INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Poisson Image Editing

Patrick Pérez, Michel Gangnet, Andrew Blake

Blondel Charlotte

Cadaux Ema

Cros Marion

Mametjanova Aijana

Introduction

La retouche de photo fait partie intégrante du quotidien : réseaux sociaux, publicités, cinéma... Mais quels algorithmes se cachent derrière cela ?

L'édition d'une image peut être **globale** ou **locale**.

Notre projet s'axe sur les modifications locales avec sélection manuelle des zones d'intérêt avec création de masques.

- L'outil mathématiques fondamental du projet : **l'équation aux dérivées partielles de Poisson** avec conditions aux limites de **Dirichlet**.

Sommaire

I. Insertion

a. Basique

On a 3 images : la cible T, la source S et son masque

Cible

Source

Masque

I. Insertion

a. Basique

But : Copier les gradients spatiaux ∇S de l'image source S dans l'image cible T, et non les valeurs de couleurs de S.

On cherche donc une image **u** solution de :

$$\min_u \int_\Omega ||\nabla u - \nabla S||^2 + \iota_K(u)$$

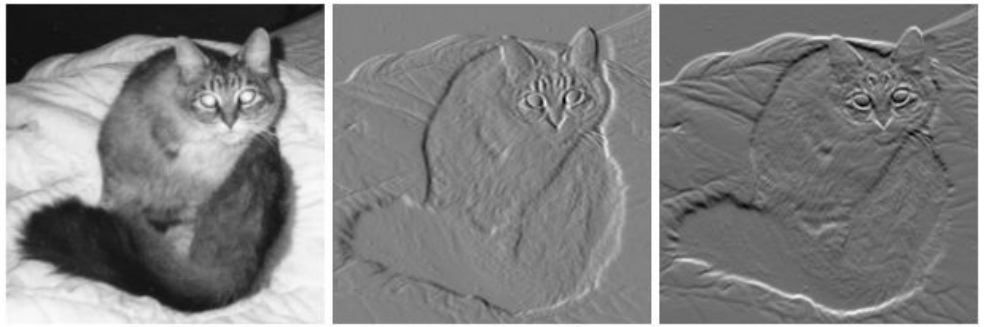
avec K l'ensemble des images qui coïncident avec la cible à l'extérieur du masque.

I. Insertion

a. Basique

Calcul du gradient et de la divergence :

- **Gradient discret** : Représente les changements d'intensité ou de couleur dans une image
- **Divergence discrète** : Opération qui prend en compte les différences entre les intensités des pixels adjacents dans une image



Calcul de la **projection** : Les pixels de l'image source correspondants aux pixels égaux à 1 du masque sont conservés tandis que ceux n'appartenant pas à cette région sont remplacés par les pixels de l'image cible

I. Insertion

a. Basique

Projection naïve :

➔ Fusionner de manière simple deux images

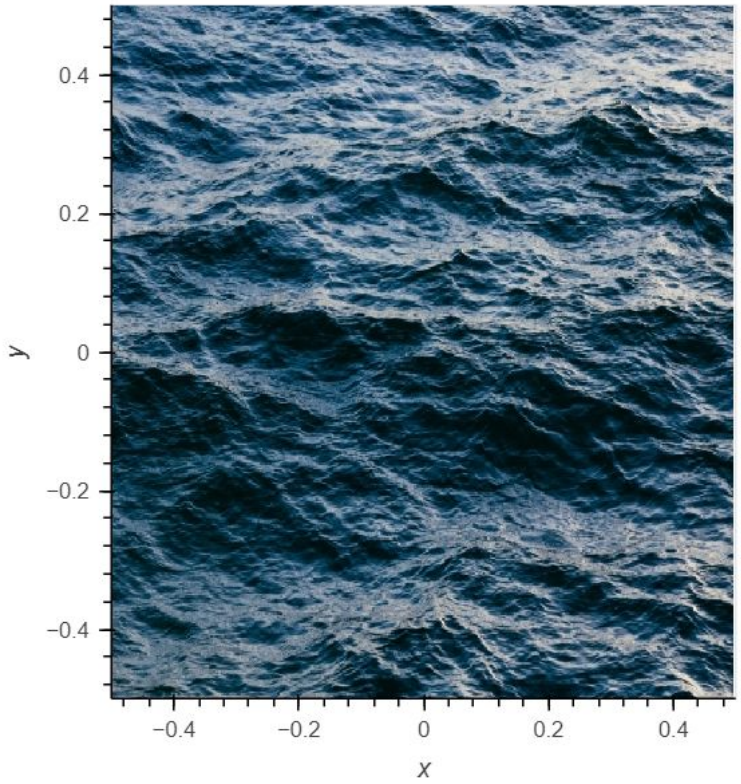
➔ Choix de l'image source, avec l'objet que l'on veut transférer (une personne, un animal etc...), son masque binaire et une image cible sur laquelle l'objet va être fusionné.

