視訊處理作業報告

HW2

Full and Fast Motion Estimation Algorithms

授課教授：柳金章

學　　生：楊憲閔

學 　號：613410047

Due date：2024/04/30

Date hand in：2024/05/08

目錄

[Technical description 3](#_Toc403519772)

[Experimental results 7](#_Toc403519773)

[Discussions 13](#_Toc403519774)

[References and Appendix 14](#_Toc403519775)

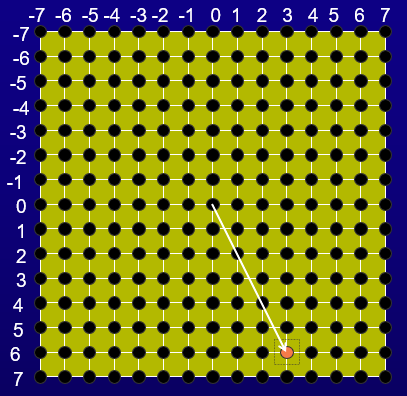
# Technical description

Motion Estimation 是視訊壓縮技術中的核心技術，主要透過「區塊比對」（Block Matching）來估計前後frame間像素的移動情形，藉由 motion vector 進行預測與壓縮。在本次作業中，我們實作了三種常見的 motion estimation 演算法，並針對不同 block 大小與搜尋範圍進行比較，方法如下：

1. Full search algorithm：

Full Search 是最精確但計算量最大的區塊比對方法。對每個 8×8 或 16×16 的區塊，會在 reference frame 的整個搜尋範圍（±8 或 ±16 像素）中搜尋所有可能的位置，計算每個候選區塊與 current block 的 Mean Absolute Difference (MAD)，以找出最佳匹配點。其優點是能找到最小誤差的 motion vector，但缺點是運算時間非常長。實作流程如下：

1. 準備前一個frame(reference frame)與現在的frame(target frame)。
2. 將target frame和reference frame中的每個mini block拆分出來(大小依需求，例如說題目要求為8x8或16x16)。
3. 以當前mini block為中心，定義一個正方形的搜索範圍(大小也依需求，例如說題目要求為8或16)
4. 對target frame中的每個mini block，在reference frame中的搜索範圍內搜尋所有可能的reference block。並對每個reference block計算它和當前mini block之間的平均絕對誤差 (mean absolute differences，MAD）。
5. 重複步驟(4)，直到搜尋完target frame中所有的mini block。並紀錄所有的motion vector傳送給編碼器，以便將它們壓縮並傳送到解碼器。
6. 在解碼器使用motion vector和reference frame來重建 frame。

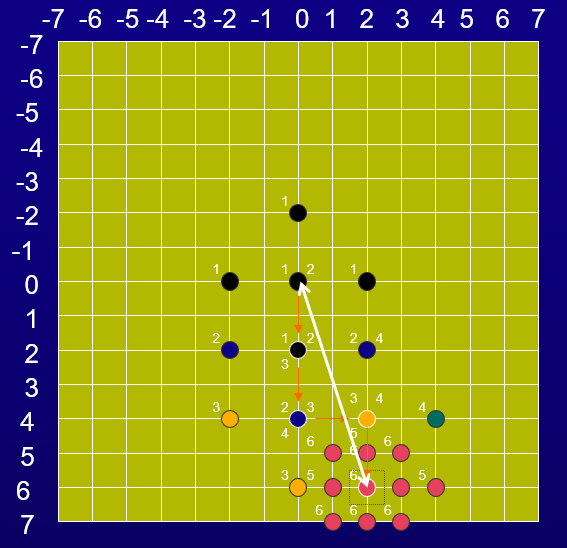


圖(1) Full search algorithm示例圖。

1. 2D logarithm search algorithm：

2D Logarithmic Search 為一種快速搜尋方法，使用十字形範圍來遞迴縮小搜尋區域。每輪搜尋會以目前的最佳位置為中心，使用step size搜尋上下左右4個點，計算 MAD並更新中心位置。當 step size 降為 1 時停止。此法大幅減少搜尋次數，提升效率，但可能錯過全域最佳解。步驟如下：

1. 透過step size找出候選位置(4點加中心)，並計算個別的MAD。
2. 利用MAD更新中心點，即中心點更新為MAD最小的候選點。
3. 計算新的中心點與舊中心點的差距作為motion vector。
4. 重複步驟(1)至(3)，直到找到最佳點並回傳motion vector。

