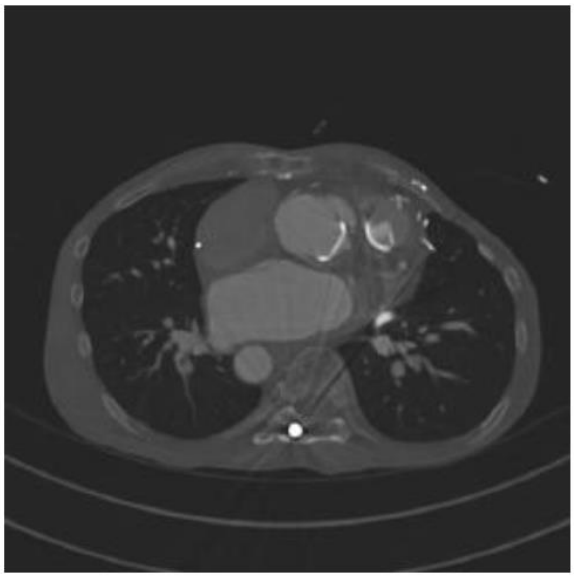
Sau bài thực hành này, sinh viên có thể

- Viết được chương trình xử lý điểm ảnh sử dụng kỹ thuật biến đổi logarith
- Viết được chương trình xử lý điểm ảnh sử dụng kỹ thuật biến đổi power law
- Viết được chương trình xử lý điểm ảnh sử dụng kỹ thuật ảnh ngược
- Viết được chương trình xử lý điểm ảnh sử dụng kỹ thuật histogram equalization
- Viết được chương trình xử lý điểm ảnh sử dụng kỹ thuật contrast stretching

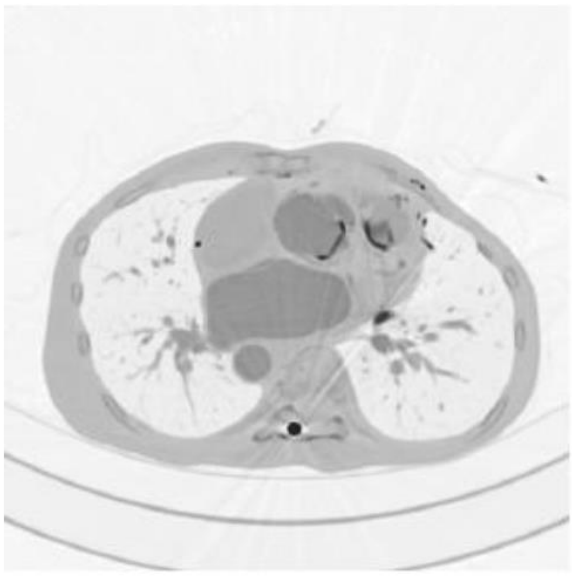
1. VIẾT CHƯƠNG TRÌNH XỬ LÝ ĐIỂM ẢNH

1.1. Biến đổi cường độ ảnh (Image inverse transformation)

Là phép biến đổi cường độ ảnh từ tối sang sáng và ngược lại.



(a) Input

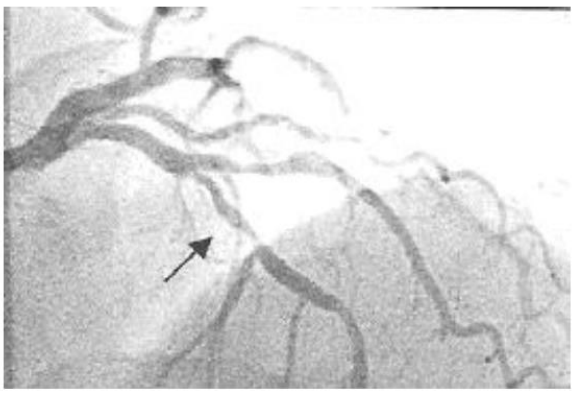


(b) Output

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')

#convert image 1 into an ndarray
im_1 = np.asarray(img)

