GrabCut

Interactive Foreground Extraction using Iterated Graph Cuts

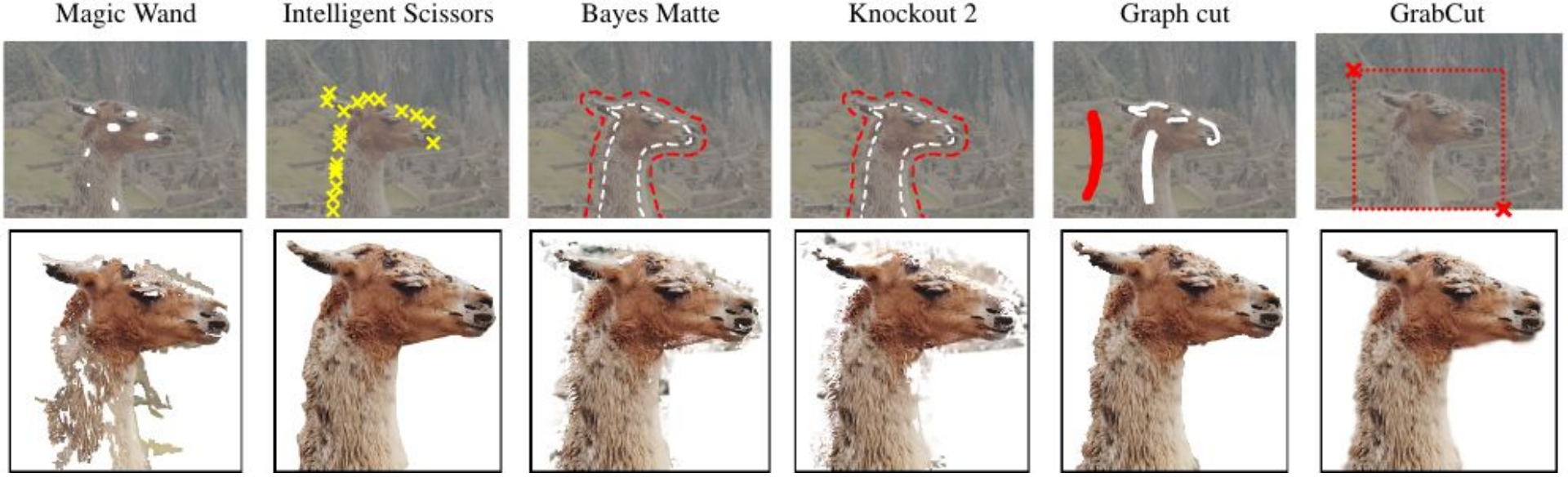



IDEES CLES

- **Trimap {TB, TU, TF}** : généralement, background et foreground sont fournis par l'utilisateur, et on cherche à calculer les "alpha values" dans la région TU

- **Alpha values** : ils décrivent la segmentation de l'image. Pour un pixel n, alpha(n) = 0 si le pixel appartient au "background", alpha(n) = 1 si le pixel appartient au foreground

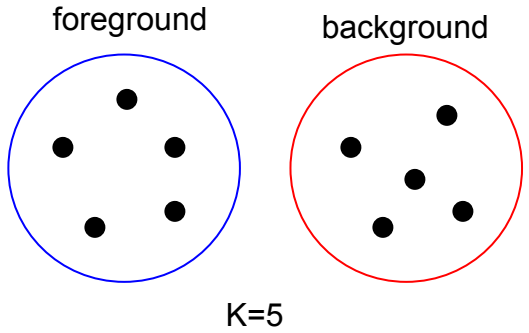
- Algorithme basé sur la recherche de coupe minimale : **minimal graph-cut**



AMELIORATIONS:

- modèle monochrome (basé sur des histogrammes) remplacé par un **modèle en couleur**, basé sur des **GMM** (Gaussian Mixture Model)
- algorithme **itératif** basé sur l'algorithme de **graph-cuts**
- interaction très faible de l'utilisateur: **labelling incomplet** (juste TB est spécifié)

Modèle de couleur



- mélanges de gaussiennes(GMM)
- 2 GMMs représentant 2 classes:
  - α=1(foreground)
  - α=0(background)
- chaque GMM contient K composantes(couleur)
- paramètre du modèle

$$\underline{\theta} = \{\pi(\alpha,k), \mu(\alpha,k), \Sigma(\alpha,k),\ \alpha = 0,1,\ k = 1 \ldots K\}\ ,$$

