An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision

FINAL RESTITUTION

Students:

- Van-Khoa NGUYEN
- Gustavo RODRIGUES

Date: April 3rd, 2020

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

OUTLINE

I. Overview
II. α - Expansion
III. α-β Swap
IV. Image Segmentation
V. Application

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Energy minimization with graph cuts

- Graph Cuts can find the optimal solution to a binary problem.
- When each pixel can be assigned many labels, an algorithmic solution can be computationally expensive.
- Graph Cuts can be used to minimize the following kind of energy law:

Constraints to the smoothing term:

$$1)V(\alpha,\beta)=0 \Leftrightarrow \alpha=\beta$$
$$2)V(\alpha,\beta)=V(\beta,\alpha)\geq 0$$
$$3)V(\alpha,\beta)\leq V(\alpha,\gamma)+V(\gamma,\beta)$$

Constraints 1 and 2: semi - metric, sufficient for α-β Swap.

All the constraints: metric, necessary for the α- expansion.

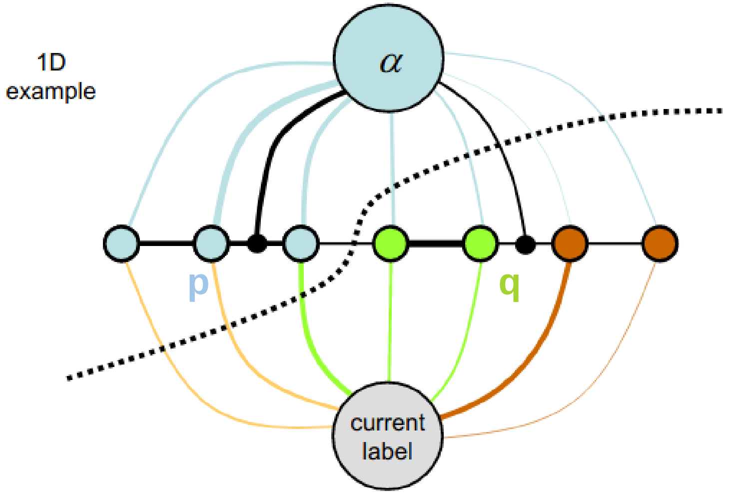
$$E(f)=\underbrace{\sum_{p\in\mathcal{P}} D_p(i_p,f_p)}_{\text{Data term}}+\underbrace{\sum_{p,q\in\mathcal{N}} V_{p,q}(f_p,f_q)}_{\text{Smooth term}}$$

Data term - assures current label f is coherent with observe data i

Smooth (regularization) term - assures overall labelling is smooth, penalizes great differences between neighboring pixels.

II. α - expansion

Algorithm



1. Start with an arbitrary labeling f
2. Set success := 0
3. For each label $\alpha \in \mathcal{L}$
   3.1. Find $\hat{f} = \arg\min E(f')$ among f' within one α-expansion of f (Section 4)
   3.2. If $E(\hat{f}) < E(f)$, set f := $\hat{f}$ and success := 1
4. If success = 1 goto 2
5. Return f

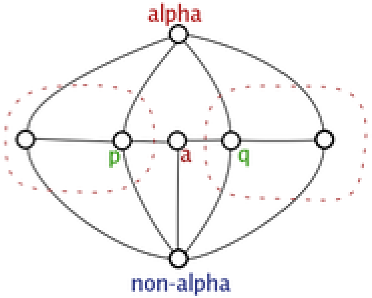
a Cycle

Obs: Being c the global minima of E(f) when the α - expansion is used is guaranteed to find a convergence in the direction of this minima within a factor of 2 in the best case.

$$2c = 2 \max_{p,q \in \mathcal{N}} \frac{\max_{\alpha \neq \beta} V(\alpha, \beta)}{\min_{\alpha \neq \beta} V(\alpha, \beta)}$$

II. α - expansion

Constructing the graph



Weights to each kind of edge in the graph constructed for the alpha expansion:

$$w(\alpha, p) = D(\alpha)$$

$$w(\bar{\alpha}, p) = D(f_p) \rightarrow if p \notin \mathcal{P}_\alpha$$

$$w(\bar{\alpha}, p) = \infty \rightarrow if p \in \mathcal{P}_\alpha$$

$$w(p, a) = V(f_p, \alpha); w(a, q) = V(f_q, \alpha)$$

$$w(p, q) = V(f_p, \alpha) when \rightarrow f_p = f_q$$

$$w(a, \overline{\alpha}) = V(f_p, f_q)$$

Rules for decision after performing a min cut algorithm:

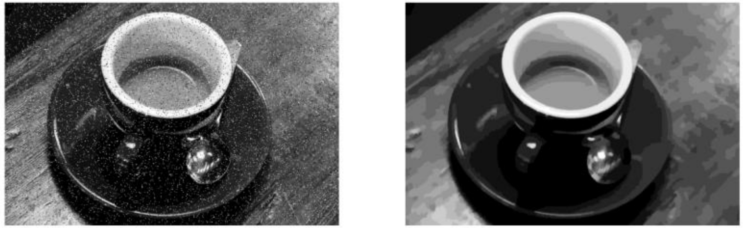
$$(p - \alpha) cut \rightarrow p \in \alpha$$

$$(p - \overline{\alpha}) cut \rightarrow p \in \overline{\alpha}$$

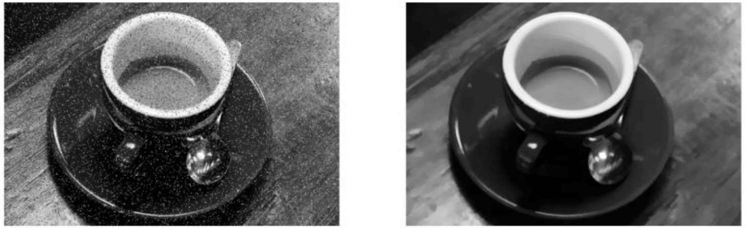
II. α - expansion

Reconstruction with α - expansion

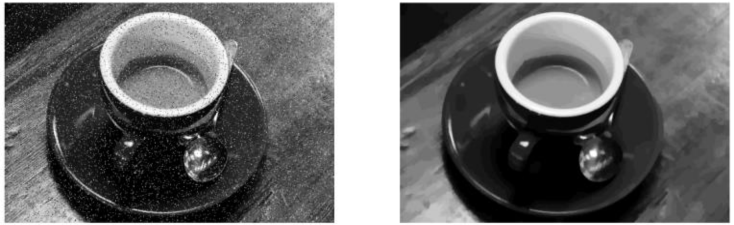
16 levels of gray - Time: 3.01s



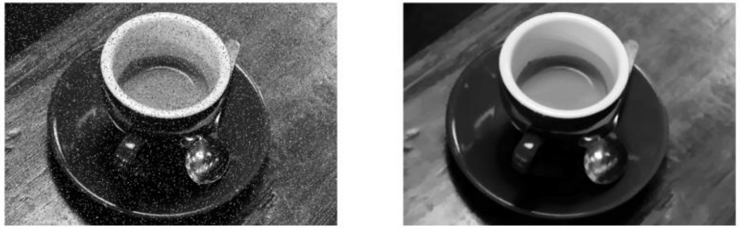
64 levels of gray - Time: 16.76s



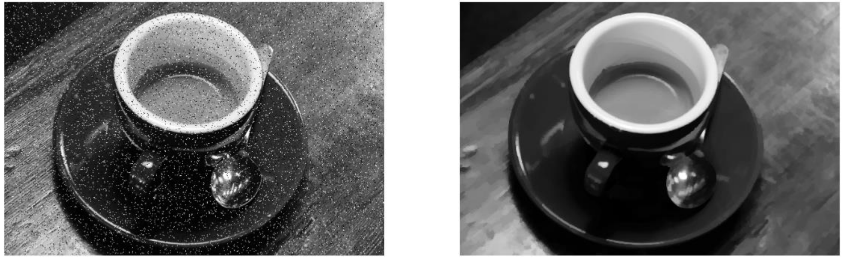
32 levels of gray - Time: 7.14s



128 levels of gray - Time: 39.52s



256 levels of gray - Time: 89.07s



Experiments conducted with images which salt and pepper noise were added.