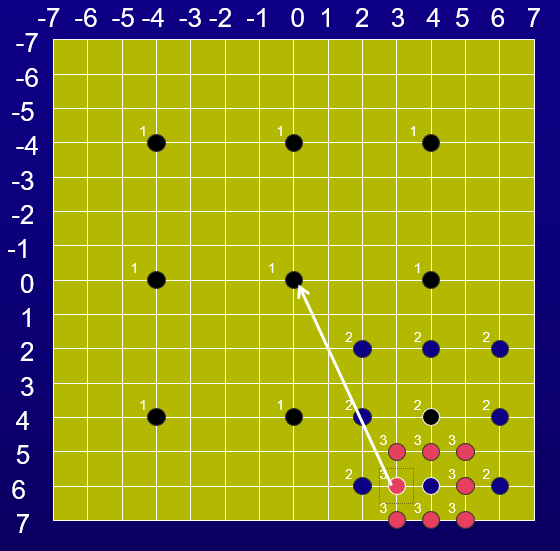


圖(2) 2D logarithm search algorithm示例圖。

1. 3-step search algorithm：

3-Step Search 是另一種快速搜尋法，以中心點利用step size搜尋九宮格周圍點，找出誤差最小的位置作為新中心，並將步長減半後繼續搜尋。相比於 2D Log Search，3-Step Search 在搜尋範圍內覆蓋更全面，因此在精度與效率間取得較佳平衡。步驟如下：

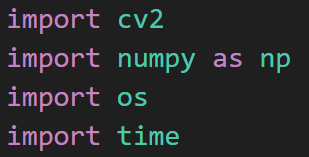
1. 定義搜尋區域(依據block size與search size)並設定step size，此決定了每次搜尋時 block移動的距離。step越大，搜尋速度就越快，但準確度則會降低。
2. 以搜尋區域中心為起始點，按照設定好的step size選出9個候選點，計算MAD來得到兩個block的相似度。
3. 利用MAD更新中心點，並將step size減半。
4. 重複步驟(2)至(3)，直到找到最佳點並回傳motion vector。



圖(3) 3-step search algorithm示例圖。

# Experimental results

1. 程式執行流程：
2. 確保已安裝相關module，本次作業使用module如下所示:



圖(4) 會使用到的module

1. 進到作業的目錄底下，會看到HW2\_test\_sequence資料夾、main.py檔及此pdf檔。點右鍵按在終端中開啟，輸入python main.py，程式即開始執行。
2. 程式會讀取兩個sequence的資料並進行作業要求的algorithm，並會一一進行不同的block size、search range等等測試，並輸出於各自的輸出資料夾中。
3. 程式執行結果：
   1. 輸出形式：

階層式架構，例如說現在我對s1做full search，block size為16、search range為8，則輸出結果(重建回來的frame)會放置於./s1/full\_search/search\_range\_8\_block\_size\_16底下，依此類推。並該次的所有PSNR結果會放置於對應的資料夾底下。

* 1. 結果比較：

以下進行不同motion estimation和search range的PSNR比較(四捨五入至小數點第2位，單位為db)。

表(1) Search range為8，block大小為8×8，不同方法對s1做motion estimation之PSNR。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1.bmp | 25.bmp | 50.bmp | 75.bmp | 90.bmp |
| Full search | 38.19 | 33.48 | 32.10 | 32.21 | 31.00 |
| 2D logarithm search | 34.82 | 31.11 | 30.37 | 30.35 | 29.68 |
| 3-step search | 36.18 | 31.69 | 31.06 | 30.95 | 30.16 |

表(2) Search range為8，block大小為16×16，不同方法對s1做motion estimation之PSNR。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1.bmp | 25.bmp | 50.bmp | 75.bmp | 90.bmp |
| Full search | 36.90 | 30.15 | 28.87 | 29.00 | 28.49 |
| 2D logarithm search | 34.22 | 28.68 | 27.70 | 27.87 | 27.48 |
| 3-step search | 36.27 | 29.49 | 28.56 | 28.45 | 28.15 |

表(3) Search range為16，block大小為8×8，不同方法對s1做motion estimation之PSNR。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1.bmp | 25.bmp | 50.bmp | 75.bmp | 90.bmp |
| Full search | 38.27 | 33.53 | 32.25 | 32.33 | 31.06 |
| 2D logarithm search | 34.65 | 30.77 | 30.27 | 30.19 | 29.48 |
| 3-step search | 35.59 | 31.81 | 30.89 | 30.95 | 30.19 |

