Fulkerson [FF62])

Let  $\mathcal{G}$  be a weighted directed graph. The max-flow in  $\mathcal{G}$  is the smallest capacity of any cuts dividing  $\mathcal{V}$  into disjoint sets  $\mathcal{S}$  and  $\mathcal{T}$ . Then  $f^* = v(C^*)$  and  $C^*$  is deduced from  $f^*$ .

# Graph cuts framework V

## Theorem 2.5

The maximum-flow (i.e. minimum-cut) problem is in **P**.

## Max-flow/min-cut algorithms

- **Augmenting paths**

Ford-Fulkerson	→	$O(mf^*)$
Edmons-Karp	→	$O(nm^2)$
Dinic	→	$O(n^2m)$
Boykov-Kolmogorov	→	$O(n^2mf^*)$ [BK04]

- **Push-relabel algorithms**

Generic push-relabel algorithm	→	$O(n^2m)$
Push-relabel with dynamic trees	→	$O(nm \log(n))$

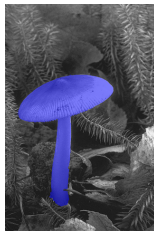
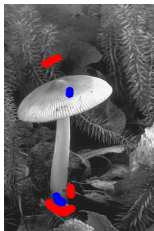
## For grid graphs

Experimental results show a complexity of  $O(n)$  (see [BK04]).

# Boykov/Jolly's energy model [BJ01] I

## Semi-automatic

- **Two kinds of seeds :**
  - Object seeds ( $\mathcal{O} \subset \mathcal{P}$ )
  - Background seeds ( $\mathcal{B} \subset \mathcal{P}$ )
- **Double role :**
  - Reduction of the feasible cuts space.
  - Computation of a fixed color model with normalized histograms.



## Boykov/Jolly's energy model [BJ01] II

## Definition

Let  $v$  be an image. The segmentation of  $v$  can be obtained by minimizing the following energy ([BJ01])

$$E(u) = \beta \cdot \underbrace{\sum_{p \in \mathcal{P}} E_p(u_p)}_{\text{Region term}} + \underbrace{\sum_{\substack{p, q \in \mathcal{P} \\ q \in \mathcal{N}(p)}} E_{p,q}(u_p, u_q)}_{\text{Boundary term}}, \quad \beta \in \mathbb{R}^+.$$

$$\begin{cases} E_p(1) = -\log \Pr(v_p | p \in \mathcal{O}) \\ E_p(0) = -\log \Pr(v_p | p \in \mathcal{B}) \end{cases} \quad \text{and} \quad E_{p,q}(u_p, u_q) = B_{p,q} \cdot |u_p - u_q|$$

where :  $B_{p,q} = \exp\left(-\frac{(v_p - v_q)^2}{2\sigma^2}\right) \cdot \frac{1}{\text{dist}(p,q)}$  where  $\text{dist}(\cdot)$  is the euclidian distance.

# Total Variation based energy models I

## Definition of $TV + L^\alpha$ models

Let  $v$  be an image. We take a level-set of a minimizer of:

$$E_\alpha(u, v) = \underbrace{\sum_{\substack{p, q \in \mathcal{P}^2 \\ q \in \mathcal{N}(p)}} w_{pq} |u_p - u_q|}_{\sim TV(u)} + \beta \cdot \|u - v\|_{L^\alpha}^\alpha, \quad \alpha \in \{1, 2\}.$$

→ Such models were successfully used in image restoration [ROF92] and video segmentation [RCD07].

# Conclusion on graph cuts I

## Conclusion on graph cuts

- (+) Low running times.
- (+) Flexible and interactive models.
- (+) Easily extensible to higher dimensions.
- (+) Yield optimal solutions for a wide range of problems.
- (-) Still too slow for very large data (3D, 4D, etc.)
- (-) Prohibitive memory usage: algorithm [BK04] allocates  $24|\mathcal{P}| + 14|\mathcal{E}_n|$  bytes.

	Connectivity 0	Connectivity 1
2D	6426	4459
3D	319	219
4D	68	45

Table 1: Maximum size of an image for which the graph fits in 2GB of RAM.

