有效的建構動態雲層模型之新技術

王宗銘
中興大學資訊科學所
CMWang@cs.nchu.edu.tw

游宗旻
中興大學資訊科學所
s9056006@cs.nchu.edu.tw

摘要

本文提出新的演算法來模擬天空之雲層。我們提出多邊形取樣演算法，以隨機變數方式產生雲層粒子來有效的控制雲層之形狀。我們延續研究學者所常使用的體積重建成像演算法，並提出一個粒子半徑擾動演算法。我們實現上述演算法、構建出一個實驗系統，並在中階電腦上測試。結果顯示，在輸入真實世界雲層的二維影像後，我們的多邊形取樣演算法能有效的控制並模擬出類似真實世界雲層之形狀、外觀。我們也能模擬出具有動物、心狀等外觀形狀之雲層。此外，粒子半徑擾動演算法能有效的改變雲層粒子之半徑，使得觀察者在靜態視覺時，雲層仍具有動態逼真之視覺效果。最後，系統每秒至少能成圖 10 個架框數(FPS)，足敷即時、互動之需求。綜言之，我們的雲層模擬演算法能同時在外觀形狀、視覺效果以及即時互動上展現優異的表現。

關鍵詞：雲層模擬、取樣演算法、擾動演算法、粒子系統、體積重建

一、前言

天空中雲層的模擬與逼真的成像在模擬應用中扮演著重要的角色；舉凡飛行模擬、戶外場景甚至電影特效皆少不了雲雲層這個物件，故長久以來對於天空雲層的模擬素為持續且盛行的研究課題[2-6, 8, 11, 15-16]。

天空中雲層的模擬必須同時考慮形狀、視覺效果、互動等三個因素。在形狀方面，我們希望能有效的控制雲層的形狀；例如，藉由輸入一張實際拍攝而得的雲層影像，能產生形狀相彷彿的三度空間雲層模型，如此可讓模擬的雲層更接近真實世界；此外，若輸入動物、心形狀等主題的 2D 影像時，也能產生形狀相彷彿的三度空間雲層模型，如此可讓模擬出來的雲層變的更生動、有趣。

在視覺效果方面，我們希望模擬出來的雲層外觀具備真實感，應儘可能與真實世界的雲層具有相同的視覺感受。此外，在做動態互動瀏覽時，有可能視點會暫時停留在某一個定點，此時視點的位置會固定不變，雲層也會僅有靜態的表現。我們希望若在此情況下，能藉由技術、技巧模擬出具有動態雲層的視覺效果。如此則不論是靜態瀏覽或互動瀏覽，雲層均有動態的視覺效果。

在互動方面，我們也希望在模擬時，可以讓使用者穿梭雲層，並且達到即時互動的效果。因此，必須能夠利用現在之硬體運算能力使成圖之能力能夠達到一定的成圖架框數(Frames Per Second, FPS)以上。如此才能符合一些相關的應用的需求，例如飛行模擬或電腦遊戲以達到更逼真、流暢的效果。

近年來對雲的模擬之文獻，由於各文獻述求的角度相異，因此，很少有文獻能夠同時滿足上述三項條件的。例如：Dabashi 忽略第一項形狀的要求，但特別著重雲層的視覺效果。故其結果能產生逆光觀測時的光柱效果(Shafts of Light)，但卻無法做即時的成圖[2, 3]。這是因為如欲如此逼真的視覺效果，往往需要龐大的運算量，故相對的無法達到即時之要求。相反的，若欲講求即時成像而以貼圖方法(Texture Mapping)來模擬雲層，其成圖結果往往不夠真實，無法展現動態的視覺效果[5, 6]。就我們所知，Harris 所提出之方法是目前文獻上能兼顧即時且具一定程度以上的真實視覺效果的研究[8]。他的方法使用粒子系統(Particle System)及體積重建成像技術(Splatting)來達成具備即時且擁有一定程度視覺真實效果的雲層模擬。然而，他的方法亦有若干缺失。首先，Harris 著重於雲層的即時視覺效果，因此，對於如何模擬雲層的外型，甚至模擬出具有特殊造型的雲層並未研究。其次，他所展示的結果僅止於靜態的模擬，亦即當視點固定不動時，雲層僅具有靜態的視覺效果。因此，在互動瀏覽時，當視點固定不動時，常感覺場景呆滯、單調。