表(4) Search range為16，block大小為16×16，不同方法對s1做motion estimation之PSNR。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1.bmp | 25.bmp | 50.bmp | 75.bmp | 90.bmp |
| Full search | 36.99 | 30.42 | 29.40 | 29.04 | 28.75 |
| 2D logarithm search | 34.18 | 28.73 | 27.75 | 27.95 | 27.59 |
| 3-step search | 36.04 | 29.59 | 28.62 | 28.43 | 28.12 |

(s2之frame非0起始，因此結果取於PSNR輸出中的i-1.bmp之結果，例如2.bmp就是看Reconstructed frame 1 PSNR這個項目)

表(5) Search range為8，block大小為8×8，不同方法對s2做motion estimation之PSNR。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 2.bmp | 15.bmp | 30.bmp |
| Full search | 36.46 | 30.03 | 30.72 |
| 2D logarithm search | 34.70 | 28.44 | 29.43 |
| 3-step search | 35.70 | 28.38 | 29.75 |

表(6) Search range為8，block大小為16×16，不同方法對s2做motion estimation之PSNR。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 2.bmp | 15.bmp | 30.bmp |
| Full search | 35.40 | 26.92 | 27.73 |
| 2D logarithm search | 34.00 | 25.96 | 27.10 |
| 3-step search | 34.85 | 26.40 | 27.04 |

表(7) Search range為16，block大小為8×8，不同方法對s2做motion estimation之PSNR。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 2.bmp | 15.bmp | 30.bmp |
| Full search | 36.54 | 30.96 | 31.03 |
| 2D logarithm search | 34.53 | 29.27 | 29.65 |
| 3-step search | 35.53 | 29.22 | 29.98 |

表(8) Search range為16，block大小為16×16，不同方法對s2做motion estimation之PSNR。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 2.bmp | 15.bmp | 30.bmp |
| Full search | 35.43 | 27.59 | 27.89 |
| 2D logarithm search | 33.85 | 27.14 | 27.17 |
| 3-step search | 34.81 | 26.65 | 27.04 |

表(9) 不同motion estimation和search range與block size在s1的執行時間(示例：(block size, search range))(四捨五入至小數點第2位)。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | (8, 8) | (8, 16) | (16, 8) | (16, 16) |
| Full search | 160.69 s | 641.30 s | 47.44 s | 181.37 s |
| 2D logarithm search | 12.33 s | 17.22 s | 3.76 s | 4.30 s |
| 3-step search | 17.87 s | 26.92 s | 5.31 s | 7.60 s |

表(9) 不同motion estimation和search range與block size在s2的執行時間(示例：(block size, search range))(四捨五入至小數點第2位)。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | (8, 8) | (8, 16) | (16, 8) | (16, 16) |
| Full search | 64.30 s | 236.16 s | 17.99 s | 65.76 s |
| 2D logarithm search | 4.58 s | 6.41 s | 1.25 s | 1.50 s |
| 3-step search | 6.93 s | 9.56 s | 1.85 s | 2.60 s |

# Discussions

Full search algorithm考慮到所有可能的點，以此找到最佳匹配點，但缺點為計算量非常大，執行時間會很久。

2D logarithm search algorithm採用十字形搜尋，只抓同水平垂直方向的4點，搜尋用的window size從大範圍的搜尋逐漸縮小搜尋範圍。好處是計算量少，執行時間短，但缺點是這種十字形搜尋有時只能找到靠近最佳點的點，而不一定可以找到最佳點。

3-step search algorithm類似於2D logarithm search algorithm，只是不同之處在於它是採用九宮搜尋，每步搜尋完後window size會減半，從大範圍的搜尋逐漸縮小搜尋範圍。計算量和執行時間比2D logarithm search algorithm略長一些(因為多抓了一些候選點)，但比依然比Full search algorithm短上許多，且和2D logarithm search algorithm，3-step search algorithm找到最佳點的機會比較大(因為有更多的候選點做選擇)。

若從search range來比較，+/- 8 pixels的計算量較少，但在快速移動的場景，也就是兩張連續影像之間有明顯變化的情況下，其計算上容易無法找到最佳點，影像還原的效果較差。+/- 16 pixels雖然計算量較大，但在快速移動的場景中容易找到最佳匹配，影像還原效果較好。

若從block size來比較，8×8在高細節區中可以得到較好的motion estimation，但計算量大，其還原影像沒有平滑效果但容易有缺陷。16×16在低細節區進行motion estimation時計算量較小，其還原影像不易有缺陷但在高細節區中容易有過度平滑的現象。

# References and Appendix

<https://www.youtube.com/watch?v=jRrs9OxLTYM>

<https://www.youtube.com/watch?v=8eZ2EA1q7As>