# HW3 Report

108062119 鄭幃謙

---

## 1. Try the two search ranges for two macroblock sizes by using the two search methods. The reference image is 40.jpg, and the target image is 42.jpg.

### Implementation

```
def fullSearch(tar, ref, row, col, searchRange):
```

```
    best = None
    minSAD = np.inf
    for i in range(-searchRange, searchRange + 1):
        for j in range(-searchRange, searchRange + 1):
            if 0 <= row + i < ref.shape[0] and 0 <= col + j < ref.shape[1]:
                nowSAD = calSAD(ref[row + i, col + j], tar[row, col])
                if nowSAD < minSAD:
                    minSAD = nowSAD
                    best = (i, j)
    return np.array(best)
```

full search的做法較單純，在search range內的blocks中找到SAD最小的作為結果。

```
def logSearch(tar, ref, row, col, searchRange):
```

```
    best = None
    minSAD = np.inf
    step = searchRange // 2
    center = (0, 0)
    while step >= 1:
        for i in [-step, 0, step]:
            for j in [-step, 0, step]:
                if i * j != 0:
                    continue
                if 0 <= row + center[0] + i < ref.shape[0] and 0 <= col +
    center[1] + j < ref.shape[1]:
                    nowSAD = calSAD(ref[row + center[0] + i, col + center[1] +
    j], tar[row, col])
                    if nowSAD < minSAD:
                        minSAD = nowSAD
                        best = (i, j)
        center = updateCenter(center, best, row, col, ref.shape)
        best = (0, 0)
        step //= 2
    for i in [-1, 0, 1]:
        for j in [-1, 0, 1]:
            if 0 <= row + center[0] + i < ref.shape[0] and 0 <= col +
    center[1] + j < ref.shape[1]:
```

```
            nowSAD = calSAD(ref[row + center[0] + i, col + center[1] + j],
tar[row, col])
            if nowSAD < minSAD:
                minSAD = nowSAD
                best = (i, j)
center = updateCenter(center, best, row, col, ref.shape)
return np.array(center)
```

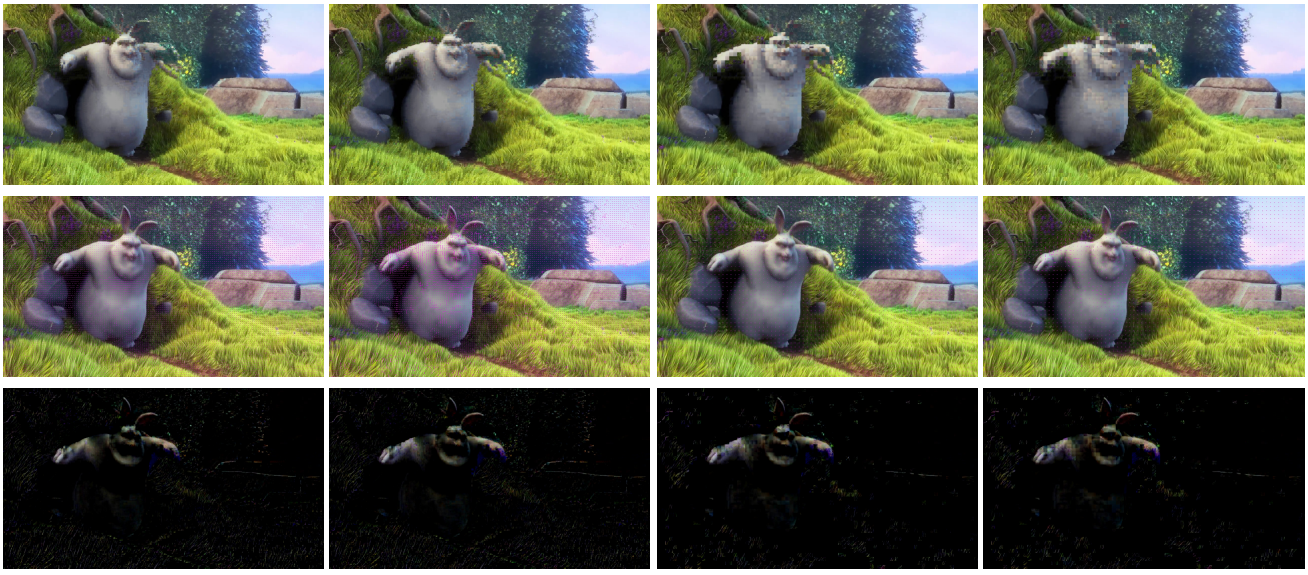
2d-logarithmic search的做法差別在於移動的center，每一輪中只檢查上下中左右五個blocks，找到最相近的更新center並縮小搜尋範圍後繼續。最後再從最近的9個block中找出最終結果。