III. α-β Swap Image Restoration

Algorithm

a Cycle

1. Start with an arbitrary labeling f of the image
2. Set success := 0
3. For each pair of labels $\{\alpha, \beta\} \in a\ set\ of\ labels\ \mathcal{L}$

   3.1. Find $\hat{f} = argminE(f')\ among\ f'$ within one $\alpha - \beta\ swap\ of\ f$ [*Graph Cut*]

   3.2. If $E(\hat{f}) < E(f), set\ f := \hat{f}$ and success := 1
4. If success = 1, go to 2
5. Return f

Image restoration: labels are all distinct pixel values in a image

III. α-β Swap Image Restoration

Graph construction and Energy specification

For each α-β Swap in a **Cycle**:

- Build a α-β swap graph for pixels having α or β density values (by weighting the t-links and n-links).

- Apply the graph cut algorithm to find a cut C yielding the minimum energy.

- The cut C will redetermine density values for the graph's pixels.

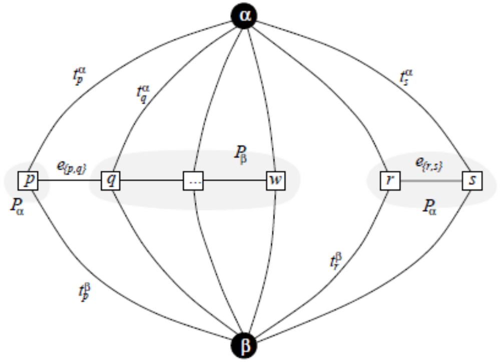


Fig 1. Graph construction in the α-β swap.

edge	weight	for
t_p^α	$D_p(\alpha) + \sum_{\substack{q \in \mathcal{N}_p \\ q \notin \mathcal{P}_{\alpha\beta}}} V_{\{p,q\}}(\alpha, f_q)$	$p \in \mathcal{P}_{\alpha\beta}$
t_p^β	$D_p(\beta) + \sum_{\substack{q \in \mathcal{N}_p \\ q \notin \mathcal{P}_{\alpha\beta}}} V_{\{p,q\}}(\beta, f_q)$	$p \in \mathcal{P}_{\alpha\beta}$
$e_{\{p,q\}}$	$V_{\{p,q\}}(\alpha, \beta)$	$\{p,q\} \in \mathcal{N}$ $p,q \in \mathcal{P}_{\alpha\beta}$

Table 1. Edge weight assignments in the α-β Swap

III. α-β Swap Image Restoration

Experience setting

Seeking a labeling f minimize the following defined energy:

Data term:

$$E_{data}(f) = \sum_{p \in P} (f_p - i_p)^2$$

$where\ i_p\ is\ the\ observed\ intensity\ of\ the\ pixel\ p$

- Tested three different potentials for the smoothness term.
- Image tests were generated with pepper & salt noises (10%,15%,25% noisy pixels)

Smoothness term:

$$E_{smooth} = \sum_{\{p,q\} \in N} V_{\{p,q\}}(f_p, f_q)$$

Linear potential

$$V_{\{p,q\}}(f_p, f_q) = abs(f_p - f_q)$$

Truncated linear potential

$$V_{\{p,q\}}(f_p, f_q) = \min(200, abs(f_p - f_q))$$

Potts

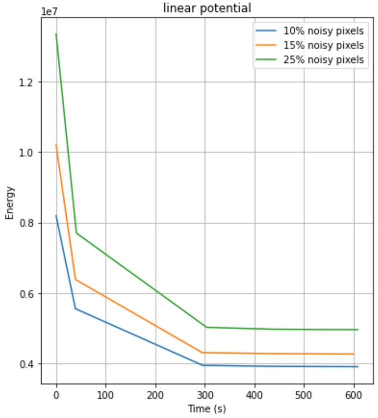
$$V_{\{p,q\}}(f_p, f_q) = \begin{cases} 255\ if\ abs(f_p - f_q)^2 > 255 \\ 0\ otherwise \end{cases}$$

The **Pymaxflow** library in python implementing our studied graph cut algorithm was used in this experience.

III. α-β Swap Image Restoration

Results

- The minimum energy value bases on the potential type on which the energy is calculated.
- The algorithm was run in **one** cycle of the α-β swap algorithm.
- After 300 s, the energy began to converge to the minimum value.
- In **one** testing cycle: the lower noisy pixels, the lower value the energy converges to.



