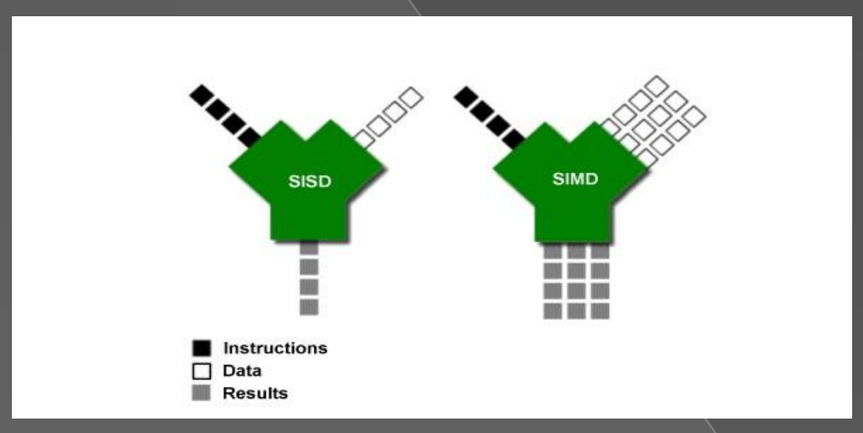
多媒體處理器實現多種演算法比較

參展人員:楊孟霖

摘要:

日常生活中，聽音樂、看影片、電腦遊戲等等都與多媒體有密切的關係，而多媒體運算通常龐大且耗時，我們這次專題目的在於如何使用單指令多資料流(SIMD)的架構將重複進行的運算以並行方式實現，以其在有限資源內對多媒體運算效率進行提升。

我們在這個專題SIMD架構採用較新的SSE指令集，來實現幾個不同的演算法，與一般方式做比較，來測試效能能獲得多大的提升，並改進找出最佳化的方式。



SIMD單指令多資料流技術 示意圖

實作內容:

1. 4x4矩陣相乘:

如圖1-1，矩陣A的第一列與矩陣B的第一行的內積結果即為矩陣C第一行第一列的資料，矩陣A的第一列與矩陣B的第二行的內積結果即為矩陣C第二行第一列的資料，以此類推，可得知相乘結果矩陣C行列的資料值取決於A的列、B的行。

