ข้อ 1

import os

import cv2

import numpy as np

def map\_to\_gray\_levels(image, levels):

    step = 256 // (levels)

    new\_image = (image // step) \* step

    return new\_image.astype(np.uint8)

def main():

    asset\_path = './asset'

    pictures = ["flower.jpg", "fractal.jpeg", "fruit.jpg"]

    for picture in pictures:

        image\_path = os.path.join(asset\_path, picture)

        image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

        if image is None:

            print("Could not open or find the image.")

            return

        gray8 = map\_to\_gray\_levels(image, 8)

        gray64 = map\_to\_gray\_levels(image, 64)

        gray128 = map\_to\_gray\_levels(image, 128)

        gray256 = map\_to\_gray\_levels(image, 256)

        pic\_name = picture.split(".")[0]

        output\_dir = f"./gray/{pic\_name}"

        os.makedirs(output\_dir, exist\_ok=True)

        # Display images

        cv2.imshow(f'{pic\_name}\_original', image)

        cv2.imshow(f'{pic\_name}\_gray8', gray8)

        cv2.imshow(f'{pic\_name}\_gray64', gray64)

        cv2.imshow(f'{pic\_name}\_gray128', gray128)

        cv2.imshow(f'{pic\_name}\_gray256', gray256)

        # Save images

        cv2.imwrite(os.path.join(output\_dir, f'{pic\_name}\_gray8.jpg'), gray8)

        cv2.imwrite(os.path.join(output\_dir, f'{pic\_name}\_gray64.jpg'), gray64)

        cv2.imwrite(os.path.join(output\_dir, f'{pic\_name}\_gray128.jpg'), gray128)

        cv2.imwrite(os.path.join(output\_dir, f'{pic\_name}\_gray256.jpg'), gray256)

        cv2.waitKey(0)

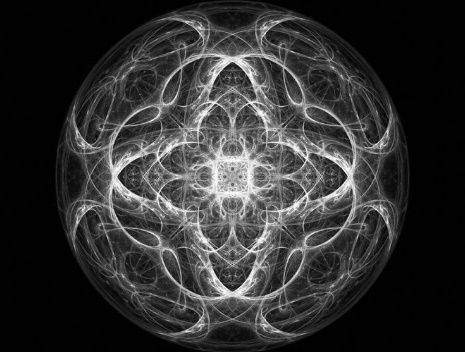
        cv2.destroyAllWindows()

if \_\_name\_\_ == '\_\_main\_\_':

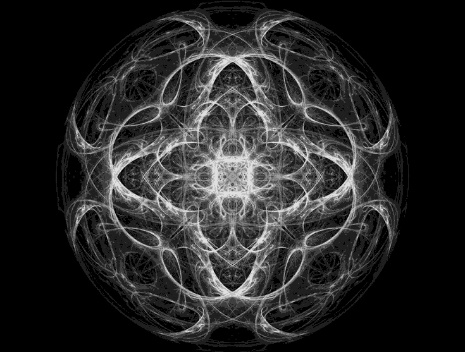
    main()