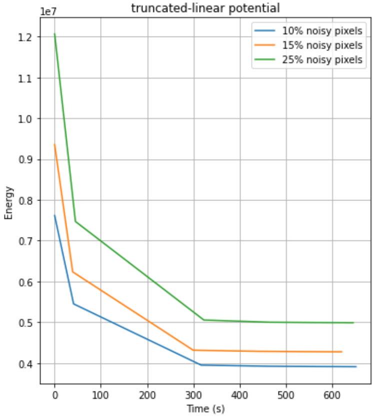
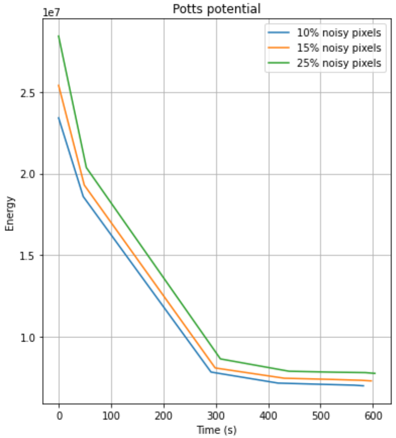
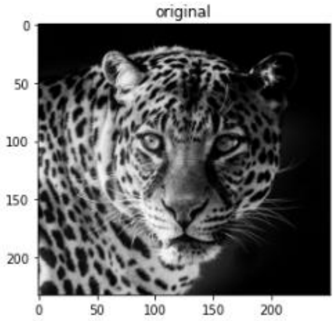


Fig 2. Energy minimization process for three different potentials.



III. α-β Swap Image Restoration

Results



	Peak signal noise ratio		
Noisy pixels	**Linear**	**Truncated linear**	**Potts**
10%	21.82	21.75	20.25
15%	24.31	24.23	19.45
25%	20.8	20.75	18.02