## State of the art I

## Banded graph cuts [LSGX05, SG06]

- **Kind of method** : heuristic, multi-resolution scheme.
- **Benefit** :  $\simeq 8x$  faster,  $\simeq 4x$  less memory usage (2D).
- **Drawback** : fail to recover thin structures and details.

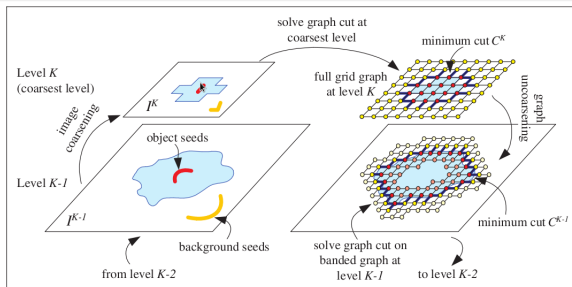


Figure 2: General working of the algorithm described in [LSGX05].

## State of the art II

## Region adjacency graphs [LSTS04, CA08]

- **Kind of method** : heuristic, coarse graphs.
- **Principle** :
  - 1 Computes a pre-segmentation  $S_0(v)$  with a low-level segmentation algorithm.
  - 2 Builds a region adjacency graph  $\mathcal{G}$  from  $S_0(v)$ .
  - 3 Computes the minimum-cut on  $\mathcal{G}$  and get the final solution  $S_{final}(v)$ .
- **Benefit** :  $\simeq 6x$  faster.
- **Drawbacks** : performances are better when over-segmentation occurs, not robust to noise.

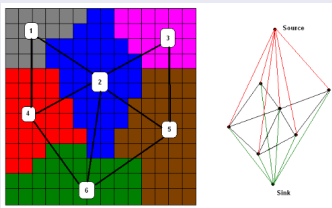


Figure 3: Region adjacency graph (left) and corresponding  $s - t$  graph (right).



## Experiment I

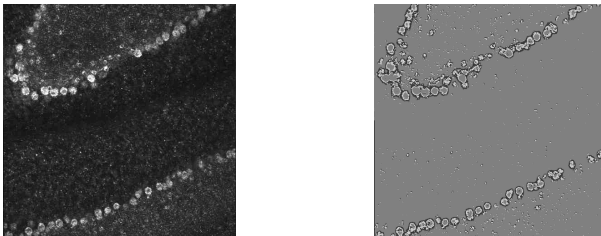


Figure 4: Left: Image of cells. Right: Flow passing through t-links (gray :  $f = 0$ ; white :  $f > 0$ ; black :  $f < 0$ ).

## Remarks

- Only few nodes are used during the max-flow computation.
- **Goal** : one would like to extract the smallest graph  $\mathcal{G}'$  from  $\mathcal{G}$  while keeping a solution  $u'$  identical (or very close) to  $u$ . Ideally, we want to maximize the reduction rate  $\rho = 1 - \frac{|\mathcal{V}'|}{|\mathcal{V}|}$  such that  $u' \simeq u$ .

## Preliminaries I

## Definitions

The graph is such that (see [KZ04]):

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

For any  $p \in \mathcal{P}$  we denote:

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

→ A node  $p$  is linked to  $s$  if  $c(p) > 0$  else  $p$  is linked to  $t$ .

Next, for any  $B \subset \mathbb{Z}^d$  and  $p \in \mathcal{P}$ , we define:

$$\tilde{B}_p = \{p + b \mid b \in B\}$$

For  $Z \subset \mathcal{P}$ , and  $B \subset \mathbb{Z}^d$ , we define the dilation of  $Z$  by  $B$  as:

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

## Preliminaries II

## Definitions

For any  $Z \subset \mathcal{P}$ , we define the maximum amount of flow coming in and out through the n-links by:

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

The maximum amount of flow passing through the t-links and the flow orientation is defined by:

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

where the  $\text{sign}(\cdot)$  function is defined as:

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

Building  $\mathcal{G}'$ 

## Intuitive idea [LML10a, LML10b]

Let  $B \subset \mathbb{Z}^d$ . We remove  $Z$  from the nodes of  $\mathcal{G}$  under the condition :

$$\begin{cases} O(\tilde{Z}_B) = +|\tilde{Z}_B| & \text{and } A(\tilde{Z}_B \setminus Z) \geq P_{out}(\tilde{Z}_B), \text{ or} \\ O(\tilde{Z}_B) = -|\tilde{Z}_B| & \text{and } A(\tilde{Z}_B \setminus Z) \geq P_{in}(\tilde{Z}_B). \end{cases} \quad (4)$$

- Building such a set  $Z$  is done by testing each pixel  $z$  of  $Z$ .
- The conjunction of (4) for any  $z \in Z$  implies (4) for  $Z$ .