#inversion operation
im_2 = 255 - im_1

#convert image 2 from ndarray to image
new_img = Image.fromarray(im_2)

img.show()

plt.imshow(new_img)
plt.show()
```

1.2. Thay đổi chất lượng ảnh với Power law (Gamma-Correction)

Dùng để tăng chất lượng của ảnh



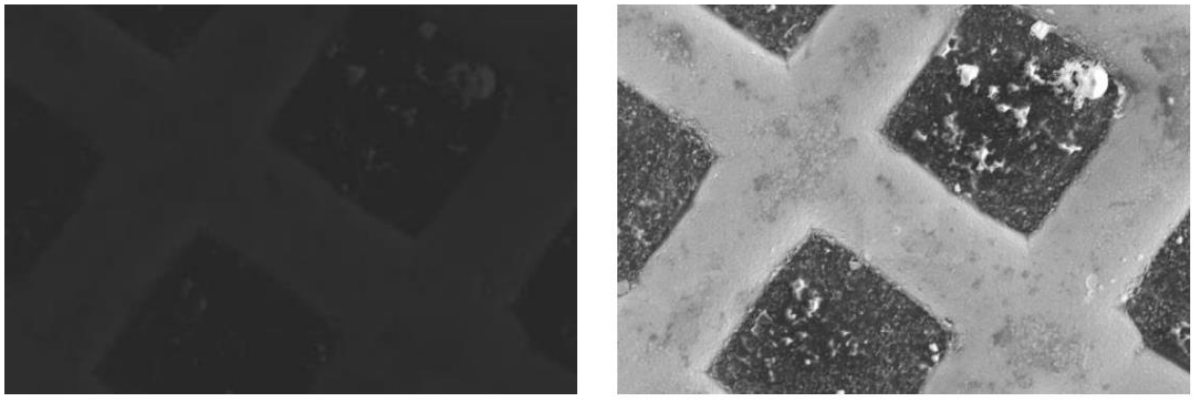
(a) Input image.

(b) Gamma corrected image with $\gamma = 0.5$.

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')

#convert image 1 into an ndarray
im_1 = np.asarray(img)

#init gamma
gamma = 0.5

#convert ndarray from int to float
b1 = im_1.astype(float)

#find maximum value in b1
b2 = np.max(b1)

#b3 is normalized
b3 = b1/b2

#b2 gamma correction exponent is computed
b2 = np.log(b3) * gamma

#gamma correction is computed
c = np.exp(b2) * 255.0

#c1 is converted to type int
c1 = c.astype(int)

d = Image.fromarray(c1)

img.show()
d.show()
plt.imshow(d)
plt.show()
```

Yêu cầu: sinh viên thay đổi giá trị gamma = 5 và xem sự thay đổi

1.3. Thay đổi cường độ điểm ảnh với Log Transformation



(a) Input (b) Output

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')

#convert image 1 into an ndarray
im_1 = np.asarray(img)

#convert ndarray from int to float
b1 = im_1.astype(float)

#find maximum value in b1
b2 = np.max(b1)

#performing the log transformation
c = (128.0 * np.log(1 + b1))/np.log(1 + b2)

#c1 is converted to type int
c1 = c.astype(int)

d = Image.fromarray(c1)

img.show()
d.show()
plt.imshow(d)
plt.show()
```

1.4. Histogram equalization

Histogram (lược đồ xám) là biểu đồ tần xuất thống kê số lần xuất hiện các mức sáng trong ảnh. Mục đích là cải tiến độ tương phản hai vùng sáng tối của ảnh. Trong ảnh grayscale, giá trị

intensive của ảnh nằm [0, *L*-1]. Ảnh càng tối khi giá trị đi vào vùng low gray, ảnh càng sáng khi giá trị đi vào vùng high gray. Hàm sử dụng xác suất, CDF(Cumulative Distribution) để tính.

32	41	52	65	70
60	35	45	66	69
38	65	35	33	39
41	68	75	71	73
37	38	57	58	65

Gray level value	Probability	CDF as C	h(u)
32	2/25	2/25	11
33	1/25	3/25	22
35	3/25	6/25	54
38	3/25	9/25	85
41	3/25	12/25	117
45	1/25	13/25	128
52	2/25	15/25	149
57	1/25	16/25	160
60	1/25	17/25	170
63	2/25	19/25	192
65	5/25	24/25	245
66	1/25	25/25	255

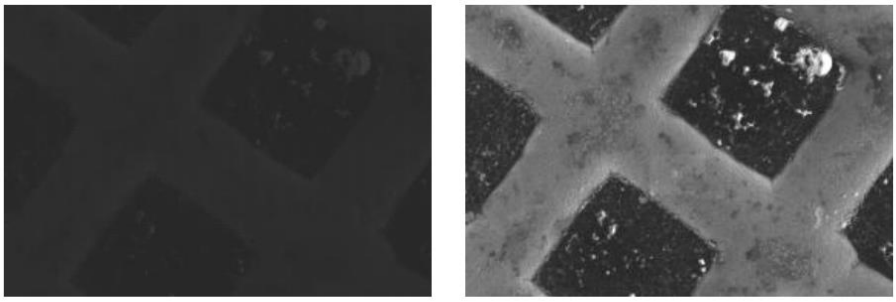
(a) Input image.

(c) Output image.

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')
#convert image 1 into an ndarray
im1 = np.asarray(img)
#convert 2D ndarray from 1D array
b1 = im1.flatten()
#histogram and bin are computed
hist, bins = np.histogram(im1, 256, [0, 255])
#cumulative distribution function is computed
cdf = hist.cumsum()
# places where cdf=0 is masked or ignored and
# rest is stored in cdf_m
cdf_m = np.ma.masked_equal(cdf, 0)
#histogram equalization is performed
num_cdf_m = (cdf_m -cdf_m.min()) * 255
den_cdf_m = (cdf.max() - cdf_m.min())
cdf_m = num_cdf_m/den_cdf_m
# the masked places in cdf_m are now 0
cdf = np.ma.filled(cdf_m, 0).astype('uint8')
# cdf values are assigned in the flattened array
im2 = cdf[b1]
# im2 is 1D so we use reshape command to
# make it into 2D
im3 = np.reshape(im2, im1.shape)
# converting im3 to an image
im4 = Image.fromarray(im3)
img.show()
im4.show()
plt.imshow(im4)
plt.show()
```