本文提出新的演算法來模擬天空之雲層。首先，我們提出多邊形取樣演算法，以隨機變數方式產生雲層粒子來有效的控制雲層之形狀。這個演算法能模擬出類似輸入真實雲層圖像之雲層形狀及具有動物、心狀等特殊造

型外觀之雲層。其次，我們延續研究學者所常使用的體積重建成像演算法，並提出一個粒子半徑擾動演算法。前者可以產生逼真的雲層圖像。後者可將改變雲層粒子之半徑，使得觀察者在靜止視點時，雲層仍具有動態逼真之視覺效果。實驗結果顯示：我們所提的方法能在中階之個人電腦上，展現良好的互動效果。系統能成圖至少每秒 10 個架框數，符合即時互動之所需。我們認為本文所提的演算法不僅同時滿足可塑形狀、動態效果與即時互動，同時更優於 Harris 之方法，能實際運用在飛行模擬或電腦遊戲，達到逼真、流暢的視覺效果。

本文架構如下：第二節敘述相關文獻，並比較其優缺點；第三節詳述我們所提之演算法；第四節說明並分析模擬結果；第五節總結本文、提出未來研究方向。

二、相關文獻研究

雲的模擬主要分成兩個步驟：建構雲層模型(Modeling)及雲層之成像(Rendering)；前者著重於產生雲層的幾何模型；後者則是以建構出來的模型為基礎，輔以成像演算法來產生雲層之圖像。本節回顧文獻上相關此兩步驟之演算法。

雲層模型之建構方式可分為三大類：程序結構(Procedural Structures)、立體像素(Voxel)以及粒子系統(Particle Systems)。

程序結構為最早用來建構雲層之演算法[14]。它使用數學之關係式來產生出類似雲層的幾何形狀，在[4]中對此建構方式有詳細的描述。此法因為無繁複之遞迴運算，故其速度最快。然而，此法之缺點為參數控制不易，無法建構出具有特定形狀之雲層。此外，給定部分參數後，由於程序結構之特性使得其建構之結果往往無法預期。因此，本法僅適用於在大場景中建構出看似雲層之模型，但我們卻無法控制雲層之外觀形狀。

立體像素(Voxel)係使用細胞自動機的理論之概念，由 Dobashi 提出[2]。其原理為將空間分割為 n×n×n 個細胞(Cell)，給予每個細胞不同的屬性，算出每個細胞之密度，並對其成像。詳細的技巧詳見相關論文[3]。此法之優點為快速的模擬顯示動態之雲層。然其缺點為模擬出來的結果不夠精確，此乃因為該法只針對部分空間做運算之故。此外，此法也不易控制所建構的雲層之形狀。

最後，建構雲層模型之第三種方法為粒子系統，係由 Reeves 提出[15]，而最近的文獻中，Harris 也使用此種方法[8]。此種方法係直接模擬雲層內之粒子，並賦予粒子大小、位置及顏色等屬性。此法之優點為：觀念較為直接，較容易控制雲層之形狀。然而，此法亦有其缺點：必須仔細考慮粒子位置之分佈，方能達到所欲模擬之雲層形狀。此外，需要數量龐大的粒子方能產生逼真的視覺效果。

上述三種方法各有其優、缺點，使用何種方法最為適當，則需考量不同之需求目的。本研究使用粒子系統，以便於控制雲層之形狀。若不以形狀為首要目的，則可選用立體像素來達到模擬動態雲層的效果。至於程序結構則適用於大場景內需要高度即時的環境之中。

雲層之成像方法大致也可分為三大類：光線追蹤法(Ray Tracing)、程序紋理成像法(Procedural Texture)以及體積重建成像法(Splatting)。

