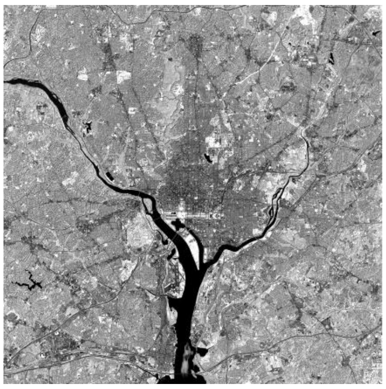
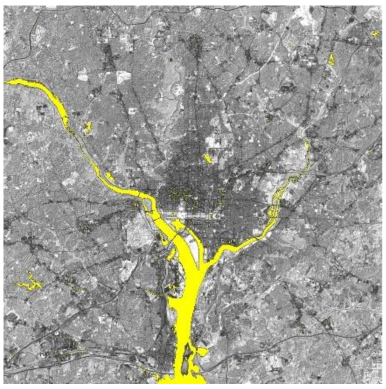
DIP homework 5

4109061012 陳柏翔

Problem 8：

```
# 4109061012 B.S.Chen
import cv2
import numpy as np

IMAGE_PATH = "homework_5/src/Fig0110(4)(WashingtonDC Band4).TIF"
TARGET_FOLDER = "homework_5/result"
TH = 40

def pseudo_color_process(img: np.ndarray, threshold: int) -> np.ndarray:
    """ Problem(a): the range of gray levels under the `threshold` will be yellow after processing,
        the other range of gray levels will be remained. """
    new_img = np.empty((img.shape[0], img.shape[1], 3))
    new_img = np.stack([img, img, img], axis=-1)
    new_img[img < threshold, :] = [0, 255, 255]  # yellow
    return new_img

if __name__ == "__main__":
    # Problem (b): using Fig. 1.10(4)
    img = np.asarray(cv2.imread(IMAGE_PATH, cv2.IMREAD_GRAYSCALE), dtype=np.uint8)
    img = pseudo_color_process(img, TH)
```

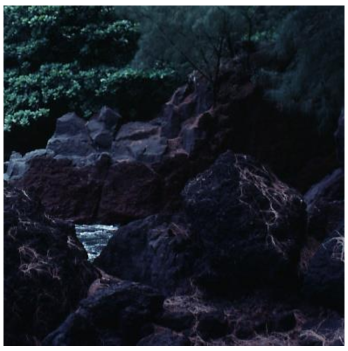

Original image

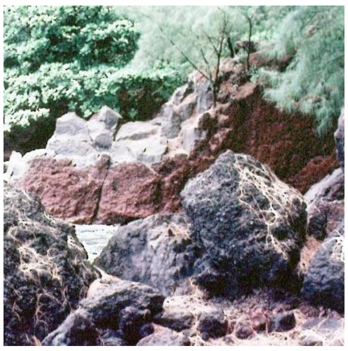

Processed image

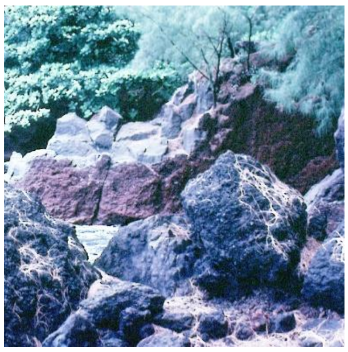
Problem 9：

```
# 4109061012 B.S.Chen
import cv2
import numpy as np

IMAGE_PATH = "homework_5/src/Fig0635(bottom_left_stream).tif"
TARGET_FOLDER = "homework_5/result"

def histogram(img: np.ndarray, value_range: tuple, normalize: bool) -> np.ndarray:
    """ Compute histogram of input image (from homework 2) """
    img = img.ravel()
    hist = np.zeros(value_range[1] - value_range[0])
    for i in range(img.shape[0]):
        hist[img[i]] += 1
    if normalize:
        hist = hist / img.shape[0]
    return hist

def hist_equalize(img: np.ndarray) -> tuple[np.ndarray, np.ndarray]:
    """ Histogram equalization (from homework 2) """
    hist = histogram(img, (0, 256), normalize=True)
    cumsum_hist = np.cumsum(hist)

    new_img = np.empty_like(img)
    for i in range(256):
        intensity = round(255 * cumsum_hist[i])
        new_img[img == i] = intensity
    return new_img
```

```
def sep_hist_equal(img: np.ndarray) -> np.ndarray:
    """ Problem (a): Histogram-equalize the R, G, and B images separately. """
    for ch in range(3):
        img[:, :, ch] = hist_equalize(img[:, :, ch])
    return img

def ave_hist_equal(img: np.ndarray) -> np.ndarray:
    """ Problem (b): Form an average histogram from the three histograms first. """
    ave_hist = histogram(img[:, :, 0], (0, 256), True) \
            + histogram(img[:, :, 1], (0, 256), True) \
            + histogram(img[:, :, 2], (0, 256), True)
    ave_hist = ave_hist / 3
    cumsum_hist = np.cumsum(ave_hist)

    trans_func = np.empty(256)
    for i in range(256):
        intensity = round(255 * cumsum_hist[i])
        trans_func[i] = intensity

    new_img = np.empty_like(img)
    for i in range(256):
        new_img[img == i] = trans_func[i]
    return new_img


if __name__ == "__main__":
    orig_img = np.asarray(cv2.imread(IMAGE_PATH, cv2.IMREAD_COLOR), dtype=np.uint8)

    img = sep_hist_equal(orig_img.copy())
    cv2.imwrite(f"{TARGET_FOLDER}/P9(a)_Result.jpg", img.astype(np.uint8))
    cv2.imwrite(f"{TARGET_FOLDER}/P9(a)_Result.tif", img.astype(np.uint8))

    img = ave_hist_equal(orig_img)
    cv2.imwrite(f"{TARGET_FOLDER}/P9(b)_Result.jpg", img.astype(np.uint8))
    cv2.imwrite(f"{TARGET_FOLDER}/P9(b)_Result.tif", img.astype(np.uint8))
```


