影像處理 HW1

學號 ：41147047S

系級 ： 資工 115

作者 ： 黃國展

主要程式邏輯 (來自 41147047S_影像處理_HW1.ipynb)

```
def ordered_dithering(img, dither_matrix):
    """
    使用給定的抖色矩陣對灰階圖像應用有序抖色 (Ordered Dithering)。

    參數:
    img (numpy.ndarray): 灰階圖像，像素值範圍為 [0, 255]。
    dither_matrix (numpy.ndarray): 抖色矩陣 (Dithering Matrix)，用於決定像素變化的閥值。

    回傳:
    binary_img (numpy.ndarray): 抖色後的二值化圖像 (黑白圖像)。
    """
    h, w = img.shape  # 取得圖像的高度 (h) 和寬度 (w)
    m_h, m_w = dither_matrix.shape  # 取得抖色矩陣的高度 (m_h) 和寬度 (m_w)

    # 根據原圖大小擴展抖色矩陣，使其覆蓋整個圖像
    # 透過 np.tile 重複填充抖色矩陣，確保大小與圖像相符
    D = np.tile(dither_matrix, (h // m_h + 1, w // m_w + 1))[:h, :w]

    # 依照抖色矩陣的閥值決定每個像素點的顏色
    # 若像素值大於對應的抖色閥值，則設為 255 (白色)，否則設為 0 (黑色)
    binary_img = np.where(img > D, 255, 0).astype(np.uint8)

    return binary_img  # 回傳二值化後的抖色圖像

def extended_dithering(img, dither_matrix):
    """
```

```
    使用擴展抖色 (Extended Dithering) 來產生 4 級灰階效果。

    參數:
    img (numpy.ndarray): 灰階圖像，像素值範圍為 [0, 255]。
    dither_matrix (numpy.ndarray): 抖色矩陣 (Dithering Matrix)。

    回傳:
    result (numpy.ndarray): 具有 4 級灰階的抖色圖像。
    """
    h, w = img.shape  # 取得圖像的高度 (h) 和寬度 (w)
    m_h, m_w = dither_matrix.shape  # 取得抖色矩陣的高度 (m_h) 和寬度
(m_w)

    # 計算量化級別 (4 級灰階)，將像素值分成 4 個等級
    Q = (img // 85).astype(np.uint8)  # 85 ≈ 255/3，將 0~255 分為 0, 85,
170, 255

    # 產生與圖像相同大小的抖色矩陣
    D = np.tile(dither_matrix, (h // m_h + 1, w // m_w + 1))[:h, :w]

    # 根據抖色矩陣進行級別調整
    # 若像素的殘差 (img - 85 * Q) 大於抖色矩陣的值，則額外增加 1 級
    I_prime = Q + np.where(img - 85 * Q > D, 1, 0)

    # 轉換回 0, 85, 170, 255 的像素範圍，適用於 4 級灰階顯示
    result = (I_prime * 85).astype(np.uint8)

    return result  # 回傳 4 級灰階的抖色圖像

def process_image(image):
    """
    將輸入的彩色圖像轉換為灰階，並套用有序抖色 (Ordered Dithering) 和擴展抖色
(Extended Dithering)。

    參數:
    image (numpy.ndarray): 彩色圖像，格式為 BGR。

    回傳:
```

```
    gray (numpy.ndarray): 轉換為灰階的圖像。
    binary_dithered (numpy.ndarray): 應用有序抖色後的黑白圖像。
    extended_dithered (numpy.ndarray): 應用擴展抖色後的 4 級灰階圖像。
    """
    # 轉換彩色圖像為灰階
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # 定義 4×4 抖色矩陣，用於有序抖色
    D2 = np.array([[0, 128, 32, 160],
                   [192, 64, 224, 96],
                   [48, 176, 16, 144],
                   [240, 112, 208, 80]], dtype=np.uint8)

    # 定義 2×2 抖色矩陣，用於擴展抖色
    D1 = np.array([[0, 56],
                   [84, 28]], dtype=np.uint8)

    # 套用有序抖色 (二值化圖像)
    binary_dithered = ordered_dithering(gray, D2)

    # 套用擴展抖色 (4 級灰階圖像)
    extended_dithered = extended_dithering(gray, D1)

    return gray, binary_dithered, extended_dithered  # 回傳處理後的圖像

def gradio_interface(image):
    """
    Gradio 介面函式，讓使用者上傳圖片並進行抖色處理。

    參數:
    image (numpy.ndarray): 上傳的彩色圖像。

    回傳:
    (tuple): 轉換後的灰階圖像、有序抖色後的黑白圖像、擴展抖色後的 4 級灰階圖像。
    """
    gray, binary_dithered, extended_dithered = process_image(image)
    return gray, binary_dithered, extended_dithered  # 回傳處理結果
```

網頁運行結果（ 使用 ./image/rick-astley.png）

以下為參考輸入以及對應結果

彩色版以及灰階處理後的圖片

partA 以及 partB 的運行結果

通過這次作業，了解了有序抖色和擴展抖色的原理和實現方法。

有序抖色 (Ordered Dithering) 使用一個固定的抖色矩陣來處理圖像，這種方法適合於將灰階圖像轉換為二值化圖像 (黑白圖像)。其優點是計算速度快，適合於需要快速處理的應用場景，但缺點是可能會產生規律性的圖案，影響圖像質量。

擴展抖色 (Extended Dithering) 則是將圖像轉換為具有多級灰階的圖像，例如 4 級灰階。這種方法能夠保留更多的圖像細節和層次感，適合於需要更高質量圖像的應用場景，但相對計算複雜度較高。