- chaque pixel *n* est classifié dans une classe(foreground ou background), noté par un attribut α_n ($\alpha_n \in \{0,1\}$)
- chaque pixel *n* est classifié plus précisément dans une composante de GMM de sa classe, noté par un attribut k_n ($k_n \in \{1,2,...,K\}$)

Energie à minimiser :

$$\mathbf{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}) = U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}) + V(\underline{\alpha}, \mathbf{z}),$$

$$U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}) = \sum_n D(\alpha_n, k_n, \underline{\theta}, z_n),$$

terme d'attache aux données

$$D(\alpha_n, k_n, \underline{\theta}, z_n) = -\log p(z_n \mid \alpha_n, k_n, \underline{\theta}) - \log \pi(\alpha_n, k_n)$$

$$D(\alpha_n, k_n, \underline{\theta}, z_n) = -\log \pi(\alpha_n, k_n) + \frac{1}{2} \log \det \Sigma(\alpha_n, k_n)$$

$$+ \frac{1}{2} [z_n - \mu(\alpha_n, k_n)]^\top \Sigma(\alpha_n, k_n)^{-1} [z_n - \mu(\alpha_n, k_n)].$$

$$V(\underline{\alpha}, \mathbf{z}) = \gamma \sum_{(m,n) \in \mathbf{C}} [\alpha_n \neq \alpha_m] \exp -\beta \|z_m - z_n\|^2 .$$

terme de régularisation

[A]=1 si A est vrai, [A]=0 si A est faux

Initialisation

- User initialises trimap T by supplying only T_B. The foreground is set to $T_F = \emptyset$; $T_U = \overline{T}_B$, complement of the background.
- Initialise $\alpha_n = 0$ for $n \in T_B$ and $\alpha_n = 1$ for $n \in T_U$.
- Background and foreground GMMs initialised from sets $\alpha_n = 0$ and $\alpha_n = 1$ respectively.

Iterative minimisation

1. *Assign GMM components to pixels:* for each n in T_U,
$$k_n := \arg\min_{k_n} D_n(\alpha_n, k_n, \theta, z_n).$$
2. *Learn GMM parameters* from data $\mathbf{z}$:
$$\underline{\theta} := \arg\min_{\underline{\theta}} U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z})$$
3. *Estimate segmentation:* use min cut to solve:
$$\min_{\{\alpha_n:\, n \in T_U\}} \min_{\mathbf{k}} \mathbf{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}).$$
4. Repeat from step 1, until convergence.
5. Apply border matting (section 4).

User editing

- *Edit:* fix some pixels either to $\alpha_n = 0$ (background brush) or $\alpha_n = 1$ (foreground brush); update trimap T accordingly. Perform step 3 above, just once.
- *Refine operation:* [optional] perform entire iterative minimisation algorithm.

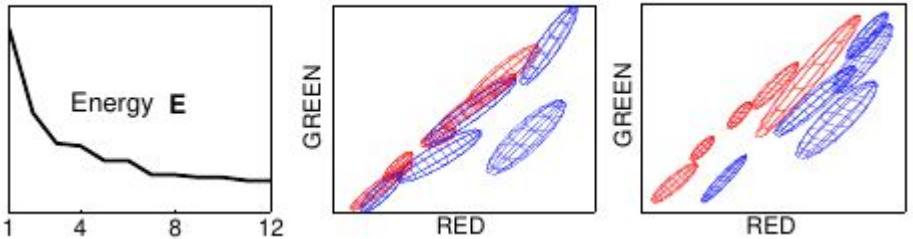
ITERATIVE ENERGY MINIMIZATION

1. mise à jour de k dans TU par minimisation de l' énergie
2. $F(k) = \{z_n : k_n = k \text{ and } \alpha_n = 1\}$

   $\mu(\alpha, k)$ = sample mean of pixel values in F(k)

   $\Sigma(\alpha, k)$ = sample covariance of pixel values in F(k)

