UE Image L3

TD-TP4 (2020)

TD-TP 4 : Analyse et traitement d'image

Exercice 1

5. Charger l' image « shapes.jpg ».

6.Transformer cette image couleur en niveaux de gris.

7.Générer l' image négative de l' image couleur.

8.Peut-on calculer le nombre d'objets dans l' image ? si oui alors comment ? faites-le.

Exercice 2

Vous avez à votre disposition 6 images de documents.

2. Calculer l' histogramme de projection. Afficher-les et commenter.

3. Évaluer le nombre de lignes.

4. Reconstituer les documents incomplets.

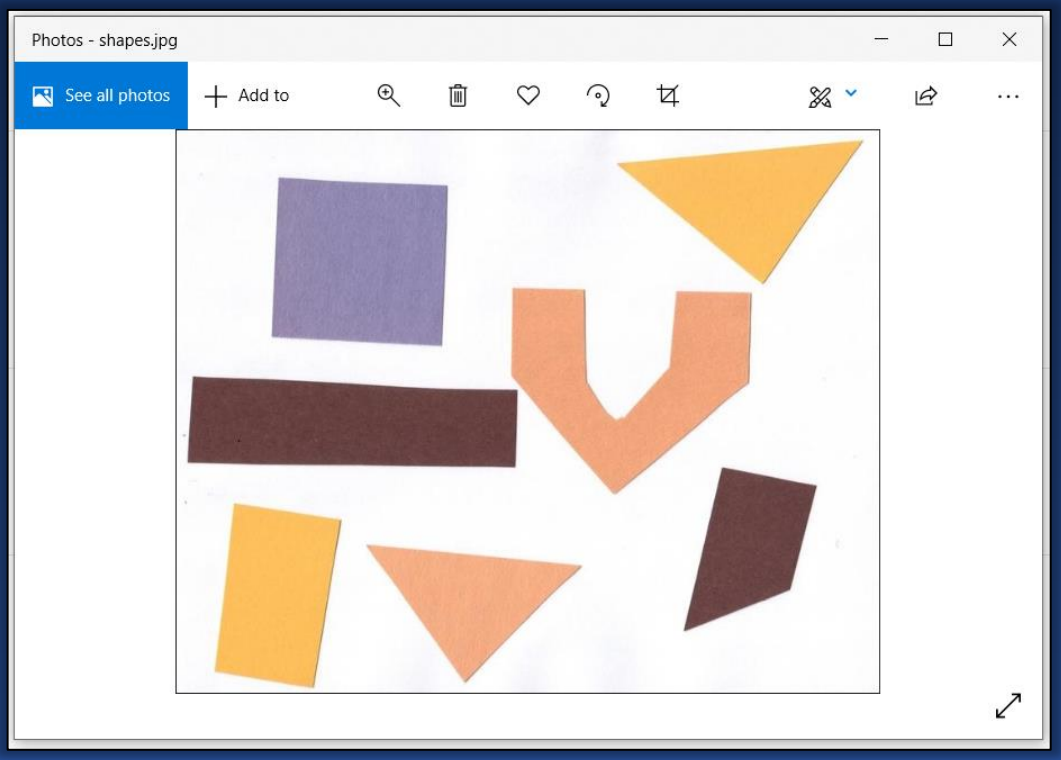
Exercice 3

3. Charger une image en niveaux de gris

4. Implémenter une fonction qui permette de calculer le produit de convolution avec un filtre de votre choix.

Exercice 1

(1) Charger l'image « shapes.jpg »




```java
import java.io.File;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ImageIcon;

import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
public class Td4Exercice1Question1 {
	public static void showImage(BufferedImage bufferedImage) throws IOException {
		JFrame frame = new JFrame("Td4Exercice1Question1");
		ImageIcon imageIcon = new ImageIcon(bufferedImage);
		JLabel jLabel = new JLabel(imageIcon);

		frame.getContentPane().add(jLabel);
		frame.pack();
		frame.setVisible(true);
	}

	public static BufferedImage loadImage(File path) {
		BufferedImage img = null;

		try {
			img = ImageIO.read(path);
		} catch (IOException e) {
			e.printStackTrace();
		}

		return img;
	}

	public static void main(String[] args) {
		File path = new File("Test_Images" + File.separator + "shapes.jpg");
		BufferedImage img = loadImage(path);

		try {
			showImage(img);
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

Exercice 1

(2) Transformer cette image couleur en niveaux de gris.