ภาพต้นฉบับ

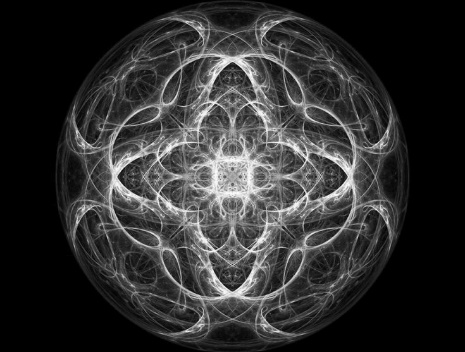




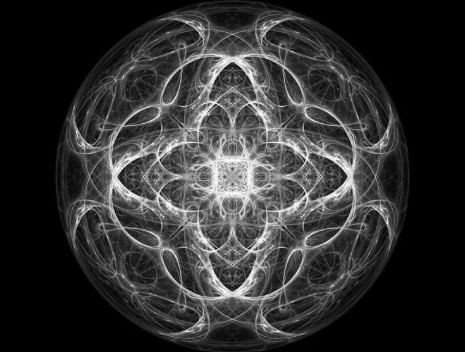
8 gray levels

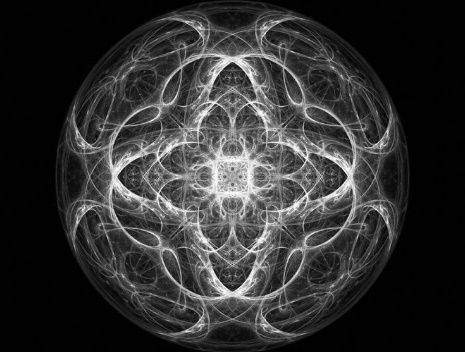


64 gray levels



128 gray levels

  
256 gray levels



อธิบายเพิ่มเติม : เมื่อปรับแต่งภาพแล้วโดยส่วนตัวจะเห็นความต่างแค่ภาพ 8 gray levels โดยสามารถสังเกตุได้คือบริเวณที่ใกล้กันจะมีสีโทนเดียวกันที่กระจุกอยู่ใกล้กัน เพราะบริเวณที่ใกล้กันจะอยู่ในช่วงของค่าใกล้ๆกันจะทำให้ถูกปรับค่าและได้ค่าออกมาเป็นค่าเดียวกัน ส่วนภาพที่เหลือจะให้ความรู้สึกเหมือนกันภาพต้นฉบับ (ที่อ่านออกมาเป็น grayscale) เพราะถ้าดูจากค่าที่ทำการ print ออกมาแล้ว ค่าที่ได้ออกมาก็มีความใกล้เคียงกันจริงๆ เพราะ 8 gray levels มีช่วงของค่าที่มากที่สุดคือ 32 ที่จะถูกปรับไปเป็นค่าเดียวกัน ส่วน 64, 128, 256 gray levels มีช่วงที่จะปรับเพียง 4, 2, 1 ตามลำดับ และค่าที่ปรับออกมาก็จะมีค่าที่ไม่ต่างกันมาก ทำให้ผลที่ได้ไม่แตกต่างกันมาก

ข้อ 2

import cv2

import numpy as np

import os

def enhance\_image(image):

    L = 256

    enhanced\_image = np.copy(image)

    h, w = image.shape

    for y in range(h):

        for x in range(w):

            if 0 <= enhanced\_image[y, x] < L/3:

                enhanced\_image[y, x] = int(5 \* L / 6)

            elif L/3 <= enhanced\_image[y, x] < 2 \* L/3:

                slope = ((L/6) - (5 \* L/6)) / ((2 \* L/3) - (L/3))

                enhanced\_image[y, x] = int((5 \* L/6) + (slope \* (enhanced\_image[y, x] - (L/3))))

            else:

                enhanced\_image[y, x] = int(L / 6)

    return enhanced\_image

def main():

    asset\_path = './asset'

    pictures = ["flower.jpg", "traffic.jpg", "tram.jpg"]

    for picture in pictures:

        image\_path = os.path.join(asset\_path, picture)

        image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

        if image is None:

            print("Could not open or find the image.")

            return

        enhanced\_image = enhance\_image(image)

        cv2.imshow("enhanced\_image", enhanced\_image)

        pic\_name = picture.split(".")[0]

        output\_dir = f"./enhanced/{pic\_name}"

        os.makedirs(output\_dir, exist\_ok=True)

        cv2.imwrite(os.path.join(output\_dir, f'{pic\_name}\_enhanced.jpg'), enhanced\_image)

        cv2.waitKey(0)

        cv2.destroyAllWindows()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

ภาพต้นฉบับ





ภาพ enhance



อธิบายเพิ่มเติม : ภาพที่ได้จะมีลักษณะคล้ายกับ negative เพราะช่วงตั้งแต่ L/3 ถึง 2L/3 มีลักษณะของ slope ที่ติดลบคล้ายๆกับ negative เพียงแค่ค่าของ slope มีค่ามากกว่าทำให้การปรับเฉดมีการปรับที่ไวกว่า และส่วนที่แตกต่างอีกเล็กน้อยคือส่วนที่มีค่าน้อยกว่า L/3 และมากกว่า 2L/3 ซึ่งจะแปลงค่าจากโทนดำเป็นโทนขาวและโทนขาวเป็นโทนดำตามลำดับ แต่จะแตกต่างกับส่วนที่เป็น slope ของฟังก์ชันคือค่าที่ได้จะเป็นค่าที่คงที่เลย

