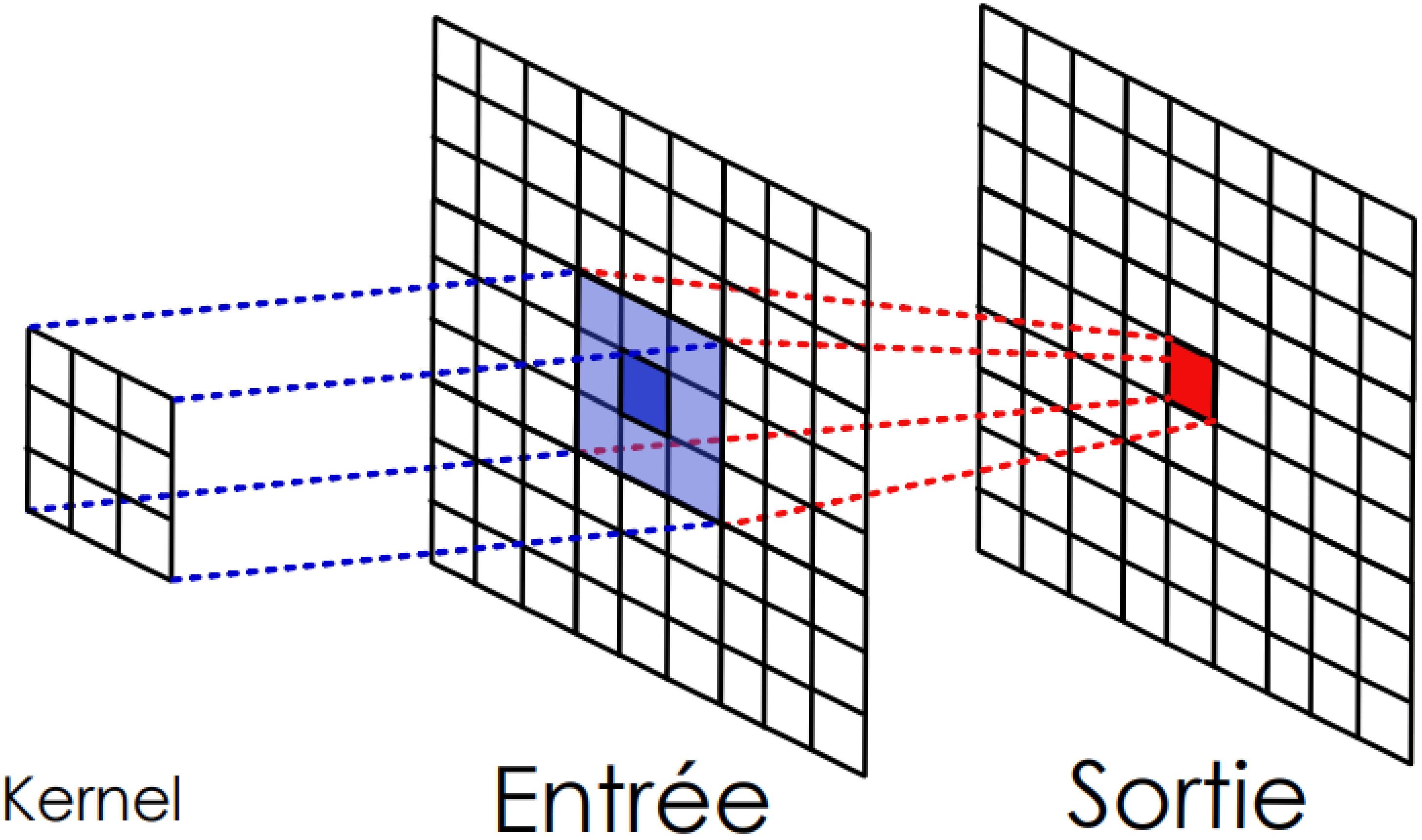
Les matrices de convolution

et leurs applications dans le cadre de la stéganographie

Par Paul Vié, Joseph Servigne, Albert Tomasi, Thibault Poux, Lukas Tabouri

1. Matrice de convolution

Qu’est qu’une matrice de convolution ?



Qu’est qu’une matrice de convolution ?