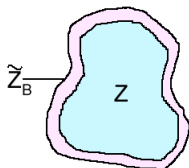


Figure 5: Illustration of condition (4).

Building  $\mathcal{G}'$  II

## A more conservative test

Consider now a square window  $B$  of size  $(2r + 1)$  ( $r > 0$ ) centered at the origin. Then, we propose a more conservative test of (4) for any  $p \in Z$ :

$$\begin{cases} c(q) \geq +\delta \cdot \gamma & \forall q \in \tilde{B}_p \\ c(q) \leq -\delta \cdot \gamma & \forall q \in \tilde{B}_p, \end{cases} \quad \text{or} \quad (5)$$

where  $\gamma \in [0, 1]$  is a parameter and  $\delta = \frac{P(B)}{(2r+1)^2-1}$ , with:

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

$$\begin{aligned} A(\tilde{B}_p \setminus \{p\}) &= \sum_{q \in \tilde{B}_p \setminus \{p\}} |c(p)| \\ &\geq \gamma \cdot [(2r + 1)^2 - 1] \cdot \delta \\ &\geq \gamma \cdot P(B) \\ &\geq \gamma \cdot P_{out}(\tilde{B}_p) \quad (\text{since } 1 \geq c_{p,q}) \end{aligned}$$

Building  $\mathcal{G}'$  III

Remind :  $\delta = \frac{P(B)}{(2r+1)^2 - 1}$  and  $c(q) \geq \delta \cdot \gamma \quad \forall q \in \tilde{B}_p$

## Remarks

- Window radius  $r$  small  $\Rightarrow \delta$  large and few tests required.
- Window radius  $r$  large  $\Rightarrow \delta$  small and a lot of tests required.

# Condition applied to energy models

## TV-based energy models

- $TV + L^1$  model :  $\forall p \in \mathcal{P}, |c(p)| \geq \delta \cdot \gamma \Leftrightarrow \beta \geq \delta \cdot \gamma$
- $TV + L^2$  model :  $\forall p \in \mathcal{P}, |c(p)| \geq \delta \cdot \gamma \Leftrightarrow \beta \cdot |v_p - \mu + \frac{1}{2}| \geq \frac{\delta \cdot \gamma}{2}$

## Boykov/Jolly's energy model [BK04]

$$\forall p \in \mathcal{P}, |c(p)| \geq \delta \cdot \gamma \Leftrightarrow \beta \cdot \left| \log \left( \frac{\Pr(v_p | p \in \mathcal{O})}{\Pr(v_p | p \in \mathcal{B})} \right) \right| \geq \delta \cdot \gamma$$

## Remark

- $\beta$  small (strong regularization)  $\Rightarrow$  we need  $\delta$  small  $\Rightarrow$  we need  $r$  large  $\Rightarrow$  we need wide bands
- $\beta$  large (small regularization)  $\Rightarrow$  we can afford  $\delta$  large  $\Rightarrow$  we can afford  $r$  small  $\Rightarrow$  we can afford narrow bands

# Algorithmic considerations I

---

**Algorithm 1** algorithm for computing  $\mathcal{G}'$

---

**INPUTS:** image  $v$ , square window  $B$  of size  $(2r + 1)$

**OUTPUTS:** reduced graph  $\mathcal{G}'$ .

```

1:  $\mathcal{G}' \leftarrow \text{allocateGraph}()$ 
2: forall  $p \in \mathcal{P}$  do
3:    $\text{deltaTestsSum} \leftarrow 0$ 
4:   forall  $q \in B$  do
5:      $\text{computeDelta}()$ 
6:     if  $c(p) \geq +\delta \cdot \gamma$  then
7:        $\text{deltaTestsSum} \leftarrow \text{deltaTestsSum} + 1$ 
8:     end-if
9:     if  $c(p) \leq -\delta \cdot \gamma$  then
10:       $\text{deltaTestsSum} \leftarrow \text{deltaTestsSum} - 1$ 
11:    end-if
12:  end-for
13:  if  $|\text{deltaTestsSum}| \neq \text{card}(B)$  then
14:    % We add node  $p$  to  $\mathcal{G}'$  and link it with its neighbors
15:  end-if
16: end-for