```java
import java.io.File;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ImageIcon;

import javax.imageio.ImageIO;

import java.awt.Color;
import java.awt.image.BufferedImage;
public class Td4Exercice1Question2 {
	public static void showImage(BufferedImage bufferedImage) throws IOException {
		JFrame frame = new JFrame("Td4Exercice1Question2");
		ImageIcon imageIcon = new ImageIcon(bufferedImage);
		JLabel jLabel = new JLabel(imageIcon);

		frame.getContentPane().add(jLabel);
		frame.pack();
		frame.setVisible(true);
	}

	public static BufferedImage loadImage(File path) {
		BufferedImage img = null;

		try {
			img = ImageIO.read(path);
		} catch (IOException e) {
			e.printStackTrace();
		}

		return img;
	}

	public static BufferedImage transformerImageCouleurEnNiveauxDeGris(BufferedImage imageCouleur) {
		int nombre_col = imageCouleur.getWidth();
		int nombre_lignes = imageCouleur.getHeight();

		BufferedImage g_img = new BufferedImage(nombre_col, nombre_lignes, BufferedImage.TYPE_3BYTE_BGR);

		/* Parcourir tous les pixels,
		 * pour chaque pixel extraire les valeurs r, g, b.
		 * La valeur du niveau de gris : la moyenne (r,g,b)/3
		 */

		for(int x = 0 ; x < nombre_col ; x++) {
			for(int y = 0 ; y < nombre_lignes ; y++) {
				int b = imageCouleur.getRGB(x, y) & 0xff;
				int g = (imageCouleur.getRGB(x,y) >> 8) & 0xff;
				int r = (imageCouleur.getRGB(x,y) >> 16) & 0xff;

				int level = (b + g + r) / 3;
				int level_c = new Color(level, level, level).getRGB();
				g_img.setRGB(x, y, level_c);
			}
		}
		return g_img;
	}

	public static void main(String[] args) {
		File path = new File("Test_Images" + File.separator + "shapes.jpg");
		BufferedImage imageCouleur = loadImage(path);
		BufferedImage imageEnNiveauxDeGris = transformerImageCouleurEnNiveauxDeGris(imageCouleur);

		try {
			showImage(imageEnNiveauxDeGris);
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

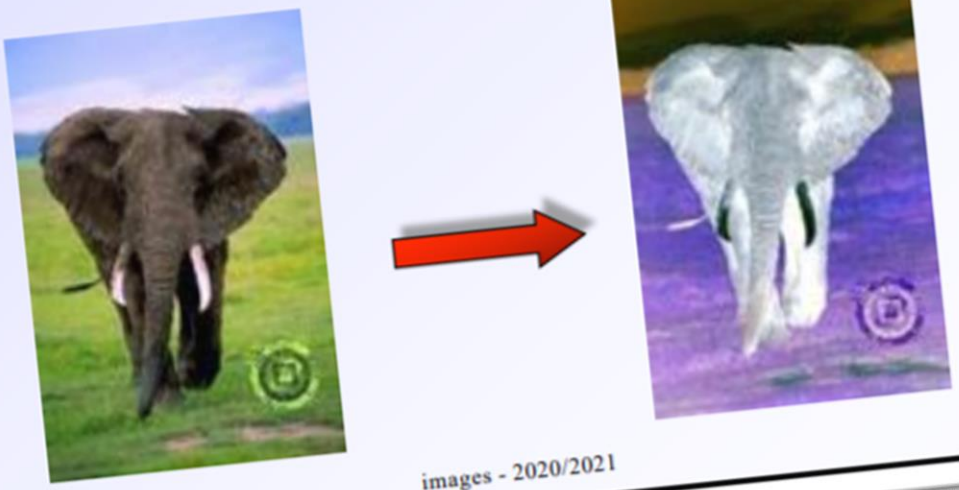
Exercice 1

(3) Générer l′ image négative de l′ image couleur.