La valeur du pixel de sortie en (x,y) s’obtient par la formule suivante

$$O(x,y) =$$
$$K(0,0) \times I(x-1,y-1) + K(0,1) \times I(x,y-1) + K(0,2) \times I(x+1,y-1) +$$
$$K(1,0) \times I(x-1,y) + K(1,1) \times I(x,y) + K(1,2) \times I(x+1,y) +$$
$$K(2,0) \times I(x-1,y+1) + K(2,1) \times I(x,y+1) + K(2,2) \times I(x+1,y+1)$$

Qu'est qu'une matrice de convolution ?

La valeur du pixel de sortie en (x,y) s'obtient par la formule suivante

$$
\begin{aligned}
O(x,y) = {} & \\
& K(0,0) \times I(x-1,y-1) + K(0,1) \times I(x,y-1) + K(0,2) \times I(x+1,y-1) + \\
& K(1,0) \times I(x-1,y) + K(1,1) \times I(x,y) + K(1,2) \times I(x+1,y) + \\
& K(2,0) \times I(x-1,y+1) + K(2,1) \times I(x,y+1) + K(2,2) \times I(x+1,y+1)
\end{aligned}
$$

Que l'on notera de cette manière par la suite :

$$O(x,y) = \sum_{i=0}^{2} \sum_{j=0}^{2} K(i,j) \times I(x-1+j, y-1+i)$$

Qu’est qu’une matrice de convolution ?

$$O(x,y) = \sum_{i=0}^{2} \sum_{j=0}^{2} K(i,j) \times I(x-1+j, y-1+i)$$

Détection de contours

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Qu'est qu'une matrice de convolution ?

$$O(x,y) = \sum_{i=0}^{2} \sum_{j=0}^{2} K(i,j) \times I(x-1+j, y-1+i)$$

Détection de contours

Amélioration de la netteté

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Qu’est qu’une matrice de convolution ?

$$O(x,y) = \sum_{i=0}^{2} \sum_{j=0}^{2} K(i,j) \times I(x-1+j, y-1+i)$$

Détection de contours

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Amélioration de la netteté

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Flou de gauss

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

1.a problème aux bords

Problème aux bords

Les noyaux de convolution demandent souvent des valeurs à l’extérieur des limites de l'image, il existe plusieurs méthodes pour pallier ce problème.

Extension

Problème aux bords

Les noyaux de convolution demandent souvent des valeurs à l'extérieur des limites de l'image, il existe plusieurs méthodes pour pallier ce problème.

Extension

Enroullage (Wrap)

Problème aux bords

Les noyaux de convolution demandent souvent des valeurs à l’extérieur des limites de l'image, il existe plusieurs méthodes pour pallier ce problème.

Extension

Enroullage (Wrap)

Miroir

Il existe aussi le Crop et le Kernel Crop que l’on ne détaillera pas ici

2. Traitement de l’image

Traitement de l'image

Détection de contours

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Traitement de l'image

Détection de contours

Amélioration de la netteté

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Traitement de l'image

Détection de contours

Amélioration de la netteté

Flou de gauss

K = $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$

2.a Filtre de Sobel

Filtre de Sobel

L'opérateur calcul le gradient de l'intensité de chaque pixel

Le gradient d'une fonction f de dimension n c'écrit :

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Filtre de Sobel

L’opérateur calcul le gradient de l'intensité de chaque pixel

Le gradient d’une fonction f de dimension n c’écrit :

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Soit A l’image source et Gx et Gy les approximations des dérivées horizontales et verticales

$$\mathbf{G_x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \mathbf{A} \quad \text{et} \quad \mathbf{G_y} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathbf{A}$$

Filtre de Sobel

L'opérateur calcul le gradient de l'intensité de chaque pixel

Le gradient d'une fonction f de dimension n c'écrit :

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Soit A l'image source et Gx et Gy les approximations des dérivées horizontales et verticales

$$\mathbf{G_x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \mathbf{A} \quad \text{et} \quad \mathbf{G_y} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathbf{A}$$

$$\mathbf{G} = \sqrt{\mathbf{G_x}^2 + \mathbf{G_y}^2}$$

On obtient donc G la norme du gradient en chaque point

Filtre de Sobel

Exemple :

