# AIV1 - TP01
# Classification automatique de textures cycliques par analyse du plan de Fourier

Suliac Lavenant et Antoine Nollet

07 January 2022

# Table des matières

# 1 128_a.png

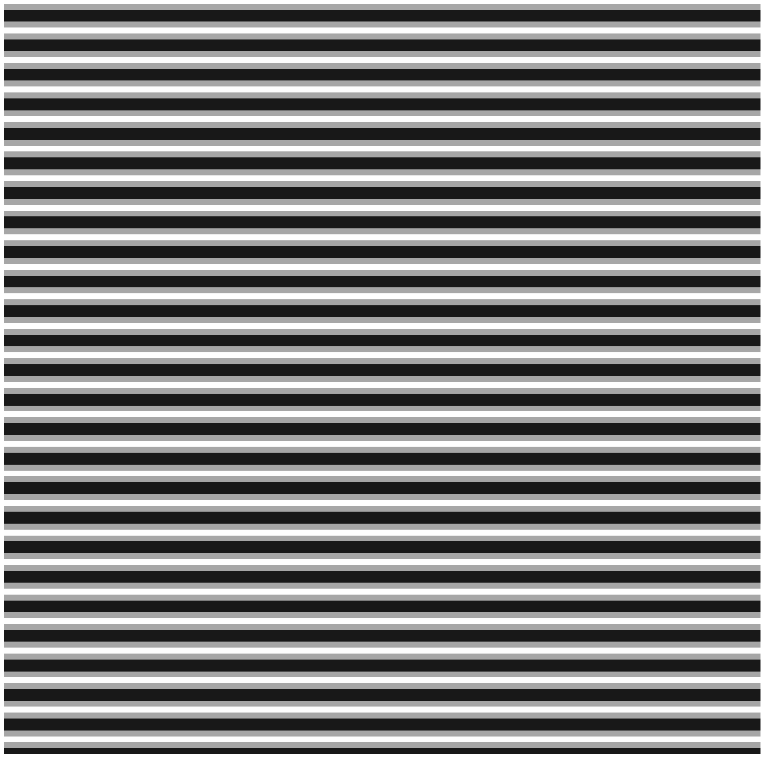


FIGURE 1 – image 128_a.png

## 1.1 Première informations de l'image

Une première analyse de cette image nous apprend qu'elle est de dimension 128 pixels par 128 pixels. La profondeur de l'image est de 8 bits, c'est-à-dire que chaque pixel a une valeur de gris allant de 0 à 255. Pour le type de données l'image est en nuance de gris.

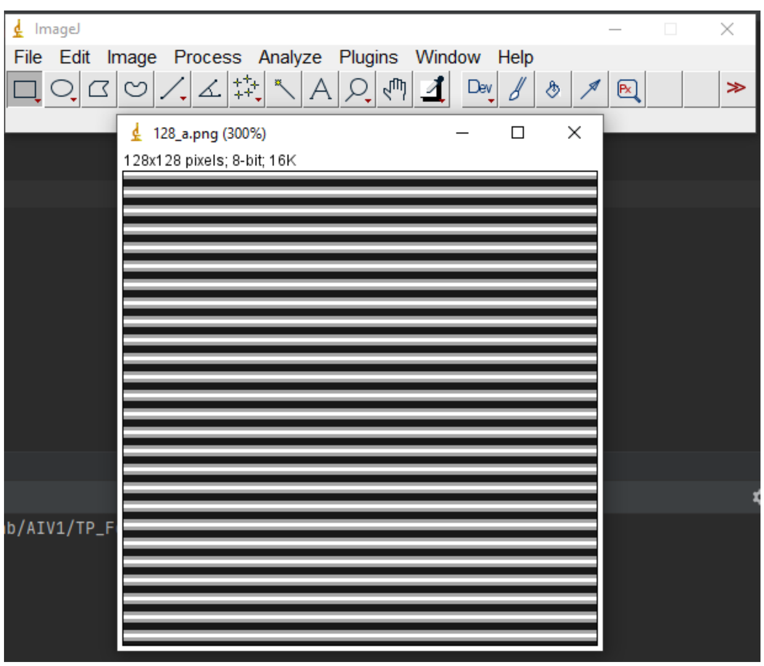



FIGURE 2 – Détails de l'image_a

## 1.2 Période en pixels du motif cyclique

img

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.9 |
| 1 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 2 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 3 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 4 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 5 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.9 |
| 6 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 7 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 8 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 9 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 10 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.9 |
| 11 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 12 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 13 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 14 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 15 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.9 |
| 16 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 17 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 18 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.09412 | 0.0 |
| 19 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.65098 | 0.6 |
| 20 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.99608 | 0.9 |