```

---



# Algorithmic considerations II

## Worst-case complexity

- Special case of convolution algorithm with a separable kernel  $\Rightarrow O(|\mathcal{P}| \cdot |B|)$ .
- Decomposing the test along dimensions  $d$  yields an optimized algorithm in  $O(|\mathcal{P}|)$  independent of  $r$ , except for image borders.

## Optimized algorithm (1)

Consider a square window  $B$  of size  $(2r + 1)$ . For any point  $p \in \mathcal{P}$ , we let:

$$g_{\delta}(p) = \begin{cases} 1 & \text{if } c(q) \geq +\delta \cdot \gamma \quad \forall q \in \tilde{B}_p, \\ -1 & \text{if } c(q) \leq -\delta \cdot \gamma \quad \forall q \in \tilde{B}_p, \\ 0 & \text{otherwise.} \end{cases}$$

$g_{\delta}(i, j)$  will denote the value of  $g_{\delta}(\cdot)$  in 2D for a point  $(i, j)$ .

# Algorithmic considerations III

## Optimized algorithm (2)

The idea is to decompose condition (5) along dimensions  $d$  by introducing a list  $M$  where each element  $M[i]$  is the sum of the tests along the lines of  $B$ :

$$M[i] = \sum_{l=-r}^{+r} g_{\delta}(i, j + l) \quad (i, j) \in \mathcal{P}.$$

Moreover, we also maintain a variable  $s(i, j)$  which is the sum of all elements in  $M$ :

$$s(i, j) = \sum_{c=-r}^{+r} M[i + c] \quad (i, j) \in \mathcal{P}.$$

Then, for any pixel  $(i, j) \in \mathcal{P}$  of the image, we first update the list  $M$  then  $s(\cdot)$ :

$$\begin{aligned} M[i + r] &\leftarrow M[i + r] - g_{\delta}(i + r, j - r - 1) + g_{\delta}(i + r, j + r) \\ s(i, j) &\leftarrow s(i - 1, j) - M[i - r - 1] + M[i + r] \end{aligned}$$

## Algorithmic considerations IV

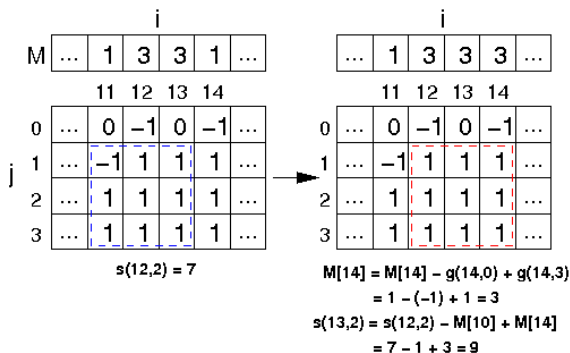


Figure 6: Illustration of the optimized algorithm on a 2D image with  $r = 1$ . Here, only the node  $p = (13, 2)$  is added to  $\mathcal{G}'$  since  $|s(13, 2)| = 3^2$ .

# Algorithmic considerations V

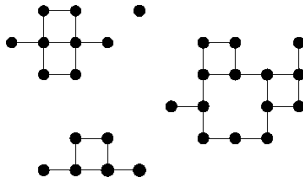
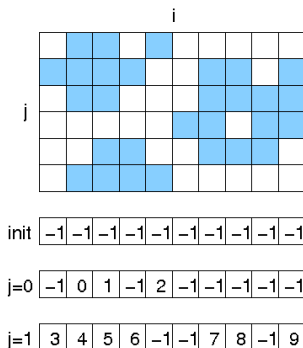
## Connecting nodes in $\mathcal{G}'$

For connecting nodes to their respective neighbors, we maintain a list  $L$  which store nodes indexes belonging to  $\mathcal{G}'$  and a counter *nodeld* indicating the last node index added. Extra memory storage is  $O(|L|^{(d-1)})$ .