Original

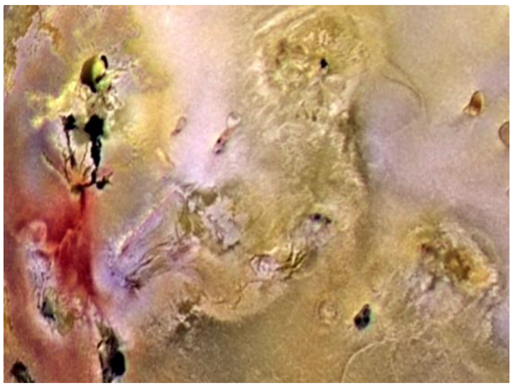

Hist. equal. (a)

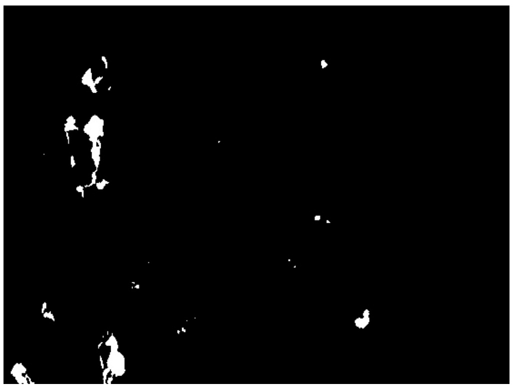

Hist. equal. (b)

由於(a)是分別對 RGB 做 histogram equalization，而(b)則是綜合考量(平均)的結果，所以(a)會相較於(b)更能看出 RGB 的色彩，(b)則是會因為 RGB 的強度較為一致顯得色彩較不鮮豔明顯。

Problem 10：

```
# 4109061012 B.S.Chen
import cv2
import numpy as np

IMAGE_PATH = "homework_5/src/Fig0628(b)(jupiter-Io-closeup).tif"
TARGET_FOLDER = "homework_5/result"
RANGE_X, RANGE_Y = [115, 135], [385, 405]

def image_segement(img: np.ndarray, sel_x: tuple, sel_y: tuple) -> np.ndarray:
    new_img = np.zeros(img.shape[0 : 2])
    sel_region = img[sel_y[0]: sel_y[1], sel_x[0]: sel_x[1], :]

    mean = np.mean(sel_region, axis=(0, 1))
    dev = np.std(sel_region, axis=(0, 1))

    th_low = mean - 1.25 * dev
    th_high = mean + 1.25 * dev
    seg = (img[:, :, 0] >= th_low[0]) & (img[:, :, 0] < th_high[0]) \
        & (img[:, :, 1] >= th_low[1]) & (img[:, :, 1] < th_high[1]) \
        & (img[:, :, 2] >= th_low[2]) & (img[:, :, 2] < th_high[2])

    new_img[seg] = 255
    return new_img

if __name__ == "__main__":
    img = np.asarray(cv2.imread(IMAGE_PATH, cv2.IMREAD_COLOR), dtype=np.uint8)
    img = image_segement(img, RANGE_X, RANGE_Y)
    cv2.imwrite(f"{TARGET_FOLDER}/P10_Result.jpg", img.astype(np.uint8))
```


Original


Segment