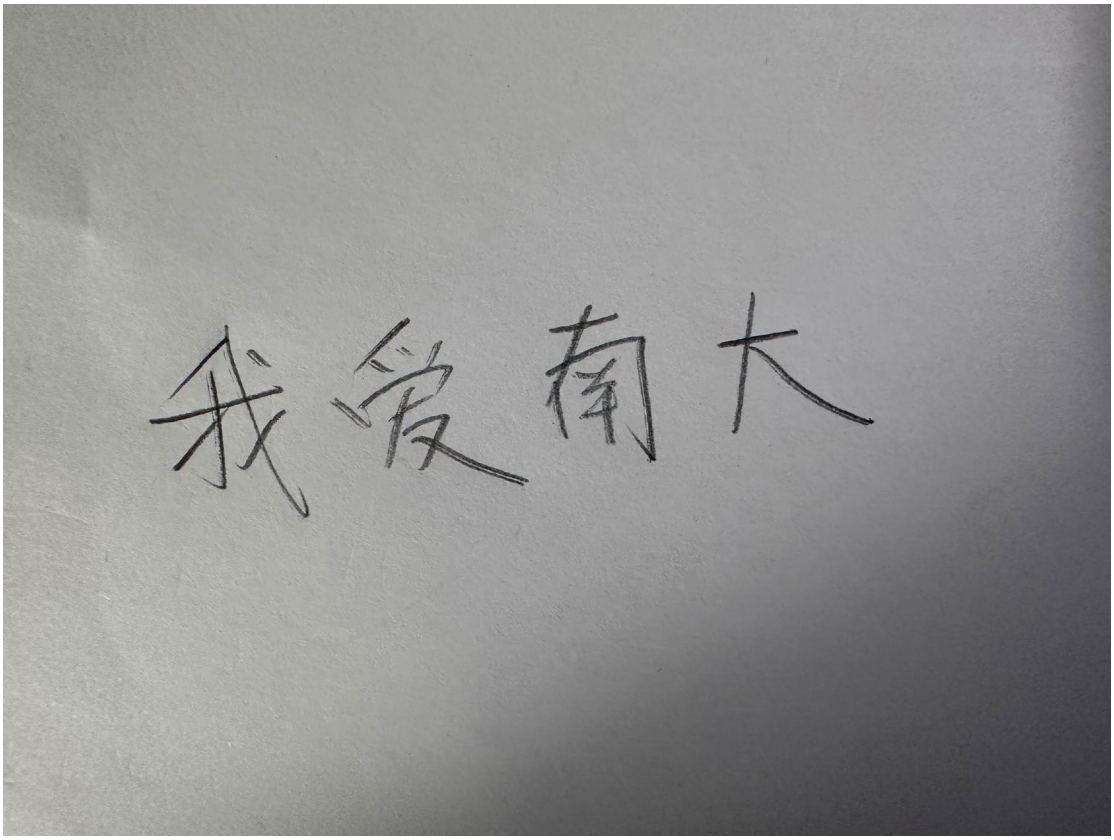
《计算机视觉》（本科，2023）作业 1

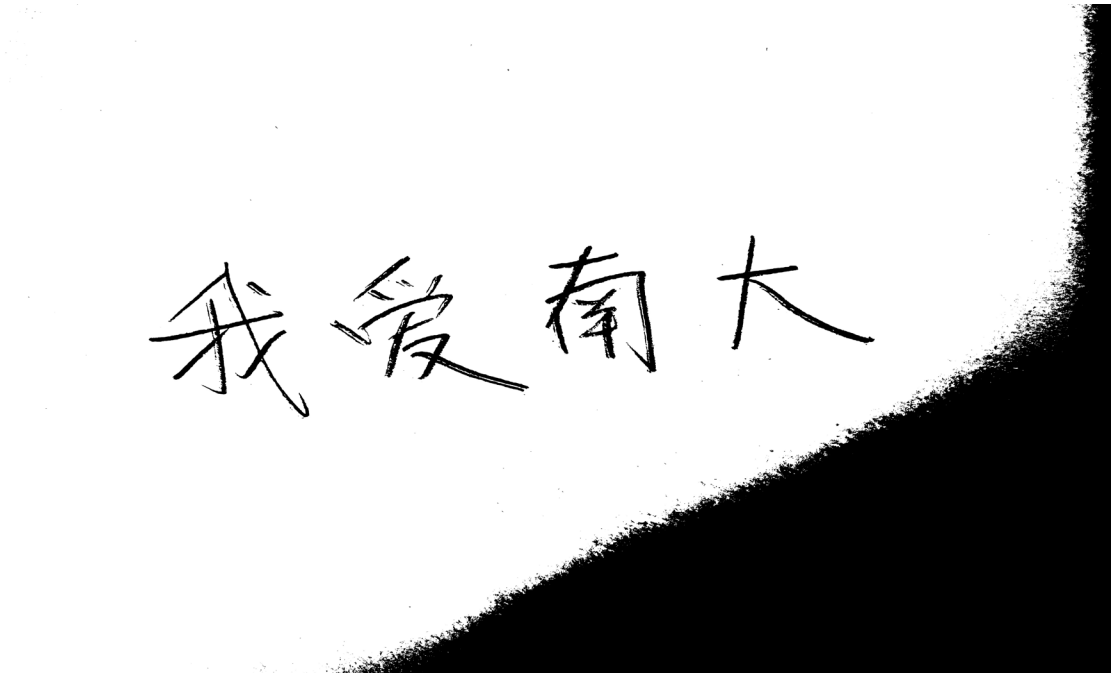
1. 将附带的彩色图像（I0）转为灰度图像（记为 I1）。

```python
import cv2


def convert_to_grayscale(image_path):
    # 读取彩色图像
    image = cv2.imread(image_path)

    # 转换为灰度图像
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    return gray_image


# 指定彩色图像路径
color_image_path = 'hw01-I0.jpeg'

# 转换为灰度图像
gray_image = convert_to_grayscale(color_image_path)

# 保存灰度图像
cv2.imwrite('hw01-I0-grey.jpeg', gray_image)
```

2. 在白纸上手写“我爱南大”四个字，拍照，转为与 I1 分辨率相同的二值图像（记为 I2）。



```python
import cv2


def convert_to_binary(image_path, width, height):
    # 读取图像
    image = cv2.imread(image_path)

    # 调整图像大小
    resized_image = cv2.resize(image, (width, height))

    # 将图像转换为灰度图像
    gray_image = cv2.cvtColor(resized_image, cv2.COLOR_BGR2GRAY)

    # 对灰度图像进行二值化处理
    _, binary_image = cv2.threshold(gray_image, 127, 255, cv2.THRESH_BINARY)

    return binary_image


# 指定原始图像路径
image_path = 'ilovenju.jpg'

# 指定目标分辨率
target_width = 5922
target_height = 3567

# 转换为二值图像
binary_image = convert_to_binary(image_path, target_width, target_height)

# 保存二值图像
cv2.imwrite('hw01-I2.jpeg', binary_image)
```