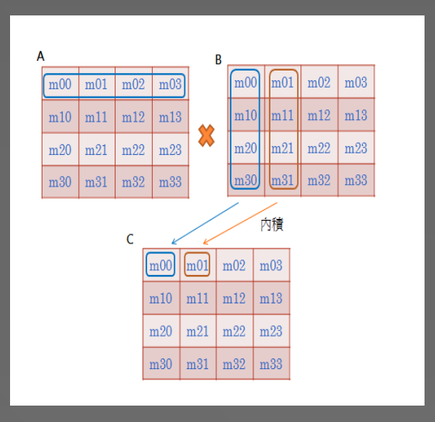


圖1-1

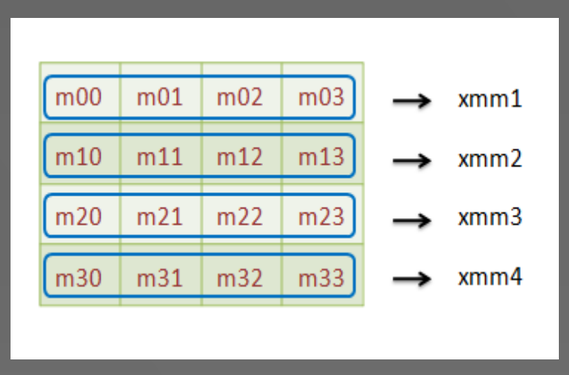


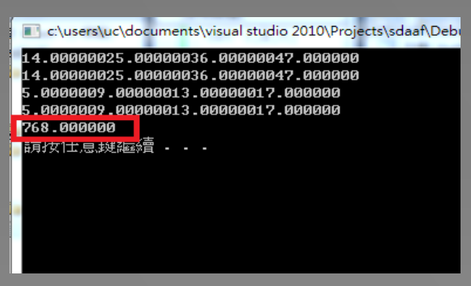
圖1-2

SSE運算暫存器為新增的8個128bits的暫存器，我們在實現方面採用單精度浮點數格式，因此一暫存器可存入4筆32bits浮點數資料。如圖1-2所示，讀取過程是將矩陣的同一列資料讀取進暫存器裡頭，因此須對圖1-1的B矩陣作轉置這個步驟才能讓A的列資料與B的行資料做內積運算。

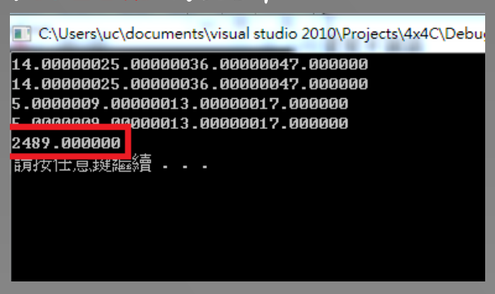
而在轉置方面，SSE指令集內有一條指令如下所示

```
_MM_TRANSPOSE4_PS (__m128 row0, __m128 row1, __m128 row2, __m128 row3)
```

此條指令可使得暫存器值直接轉置，接下來在使用SSE的dp(內積)指令即可一一算出矩陣內的值，我們在這邊測試結果可達到快C語言大約3.24倍的效率。



SSE實現結果



C語言實現結果

2. 8x8矩陣相乘:

我們這邊採用兩種方式去實作出來，第一種如圖2-1所示的方式，採用高中時所學過的矩陣相乘法，將矩陣分成4區塊，作圖上的運算方式來做矩陣的相加以及相乘。第二種方式如圖2-2則是採用施特拉森演算法用較少的矩陣相乘但相反較多的矩陣加減法來實現。

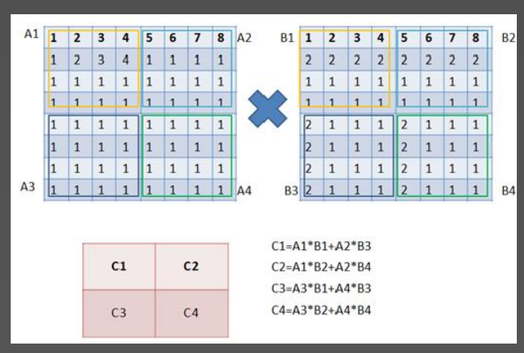


圖2-1

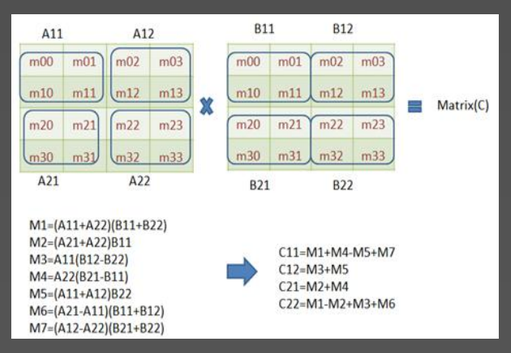


圖2-2

我們這邊區塊矩陣相乘的部分採用之前做的4x4矩陣相乘的方式，相加減則用基本的SSE運算指令add和sub指令來實現。

實作結果發現SSE一般矩陣乘法較C語言速度可達到2.47倍的效率提升，但用SSE實現施特拉森法，並無法超越一般矩陣乘法的效能。

3. 亂數排序:

我們採用圖3-1方式來實現大資料的泡沫排序，圖3-2則是實現大資料量的並行排序。

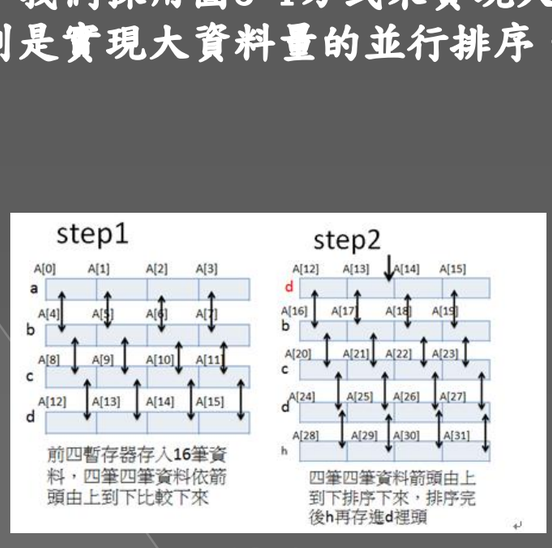


圖3-1

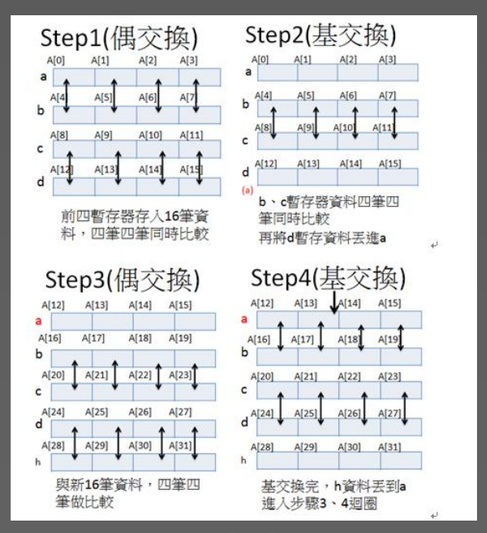
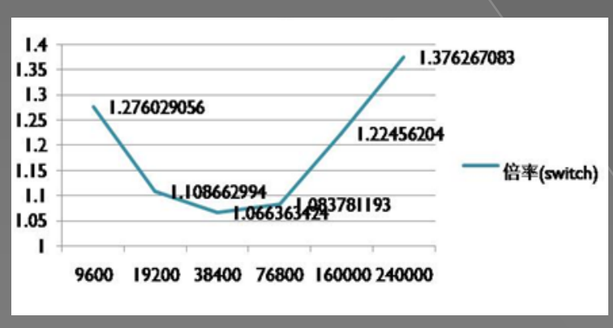
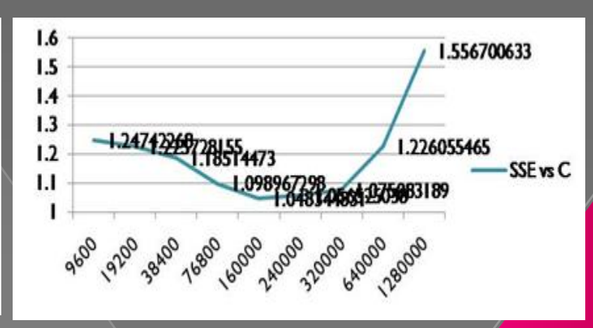


圖3-2

運算後的結果會將資料分成4等份分別排序好，最後我們在使用Merge Sort方式合併起來，下圖為SSE與C語言比較資料量與效率的關係統計出來的圖表，上圖數字為SSE的處理速率比C語言快的倍率，橫軸為資料量縱軸為執行程式時間(ms)，由圖可觀察到資料量愈龐大，倍率有上升的趨勢。



泡沫排序(SSEvsC)



並行排序(SSEvsC)

未來展望:

前面實作部份主要是掌握SSE指令集加速的方式，以及如何達到最佳加速的的效果，未來將會試著對多種關於動作識別方面的演算法進行進行分析，並且使用SSE指令集內的指令來進行改寫，動作識別為大資料量運算，期望能利用這個多媒體指令集能對動作識別演算法達到不錯的效能提升。