光線追蹤法為早期用來模擬貌似真實的自然現象之主流演算法；Kajiya 率先使用光線追蹤法來做雲層之成像[9]。光線追蹤法之原理主要是利用成像方程式(Rendering Equation)來計算粒子在空間中所受之照度影響，將其量化後，成像於螢幕上[10]。此法之優點為能夠計算出非常逼真之雲層模擬視覺效果；然其缺點在於粒子間照度的影響造成龐大的運算，相對的也就使得成像速度變慢。

Gardner 於[6]中提出程序紋理成像法來避開光線追蹤法中複雜的運算。此方法是利用紋理貼圖的技巧來產生類似“雲層”的貼圖影像，並將之以貼圖技巧(Texture Mapping)來達到成像之目的。此法之優點為成像速度是三種方法中最快的；然而其缺點在於貼圖參數不易決定，且成圖後之效果難以和同場景中其他物件互動。文獻中[5]及[13]皆利用此種方法。

最新的雲層成像演算法為體積重建成像法，此法係由 Stam 所提出的[16]。其原理是以立體像素所建構出來的細胞或粒子系統所建構出來的粒子為基礎，計算出細胞或粒子之顏色；隨後以高斯分布法(Gaussian Distribution)產生貼圖影像。在貼圖時，將貼圖影像以混色(Blend)方式貼於面向視點方向之多邊形(Front-Facing Polygon)，也就是通稱的遮擋板(Billboard)。此法以細胞或粒子為基礎來產生貼圖影像，故雖也使用貼圖技巧來減少計算時間，但具有比程序紋理成像法更佳的視覺效果。此外，體積重建成像法之優點為可以藉由硬體設備的支援（中央處理器、顯示卡），在圖像品質與計算時間兩者間達到某種程度的平衡。2000 年與 2001 年分別有兩篇論文使用此種成像演算法[3] [8]。

如前所述，我們需考量不同之需求、目的，以便能選擇最適當之方法。本文使用體積重建成像法。究其原因在於此法能兼顧圖像品

質與計算時間；此外，就目前硬體趨勢觀之，此法也已漸漸取代往昔程序貼圖成像法之地位。至於光線追蹤法部分，由於該法使用精確的計算，故能產生非常逼真之視覺效果。在部分領域，例如大氣科學中，仍然具有不可取代之地位。

三、新的天空雲層演算法

本文使用粒子系統的方式來模擬天空之雲層，此乃取其形狀容易控制且建構直觀的特性。然而，粒子系統必須透過隨機取樣的方式產生粒子的位置方能模擬出想要的雲層形狀。

粒子位置的產生在先前學者 Harris 的論文中並未曾考慮，而是由本文首先提出相關的演算法。一旦取得雲層粒子的位置分佈後，我們必須搭配給予的參數，例如粒子大小(Radius)，及粒子顏色(Color)等，以便能直接對粒子進行成像處理。

我們針對粒子系統提出多邊形取樣演算法直接以我們所需形狀的多邊形點集合為輸入，針對此多邊形作取樣，以得到我們所需的點集合，亦即我們所需雲的模型。

3.1 多邊形取樣演算法

本節，我們提出新的雲層模型模擬的演算法，多邊形取樣演算法。此演算法原理為在多邊形之四角邊界(Bounding Rectangle)中取出一點，然後以 Antonio 於 1992 年所提之網格點測試法(Grid Method) [1]測試點是否在目標形狀中，重複此一步驟直至取完所需點數，如圖 4.1 中。此演算法步驟為：

1. 以一個多邊形之點集合為輸入檔。
2. 找出此多邊形之四角邊界，並在此邊界範圍之內任取一點。
3. 以 Grid 演算法檢查此點是否在多邊形範圍之中若是則取若不是則不取。
4. 回到步驟一，直到取得所需點數。