1.5. Thay đổi ảnh với Contrast Stretching

Tương tự như Histogram Equalization nhưng thay đổi giá trị pixel thay vì dùng xác suất, CDF để tính như Histogram.

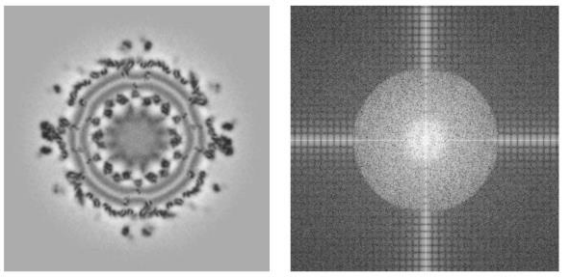


(a) Input image. (b) Output image.

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')
#convert image l into an ndarray
iml = np.asarray(img)
# finding the maximum and minimum pixel values
b = iml.max()
a = iml.min()
print(a, b)
# converting iml to float
c = iml.astype(float)
# contrast stretching transformation
im2 = 255 * (c - a)/(b - a)
#im2 is converted from ndarray to image
im3 = Image.fromarray(im2)
img.show()
im3.show()
plt.imshow(im3)
plt.show()
```

1.6. Biến đổi Fourier

Dùng để biến đổi ảnh theo miền tần suất. Biến đổi Fourier được sử dụng trong image filter, image compression, image enhancement, image restoration, image analysis, image reconstruction.



(a) Input for FFT. (b) Output of FFT.

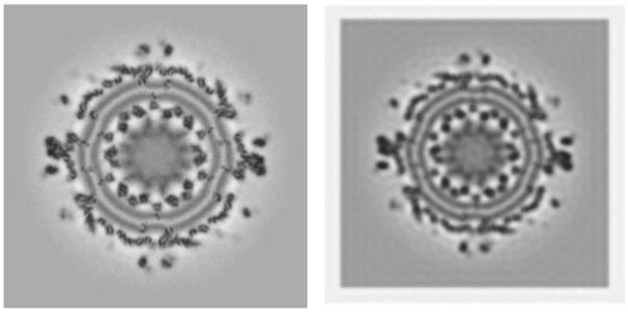
1.6.1. Biến đổi ảnh với Fast Fourier

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')
#convert image 1 into an ndarray
iml = np.asarray(img)
# performing FFT
c = abs(scipy.fftpack.fft2(iml))
# shifting the Fourier frequency image
d = scipy.fftpack.fftshift(c)
d = d.astype(float)
#im2 is converted from ndarray to image
im3 = Image.fromarray(d)
img.show()
im3.show()
plt.imshow(im3)
plt.show()
```

1.6.2. Lọc ảnh trong miền tần suất

Có hai dạng là lowpass và highpass. Lowpass sử dụng những điểm ảnh có tần suất thấp từ biến đổi Fourier. Lowpass dùng để làm mịn và khử nhiễu ảnh. Highpass sử dụng những điểm ảnh có tần suất cao từ biến đổi Fourier. Highpass dùng để làm sắc biên của ảnh.

Butterworth Lowpass Filter



(a) Input for lowpass filters. (c) Output of BLPF.

