Département MI Centre Universitaire de Mila

Traitement d'images Dr. B Maamar

Solution travaux pratiques N°4

```
import numpy as np
import matplotlib.pyplot as plt

# ----- 1a Visualisation de la partie réelle et imaginaire de la transfo
n = 20
a = np.zeros(n)
a[1] = 1
print(a)
plt.subplot(311)
plt.plot( np.append(a, a[0]) )
A = np.fft.fft(a)
print('la valeur de A est: ', np.round(A,2))
B = np.append(A, A[0])
print('la valeur de B est: ', np.round(B,2))
plt.subplot(312)
preal =  np.real(B)
print('la partie reelle est: ', np.round(preal,2))
plt.plot(preal)
plt.ylabel("partie reelle")
plt.subplot(313)
pimag = np.imag(B)
print('la partie imaginaire est: ', np.round(pimag,2))
plt.plot(pimag)
plt.ylabel("partie imaginaire")
plt.show()

# ----------------- Créer un signal sinusoïdale 1D :
x = np.arange(-500, 501, 1)
wavelength = 200
y = np.sin(2 * np.pi * x / wavelength)
plt.plot(x, y)
plt.show()

# ----------------- Réseau sinusoïdal 1D à un réseau sinusoïdal 2D:
x = np.arange(-500, 501, 1)
X, Y = np.meshgrid(x, x)
wavelength = 200
grating = np.sin(2 * np.pi * X / wavelength)
plt.set_cmap("gray")
plt.imshow(grating)
plt.show()

# Si vous souhaitez créer un réseau avec une autre orientation
x = np.arange(-500, 501, 1)
X, Y = np.meshgrid(x, x)
wavelength = 200
angle = np.pi / 9
grating = np.sin(
    2*np.pi*(X*np.cos(angle) + Y*np.sin(angle)) / wavelength
)
plt.set_cmap("gray")
plt.imshow(grating)
plt.show()
```

```python
# ----------------- déterminons leur transformée de Fourier à l'aide de NumPy
x = np.arange(-500, 501, 1)
X, Y = np.meshgrid(x, x)
wavelength = 200
angle = 0
grating = np.sin(2*np.pi*(X*np.cos(angle) + Y*np.sin(angle)) / wavelength)
plt.set_cmap("gray")
plt.subplot(121)
plt.imshow(grating)
# -----------------  Calculet la transformée de Fourier
ft = np.fft.ifftshift(grating)
ft = np.fft.fft2(ft)
ft = np.fft.fftshift(ft)
plt.subplot(122)
plt.imshow(abs(ft))
plt.xlim([480, 520])
plt.ylim([520, 480])  # Note, order is reversed for y
plt.show()
# -----------------  doubler la fréquence

x = np.arange(-500, 501, 1)
X, Y = np.meshgrid(x, x)
wavelength = 100
angle = 0
grating = np.sin(2*np.pi*(X*np.cos(angle) + Y*np.sin(angle)) / wavelength)
plt.set_cmap("gray")
plt.subplot(121)
plt.imshow(grating)
ft = np.fft.ifftshift(grating)
ft = np.fft.fft2(ft)
ft = np.fft.fftshift(ft)

plt.subplot(122)
plt.imshow(abs(ft))
plt.xlim([480, 520])
plt.ylim([520, 480])  # Note, order is reversed for y
plt.show()
# ------ pivoter ce réseau sinusoïdal de 20°. Cela fait pi/9 radians :
 x = np.arange(-500, 501, 1)
 X, Y = np.meshgrid(x, x)
 wavelength = 100
 angle = np.pi/9
 grating = np.sin(2*np.pi*(X*np.cos(angle) + Y*np.sin(angle)) / wavelength)
 plt.set_cmap("gray")
 plt.subplot(121)
 plt.imshow(grating)

 ft = np.fft.ifftshift(grating)
 ft = np.fft.fft2(ft)
 ft = np.fft.fftshift(ft)

 plt.subplot(122)
 plt.imshow(abs(ft))
 plt.xlim([480, 520])
 plt.ylim([520, 480])  # Note, order is reversed for y
 plt.show()
```

```python
# -----------------  convertir une image en sa transformée de Fourier :
def calculate_2dft(input):
    ft = np.fft.ifftshift(input)
    ft = np.fft.fft2(ft)
    return np.fft.fftshift(ft)
image_filename = "D:\img\Objet1.png"
image = plt.imread(image_filename)
image = image[:, :, :3].mean(axis=2)  # Convert to grayscale
plt.set_cmap("gray")
ft = calculate_2dft(image)

plt.subplot(121)
plt.imshow(image)
plt.axis("off")
plt.subplot(122)
plt.imshow(np.log(abs(ft)))
plt.axis("off")
plt.show()

#La transformée de Fourier inverse

x = np.arange(-500, 501, 1)
X, Y = np.meshgrid(x, x)
wavelength = 100
angle = np.pi/9
grating = np.sin(2*np.pi*(X*np.cos(angle) + Y*np.sin(angle)) / wavelength
)
plt.set_cmap("gray")
plt.subplot(131)
plt.imshow(grating)
plt.axis("off")
ft = np.fft.ifftshift(grating)
ft = np.fft.fft2(ft)
ft = np.fft.fftshift(ft)

plt.subplot(132)
plt.imshow(abs(ft))
plt.axis("off")
plt.xlim([480, 520])
plt.ylim([520, 480])

# -----------------  inverse transformée de Fourier
ift = np.fft.ifftshift(ft)
ift = np.fft.ifft2(ift)
ift = np.fft.fftshift(ift)
ift = ift.real  # Take only the real part

plt.subplot(133)
plt.imshow(ift)
plt.axis("off")
plt.show()
```

```python
# Transformée de radon
from skimage.data import shepp_logan_phantom
from skimage.transform import radon, rescale
image =  shepp_logan_phantom()
image = rescale(image, scale=0.4, mode='reflect', channel_axis=None)
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 4.5))
ax1.set_title("Original")
ax1.imshow(image, cmap=plt.cm.Greys_r)
theta = np.linspace(0.0, 180.0, max(image.shape), endpoint=False)
sinogram = radon(image, theta=theta)
dx, dy = 0.5 * 180.0 / max(image.shape), 0.5 / sinogram.shape[0]
ax2.set_title("Radon transform\n(Sinogram)")
ax2.set_xlabel("Projection angle (deg)")
ax2.set_ylabel("Projection position (pixels)")
ax2.imshow(sinogram, cmap=plt.cm.Greys_r, extent=(-dx, 180.0 + dx, -dy, sinogram.shape[0] + dy), aspect='auto',)
fig.tight_layout()
plt.show()
```