Figure 3 – Tableau numpy de l'image 128_a

En observant l'image mais aussi en observant son tableau numpy associé nous observons sur cette image un motif cyclique. Nous pouvons déduire de ce motif une période de 5 pixels.

Figure 4 – motif cyclique de 5pixel de 128_a.png

## 1.3 Discret Fourier Transform

On applique une Discret Fourier Transform sur notre image 128_a et on obtient un tableau numpy de dimension 128 par 128 (les même que l'image initiale).

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 8178.94... | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 1 | 31.87964 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 2 | 31.74361 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 3 | 31.51406 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 4 | 31.18659 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 5 | 30.75475 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 6 | 30.20973 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 7 | 29.53985 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 8 | 28.72996 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 9 | 27.76045 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 10 | 26.60601 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 11 | 25.23372 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 12 | 23.60037 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 13 | 21.64848 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 14 | 19.30018 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 15 | 16.44769 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 16 | 12.93784 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 17 | 8.54568 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 18 | 2.92738 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 19 | -4.46981 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 20 | -14.59400 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 21 | -29.21839 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 22 | -52.08703 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 23 | -92.70500 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 24 | -184.341... | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 25 | -582.061... | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 26 | 1009.73... | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |

Figure 5 – tableau numpy de la DFT réel de 128_a

En analysant ce tableau on remarque que les coefficients à partie réelle non nulle ce situe verticalement dans la colonne 0. Grâce au coefficient maximal de la DFT on remarque qu'elle est centrée en (0,0) alors que l'on voudrait l'avoir centrée au centre du tableau, soit en (64,64).

On peut justifier le fait que les valeurs sont sur une et même colone du fait que le motif est horizontal et identique pour chaque colonne.

Ce coefficient maximal est de valeur 8178.94917. Il s'agit de la somme de tous les coefficients du tableau.

On va donc maintenant recentrer cette DFT grace a une fonction python en décalant les valeurs obtenues dans les deux directions. On obtient maintenant un tableau bien centrer.

| | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 51 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 21.64848 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 52 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 23.60037 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 53 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 25.23372 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 54 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 26.60601 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 55 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 27.76045 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 56 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 28.72996 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 57 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 29.53985 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 58 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 30.20973 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 59 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 30.75475 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 60 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.18659 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 61 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.51406 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 62 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.74361 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 63 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.87964 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 64 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 8178.94... | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 65 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.87964 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 66 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.74361 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 67 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.51406 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 68 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 31.18659 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 69 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 30.75475 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 70 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 30.20973 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 71 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 29.53985 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 72 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 28.72996 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 73 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 27.76045 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 74 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 26.60601 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 75 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 25.23372 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 76 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 23.60037 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 77 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 21.64848 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |

Figure 6 – tableau numpy de la DFT réel de 128_a centré

On normalise notre DFT par le produit de ses dimensions car on ne voudrait pas que les degrés de présence des fréquences possèdent une valeur supérieure au nombre de pixels de l'image initiale, notre signal initial. En effet, les fréquences supérieures au produit des dimensions de l'image ne peuvent exister dans le contexte de cette image. Ainsi, nous avons des valeurs avec lesquelles travailler plus facilement.

On obtient donc un tableau normalisé :

| | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 51 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00132 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 52 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00144 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 53 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00154 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 54 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00162 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 55 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00169 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 56 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00175 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 57 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00180 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 58 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00184 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 59 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00188 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 60 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00190 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 61 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00192 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 62 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00194 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 63 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00195 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 64 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.49920 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 65 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00195 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 66 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00194 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 67 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00192 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 68 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00190 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 69 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00188 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 70 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00184 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 71 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00180 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 72 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00175 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 73 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00169 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 74 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00162 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 75 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00154 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 76 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00144 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| 77 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00132 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |

Figure 7 – tableau numpy de la DFT réel de 128_a centré normalisé

## 1.4 Module du DFT

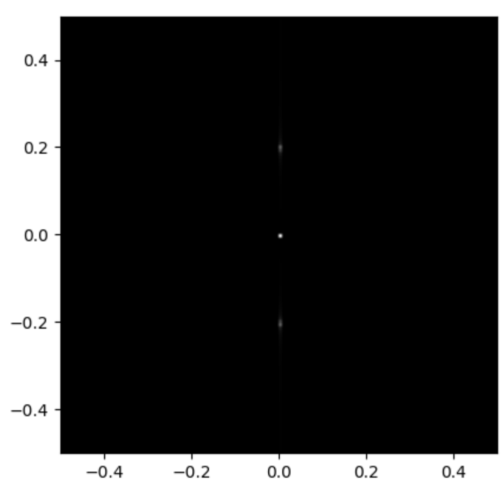


Figure 8 – module de 128_a sous la forme d'une image en niveaux de gris

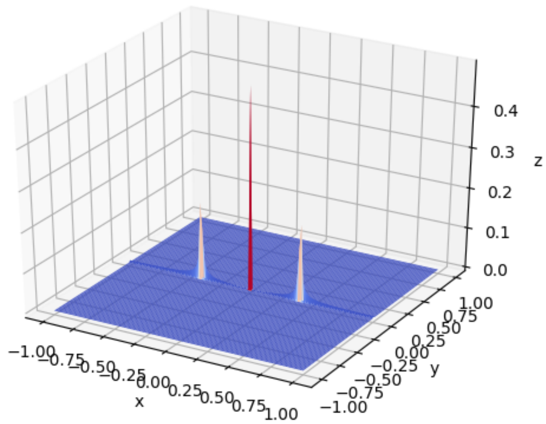



Figure 9 – module de 128_a sous la forme d'une image en niveaux de gris

Le module représente l'importance des motifs périodiques (le motif cyclique) de l'image (128_a.png) dans la direction du vecteur (0,0)–(fx,fy) et de fréquence proportionnelle à sa norme. Ainsi, dans ce cas, on constate que les 3 "pics" de notre module du DFT sont de valeurs absolues inférieurs à 0.25, alors les fréquences spatiales seront considérées comme relativement basses. Les pics sont présents verticalement, alors les perturbations sont verticales.

## 2 128_e.png

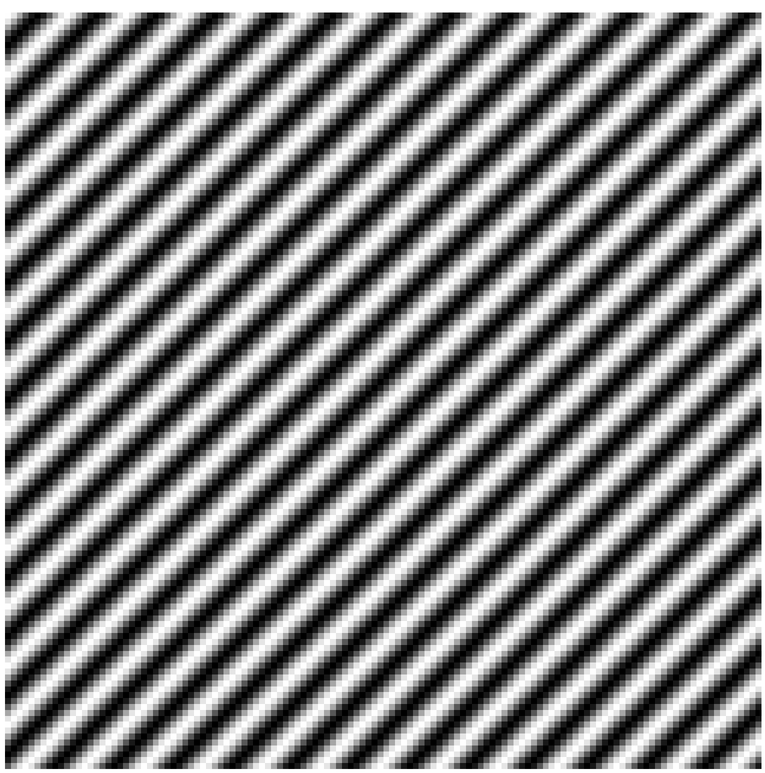


Figure 10 – image 128_e.png

## 2.1 Première informations de l'image

Une première analyse de cette image nous apprend qu'elle est de dimension 128 pixels par 128 pixels. La profondeur de l'image est de 8 bits, c'est-à-dire que chaque pixel a une valeur de gris allant de 0 à 255. L'image est au format PNG, en niveaux de gris.