Image négative

- Sur une image couleur

s(r,g,b) devient de couleur (255-r, 255-g, 255-b)



```java
import java.io.File;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ImageIcon;

import javax.imageio.ImageIO;

import java.awt.Color;
import java.awt.image.BufferedImage;
public class Td4Exercice1Question3 {
	public static void showImage(BufferedImage bufferedImage) throws IOException {
		JFrame frame = new JFrame("Td4Exercice1Question2");
		ImageIcon imageIcon = new ImageIcon(bufferedImage);
		JLabel jLabel = new JLabel(imageIcon);

		frame.getContentPane().add(jLabel);
		frame.pack();
		frame.setVisible(true);
	}

	public static BufferedImage loadImage(File path) {
		BufferedImage img = null;

		try {
			img = ImageIO.read(path);
		} catch (IOException e) {
			e.printStackTrace();
		}

		return img;
	}

	public static BufferedImage genererImageNegativeDeImageCouleur(BufferedImage imageCouleur) {
		int nombre_col = imageCouleur.getWidth();
		int nombre_lignes = imageCouleur.getHeight();

		BufferedImage new_img = new BufferedImage(nombre_col, nombre_lignes, BufferedImage.TYPE_3BYTE_BGR);

		for(int x = 0 ; x < nombre_col ; x++) {
			for(int y = 0 ; y < nombre_lignes ; y++) {
				int b = imageCouleur.getRGB(x, y) & 0xff;
				int g = (imageCouleur.getRGB(x,y) >> 8) & 0xff;
				int r = (imageCouleur.getRGB(x,y) >> 16) & 0xff;

				int new_color = new Color(255 - r, 255 - g, 255 - b).getRGB();
				new_img.setRGB(x, y, new_color);
			}
		}
		return new_img;
	}

	public static void main(String[] args) {
		File path = new File("Test_Images" + File.separator + "shapes.jpg");
		BufferedImage imageCouleur = loadImage(path);
		BufferedImage imageEnNiveauxDeGris = genererImageNegativeDeImageCouleur(imageCouleur);

		try {
			showImage(imageEnNiveauxDeGris);
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

Exercice 1

(4) Peut-on calculer le nombre d'objets dans l' image ? si oui alors comment ? faites-le.

...

Exercice 3

(1) Charger une image en niveaux de gris

```java
import java.io.File;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.ImageIcon;

import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
public class Td4Exercice3Question1 {
	public static void showImage(BufferedImage bufferedImage) throws IOException {
		JFrame frame = new JFrame("Td4Exercice3Question1");
		ImageIcon imageIcon = new ImageIcon(bufferedImage);
		JLabel jLabel = new JLabel(imageIcon);

		frame.getContentPane().add(jLabel);
		frame.pack();
		frame.setVisible(true);
	}

	public static BufferedImage loadImage(File path) {
		BufferedImage img = null;

		try {
			img = ImageIO.read(path);
		} catch (IOException e) {
			e.printStackTrace();
		}

		return img;
	}

	public static void main(String[] args) {
		File path = new File("Test_Images" + File.separator + "shapesGray.jpg");
		BufferedImage img = loadImage(path);