Table 2. PSNR image restoration

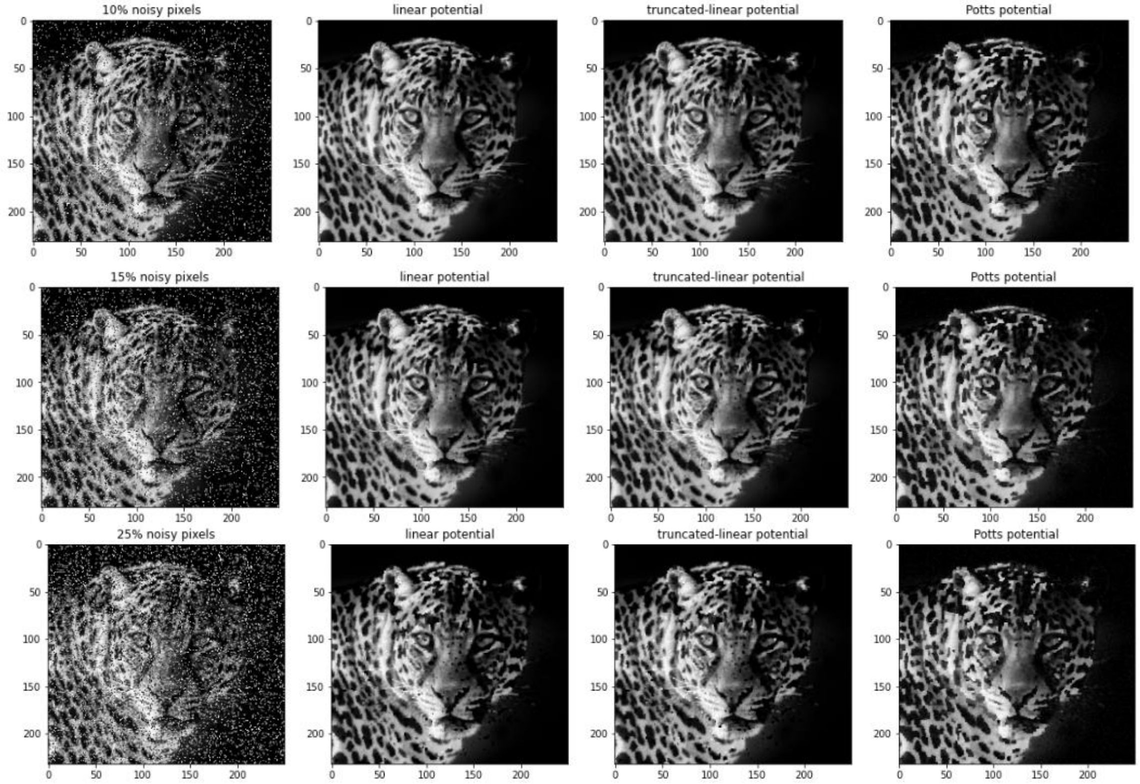


Fig 3. Image restoration results

Experiment setting

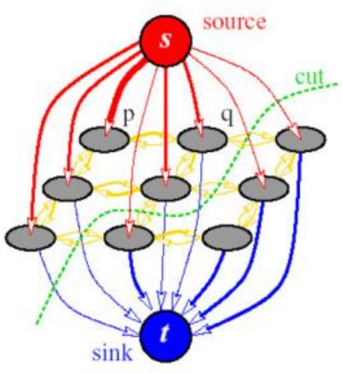


Fig . Image segmentation graph representation

- Two labels: background (sink), foreground (source).
- Seeds added as hard constraints => reduce large feasible solutions.

Region term:

pixel	$R_p(obj)$	$R_p(bkg)$
bkg seed	0	K
obj seed	K	0
others	0	0

Boundary term:

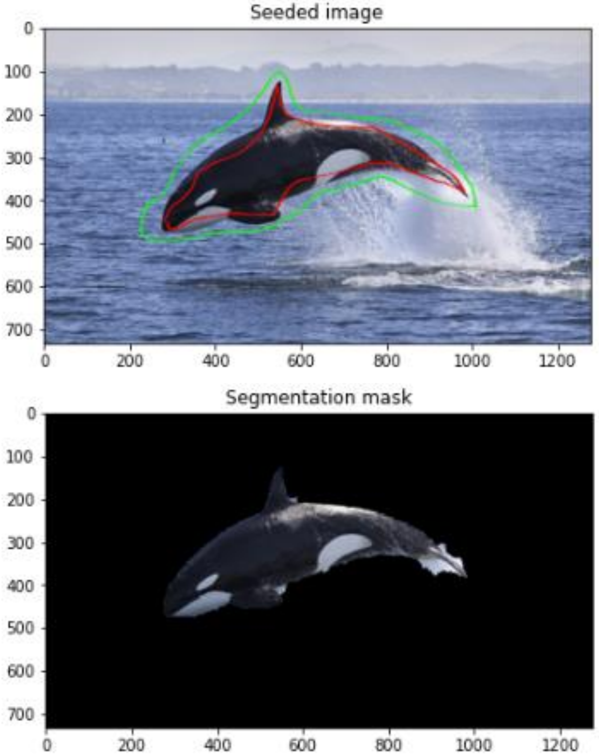
$$B_{\{p,q\}\in N} = \frac{1}{1 + \|p-q\|_2^2}$$

- p, q are neighboring pixels (RGB).
- Bpq small when p,q are differents.
- K = max(Bpq) ∀ p,q ∈ N.
- Bpq = Bqp.

The **Pymaxflow** library in python implementing our studied graph cut algorithm was used in this experience.

IV. Image Segmentation

Results



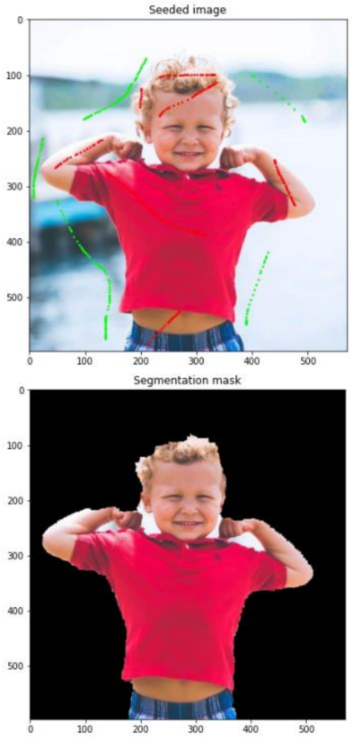


	Image size	Segmentation time (s)
hat	(181,229,3)	1.20
baby	(596,571,3)	10.15
whale	(733,1277,3)	26.85