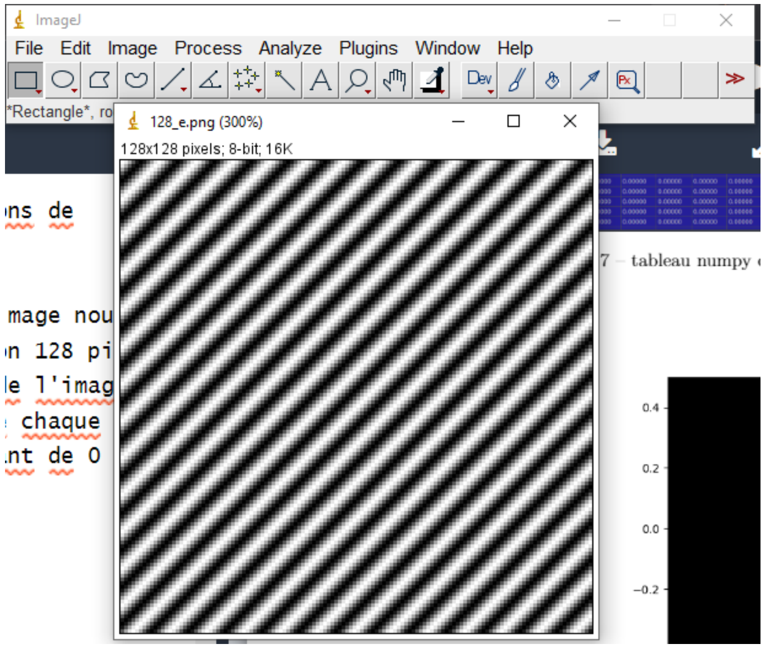
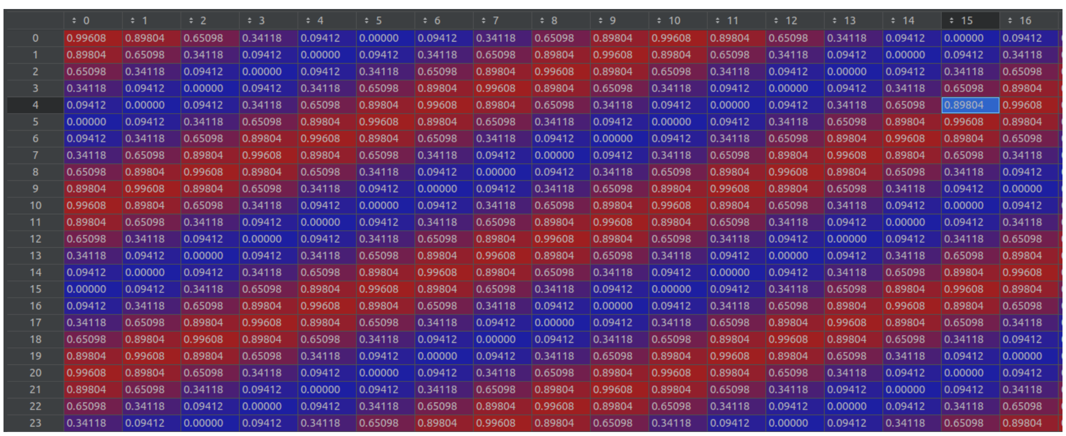



FIGURE 11 – Détails de l'image_e

## 2.2 Période en pixels du motif cyclique



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 |
| 1 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 |
| 2 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 |
| 3 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 |
| 4 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 |
| 5 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 |
| 6 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 |
| 7 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 |
| 8 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 |
| 9 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 |
| 10 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 |
| 11 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 |
| 12 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 |
| 13 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 |
| 14 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 |
| 15 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 |
| 16 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 |
| 17 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 |
| 18 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 |
| 19 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 |
| 20 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 |
| 21 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 |
| 22 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 |
| 23 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 | 0.99608 | 0.89804 | 0.65098 | 0.34118 | 0.09412 | 0.00000 | 0.09412 | 0.34118 | 0.65098 | 0.89804 |