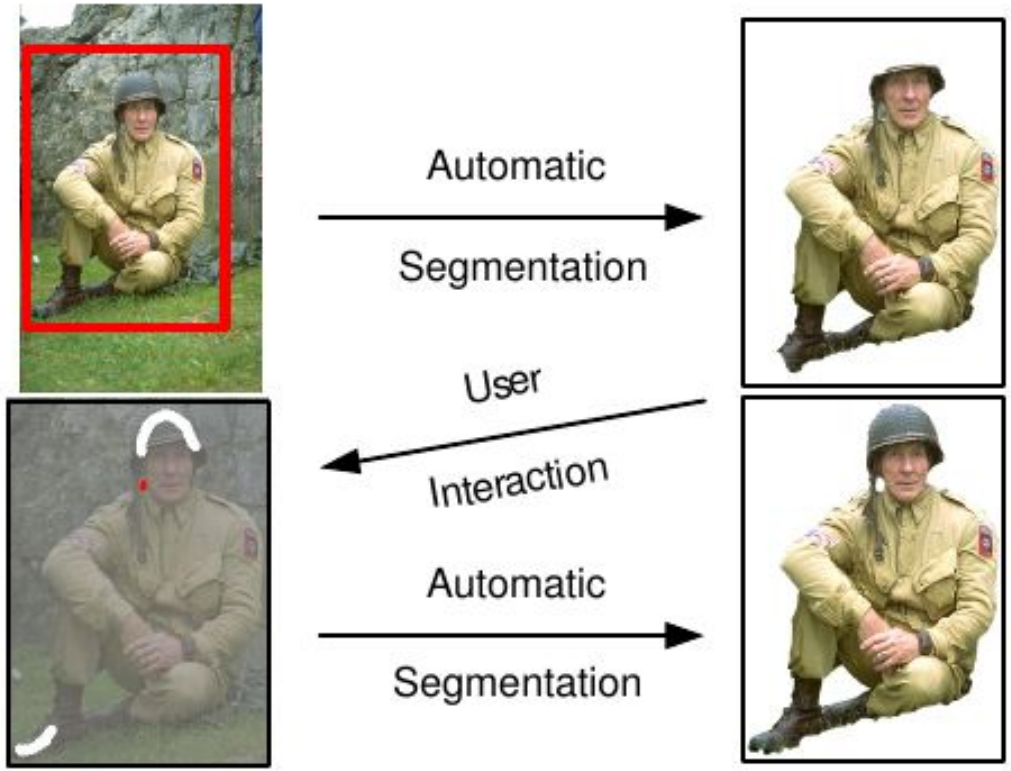
   $\pi(\alpha, k) = |F(k)| / \sum_k |F(k)|$
3. mise à jour de alpha dans TU par recherche de coupe minimale



On a **convergence** vers un minimum local

USER INTERACTION

Dans l’initialisation, seulement les labels de TB sont définitifs



Après l'exécution de l’algorithme, l’utilisateur peut donner des **labels définitifs** au background (rouge) et au foreground (blanc)

Il n’est pas nécessaire de marquer les objets entièrement, car l’algorithme est relancé et les labels se propagent selon ce que l’utilisateur a marqué

Amélioration des contours

1. border matting:
transformer les valeurs binaires de α en valeurs continues

$$\alpha_n = g\left(r_n; \Delta_{t(n)}, \sigma_{t(n)}\right)$$

Parameter values $\Delta_1, \sigma_1, \ldots, \Delta_T, \sigma_T$ are estimated by minimizing the following energy function using DP over t:

$$E = \sum_{n \in T_U} \tilde{D}_n(\alpha_n) + \sum_{t=1}^{T} \tilde{V}(\Delta_t, \sigma_t, \Delta_{t+1}, \sigma_{t+1}) \qquad (12)$$

$$\tilde{V}(\Delta, \sigma, \Delta', \sigma') = \lambda_1 (\Delta - \Delta')^2 + \lambda_2 (\sigma - \sigma')^2,$$

$$\tilde{D}_n(\alpha_n) = -\log \mathbf{N}\left(z_n; \mu_{t(n)}(\alpha_n), \Sigma_{t(n)}(\alpha_n)\right)$$