		try {
			showImage(img);
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

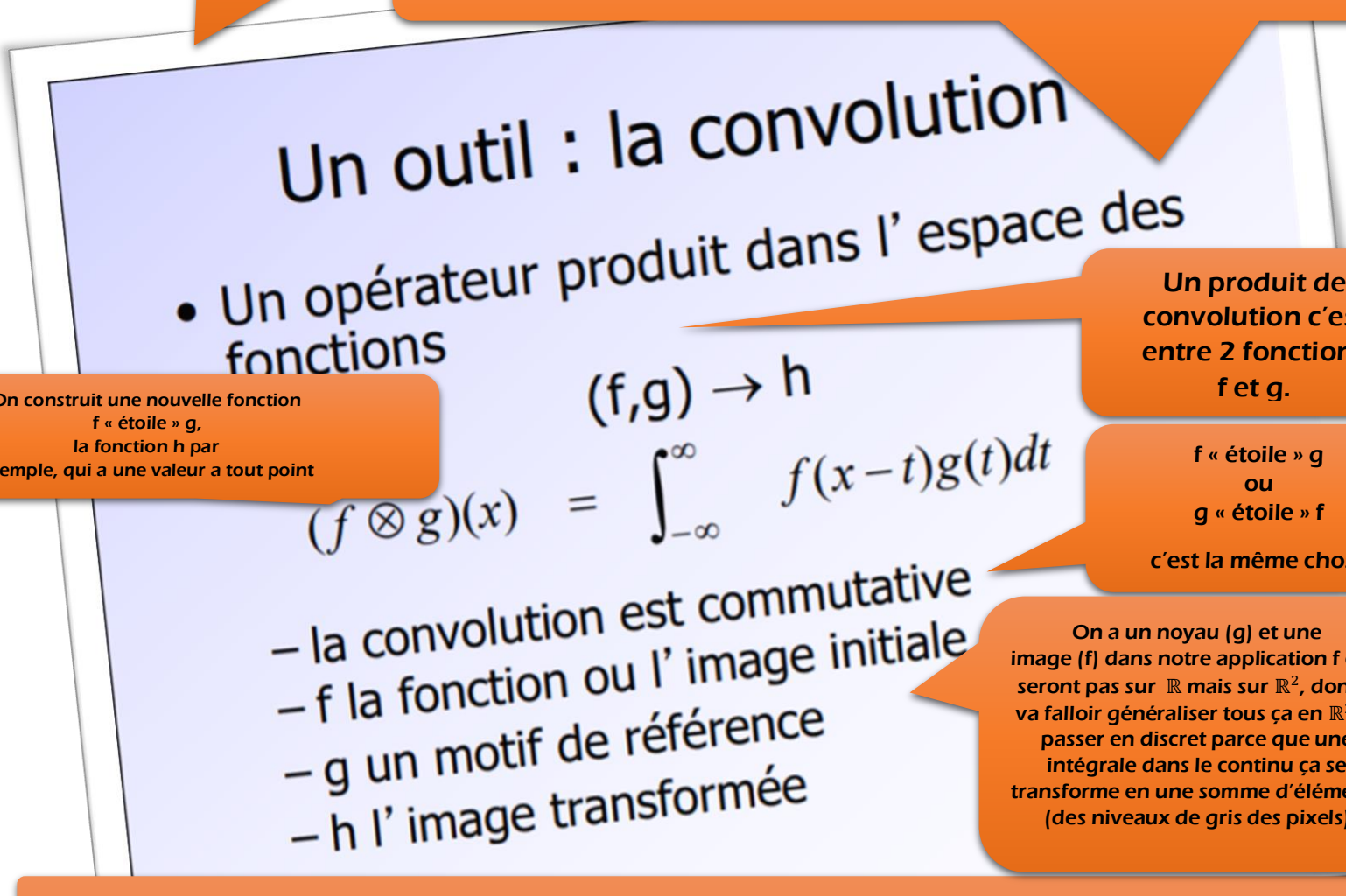
Exercice 3

(2) Implémenter une fonction qui permette de calculer le produit de convolution avec un filtre de votre choix.

..

Un outil mathématique : le produit de convolution

Le produit de convolution est défini sur l'espace des fonctions



Un produit de convolution c'est entre 2 fonctions f et g.

On construit une nouvelle fonction f « étoile » g, la fonction h par exemple, qui a une valeur a tout point

f « étoile » g
ou
g « étoile » f

c'est la même chose

On a un noyau (g) et une image (f) dans notre application f et g seront pas sur $\mathbb{R}$ mais sur $\mathbb{R}^2$, donc il va falloir généraliser tous ça en $\mathbb{R}^2$ et passer en discret parce que une intégrale dans le continu ça se transforme en une somme d'éléments (des niveaux de gris des pixels)