Entrée A

$$\mathbf{G_x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \mathbf{A} \quad \text{et} \quad \mathbf{G_y} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathbf{A}$$

$$\mathbf{G} = \sqrt{\mathbf{G_x}^2 + \mathbf{G_y}^2}$$

3.b Inversion de la convolution

Méthode naïve

Soit la matrice de convolution K et A l’image l’entrée

$$K = \begin{matrix} 1 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 1 \end{matrix}$$

$Kbis = -K$, et $Kbis(centre) = \sum_{i=0}^{2} \sum_{j=0}^{2} K(i,j)$

Ainsi $Kbis = \begin{matrix} -1 & -2 & -1 \\ -2 & 15 & -2 \\ -1 & -2 & -1 \end{matrix}$

c’est de cette manière que fonctionne les filtres pour améliorer la netteté

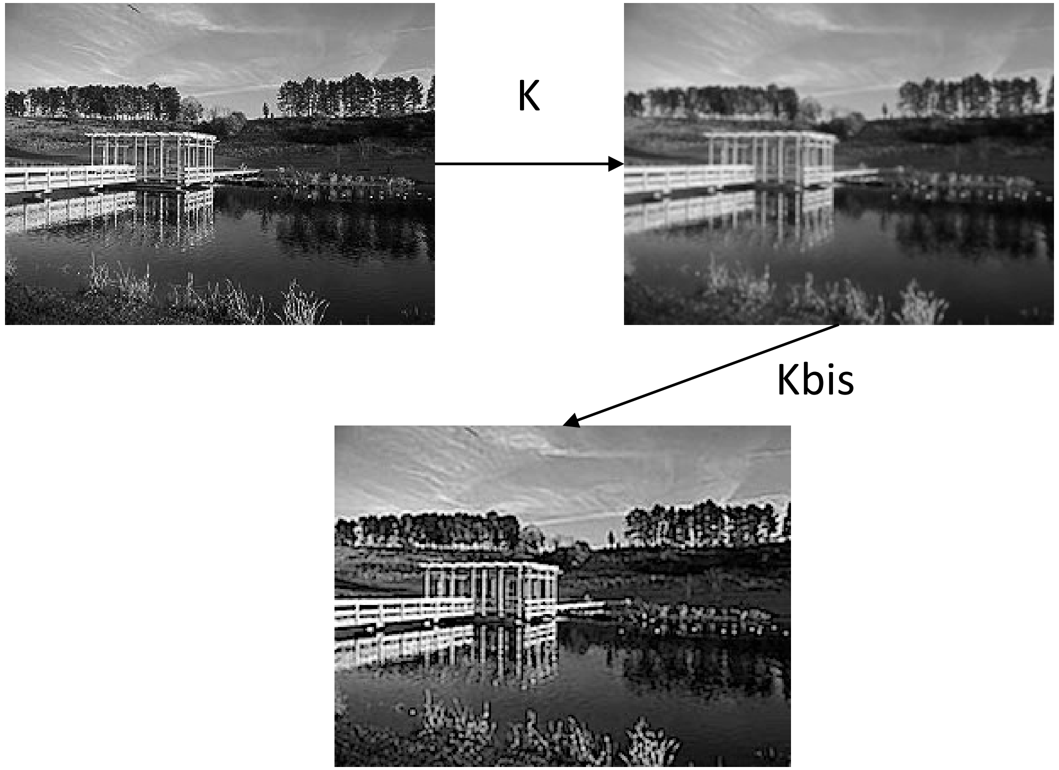
Méthode naïve

Soit la matrice de convolution K et A l'image l'entrée

$$K = \begin{matrix} 1 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 1 \end{matrix}$$

$Kbis = -K$, et $Kbis(centre) = \sum_{i=0}^{2} \sum_{j=0}^{2} K(i,j)$

$$\text{Ainsi } Kbis = \begin{matrix} -1 & -2 & -1 \\ -2 & 15 & -2 \\ -1 & -2 & -1 \end{matrix}$$