FIGURE 12 – Tableau numpy de l'image 128_e

En observant l'image mais aussi en observant son tableau numpy associé nous observons sur cette image un motif cyclique, ce motif est diagonal. Ainsi, il y a donc pas un mais deux motifs cycliques : un en x et un en y. Grâce à ImageJ, on sait que ces motifs ont des périodes de 10 pixels

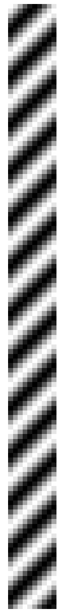


FIGURE 13 – motif cyclique en x de 128_e.png

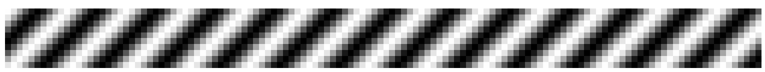
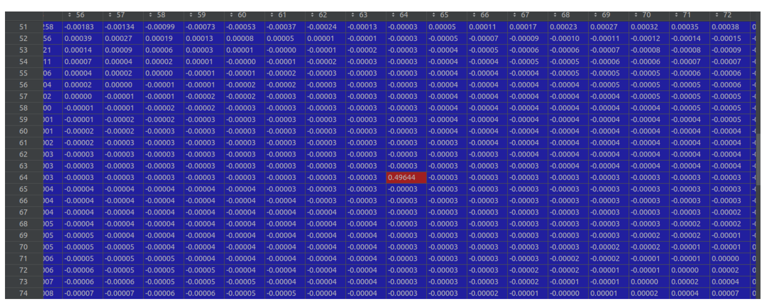


FIGURE 14 – motif cyclique en y de 128_e.png

### 2.3 Discret Fourier Transform

On applique une Discret Fourier Transform sur notre image 128_e et on obtient un tableau numpy de dimension 128 par 128 (les même que l'image initiale).



| | | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 51 | 258 | -0.00183 | -0.00134 | -0.00099 | -0.00073 | -0.00053 | -0.00037 | -0.00024 | -0.00013 | -0.00003 | 0.00005 | 0.00011 | 0.00017 | 0.00023 | 0.00027 | 0.00032 | 0.00035 | 0.00038 |
| 52 | 56 | 0.00039 | 0.00027 | 0.00019 | 0.00013 | 0.00008 | 0.00005 | 0.00001 | -0.00001 | -0.00003 | -0.00005 | -0.00007 | -0.00009 | -0.00010 | -0.00011 | -0.00012 | -0.00014 | -0.00015 |
| 53 | 21 | 0.00014 | 0.00009 | 0.00006 | 0.00003 | 0.00001 | -0.00000 | -0.00001 | -0.00002 | -0.00003 | -0.00004 | -0.00005 | -0.00006 | -0.00006 | -0.00007 | -0.00008 | -0.00008 | -0.00009 |
| 54 | 11 | 0.00007 | 0.00004 | 0.00002 | 0.00001 | -0.00000 | -0.00001 | -0.00002 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00005 | -0.00005 | -0.00006 | -0.00006 | -0.00007 | -0.00007 |
| 55 | 06 | 0.00004 | 0.00002 | 0.00000 | -0.00001 | -0.00001 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00005 | -0.00005 | -0.00005 | -0.00006 | -0.00006 |
| 56 | 04 | 0.00002 | 0.00000 | -0.00001 | -0.00001 | -0.00002 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00005 | -0.00005 | -0.00005 | -0.00006 |
| 57 | 02 | 0.00000 | -0.00001 | -0.00001 | -0.00002 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00005 | -0.00005 | -0.00005 |
| 58 | 00 | -0.00001 | -0.00001 | -0.00002 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00005 | -0.00005 |
| 59 | 001 | -0.00001 | -0.00002 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00005 |
| 60 | 001 | -0.00002 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 |
| 61 | 002 | -0.00002 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 |
| 62 | 003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 |
| 63 | 003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 |
| 64 | 003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | 0.49644 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 |
| 65 | 004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 |
| 66 | 004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 |
| 67 | 004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00002 |
| 68 | 005 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00002 | -0.00002 |
| 69 | 005 | -0.00005 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00002 | -0.00002 | -0.00001 |
| 70 | 005 | -0.00005 | -0.00005 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00002 | -0.00002 | -0.00001 | -0.00001 |
| 71 | 006 | -0.00005 | -0.00005 | -0.00005 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00003 | -0.00002 | -0.00002 | -0.00001 | -0.00001 | 0.00000 |
| 72 | 006 | -0.00006 | -0.00005 | -0.00005 | -0.00005 | -0.00004 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00002 | -0.00002 | -0.00001 | -0.00001 | 0.00000 | 0.00002 |
| 73 | 007 | -0.00006 | -0.00006 | -0.00005 | -0.00005 | -0.00005 | -0.00004 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00003 | -0.00002 | -0.00001 | -0.00001 | 0.00000 | 0.00002 | 0.00004 |
| 74 | 008 | -0.00007 | -0.00007 | -0.00006 | -0.00006 | -0.00005 | -0.00005 | -0.00004 | -0.00004 | -0.00003 | -0.00003 | -0.00002 | -0.00001 | -0.00000 | 0.00001 | 0.00002 | 0.00004 | 0.00007 |