3. 灰度图像每个像素的灰度值为 1 个字节（8 位），按照从低到高记为 L1、L2、…、L8。将 I1 中每个像素的 L1、L2、…、L8 分别用 I2 替换。对结果进行分析。

```python
import cv2
import numpy as np
import tqdm


def replace_pixel_values(image1_path, image2_path):
    # 读取I1和I2的灰度图像
    image1 = cv2.imread(image1_path, cv2.IMREAD_GRAYSCALE)
    image2 = cv2.imread(image2_path, cv2.IMREAD_GRAYSCALE)

    # 获取图像宽度和高度
    height, width = image1.shape
    result_images = []
    for k in range(8):
        binary_value = 1 << k
        # 创建结果图像，与输入图像相同大小
        result_image = np.zeros((height, width), dtype=np.uint8)

        # 遍历图像的每个像素
        for i in tqdm.tqdm(range(height)):
            for j in range(width):
                # 获取I2中当前像素的灰度值
                pixel_value = image2[i, j]

                # 获取I1中当前像素的灰度值
                replacement_value = image1[i, j]
                # 检查I2当前像素的灰度值中的该位是否为1
                if pixel_value & binary_value:
                    # 将I1对应像素的灰度值中的该位设置为1
                    replacement_value |= binary_value
                else:
                    # 将I1对应像素的灰度值中的该位设置为0
                    replacement_value &= ~binary_value

                # 将替换后的灰度值赋给结果图像的当前像素
                result_image[i, j] = replacement_value
        result_images.append(result_image)
    return result_images

# 指定I1和I2的灰度图像路径
image1_path = 'hw01-I1.jpeg'
image2_path = 'hw01-I2.jpeg'

# 执行像素值替换
result_images = replace_pixel_values(image1_path, image2_path)

for i in range(8):
    # 保存结果图像
    cv2.imwrite('question3/hw01-I1-{}.jpeg'.format(i), result_images[i])
```

替换后的图像将展现 I2 的灰度值特征，但仍然保留了 I1 的结构和分布。同时替换的位数越低，结果图片与 I2 就越相似。

4. 将附带彩色图像 I0 的 R、G、B 通道中某个或某几个通道做与问题 3 类似的处理。对结

果进行分析。

```python
def replace_pixel_values(image1_path, image2_path,channel):
    # 读取I1和I2的灰度图像
    image1 = cv2.imread(image1_path)
    image2 = cv2.imread(image2_path, cv2.IMREAD_GRAYSCALE)
    b, g, r = cv2.split(image1)
    channel_pixels = eval(channel)
    # 获取图像宽度和高度
    height, width = channel_pixels.shape
    result_images = []
    for k in range(8):
        binary_value = 1 << k
        # 创建结果图像，与输入图像相同大小
        result_image = np.zeros((height, width), dtype=np.uint8)

        # 遍历图像的每个像素
        for i in tqdm.tqdm(range(height)):
            for j in range(width):
                # 获取I2中当前像素的灰度值
                pixel_value = image2[i, j]

                # 获取I1中当前像素的灰度值
                replacement_value = channel_pixels[i, j]
                # 检查I2当前像素的灰度值中的该位是否为1
                if pixel_value & binary_value:
                    # 将I1对应像素的灰度值中的该位设置为1
                    replacement_value |= binary_value
                else:
                    # 将I1对应像素的灰度值中的该位设置为0
                    replacement_value &= ~binary_value

                # 将替换后的灰度值赋给结果图像的当前像素
                result_image[i, j] = replacement_value
        if channel == 'b':
            result_image = cv2.merge((result_image, g, r))
        elif channel == 'g':
            result_image = cv2.merge((b, result_image, r))
        elif channel == 'r':
            result_image = cv2.merge((b, g, result_image))
        else:
            raise ValueError("Invalid channel. Please use 'b', 'g', or 'r'.")
        result_images.append(result_image)
    return result_images

# 指定I1和I2的灰度图像路径
image1_path = 'hw01-I0.jpeg'
image2_path = 'hw01-I2.jpeg'

# 执行像素值替换
result_images = replace_pixel_values(image1_path, image2_path,'r')

for i in range(8):
    # 保存结果图像
    cv2.imwrite('question4/hw01-I0-{}.jpeg'.format(i), result_images[i])
```


南京大学


南京大学


南京大学


南京大学


南京大学


南京大学

与题目三一样替换后的图像将展现 I2 的灰度值特征，但仍然保留了 I0 的结构和分布。同时替换的位数越低，结果图片与 I2 就越相似。同时，由于我替换的是 R 通道，所以随着位数降低，I0 的 R 通道特征更为明显，且与 I2 的特征更为相似。