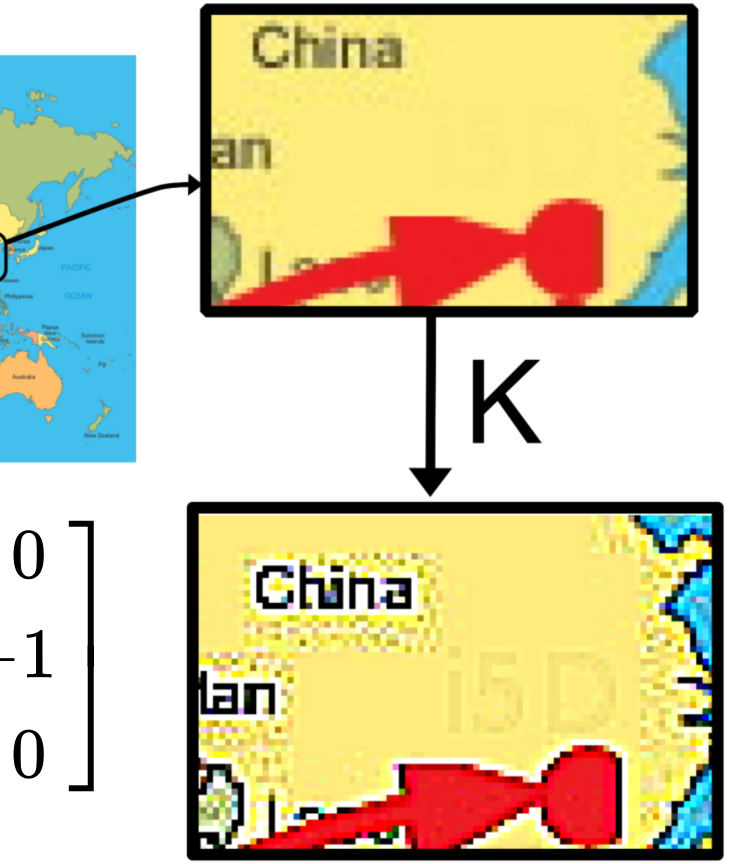


3. Stéganographie

Qu’est que la stéganographie ?