**Result**

1. full r8b8 / r16b8 / r8b16 / r16b16

基本上full search的效果都滿好的，range大小的影響不太明顯，但是blocksize較大會使得圖片有移動處有更明顯的方塊感。

2. 2d-log r8b8 / r16b8 / r8b16 / r16b16



2d-logarithmic search的計算速度比full search快非常多，不過結果明顯看得出較差。在range較小時常常找不到最佳匹配塊，比較細節的地方（如兔子耳朵）會因為搜尋機制的關係找到差很多的方塊。在blocksize較大的結果就更差了。

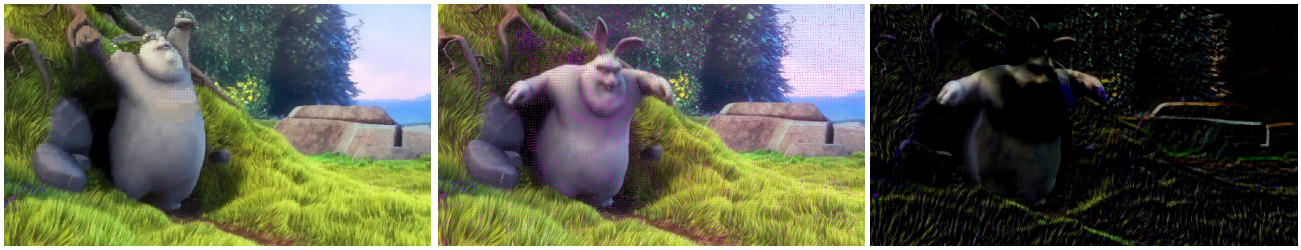
**SAD and PSNR values**

```
full_r08_b08 -- SAD: 18139668, PSNR: 31.8711
full_r16_b08 -- SAD: 17359539, PSNR: 32.0220
full_r08_b16 -- SAD: 24655644, PSNR: 30.9632
full_r16_b16 -- SAD: 24299122, PSNR: 30.9872
2d_r08_b08 -- SAD: 21934110, PSNR: 31.3365
2d_r16_b08 -- SAD: 21930084, PSNR: 31.3133
2d_r08_b16 -- SAD: 26948698, PSNR: 30.7740
2d_r16_b16 -- SAD: 27269661, PSNR: 30.6927
```

可以看出full search整體比2d-logarithmic search更好，而blocksize越小越好，search range越大越好。

## 2. Try the full search method with search range p=8 and macroblock sizes = 8x8. The reference image is 40.jpg, and the target image is 51.jpg.

**Result**



因為移動的範圍過大而搜尋範圍過小無法找到匹配塊，雖然圖形輪廓上大致能看出來但細節失真很嚴重。

**PSNR value**

```
SAD: 19796761, PSNR: 31.6623
```

與full_r8_b8相比，其實只略差一些。可見full search效果很好。

## 3. Analyze the time complexity.

**Time complexity**

full search的time complexity應為 $O(p^2)$，而2d-logarithmic search的time complexity應為 O(log(p))，p為search range。

**Execution time**

```
full_r08_b08 -- runtime: 12.2774
full_r16_b08 -- runtime: 43.5006
full_r08_b16 -- runtime: 3.4654
full_r16_b16 -- runtime: 11.1729
2d_r08_b08 -- runtime: 1.1816
2d_r16_b08 -- runtime: 1.4203
2d_r08_b16 -- runtime: 0.3458
2d_r16_b16 -- runtime: 0.4134
```

基本上算是符合預期，full search的部分range變為2倍時runtime約變為4倍。2d-logarithmic search的部分range從8變為16時runtime約變為1.3倍 ~= $\log_2{16}$ / $\log_2{8}$ = 4 / 3。