➔ Projection simple de l'image source sur l'image cible grâce au masque

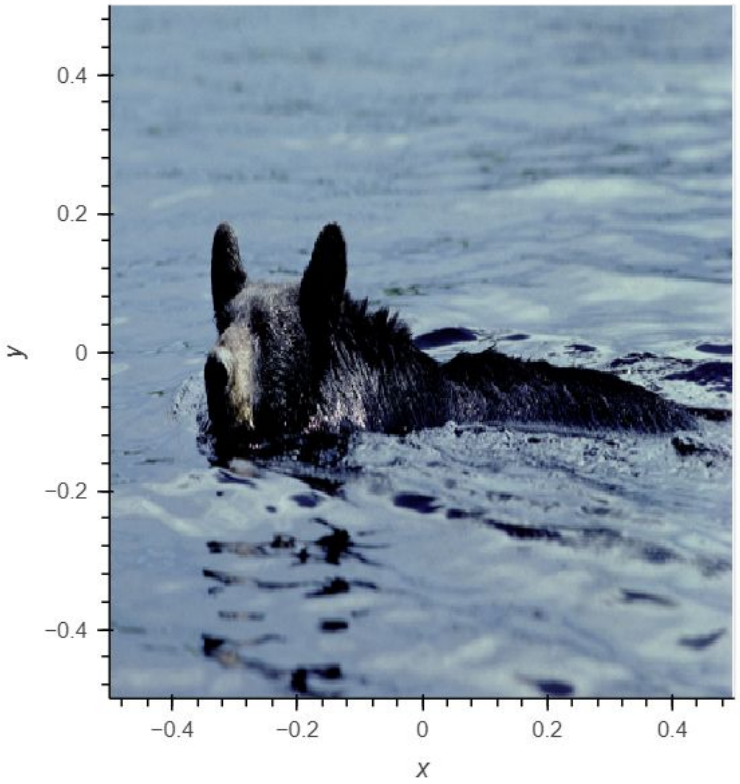
I. Insertion

a. Basique

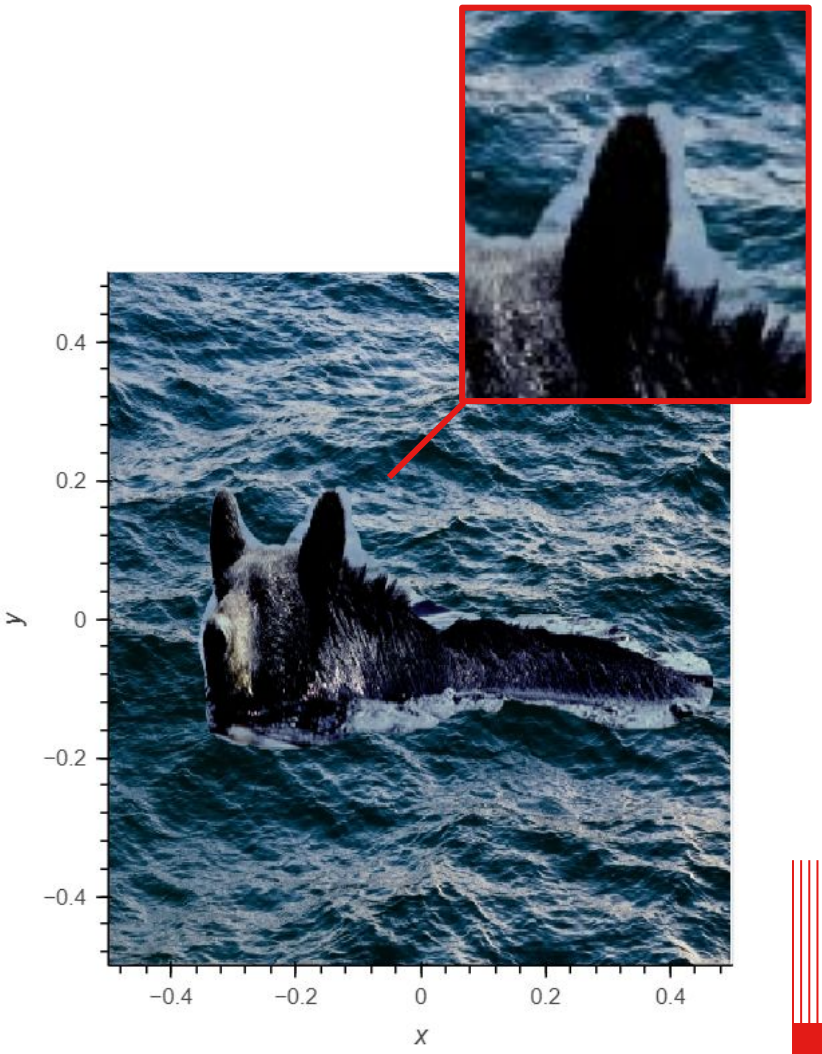
Projection naïve :



Cible



Source



Résultat

I. Insertion

a. Basique

Forward Backward Poisson :

Ajuste l'image source pour qu'elle corresponde mieux à l'image cible tout en conservant les contraintes apportées par le masque.

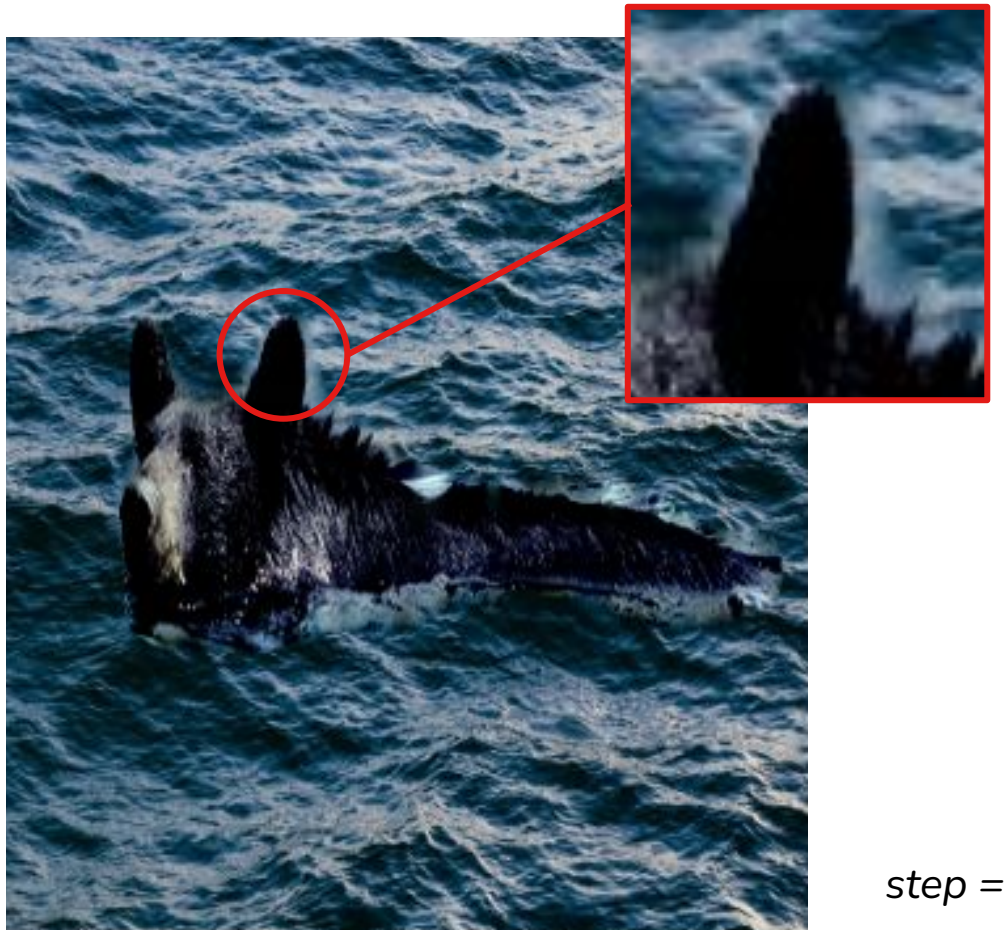
- ➔ Calcul d’une mise à jour de x en utilisant la formule $x-s*\nabla f(x,y)$
- ➔ Projection de cette mise à jour sur l'image cible

```python
def FBPoissonEditing(targ,sour,ma,step,Niter):

    f = []

    # Copie de l'image source
    x = np.copy(sour)

    # Calcule le gradient de l'image source
    y = Gradient(sour)

    for i in range(Niter):

        # Calcule valeur temporaire en utilisant le gradient
        temp = x - step * GradientFonc(x,y)

        # Projete la valeur temporaire sur l'image cible en utilisant le masque
        x = Proj(temp,ma,targ)

        # Calcule les gradients de l'image modifiée
        f1,f2 = Gradient(x - targ)*ma

        # Calcule et stocke la norme L2 des gradients
        f.append(np.linalg.norm(f1,2) + np.linalg.norm(f2,2))

    return np.clip(x,0,255), f[:10]
```