Matrice de convolution pour la stéganographie

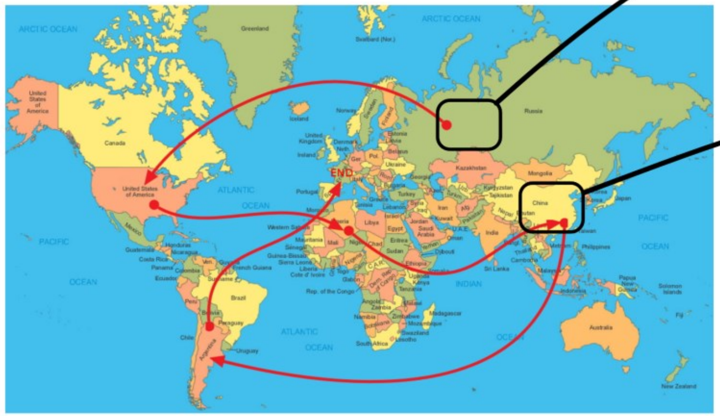
Issu d'un challenge de cybersécurité

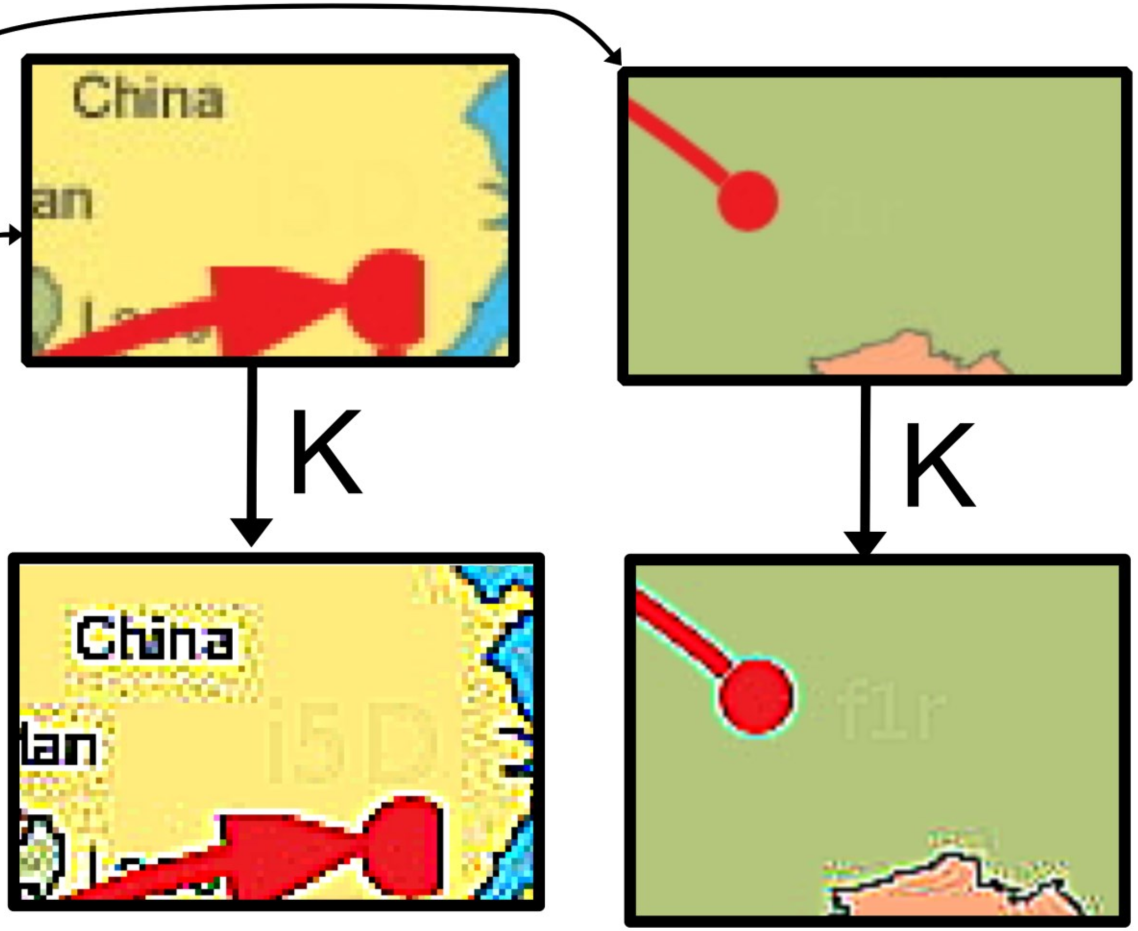


$$K = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Matrice de convolution pour la stéganographie

Issu d'un challenge de cybersécurité





$$K = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Matrice de convolution pour la stéganographie

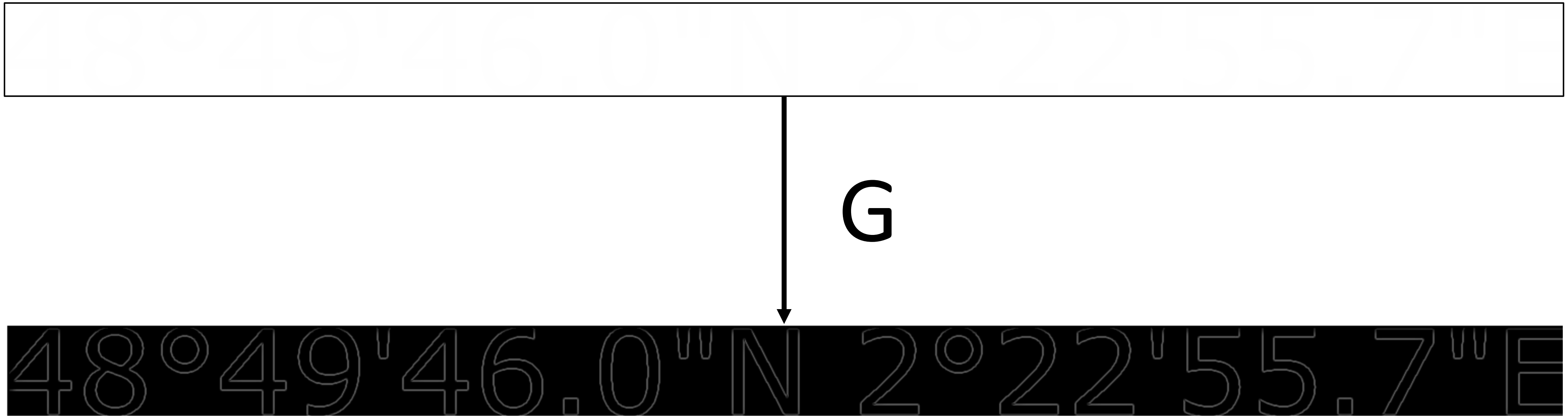
Par le filtre de Sobel

La couleur du texte est (254,254,254), on le voit donc pas

48°49'46.0"N 2°22'55.7"

