HW3

111550143 林彥佑

* Implementation

particleRelaxation():

particleRelaxation()函式負責實作流體模擬中的粒子鬆弛處理，以避免粒子之間的重疊。

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

AI 產生的內容可能不正確。

根據每個粒子的位置，計算其所屬的relaxation cell，並將粒子index放入對應的cell中。

一張含有 文字, 螢幕擷取畫面, 軟體, 陳列 的圖片

AI 產生的內容可能不正確。

進行iterations次的疊代，對於每個relaxation cell中的粒子，掃描該cell周圍3×3的鄰近cell，以找出可能與之接觸的粒子，檢查粒子間距是否小於 2 \* particle\_radius，若是則視為碰撞。

使用公式

一張含有 圖表, 圓形, 文字, 行 的圖片

AI 產生的內容可能不正確。將兩粒子沿其連線方向等量推開。

transferVelocities(bool to\_cell):

這個函式實作了PIC/FLIP模擬中粒子與MAC格點速度場之間的雙向速度轉換，根據to\_cell執行particle to cell或cell to particle的轉換邏輯。

一張含有 文字, 螢幕擷取畫面, 軟體, 作業系統 的圖片

AI 產生的內容可能不正確。

首先，reset所有cell速度以及cell types，不是SOLID先設為AIR，再判斷該cell是否有粒子，設為FLUID。

一張含有 文字, 螢幕擷取畫面, 陳列, 軟體 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 軟體, 陳列 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

AI 產生的內容可能不正確。

calculate the 4 cells that should contribute to the biliear interpolation of each particle. 照講義公式，先-l/2得出最左下角的(I, j)，得出i0, j0, i1, j1，然後計算deltax, deltay，我有點被講義給的公式誤導，我原本是分成x和sample\_x去計算，x沒有也減去l/2，導致粒子亂飛，再來如果照講義公式，sample\_x會被除以兩次l，這是完全錯誤的，我不知道我是否誤解講義的意思，講義上是否在算floor(sample\_x/l)後要\*l，雖說影片有講到比較正確的算法，但講義還是有混淆到我。得出delta之後套公式

一張含有 文字, 字型, 圖表, 螢幕擷取畫面 的圖片

AI 產生的內容可能不正確。

得出w1~w4，將該粒子的速度分別加權加總到這四個格點的對應速度分量上（x或y），並在 weight\_sum中記錄該格點累積了多少權重，最後將格點的速度除以其累積權重，完成加權平均，並同時備份至prev\_cell\_velocities供下一步FLIP使用。

to\_cell=false，將格點上的速度重新插值還原回粒子，使粒子能夠接續進行運動，對於每個粒子，我們同樣根據其位置與格點進行biliear interpolation，要排除兩cell皆為AIR的速度干擾，用isValid去判斷是否採用速度，使用有效的格點速度計算

PIC速度：直接插值得到的新速度。

FLIP速度：透過cell\_velocities - prev\_cell\_velocities取得的差值，加回粒子原速度。

最終將PIC與FLIP結果依據flip\_ratio混合，更新回粒子速度。

一張含有 文字, 字型, 螢幕擷取畫面, 行 的圖片

AI 產生的內容可能不正確。

updateDensity():

updateDensity()的目的是計算每個格點的密度（cell density），作為下一步 solveIncompressibility()判斷是否需要壓縮修正的依據。

一張含有 文字, 螢幕擷取畫面, 軟體, 陳列 的圖片

AI 產生的內容可能不正確。 一張含有 文字, 螢幕擷取畫面, 字型 的圖片

AI 產生的內容可能不正確。

一開始將所有格點的密度cell\_densities[i][j]歸零，對每個粒子先將其位置調整為落在格點中心座標（即 x -= 0.5 \* cell\_dim, y -= 0.5 \* cell\_dim），接著，根據粒子位置找到其對應的上下左右四格，利用biliear interpolation計算四個權重w1~w4，並將每個粒子對應的權重加總至四個格點的密度，僅在 particle\_rest\_density == 0.0時計算rest\_density，掃描所有格點，若該格點型態為FLUID，則累加其密度值並統計數量。

一張含有 文字, 字型, 白色, 行 的圖片

AI 產生的內容可能不正確。

solveIncompressibility():

solveIncompressibility()的目的是讓整個流體系統滿足不可壓縮性條件，也就是確保每個流體格子的流入與流出速率相等。

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

AI 產生的內容可能不正確。

外層使用iterations迴圈對整體流場進行多次修正，使每個格子的divergence趨近0，內層針對所有fluid格子逐一處理，要判斷該格的鄰居是否是SOLID，計算上下左右格非SOLID個數，並得出上下左右速度，套過公式:



得出divergence，若density\_correction == true，額外加入修正項

一張含有 字型, 筆跡, 文字, 書法 的圖片

AI 產生的內容可能不正確。

修正量會根據目前divergence、鄰近可流通的格子數（上下左右非 solid）與over\_relaxation 係數決定

一張含有 字型, 白色, 文字, 符號 的圖片

AI 產生的內容可能不正確。

左/下方向的速度加上correction，右/上方向的速度減去correction

* Results & Discussion

cell\_dim:

cell\_dim=10:

正常，如demo影片所示。

cell\_dim=1:

一張含有 螢幕擷取畫面, 軟體, 多媒體軟體, 文字 的圖片

AI 產生的內容可能不正確。一張含有 文字, 螢幕擷取畫面 的圖片

AI 產生的內容可能不正確。

每格只包含少量粒子，造成每格的密度估計較不穩定（浮動大）。在FLIP模型中，粒子速度從格點插值回來的品質變差。細節多、可以模擬出更精細的流體動態，但初始流速慢。

cell\_dim=50

每格涵蓋大量粒子，導致密度估計過於模糊。粒子的變化趨於平均化，流體行為顯得「鈍化」。流動區域會看起來像硬塊（因為速度修正範圍變少）。

一張含有 文字, 螢幕擷取畫面, 多媒體軟體, 編輯 的圖片

AI 產生的內容可能不正確。一張含有 文字, 螢幕擷取畫面, 多媒體軟體, 軟體 的圖片

AI 產生的內容可能不正確。

FLIP ratio:

當 flip\_ratio = 0.0，完全採用 PIC 方法。

當 flip\_ratio = 1.0，完全採用 FLIP 方法。

flip\_ratio = 0.0

粒子速度每次都被格點速度完全取代，模擬非常穩定，不容易跳動或爆炸，多粒子一起行動。

穩定、不抖動，水波紋或渦流會快速消散。

flip\_ratio = 1.0

粒子速度每次都加上從格點推回的速度變化量（delta）。

能量與動量保留較完整，流動細節更自然，但容易因誤差累積導致粒子亂飛，尤其在格點密度變化大或靠近邊界時，數值震盪會放大，粒子會出現單獨一顆飛的現象。

保留細節、模擬活潑、易抖動、不穩定

Stiffness coefficient:

stiffness\_coefficient = 0.0

若粒子因邊界或碰撞集中於一格，則該格密度會異常高，無機制將其膨脹釋放，速度場穩定。

stiffness\_coefficient = 1.0

正常

stiffness\_coefficient = 100.0

密度只要略高於靜態密度就會被強力推開，模擬畫面中會看到粒子不斷抖動，像爆炸一樣四散。數值震盪、易發散、不穩定

一張含有 文字, 螢幕擷取畫面, 地圖 的圖片

AI 產生的內容可能不正確。