FIGURE 15 – tableau numpy de la DFT réel de 128_e centré normalisé

Il y a des motifs en x et en y, ayant les mêmes périodes de 10pixels, alors les valeurs sont mieux répartis que pour l'image précédente.

## 2.4 Module du DFT

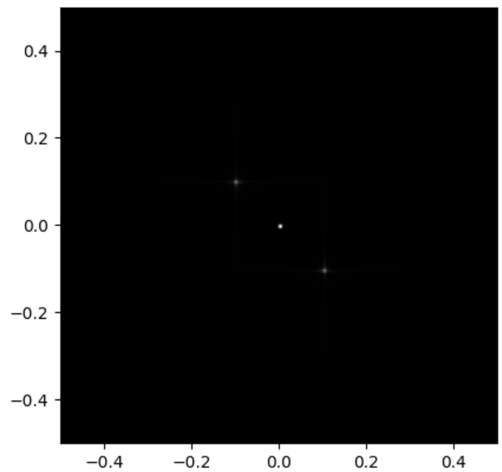



FIGURE 16 – module de 128_e sous la forme d'une image en niveaux de gris

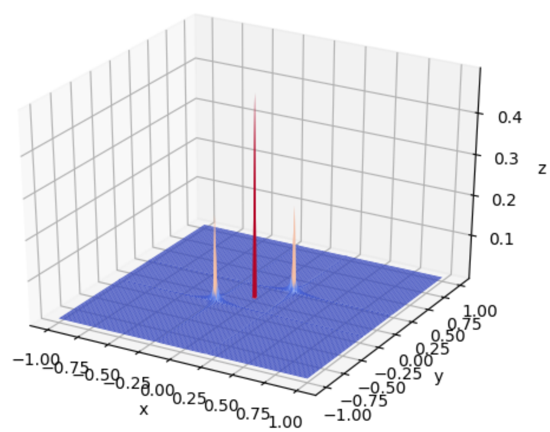



FIGURE 17 – module de 128_a sous la forme d'une image en niveaux de gris

Le module représente l'importance des motifs périodiques (le motif cyclique) de l'image (128_e.png) dans la direction du vecteur (0,0)–(fx,fy) et de fréquence proportionnelle à sa norme. Ainsi, dans ce cas, on constate que les 3 "pics" de notre module du DFT sont de valeurs absolues inférieurs à 0.25, alors les fréquences spatiales seront considérées comme relativement basses. Les pics sont présents en diagonales du haut-gauche au bas-droite, alors les perturbations suivent cette diagonale.

# 3 Fonctions

## 3.1 les 3 maxima du module de la DFT

```python
def getModuleOfDftOfImg(image):
        # fft de l'image
        fft = np.fft.fft2(image)  # tableau centrer en 0,0
        fftc = np.fft.fftshift(fft)  # centrage du fft
        fftcn = fftc / (len(image) * len(image[0]))  # normalisation par le produit des dimmensions de l'image
        module = np.abs(fftcn)

        return module

def get3MaximaOfModuleOfDftOfImg(image):
        maxima3 = []
        module = getModuleOfDftOfImg(image)

        ind = np.unravel_index(np.argsort(module, axis=None), np.shape(module))
        for i in range(1,4):
                maxima3.append([ind[0][len(ind[0])-i], ind[1][len(ind[1])-i]])

        return maxima3
```

Pour décomposer le code on a décidé de faire une première fonction getModuleOfDftOfImg qui renvoie le module DFT d'une image.