Matrice de convolution pour la stéganographie

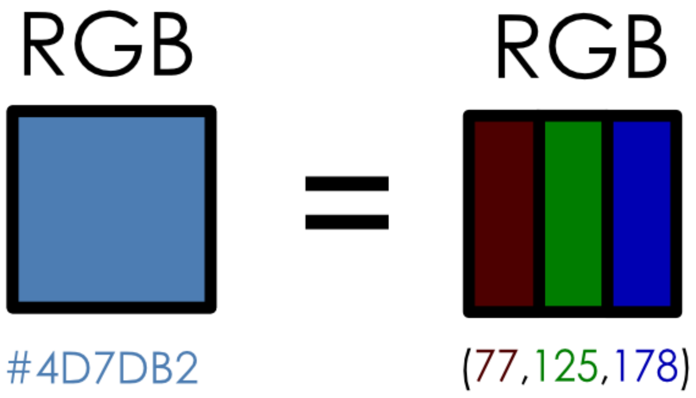
Par le filtre de Sobel



3.a LSB

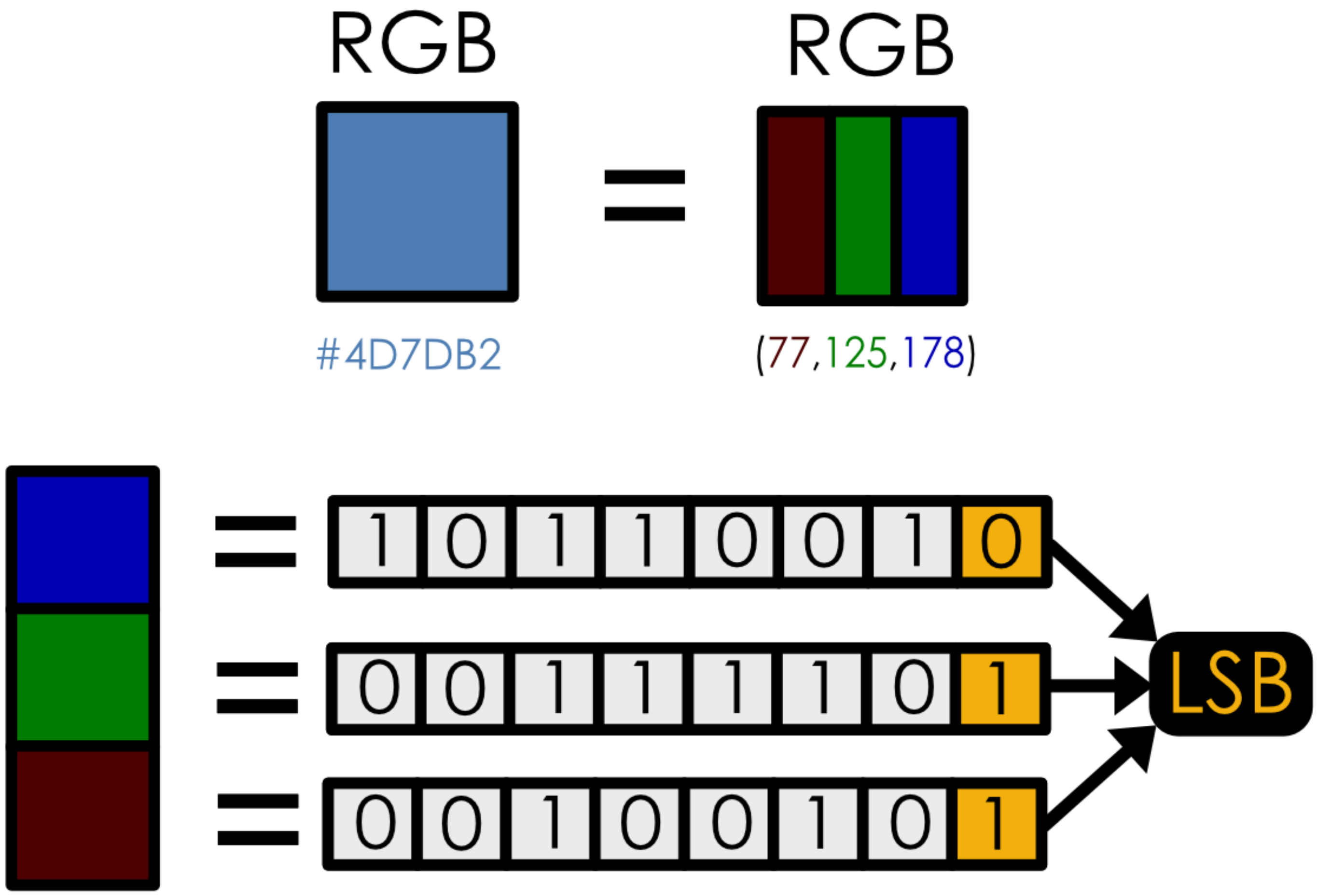
LSB

Pour bits de poids faible (Least Significant Bit en anglais)



LSB

Pour bits de poids faible (Least Significant Bit en anglais)