I. Insertion

a. Basique

Forward Backward Poisson :



- Moins de défauts
- Bords de l’image source moins épais
- Temps de compilation de ~50s

step = 1/4 et niter=300

I. Insertion

a. Basique

FISTA Poisson :

Ajuste l'image source pour qu'elle corresponde mieux à l'image cible tout en conservant les contraintes apportées par le masque.

- ➔ Calcul d'une mise à jour de x en utilisant la formule (x+n/(α + n)*e) - s*∇ f(x,y) avec e la différence entre les itérations successives
- ➔ Projection de cette mise à jour sur l'image cible
- ➔ Mise à jour de e

```
# Met à jour temp avec une composante de la dernière erreur
temp = x + Niter/(alpha+Niter) * e

# Met à jour x en utilisant la méthode FISTA et la fonction de gradient
x = temp - step * GradientFonc(temp,y)

# Projet x sur l'image cible en utilisant le masque
x = Proj(x,ma,targ)

# Met à jour e avec la différence entre x actuel et x précédent (xp)
e = x - xp
```

I. Insertion

a. Basique

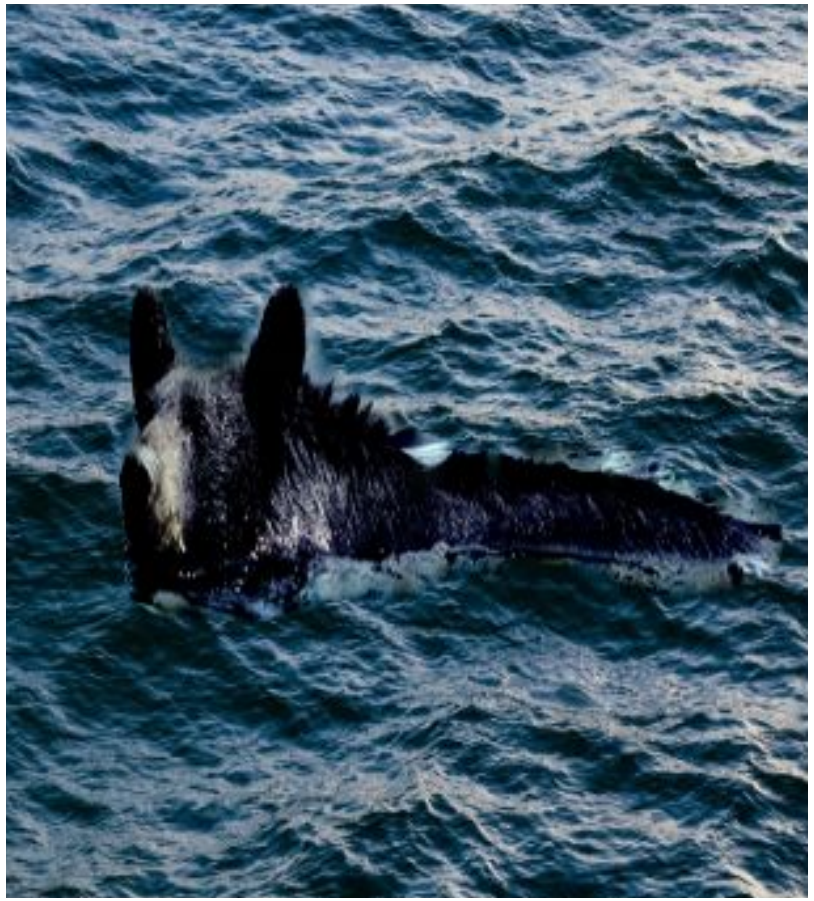
FISTA Poisson :