3.1.1 控制點的分布

使用 Grid Method 檢驗點是否在多邊形後，接下來便是要決定點的分布。以積雲來說，雲粒子的分布我們認為是內部密集而外部稀疏，如此其邊緣才會有所需的絮狀效果。然而，以隨機取點方式所得之點分布卻為接近平均分布的狀態，因此我們對產生的點做位移以達到上述之目的，我們採用的方式為在取出每一點時，算出取出之點與四角邊界中心點 X、Y 軸分量之距離，依此值與 1/2 對角線長度之比值算出此點之位移量，方式如下：

1. 算出四角邊界的中心點 center (x_{cen}, y_{cen}) 以及與四角之間的距離，即四角邊界範圍之內與 center (x_{cen}, y_{cen}) 的最大距離 Maxdis=1/2*對角線長度。

2. 定出最大位移量 Maxshiftx, Maxshifty，並以隨機變數取點 p (x_1, y_1)，計算 center (x_{cen}, y_{cen}) 以及 p (x_1, y_1) 的 x 距離 Nowdistx=$|x_{cen} - x_1|$，y 距離 Nowdisty=$|y_{cen} - y_1|$ ，以及 center 指向 p 之單位方向 (*normalx,normaly*) 向量，因此新的 p (x_1^*, y_2^*) 點座標如式(1)：

$$\begin{cases} x_1^* = x_1 + Nowdistx \,/\, Maxdisx * Maxshiftx * normalx \\ y_1^* = y_1 + Nowdisty \,/\, Maxdisy * Maxshifty * normaly \end{cases} \quad (1)$$

如此一來，就能達到偏於四角邊界外側的點便會往四角邊界收斂，而達到愈接近於中心點點愈密集的目的。

3.1.2 二維取樣點延伸至三維空間

3.1.1 所述之演算法是說明如何於 2 維平面中取得所需成像的 3 維點集合。由多邊形取樣演算法取出來之點，Z 值皆為 0，而無第三維資訊。因此，我們必須給與所取得之點 Z 值，若直接以隨機分布方式來取，則其側面會產生不自然的長條柱狀。因此，我們必須給予一個合理且虛構的第三維厚度給它，如圖 4.3 之成像結果。一般來說，在側面來看，我們希望其形狀為不規則形，因此我們使用隨機分布配合機率方式來控制此形狀，使其不至於成為長條柱狀，同樣地，我們希望愈外層點愈稀疏。因此，我們可將雲層之厚度分區，並使用一個機率值 來決定該點應落於哪一區，我們給予的機率值愈接近中心區域處則機率值愈高，如圖 3.1 所示，做法如下：

給予一個隨機變數 ε ,$0 \le \varepsilon \le 1$

若 ε 落於 ProPart N

則於 範圍 RangPart N 中取樣

其中由內而外之機率為

ProPart N(N$\in 1...n$)

且 ProPart K> ProPart K+1

其中由內而外之分布範圍為

RangePart N(N$\in 1...n$)

且 RangePart K> RangePart K+1

如此一來便可使其有厚度並達到內部密

集外部稀疏的目的了。

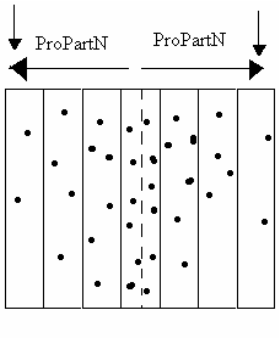

圖 3.1 二維延伸三維示意圖

3.2 即時動態之視覺效果

成像方面，我們則是引用 Harris 在 2001 年所提出的文獻[8]中所使用的體積重建成像演算法，此法的優點為能夠產生機真效果之雲層模型且能夠得到及時城圖之運算效能，此演算法在[8]有詳細的描述。另外，我們發展出一個粒子半徑擾動演算法，來達到動態的視覺效果。

若純以文獻[8]之演算法來做雲層的成像，則在做即時場景瀏覽時雲層粒子僅會呈現出靜態的效果而令場景顯得呆板、單調。故此，為了使模擬出來的雲層有即時動態的視覺效果，我們提出一個粒子半徑擾動演算法。