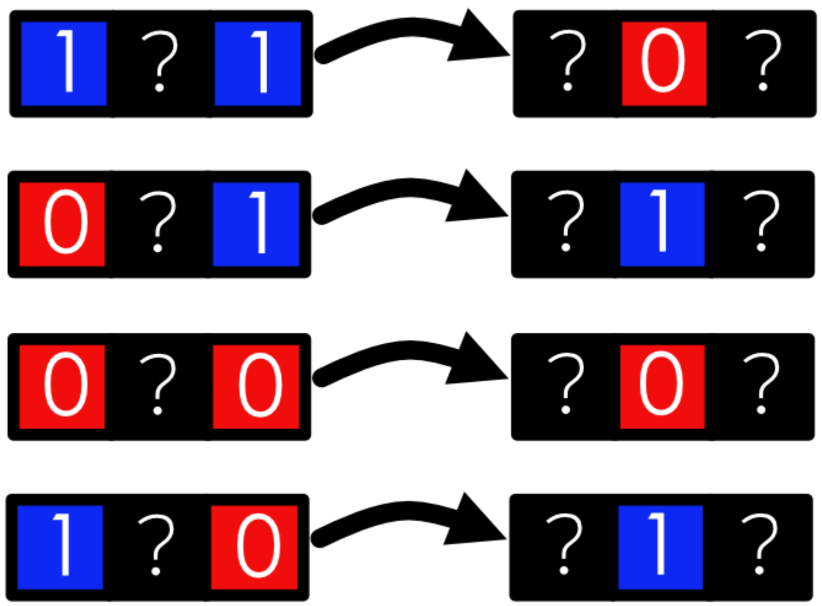


3.a.a Les matrices de convolution et le LSB

LSB + Convolution

$$K = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

LSB



~ Xor

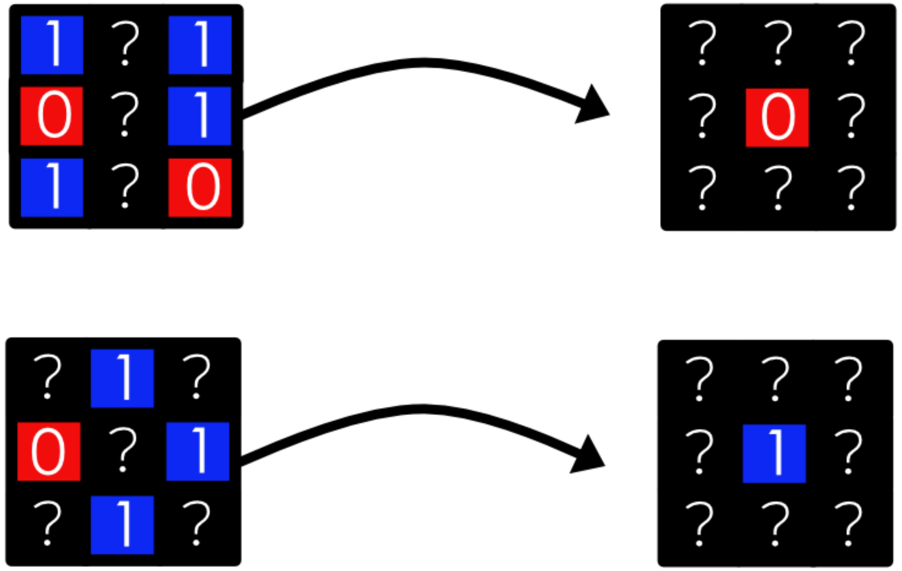
LSB + Convolution

LSB De l'input

LSB De l'output

$$K = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$



3.c Blind Kernel

Méthode en Blind Kernel

Soit A l’image l’entrée, B l’image de sortie

On cherche la matrice K qui a servi pour faire
$Conv(A,K)=B$

Méthode en Blind Kernel

Soit A l’image l’entrée, B l’image de sortie

On cherche la matrice K qui a servi pour faire

$Conv(A,K)=B$

A l’aide du domaine de Fourier, il suffit d’écrire

$$K = F^{-1}\left(\frac{F(B)}{F(A)}\right)$$

Pour obtenir une approximation de la matrice de convolution utilisé

Méthode en Blind Kernel

Pourquoi la transformée de Fourier ?

Méthode en Blind Kernel

Pourquoi la transformée de Fourier ?

$$F(A * K) = F(A)F(K) = F(B)$$

Et Donc que $F(K) = F(B)/F(A)$

Ainsi

$$K = F^{-1}(\frac{F(B)}{F(A)})$$

Méthode en Blind Kernel

Soit A l’image l’entrée, B l’image de sortie

On cherche la matrice qui a servi pour faire

$$Conv(A,K)=B$$