Notre deuxième fonction, "get3MaximaOfModuleOfDftOfImg", est celle permettant de récuperer les coordonnées des trois maxima du module dans le plan de Fourier de l'image. Elle commence par récuperer le module du DFT de l'image puis tri, grâce à "np.unravel_index(np.argsort(module, axis=None), np.shape(module))", chaque coordonnée par rapport à l'ordre croissant de leur valeurs correspondantes. On récupère par la suite les trois dernières que l'on retourne sous forme de tableau.

## 3.2 Récupération de la texture d'une image

```python
def getTextureType(image):

        max = get3MaximaOfModuleOfDftOfImg(image) # [[x,y],[x,y],[x,y]]
        if max[0][0] == max[1][0] == max[2][0]:
                print("La texture est verticale")
        elif max[0][1] == max[1][1] == max[2][1]:
                print("La texture est horizontale")
        else:
                print("La texture est diagonale")
```

Le principe de cette fonction qui reconnait les textures des images, c'est qu'on récupère les coordonnées des 3 maxima du module de la DFT de l'image (grâce à la fonction "get3MaximaOfModuleOfDftOfImg". Si ces trois maxima sont alignés verticalement, alors la perturbation est verticale, alors on déduit que les lignes de l'image sont horizontales. Si ils sont alignés horizontalement, alors la perturbation est horizontale et donc les lignes de l'image sont verticales. Sinon, on déduit que l'image est diagonale.

# 4 Le code final

```python
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D  # pour l'affichage de surfaces 3D


def printSizeOfImage(image):
        print("dimensions : " + str(len(image)) + "*" + str(len(image[0])))
```

```python

def getModuleOfDftOfImg(image):
    # fft de l'image
    fft = np.fft.fft2(image)  # tableau centrer en 0,0
    fftc = np.fft.fftshift(fft)  # centrage du fft
    # normalisation par le produit des dimmensions de l'image
    fftcn = fftc / (len(image) * len(image[0]))
    module = np.abs(fftcn)

    return module

def showDFT(image):
    # Visualisation de l'image
    plt.figure()
    plt.imshow(image, cmap='gray')

    # fft de l'image
    module = getModuleOfDftOfImg(image)

    # Visualisation du module du fft sous forme d'image en niveaux de gris
    plt.figure()
    plt.imshow(module, cmap='gray', extent=(-0.5, 0.5, -0.5, 0.5))

    # Definition d'une grille de trace (taille des axe en fonction des dimmensions de l'image)
    x, y = np.mgrid[-1.0:1.0:(len(image) + 0j), -1.0:1.0:(len(image[0]) + 0j)]
    z = module #notre z corespond au module du fft

    # Visualisation de la fonction en 3D
    plt.figure()
    axes = plt.gca(projection=Axes3D.name)
    axes.set_xlabel('x')
    axes.set_ylabel('y')
    axes.set_zlabel('z')
    axes.plot_surface(x, y, z, cmap='coolwarm', rstride=1, cstride=1, antialiased=True)

    # Affichage des figures matplotlib a l'ecran
    plt.show()

def get3MaximaOfModuleOfDftOfImg(image):
    maxima3 = []
    module = getModuleOfDftOfImg(image)

    ind = np.unravel_index(np.argsort(module, axis=None), np.shape(module))
    for i in range(1,4):
        maxima3.append([ind[0][len(ind[0])-i], ind[1][len(ind[1])-i]])

    return maxima3

def getTextureType(image):

    max = get3MaximaOfModuleOfDftOfImg(image) # [[x,y],[x,y],[x,y]]
    if max[0][0] == max[1][0] == max[2][0]:
        print("La texture est verticale")
    elif max[0][1] == max[1][1] == max[2][1]:
        print("La texture est horizontale")
    else:
        print("La texture est diagonale")



image = plt.imread("textures2/512_d.png")

showDFT(image)

getTextureType(image)

printSizeOfImage(image)
```