- Bords de l’image source moins épais
- Temps de compilation de ~50s

step = 1/10 et niter = 300

I. Insertion

a. Basique

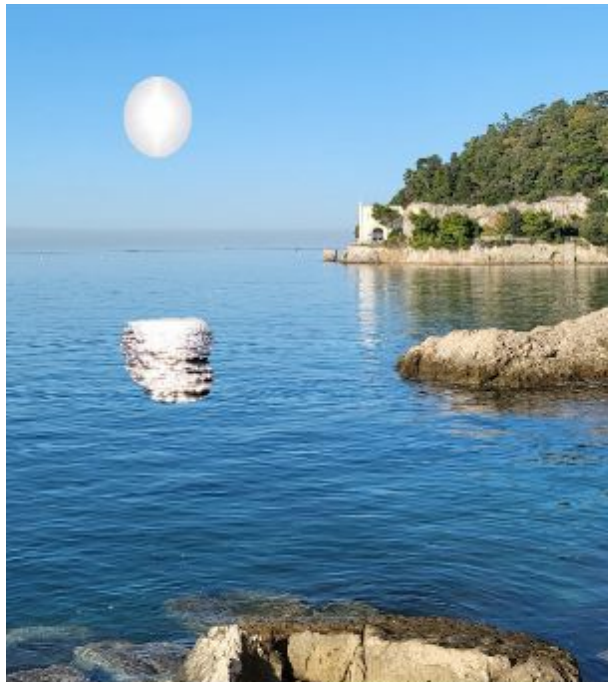


Forward Backward

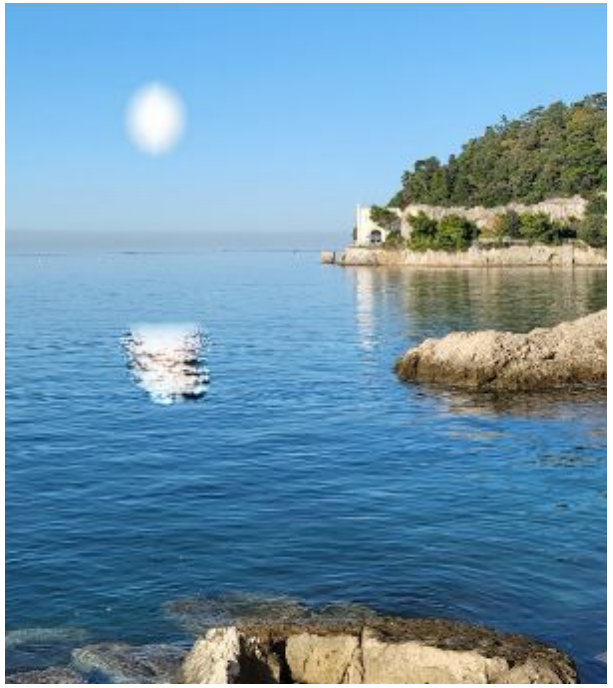
FISTA

I. Insertion

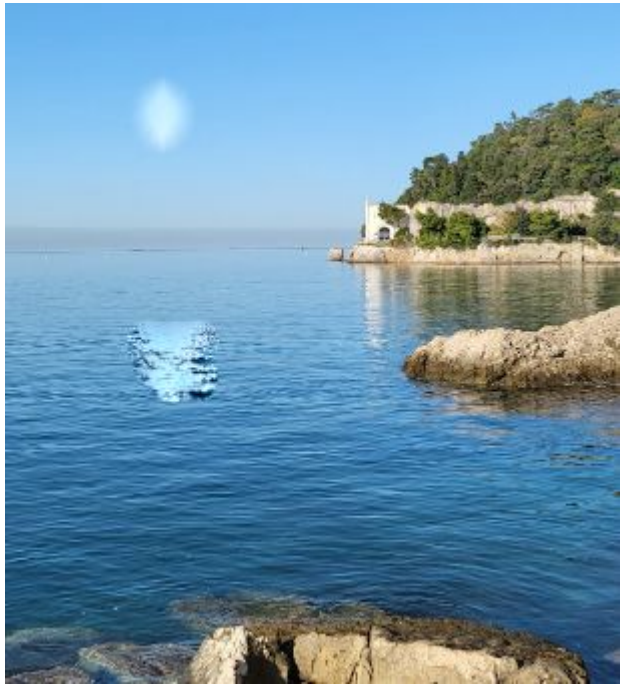
a. Basique



Naïve



Forward Backward



FISTA

I. Insertion

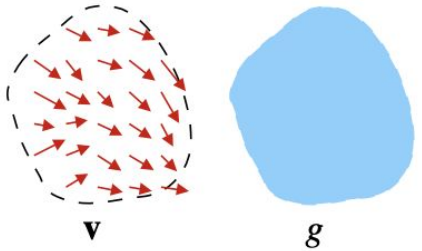
a. Basique

Forward Backward VS FISTA :

Forward Backward	FISTA
Bordures peu présentes Préserve les couleurs de l'image source en priorité Temps de compilation ~50s	Bordures peu présentes Mélange les couleurs de l'image source avec celles de l'image cible Temps de compilation ~50s, souvent plus rapide à 1s près

I. Insertion

b. Avec trous



But : copier l'objet avec des trous de l'image source dans l'image cible

Méthode : mixed seamless cloning = seamless cloning avec un champ de guidage comme suit :

$$\text{for all } \mathbf{x} \in \Omega, \mathbf{v}(\mathbf{x}) = \begin{cases} \nabla f^*(\mathbf{x}) & \text{if } |\nabla f^*(\mathbf{x})| > |\nabla g(\mathbf{x})|, \\ \nabla g(\mathbf{x}) & \text{otherwise.} \end{cases}$$