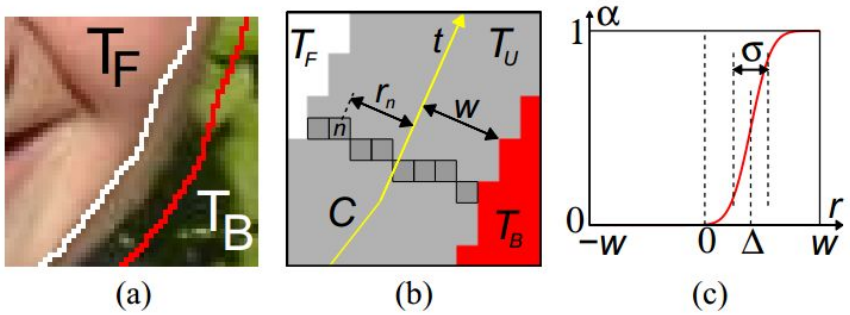
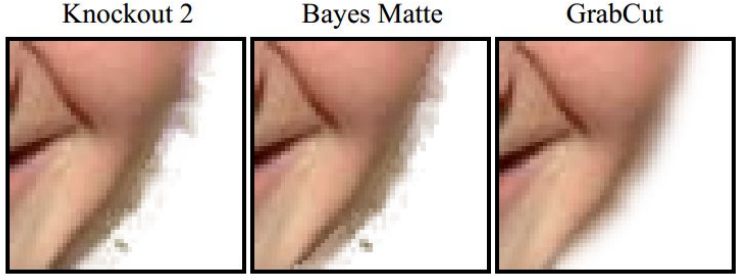


Figure 6: **Border matting.** (a) Original image with trimap overlaid. (b) Notation for contour parameterisation and distance map. Contour C (yellow) is obtained from hard segmentation. Each pixel in T_U is assigned values (integer) of contour parameter t and distance r_n from C. Pixels shown share the same value of t. (c) Soft step-function for α-profile g, with centre Δ and width σ.



2. foreground pixel stealing:
restaurer la fuite de couleur de background en volant les pixel foreground

POINTS FORTS

- Prise en compte de la **couleur**
- **Faible interaction** de l'utilisateur
- Bonne **qualité des contours**
- grâce aux **K composants** de chaque GMM (2K composants au total), on peut détecter plus des variations de couleur dans le foreground et le background

POINTS FAIBLES

- Plus **lent et complexe** que ses concurrents
- Le niveau d'interaction de l'utilisateur depend de l'image. Par exemple, besoin d'interaction supplémentaires si:
  - transition avec peu de contraste entre le foreground et le background
  - camouflage (chevauchement en couleur entre le foreground et le background)
  - le background contenu dans le rectangle rouge n'est pas bien représenté par la zone TB

CONCLUSION





No User Interaction

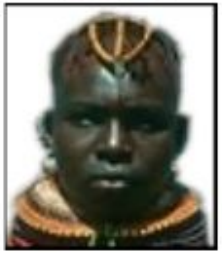










UNE AUTRE APPROCHE : Méthode **markovienne** et optimisation par **graph-cut**

Critère MAP (maximiser $P(X|Y) \propto P(Y|X)P(X)$)

<=>

minimiser

$$\mathcal{U}(x|y) = \sum_i V_c(y_i|x_i) + \sum_{(i,j)} \beta(x_i - x_j)^2$$

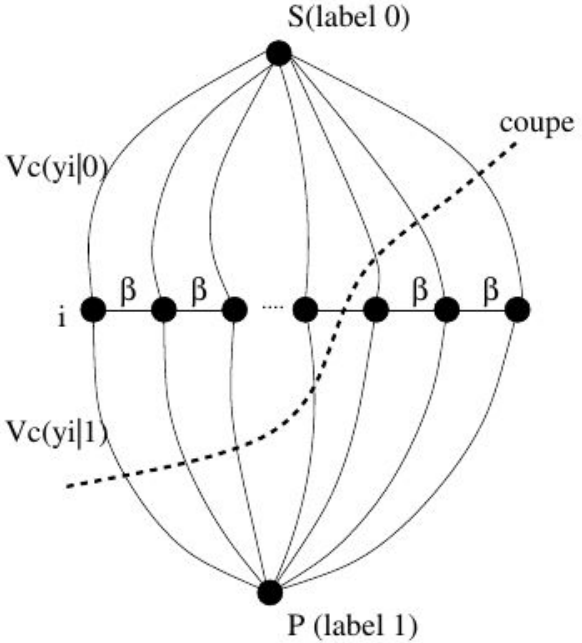
terme d'attache aux données + potentiel des cliques pour le modèle d'Ising

<=>

minimiser

$$cut(E_S, E_P) = \sum_{i \in E_S} V_c(y_i|1) + \sum_{i \in E_P} V_c(y_i|0) + \sum_{(i \in E_s, j \in E_P)} \beta$$

xi = 1 pour i ∈ ES , xi = 0 pour i ∈ EP



BIBLIOGRAPHIE

- C. Rother, V. Kolmogorov, A. Blake: Grab Cut Interactive foreground extraction using iterated graph-cuts, SIGGRAPH 2004