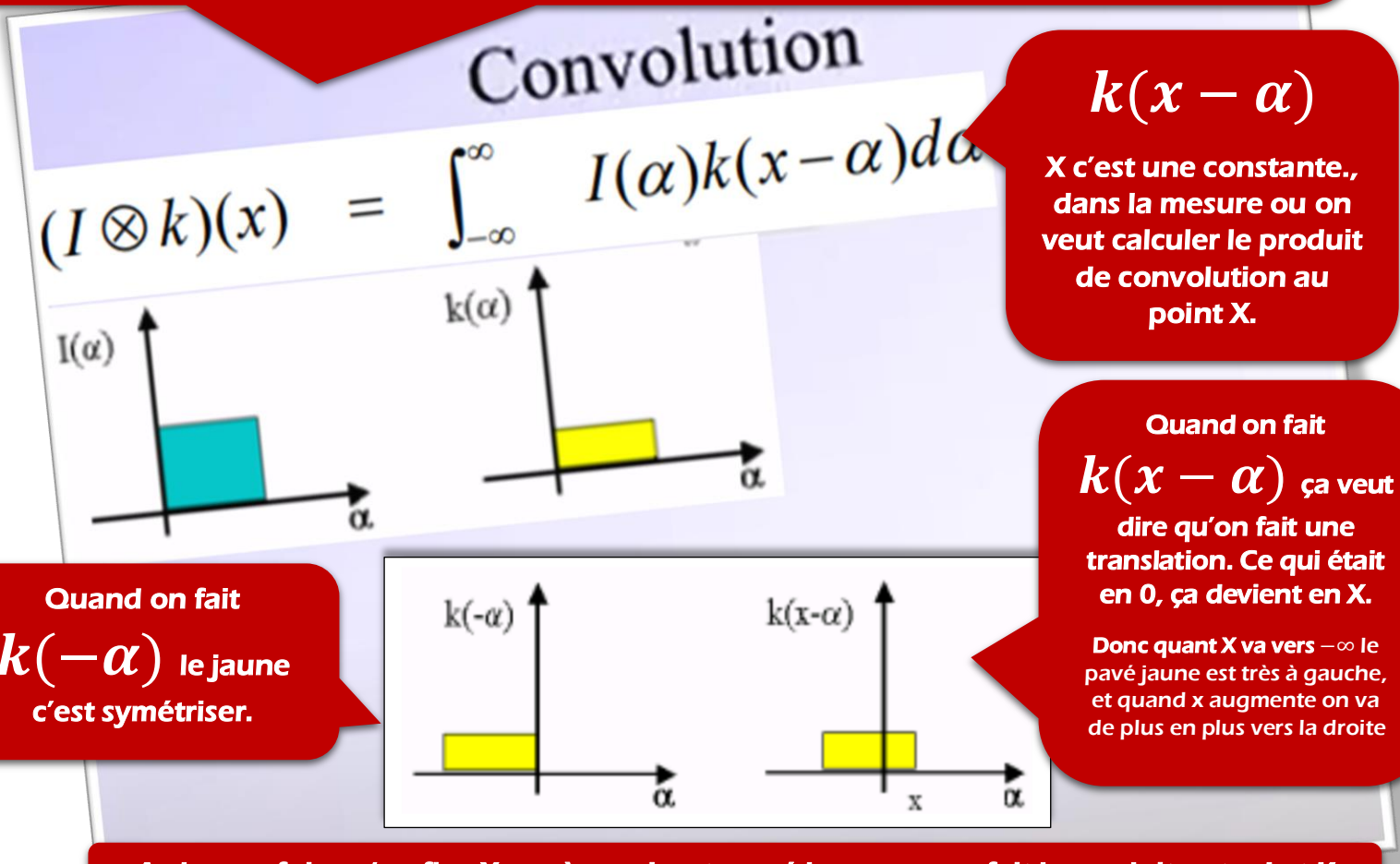
On général quand on définit le produit de convolution, ce sont des fonctions qui sont définie sur $\mathbb{R}$, ça veut dire que f est définit sur $\mathbb{R}$ entre $-\infty$ et ∞, g est définit sur $\mathbb{R}$ entre $-\infty$ et ∞, et on fait le produit de f par g : donc la variable d'intégration c'est t, on utilise pas f en t mais en x-t. x est la valeur sur laquelle on est en train de travailler,
donc on fait le produit de f et de g mais pas au même point : la fonction f est centré en x et on la symétrise en -t

On intègre ca sur $-\infty$ et ∞.
Si on a une intégration entre $-\infty$ et ∞, il faut se poser la question de la convergence, mais nous notre objectif est que f soit une image (et une image c'est toujours fini, sa a un certain nombre de lignes et de colonnes), donc l'intégral il sera pas entre $-\infty$ et ∞.

g c'est notre petit noyau, ça veut dire qu'on va s'intéresser au voisinage de x, ce qui veut dire que le support de g sera tous petit, donc notre intégrale il sera très simple, donc on aura pas à se poser la question de convergence..