I. Insertion

b. Avec trous

Source

Cible

Masque utilisé

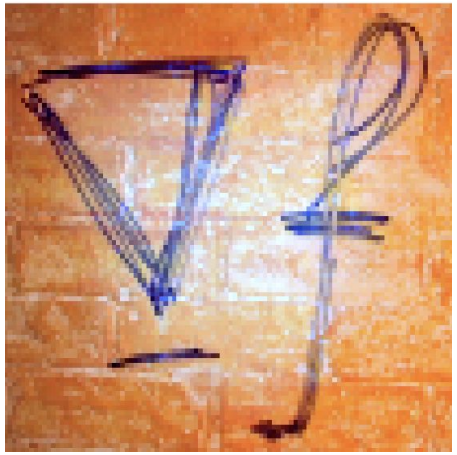
Masque élaboré

I. Insertion

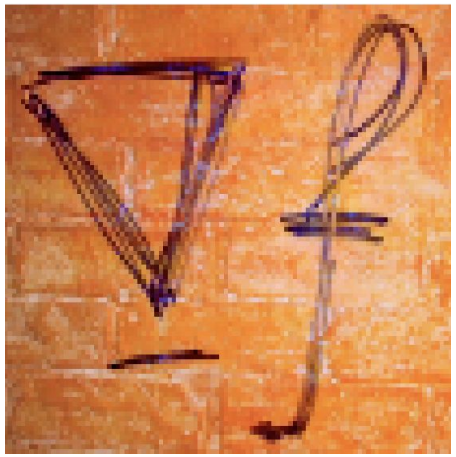
b. Avec trous

Mixed seamless cloning

FB

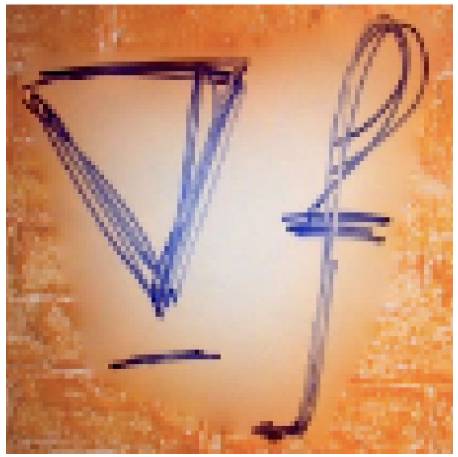


FISTA

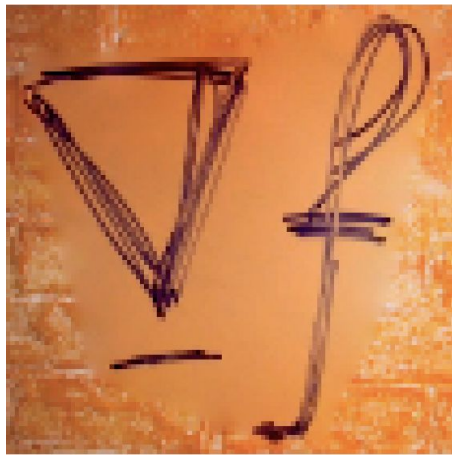


Seamless cloning

FB



FISTA



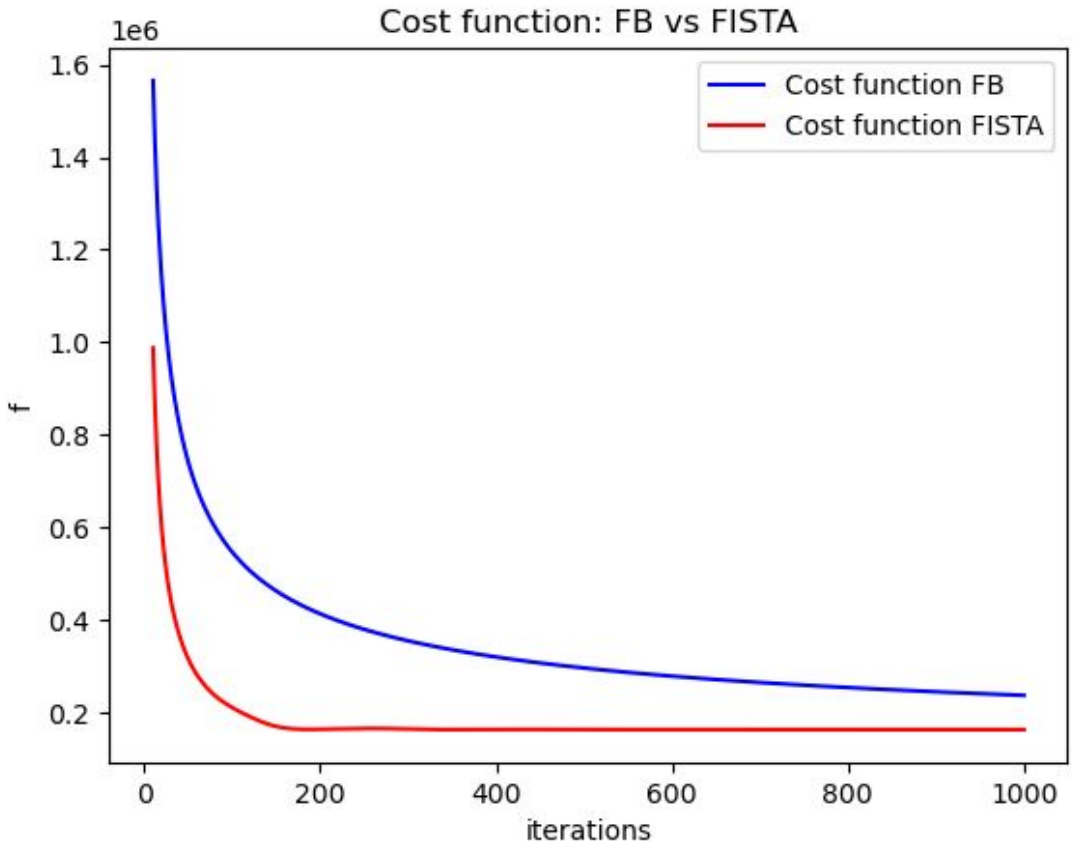
* step = 1/8, alpha = 1, 1000 itérations

I. Insertion

b. Avec trous

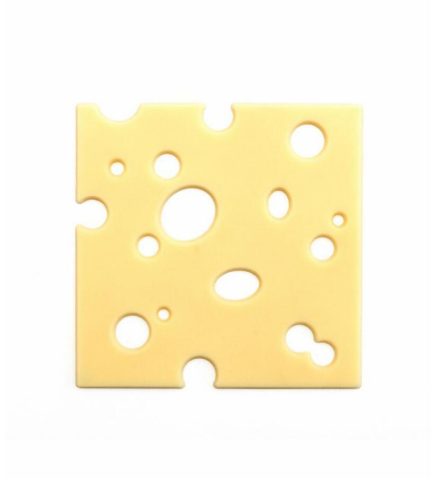
On utilise l'algorithme FISTA utilisé en TP en modifiant alpha :

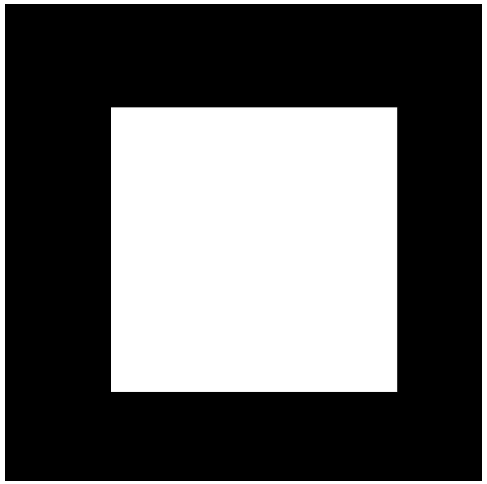
$$\alpha_{k+1} = \frac{1 + \sqrt{1 + 4\alpha_k^2}}{2}$$