In 2D, we apply the procedure below:

- 1 We initialize all elements of  $L$  to -1.
- 2  $nodeld \leftarrow 0$ .
- 3 For any point  $(i, j) \in \mathcal{P}$ 
  - If condition (5) is false for  $(i, j)$ :
    - We add current node to  $\mathcal{G}'$ .
    - If  $L[j] \geq 0 \Rightarrow addEdge(nodeld, L[j])$ .
    - If  $L[j - 1] \geq 0 \Rightarrow addEdge(nodeld, L[j - 1])$ .
    - $L[j] \leftarrow nodeld$ .
    - $nodeld \leftarrow nodeld + 1$ .
  - If condition (5) is true for  $(i, j)$ :
    - $L[j] \leftarrow -1$ .

## Algorithmic considerations VI



**Figure 7:** Illustration of building a graph  $\mathcal{G}'$  (right) from a 2D image (left). Blue squares correspond to nodes to add.

# General information I

## Information on tests

- Experiments performed on an Athlon Dual Core 6000+ with 2GB RAM.
- Times are averaged over 10 runs.
- Max-flow algorithm of Boykov/Kolmogorov [BK04] in v3.0.
- All tests are performed in connectivity 1.
- Segmentations are stored using sparse domains.

## Next sections

- 1 Study of the influence of the window radius  $r$  and  $\gamma$  with a  $TV + L^2$  model.
- 2 Reduction results using a  $TV + L^2$  model.
- 3 Reduction results using a Boykov/Jolly's model.

Influence of  $r$  and  $\gamma$  parameters I

## Images

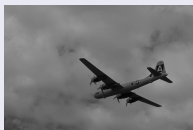


Image "plane"

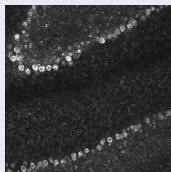


Image "cells"

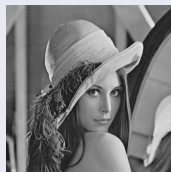


Image "lena"



Volume "woman"

Image	Size	Memory	Min	Max	Mean	Sampling
"plane"	1443 × 963	5.3 MB	0.0	179.0	117.71	3.0x
"cells"	1536 × 1536	9.0 MB	0.0	60.0	8.11	3.0x
"lena"	2048 × 2048	16.0 MB	14.0	255.0	116.77	4.0x
"woman"	211 × 172 × 92	12.7 MB	10.0	255.0	110.46	0.6x

Table 2: Table summarizing characteristics of images used for tests

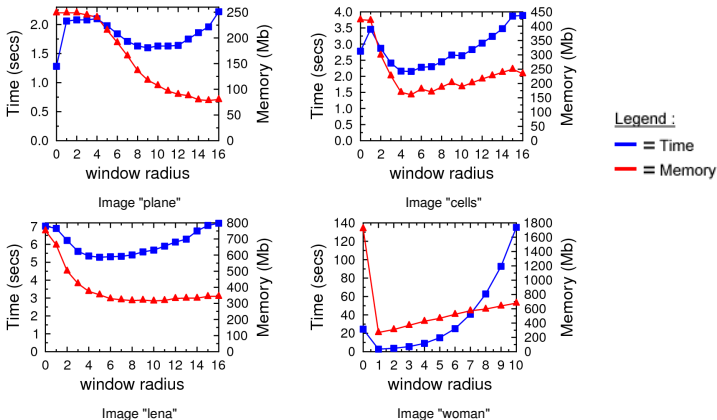
Influence of  $r$  and  $\gamma$  parameters II

Figure 8: Influence of window radius  $r$  for segmenting images with a  $TV + L^2$  energy model. Standard graph cuts correspond to  $r = 0$ .