ข้อ 3

import cv2

import os

import numpy as np

def enhance\_image(image, c, gamma):

    enhanced\_image = np.power(image / 255.0, gamma)

    enhanced\_image = np.uint8(255 \* c \* enhanced\_image)

    return enhanced\_image

def main():

    asset\_path = './asset'

    pictures = ["cartoon.jpg", "scenery1.jpg", "scenery2.jpg"]

    for picture in pictures:

        image\_path = os.path.join(asset\_path, picture)

        image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

        if image is None:

            print("Could not open or find the image.")

            return

        c\_values = [0.5, 1, 2]

        gamma\_values = [0.4, 2.5]

        for c in c\_values:

            for gamma in gamma\_values:

                enhanced\_image = enhance\_image(image, c, gamma)

                cv2.imshow(f"enhanced\_image c : {c} gamma : {gamma}", enhanced\_image)

                pic\_name = picture.split(".")[0]

                output\_dir = f"./enhanced\_powerlaw/{pic\_name}"

                os.makedirs(output\_dir, exist\_ok=True)

                cv2.imwrite(os.path.join(output\_dir, f'{pic\_name}\_c{c}\_gamma{gamma}.jpg'), enhanced\_image)

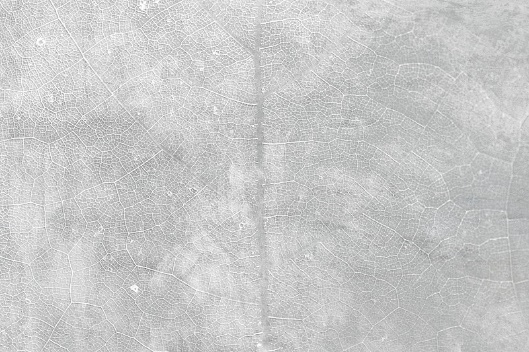
                cv2.waitKey(0)

                cv2.destroyAllWindows()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

ภาพต้นฉบับ



ภาพ enhance power-law c = 0.5 ɣ = 0.4



ภาพ enhance power-law c = 0.5 ɣ = 2.5



ภาพ enhance power-law c = 1 ɣ = 0.4



ภาพ enhance power-law c = 1 ɣ = 2.5



ภาพ enhance power-law c = 2 ɣ = 0.4



ภาพ enhance power-law c = 2 ɣ = 2.5



อธิบายเพิ่มเติม : ภาพที่ได้เมื่อ

c = 0.5 ɣ = 0.4 | ภาพที่ได้จะอยู่ในโทนดำกว่าภาพต้นฉบับ

c = 0.5 ɣ = 2.5 | ภาพที่ได้จะอยู่ในโทนดำกว่าภาพต้นฉบับ และก็โทนดำกว่า c = 0.5 ɣ = 0.4

c = 1 ɣ = 0.4 | ภาพที่ได้จะอยู่ในโทนขาวกว่าภาพต้นฉบับ

c = 1 ɣ = 2.5 | ภาพที่ได้จะอยู่ในโทนดำกว่าทั้งภาพต้นฉบับ และ c = 1 ɣ = 0.4

c = 2 ɣ = 0.4 | จะมีการแยกสีที่ชัดเจนกว่าภาพต้นฉบับส่วนที่ดำจะดำกว่า ส่วนที่ขาวก็จะขาวกว่า

c = 2 ɣ = 2.5 | ภาพที่ได้จะอยู่ในโทนดำกว่าภาพต้นฉบับ

โดยถ้าจะอธิบายภาพที่ได้จากสมการคือ ค่า c จะทำการปรับค่าของภาพให้เพิ่มลดได้โดย c เป็นค่าคงที่ กล่าวคือถ้าเพิ่มก็จะเพิ่มทั้งภาพถ้าลดก็จะลดทั้งภาพ ส่วนค่า จะทำให้ภาพสว่างขึ้นถ้าค่าน้อยกว่า 1 และจะทำให้ภาพดำลงถ้าค่า มากกว่า 1