I. Insertion

b. Avec trous





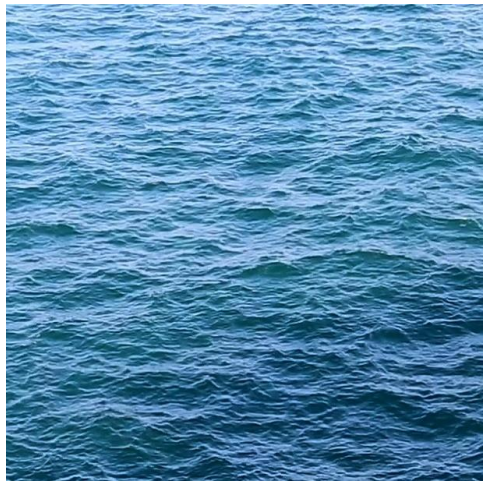
FB

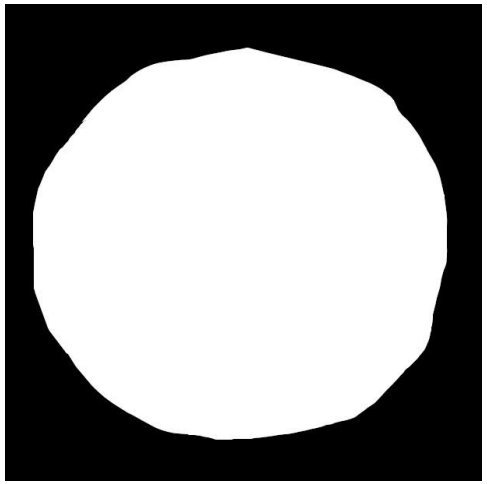
FISTA

* step = 1/8, alpha = 1, 1000 itérations

I. Insertion

b. Avec trous





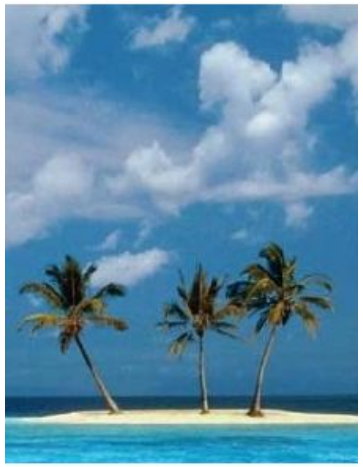
FB

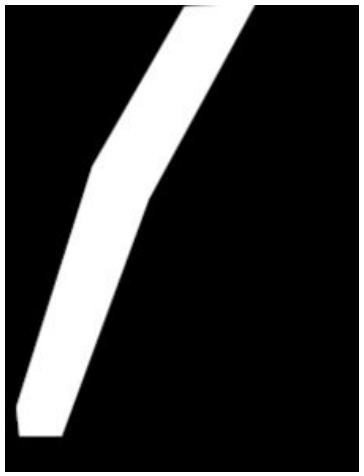
FISTA

* step = 1/8, alpha = 1, 1000 itérations

I. Insertion

b. Avec trous





FB

FISTA

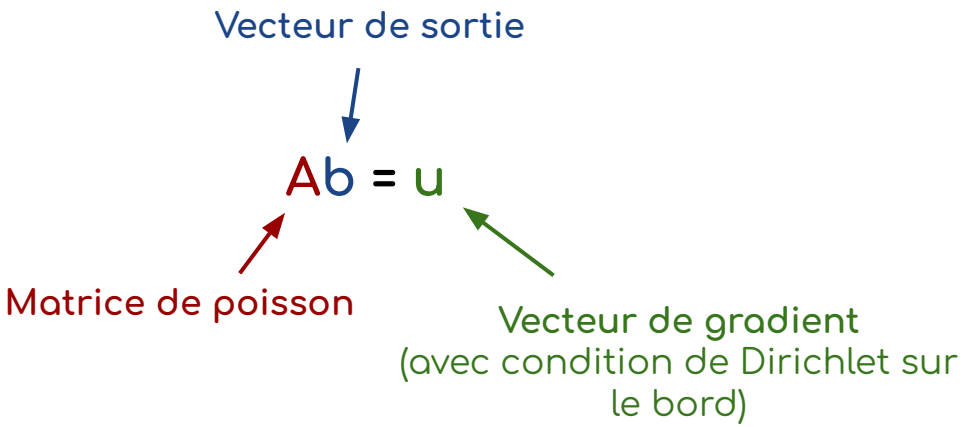
* step = 1/8, alpha = 1, 1000 itérations

I. Modification d’image

a. Couleur

Résolution du système de Poisson

La résolution de l'équation de Poisson peut être interprétée comme un problème de minimisation.



Objectif: Calculer la fonction dont le gradient est le plus proche, en norme L2, d'un champ vectoriel de guidage.

I. Modification d'image

a. Couleur

Résolution du système de Poisson

N :
Nombre de pixels non-nuls du masque

$$\underbrace{\begin{pmatrix} 4 & -1 & 0 & \cdots & -1 \\ -1 & 4 & -1 & 0 & \cdots \\ \cdots & \ddots & \ddots & \ddots & \cdots \\ \cdots & 0 & -1 & 4 & -1 \\ -1 & \cdots & 0 & -1 & 4 \end{pmatrix}}_{N} \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ \vdots \\ b_N \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ \vdots \\ \vdots \\ u_N \end{pmatrix}$$

Schéma à 5 points de la méthode des **différences finies**

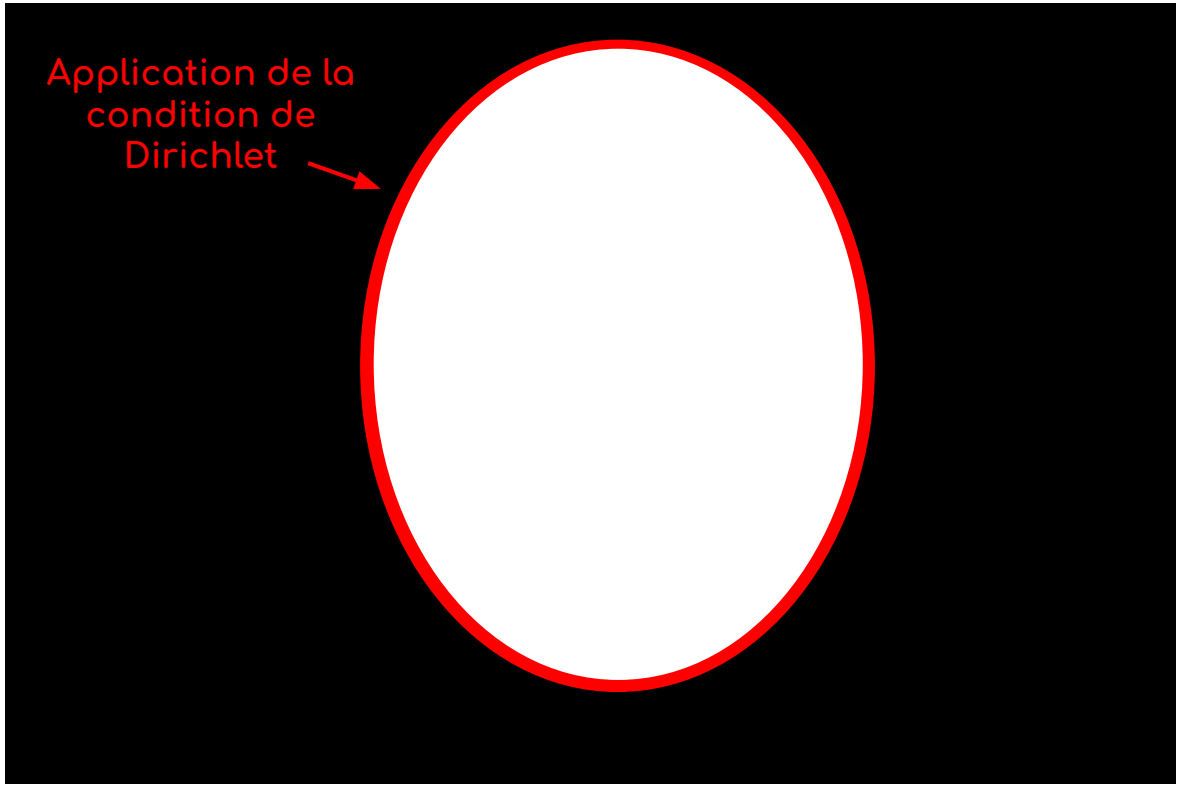
Discrétisation au second ordre de l'équation de **Poisson**