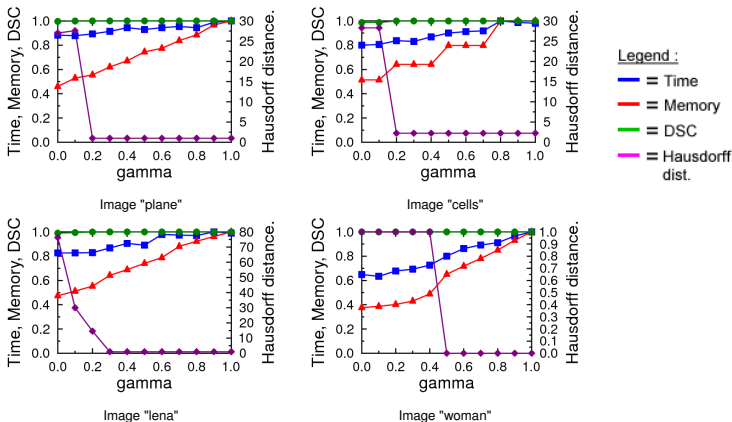
Influence of  $r$  and  $\gamma$  parameters III

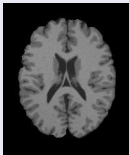
Figure 9: Influence of  $\gamma$  parameter for segmenting images with a  $TV + L^2$  energy model. The Window radius  $r$  is chosen to minimize both time and memory usage.

Results with  $TV + L^2$  model I

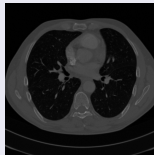
## Images



Image "book"



Volume "brain" + noise 3%



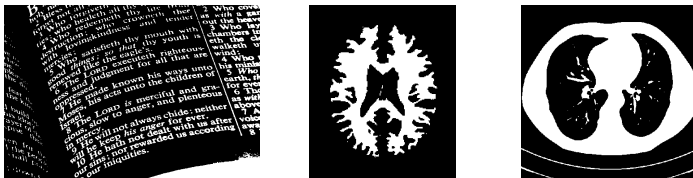
Volume "ct-thorax-0.8"

Image	Size	Memory	Min	Max	Mean	Sampling
"book"	3012 × 2048	25.53 MB	0.0	242.0	148.28	1.0x
"brain"	181 × 217 × 181	27.12 MB	0.0	173.0	26.06	1.0x
"ct-thorax-0.8"	409 × 409 × 252	160.81 MB	0.0	255.0	33.07	0.8x

Table 3: Table summarizing information on images used for tests.

## Information on tests

- Model's parameters are optimized for better visualization.
- Window radius is chosen such that memory usage is minimized while  $\gamma = 1$ .

Results with  $TV + L^2$  model II

	Original		Our algorithm	
Image	Time	Memory	Time	Memory ( $\rho$ )
"book"	5.05	1.07 GB	1.92	94.68 MB (91.36%)
"brain"	/	3.59 GB	7.31	434.33 MB (86.80%)
"ct-thorax-0.8"	/	21.38 GB	22.81	1.43 GB (91.85%)

Figure 10: Speed (secs) and memory usage compared to standard graph cuts for segmenting 2D/3D images with a  $TV + L^2$  energy model. Top row shows the segmentation results where object part correspond to white area.

## Results with Boykov/Jolly's model [BJ01]

## Images

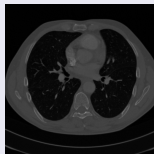
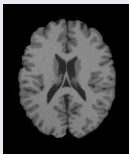


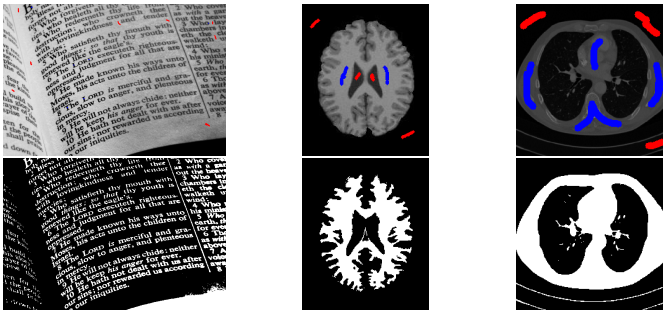
Image	Size	Memory	Min	Max	Mean	Sampling
"book"	3012 × 2048	25.53 MB	0.0	242.0	148.28	1.0x
"brain"	181 × 217 × 181	27.12 MB	0.0	173.0	26.06	1.0x
"ct-thorax-0.48"	245 × 245 × 151	160.81 MB	0.0	255.0	33.07	0.8x

Table 4: Table summarizing information on images used for tests.