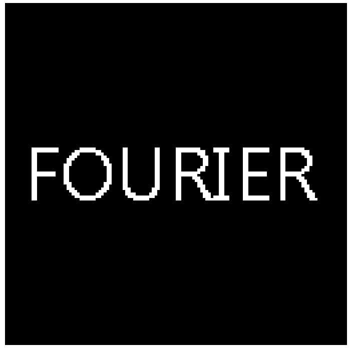
il suffit donc de réécrire l’équation précédente en python

```
result = fft.fftshift(fft.irfft2((fft.rfft2(b)/fft.rfft2(A))))
```

Pour obtenir une approximation de la matrice de convolution utilisé

A

K



B

Méthode en Blind Kernel

Soit A l’image l’entrée, B l’image de sortie

On cherche la matrice qui a servi pour faire

$$Conv(A,K)=B$$

il suffit donc de réécrire l’équation précédente en python

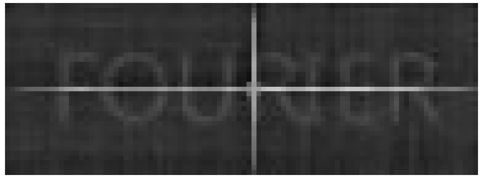
```
result = fft.fftshift(fft.irfft2((fft.rfft2(b)/fft.rfft2(A))))
```

Pour obtenir une approximation de la matrice de convolution utilisé

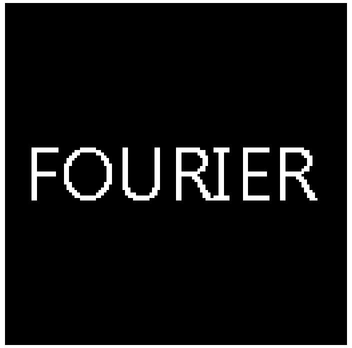
A

B

résultat



K



Conclusion

Bibliographie

TRANSFORMEE DE FOURIER ET APPLICATIONS Edoardo Provenzi

Inverse Image Filtering with Conjugate Gradient Presented at BOOM 2003 by Zhengyun Zhang

Inverse Kernels for Fast Spatial Deconvolution by Li Xu, Xin Tao, and Jiaya Jia

Types of Convolution Kernels : Simplified, by Prakhar Ganesh, published in Towards Data Science

Liens externes :

https://docs.gimp.org/2.8/en/plug-in-convmatrix.html
https://mathinfo.alwaysdata.net/2016/11/filtres-de-convolution/