I. Modification d’image

a. Couleur

Condition de Dirichlet

La condition de Dirichlet assure une stabilité lors de l'édition d'images à proximité des limites.



I. Modification d’image

a. Couleur

Obtention du vecteur de gradient

Le **gradient** d’un pixel du masque est obtenu en additionnant les valeurs de ses pixels voisins si ceux-ci appartiennent au masque.

$$Gradient_{pixel} = \sum_{i \in Voisinsdupixel} 1_{voisins \in masque} \times (Valeurpixelvoisin)$$

$$u_i^k = Dirichlet - Gradient_{pixel} + \sum_{i \in Voisinsdupixel} 1_{voisins \in masque} \times (Valeurpixel)$$

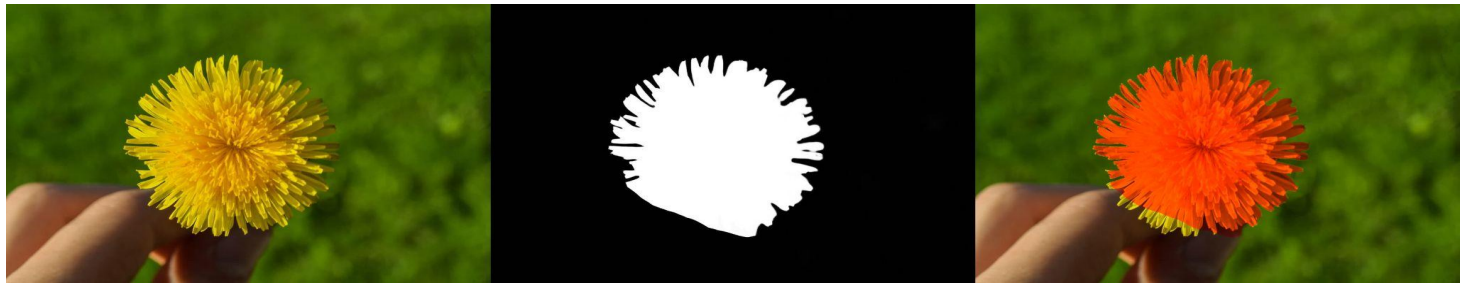
Le **vecteur b** pour chaque canal pour un pixel de l’image est obtenu en ajoutant à la condition de Dirichlet la valeur du pixel par rapport au nombre de voisins appartenant au masque et en lui soustrayant le gradient du pixel.

I. Modification d'image

a. Couleur

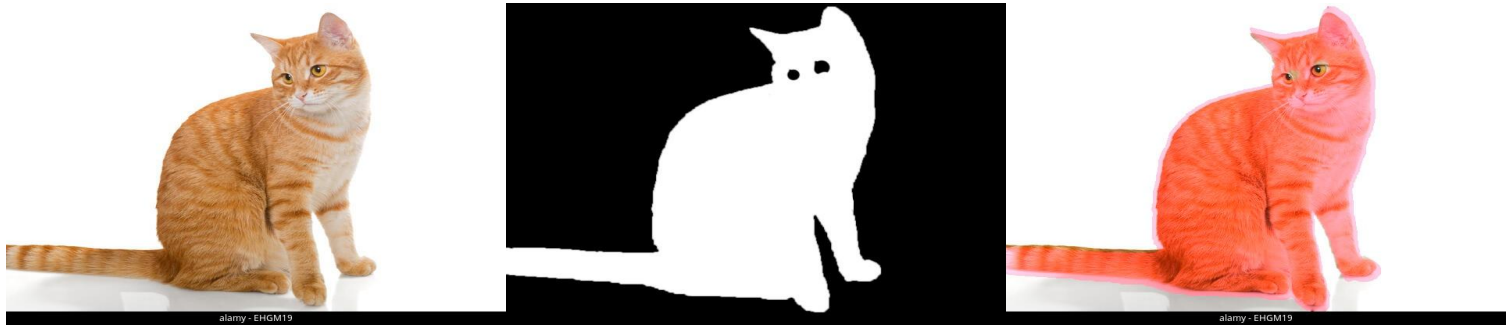
Résultats

R : 1.5 G : 0.5 B : 0.5



R : 2 G : 0.25 B : 0.25

R : 1.6 G : 0.8 B : 0.9



II. Modification d’image

b. Face flattening

1. Application

L'anonymisation des visages protège la vie privée des individus et garantit le respect des règles de confidentialité

- En **sciences sociales** et en **recherche clinique** : Anonymisation des participants aux études.
- Entraînement des **algorithmes de reconnaissance faciale** : Anonymisation des bases de données.
- Diffusion d'images provenant de caméras de **surveillance publiques** : Protection de la vie privée des citoyens.

Polémiques : En 2021, la police a utilisé ClearView AI pour traquer les criminels.
Préoccupations dans la population sur le droit de la police d'utiliser des données contenant des personnes à des fins de surveillance criminelle.