## Information on tests

- Model's parameters are optimized for better visualization.
- Window radius is chosen such that memory usage is minimized while  $\gamma = 1$ .
- Object seeds and background seeds were placed by hand.

## Results with Boykov/Jolly's model [BJ01] II



	Original		Our algorithm	
Image	Time	Memory	Time	Memory ( $\rho$ )
"book"	5.58	1.08 GB	3.25	231.25 MB (78.5%)
"brain"	/	3.59 GB	9.02	734.64 MB (78.9%)
"ct-thorax-0.48"	/	4.58 GB	8.25	606.27 MB (83.6%)

Figure 11: Speed (secs) and memory usage compared to standard graph cuts for segmenting 2D/3D images with a Boykov/Jolly's energy model [BJ01]. Top and middle rows show respectively the seeds and the segmentation results where object part correspond to white area.

# Conclusion

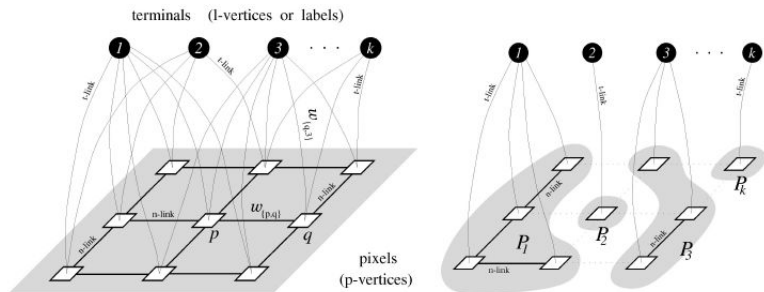
## Conclusion

- (+) Experimental results show important reduction rates.
- (+) Exact or approximate solutions can be obtained simply by tuning  $\gamma$ .
- (+) Reduction principle is easily extensible to higher dimensions.
- (-) Dependency between reductions rates and model's parameters.
- (-) Noise sensibility.

## Future work

- Evaluate results for segmenting lung tumors in TDM images.
- Extend implementation to color images.
- Application to the segmentation of lung tumors in TDM/PET images.
- Prove theoretical exactness of the reduction when  $\gamma = 1$ .
- Investigate other methods for solving the multi labels problem.

## Conclusion

Figure 12: An example of multiway cut on a  $3 \times 3$  grid graph.

# References I

- [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 PAMI*, 26(9):1124–1137, 2004.
- [CA08] C. Cigla and A.A. Alatan.  
Region-based image segmentation via graph cuts.  
*In ICIP*, pages 2272–2275, 2008.
- [FF62] L. Ford and D. Fulkerson.  
*Flows in network*.  
Princeton University Press, 1962.
- [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.



## References II

- [KZ04] V. Kolmogorov and R. Zabih.  
What energy functions can be minimized via graph cuts?  
*IEEE Transactions on PAMI*, 26(2):147–159, 2004.
- [LML10a] N. Lermé, F. Malgouyres, and L. Létocart.  
Reducing graphs in graph cut segmentation.  
In *ICIP*, 2010.
- [LML10b] N. Lermé, F. Malgouyres, and L. Létocart.  
Réduction de "vision graph".  
Patent No. PB0091FR, January 2010.
- [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] Yin. Li, Jian. Sun, Chi-Keung. Tang, and Heung-Yeung. Shum.  
Lazy snapping.  
*ACM Transactions on Graphics*, 23(3):303–308, 2004.

## References III

- [RCD07] F. Ranchin, A. Chambolle, and F. Dibos.  
*Total Variation Minimization and Graph Cuts for Moving Objects Segmentation*, pages 743–753.  
2007.
- [ROF92] L. Rudin, S. Osher, and E. Fatemi.  
Nonlinear total variation based noise removal algorithms.  
*Phys. D*, 60(1-4):259–268, 1992.
- [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.