I c'est l'image (Image), k c'est le noyau (Kernel).

ce qui fait faire c'est le produit de la partie de I (la partie en vert), par la partie en jaune. Le problème : la partie en jaune c'est $k(\alpha)$, mais dans la formule il est écrit $k(x-\alpha)$



$k(x-\alpha)$

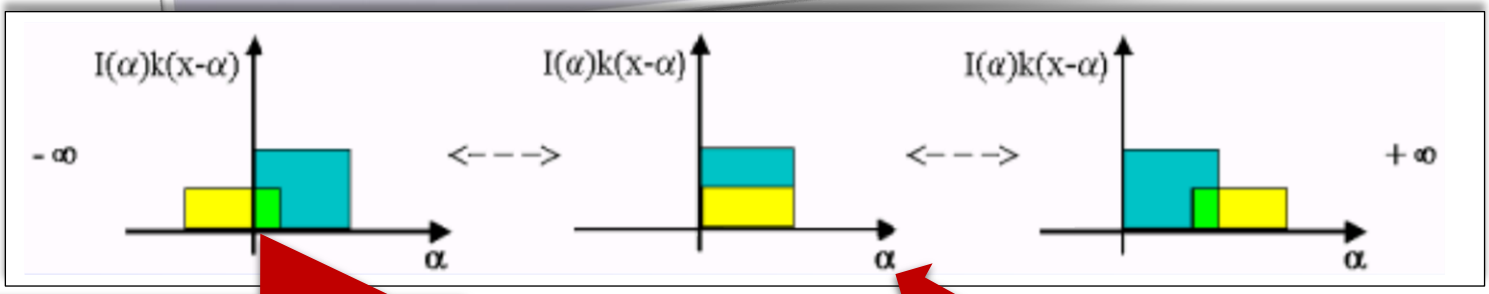
X c'est une constante., dans la mesure ou on veut calculer le produit de convolution au point X.

Quand on fait $k(-\alpha)$ le jaune c'est symétriser.

Quand on fait $k(x-\alpha)$ ça veut dire qu'on fait une translation. Ce qui était en 0, ça devient en X.

Donc quant X va vers $-\infty$ le pavé jaune est très à gauche, et quand x augmente on va de plus en plus vers la droite

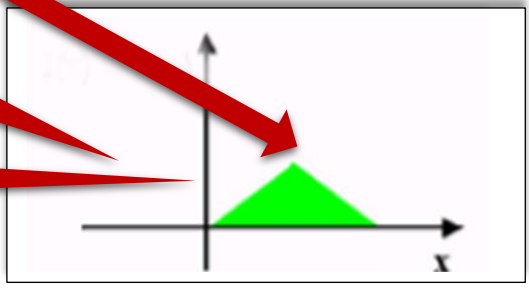
A chaque fois qu'on fixe X, après avoir retourné le noyau, on fait le produit entre I et K



Le produit = l'air dans la partie verte.

Quand x est négatif le produit est nul

Quand x devient positif, le produit augmente



Quand on fait

$k(-\alpha)$ le jaune c'est symétriser.

Convolution discrète

$$f \otimes g(i,j) = \sum_{\alpha=-\infty}^{+\infty} \sum_{\beta=-\infty}^{+\infty} f(i-\alpha, j-\beta) \cdot g(\alpha,\beta)$$

- Une image a un support borné et est définie par une matrice de valeurs $(f_{ij})_{ij}$ où i est l'indice de ligne et j indice de colonne
- Si le support de la fonction de référence est un carré de côté 2p+1 centré à l' origine

$$f \otimes g(i,j) = \sum_{\alpha=-p}^{+p} \sum_{\beta=-p}^{+p} f_{i-\alpha,j-\beta} \cdot g(\alpha,\beta) = \sum_{\alpha=-p}^{+p} \sum_{\beta=-p}^{+p} f_{i-\alpha,j-\beta} \cdot a_{\alpha,\beta}$$

....