→ Problèmes **éthiques**, **juridiques** et **scientifiques**.
→ Risque d’utilisation abusive des bases de données.

Comment anonymiser les données suffisamment tout en préservant leur utilité ?

II. Modification d’image

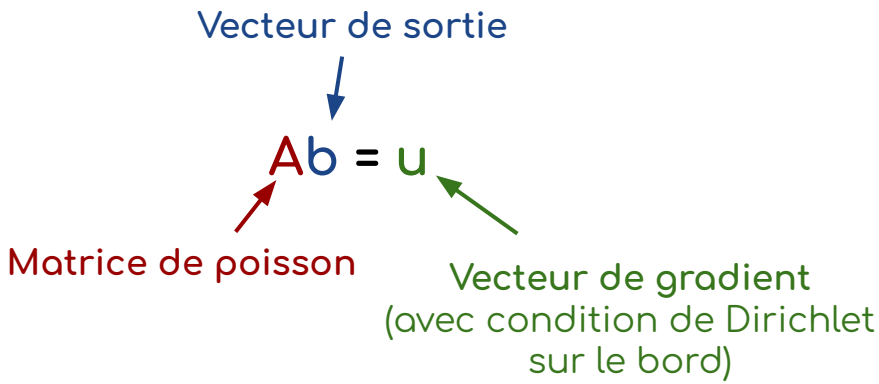
b. Face flattening

2. Résolution du système de Poisson

Les visages fournissent une représentation directe de l'identité des êtres humains.
→ Cela en fait l’un des types d'informations les plus **complexes**.

Pour **préserver les caractéristiques faciales** des données, on décide de préserver les yeux, les lèvres et le nez de l’image originale.

Le **face flattening** est réalisé en résolvant le **système de Poisson** décrit précédemment :



II. Modification d'image

b. Face flattening

3. Obtention du vecteur du gradient

Le **gradient** d'un pixel du masque est obtenue additionnant les valeurs de ses pixels voisins si ceux-ci appartiennent au masque et au contour de l'image.

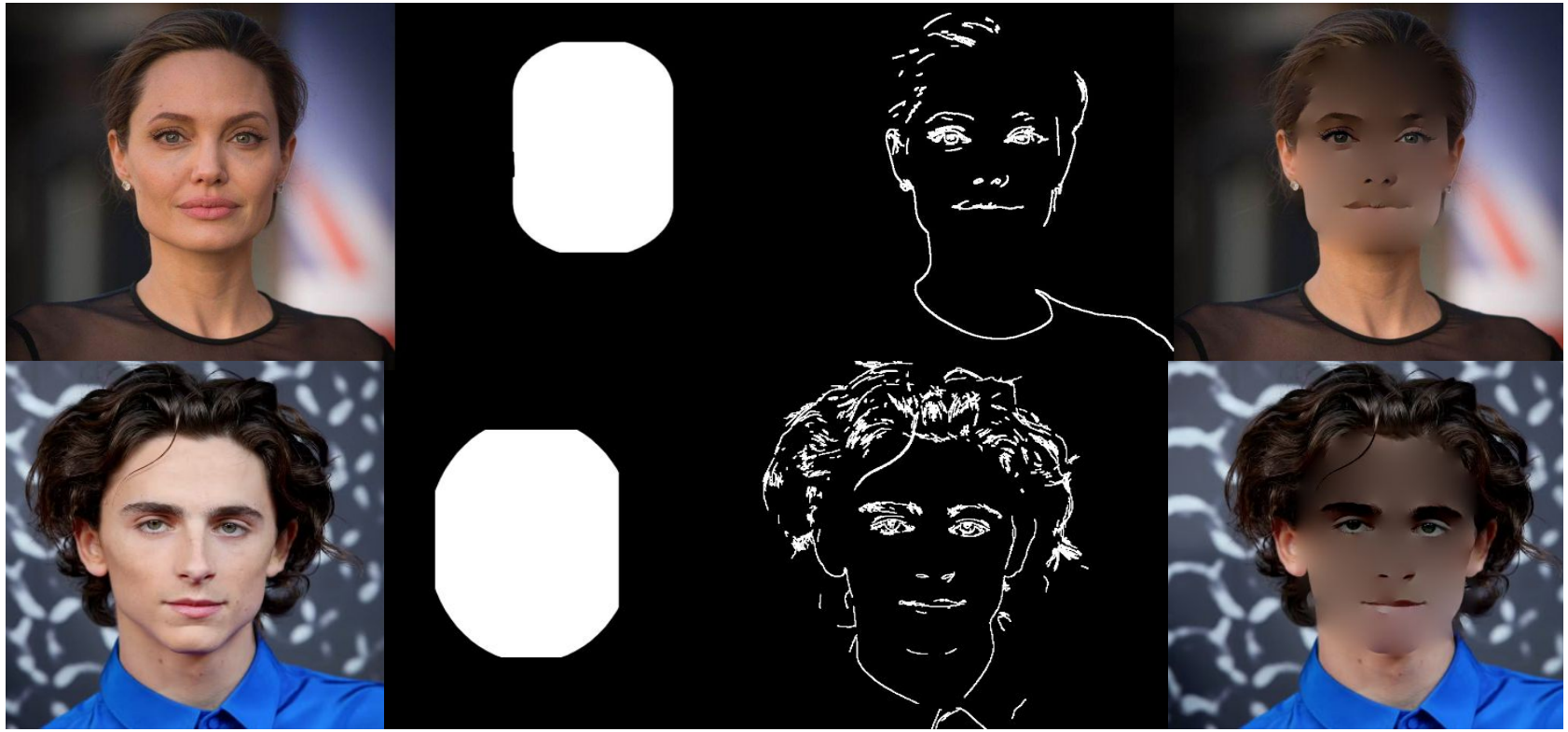
$$\text{Gradient}_{\text{pixel}} = \sum_{i \in \text{Voisins du pixel}} \mathbb{1}_{\text{voisin } \in \text{masque}} \, \mathbb{1}_{\text{voisin } \in \text{contour}} \times (\text{Valeur du pixel} - \text{Valeur du voisin})$$

II. Modification d’image

b. Face flattening

3. Résultats et limites

Meilleurs résultats :



II. Modification d’image

b. Face flattening

3. Résultats et limites

✓ Lunettes

✗ Barbe

✗ Texture de peau et rides