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')
#convert image 1 into an ndarray
im1 = np.asarray(img)
# performing FFT
c = abs(scipy.fftpack.fft2(im1))
# shifting the Fourier frequency image
d = scipy.fftpack.fftshift(c)
#intializing variables for convolution function
M = d.shape[0]
N = d.shape[1]
# H is defined and values in H are initialized to 1
H = np.ones((M, N))

center1 = M/2
center2 = N/2
d_0 = 30.0 #cut-off radius
t1 = 1 #the order of BLPF (Butterworth Lowpass Filter)
t2 = 2 * t1

# defining the convolution function for BLPF
for i in range(1, M):
    for j in range(1, N):
        r1 = (i - center1)**2 + (j - center2)**2
        # euclidean distance from origin is computed
        r = math.sqrt(r1)
        # using cut-off radius to eliminate high frequency
        if r > d_0:
            H[i, j] = 1/(1 + (r/d_0)**t1)
#H is converted from ndarray to image
H = H.astype(float)
H = Image.fromarray(H)
# performing the convolution
con = d * H
# computing the magnitude of the inverse FFT
e = abs(scipy.fftpack.ifft2(con))
#e is converted from ndarray to image
e = e.astype(float)
im3 = Image.fromarray(e)

img.show()
im3.show()
plt.imshow(im3)
plt.show()
```

Butterworth highpass Filter

```
from PIL import Image
import math
import scipy
import numpy as np
import imageio.v2 as iio
import matplotlib.pylab as plt

#open a grayscale image
img = Image.open('world_cup.jpg').convert('L')
#convert image 1 into an ndarray
iml = np.asarray(img)
# performing FFT
c = abs(scipy.fftpack.fft2(iml))
# shifting the Fourier frequency image
d = scipy.fftpack.fftshift(c)
#intializing variables for convolution function
M = d.shape[0]
N = d.shape[1]
# H is defined and values in H are initialized to 1
H = np.ones((M, N))

center1 = M/2
center2 = N/2
d_0 = 30.0 #cut-off radius
t1 = 1 #the order of BLPF (Butterworth Lowpass Filter)
t2 = 2 * t1

# defining the convolution function for BLPF
for i in range(1, M):
    for j in range(1, N):
        r1 = (i - center1)**2 + (j - center2)**2
        # euclidean distance from origin is computed
        r = math.sqrt(r1)
        # using cut-off radius to eliminate high frequency
        if r > d_0:
            H[i, j] = 1/(1 + (r/d_0)**t2)
#H is converted from ndarray to image
H = H.astype(float)
H = Image.fromarray(H)
# performing the convolution
con = d * H
# computing the magnitude of the inverse FFT
e = abs(scipy.fftpack.ifft2(con))
#e is converted from ndarray to image
e = e.astype(float)
im3 = Image.fromarray(e)

img.show()
im3.show()
plt.imshow(im3)
plt.show()
```

2. BÀI TẬP

1. Viết chương trình tạo menu cho phép người dùng chọn các phương pháp biến đổi ảnh như sau:

- Image inverse transformation

- Gamma-Correction

- Log Transformation

- Histogram equalization

- Contrast Stretching

Khi người dùng ấn phím I, G, L, H, C thì chương trình sẽ thực hiện hàm tương ứng cho các hình trong thư mục exercise. Lưu và hiển thị các ảnh đã biến đổi.

2. Viết chương trình tạo menu cho phép người dùng chọn các phương pháp biến đổi ảnh như sau:

- Fast Fourier

- Butterworth Lowpass Filter

- Butterworth Highpass Filter

Khi người dùng ấn phím F, L, H thì chương trình sẽ thực hiện hàm tương ứng cho các hình trong thư mục exercise. Lưu và hiển thị các ảnh đã biến đổi.

3. Viết chương trình thay đổi thứ tự màu RGB của ảnh trong thư mục exercise và sử dụng ngẫu nhiên một trong các phép biến đổi ảnh trong câu 1. Lưu và hiển thị ảnh đã biến đổi.

4. Viết chương trình thay đổi thứ tự màu RGB của ảnh trong thư mục exercise và sử dụng ngẫu nhiên một trong các phép biến đổi ảnh trong câu 2. Nếu ngẫu nhiên là phép Butterworth Lowpass thì chọn thêm Min Filter để lọc ảnh. Nếu ngẫu nhiên là phép Butterworth Highpass thì chọn thêm Max Filter để lọc ảnh. Lưu và hiển thị ảnh đã biến đổi.