HW3 Mandelbulb GPU 優化報告

1. 作業背景

- **目標：** 將提供的 CPU 版 Mandelbulb 程式 (hw3_cpu.cpp) 改寫成 CUDA 版 (hw3.cu)，維持原有 CLI 輸入與 PNG 輸出行為。

2. 初始狀況

- 大型測試耗時 25～40 秒，小型測試受排程影響偶有 timeout。

- nvprof 顯示 **achieved occupancy 約 0.46**、每次 kernel 執行 **10^{12}～10^{13} 指令**，顯示計算密集且暫存器為瓶頸。

3. 優化步驟與實作想法

1. md()（距離估計）重構

- 使用 sincosf 取代重複的三角函數呼叫。
- 將固定 power=8 的 powf 改為 $r^2/r^4/r^6/r^7/r^8$ 乘法展開。
- 實作時需注意 r 可能為零或非常小，因此加上倒數保護與 clamp。
- **原因：** 減少每步 FLOP 與暫存器壓力，使迭代更快。

2. softshadow() 早期終止

- 當遮蔽度已接近 1 (res ≥ 0.99) 且距離較遠 (h ≥ 0.5、t ≥ 5)，直接回傳 1。
- **想法：** 光線已遠離障礙物，繼續迴圈只是空轉，提早返回可節省大量步數。

3. 光照與著色標量化

- 在 render_kernel 中將 vec3 的暫存變數改用純 scalar（例如 col_r、

col_g、col_b），並改寫 vpow。

- **想法：** 縮短暫存器生命週期、避免臨時物件造成的額外開銷。

4. **暫存器與佔用率提示**

- 加入 __launch_bounds__(256, 4) 與 -maxrregcount=44，提醒編譯器嘗試降低寄存器使用。

- 雖然 ptxas 最終仍使用 48 reg/thread，但後續重構保留空間。

4. 面臨的課題與處理方式

課題 影響 解法

nvprof 緩衝不足導致 testcase08 metrics 缺失 難以比較優化前後 分測資分批執行 nvprof（考慮改用 Nsight Compute）。

寄存器數始終停在 48 SM 同時僅能排 4 個 block 經由 shading scalar 化縮短 lifetime；未來可再拆函式或改 float3。

小測資偶爾超時 測環境排程導致誤判 使用 judge/多次測試確認真實 kernel 時間。

5. 效能比較（節錄）

Testcase 原始時間 (sec) 優化後 (sec)

05 (4096^2) ~24.98 **9.15**

06 (4096^2) ~28.9 **16.6**

07 (4096^2) ~33.2 **9.5**

08 (4096^2) ~41.9 **20.1**

00–04 受排程影響 **<5 s**（judge 報告為 0.x sec）

hw3-judge 最佳紀錄由 {4 13.83} 提高到 {4 12.57}。

6. 未來持續優化方向

1. **寄存器重構：** 將 shading 分拆成 helper 函式或使用 float3 搭配手寫向量計算，讓寄存器實際降至 44 以下，再配合 __launch_bounds__ 以提高 SM occupancy。

2. **動態 ray marching：** 在 trace() 加入 miss 半徑/自適應步長，以減少空場景迴圈數。

3. **使用 Nsight Compute：** 可避免 nvprof 緩衝問題，獲得更多硬體指標。