Table 3. Computational time

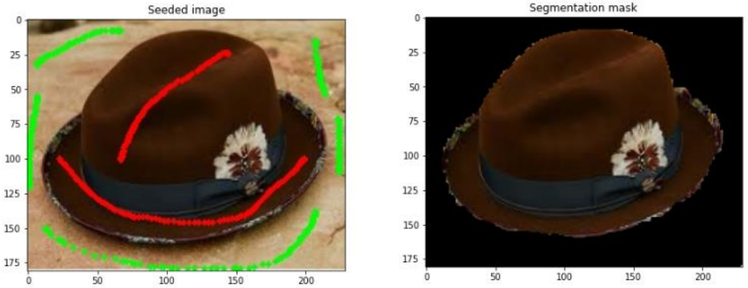
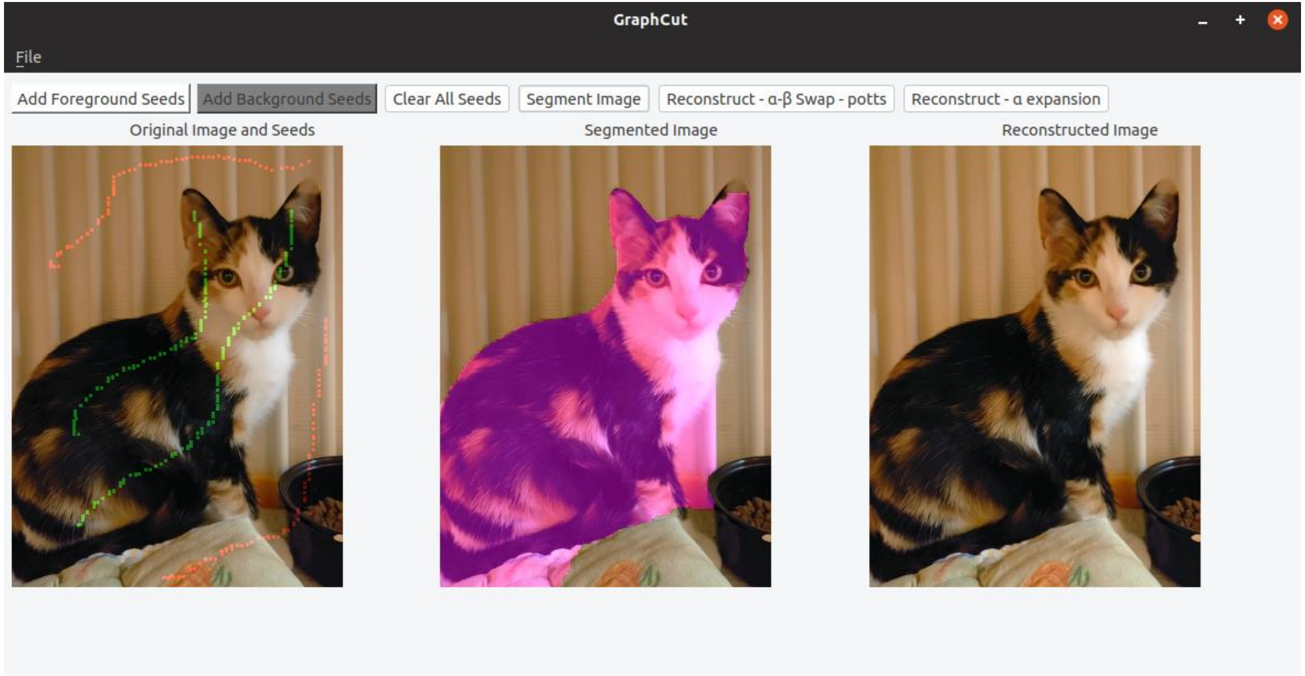


Fig 4. Seeded images and segmentation masks.

V. Application



An application to illustrate the different algorithms is proposed in an interactive application.

The requirements are the following:

- pyQT 4
- Pillow
- cv2
- Numpy
- Pymaxflow

Obs: The two different reconstructions may take a while to complete the execution. All reconstructions are performed with the grayscale image.

To install all requirements is necessary to execute in a terminal:
apt install python3-pyqt4
pip install -r requirements.txt
To launch the application is sufficient to execute: python3 Graph_cut_UI.py