此演算法雖相當直覺，但效果卻相當良好。我們在每一次成像部分重新給予與原初始之粒子半徑可容許範圍之內新的雲粒子半徑，亦即於每一次重新成像時，以隨機變數的方式重新給予雲粒子半徑，計算如式(2)：

$$R_k = R_{k-1} + threshold * (2 * \varepsilon - 1) \qquad (2)$$

其中 $0 < \varepsilon < 1$ 之隨機變數,threshold 為最大擾動之值。

門檻值(Threshold)之值可依場景狀況而有所不同。一般說來，若視點離觀察雲層較遠，我們可給予較大門檻值來加大擾動粒子半徑；若視點離觀察雲層較近，則給予較小的門檻值來縮小擾動範圍，藉此保持其各架框間的連續性。如此一來，雖然連續之第 K 個架框與第 K-1 個架框其形狀外觀相似，但其內部粒子結構已擾動，因此當連續播放時能夠產生動態之視覺效果，如圖 4.4 所得視覺之效果。

四、結果與討論

本節我們將分別對模型模擬以及成像部分做時間、空間以及效果上做測試。在模型建構方面，我們所測試的電腦 CPU 為 Pentium IV 1.5G，記憶體為 256 MB SDRAM。我們做出兩個模型來比較各種取樣及成圖時間。圖 4.1 為以多邊形取樣方式從所攝影之影像取其形狀所得之效果，其取樣時間僅花了 0.093 秒。圖 4.2 為任意繪出之形狀，使用多邊形取樣並成圖之效果，其取樣時間花了 0.047 秒。

由上述結果可知多邊型取樣法取樣時間相當迅速幾乎不需讓使用者等待。此外，使用此法可依自己的喜愛做出任意形狀的雲，並對產生之 2D 圖形延伸出雲的模型。此為先前各研究學者所忽略或無法達成之特點。

在成像方面，我們所使用來測試的顯示卡等級為中階等級之 Nvidia TNT2 Model 64 32MB Video RAM，我們並模擬出兩個場景來做比較。以圖 4.3 之場景，此場景約 9600 個粒子，粒子半徑約為 20~130 之間，繪於 500*400 之視窗大小，並不使用任何加速之程式技巧，可得到預算時間 0.719 秒的速度，執行速度約為 7~12 fps，圖 4.4 之場景為較大之場景，其粒子數目約為 15000 個粒子，粒子半徑約為 20~130 之間，繪於 500*400 之視窗大小，並不使用任何加速之程式技巧，可得到預算時間 1.013 秒的速度，執行速度約為 6~10 fps。

由實驗可知，影響執行效能的因素包括場景粒子數目、粒子半徑大小以及雲層分布，其粒子數與即時效能之關係如圖 4.5 所示。如此關係曲線所示，當場景隨粒子數增加而 FPS 遞減。而其平均粒子半徑與即時效能關係如圖 4.6 所示，如關係曲線所示，場景平均半徑增加則 FPS 遞減。圖 4.4 為連續四個播放中的架框，其顯示出雲層粒子使用動態半徑與 Harris 使用固定半徑之比較，以動態半徑可使每個架框之成像結果產生些微異動，構成動態視覺效果。我們可以根據 RMS (Root-Mean-Square) [17]之定義將每個架框之些微異動予以量化表示。

圖 4.7 顯示量化後之結果。此圖表顯示出各個架框間 RMS 皆維持在 12 上下，相對於 Harris 靜態之雲層展示，此數值既可維持原來之形狀，又能夠產生動態之視覺效果。

上述結果明顯可以看出我們不但在即時互動方面能夠以中階電腦的能力達到即時之效果，而且在視覺效果方面，使用我們所提出的動態半徑方法優於 Harris 原先使用之固定半徑方法，也顯示出我們的成像演算法能夠於

中階電腦上得到動態且即時之視覺效果。

綜合上述所得之結果，本文所提出之方法，在模型建構、即時互動以及視覺效果三方面與先前研究學者 Harris 所提出之方法相比，成效更為周延、良好。

五、結論及未來工作

本文提出新的演算法來模擬天空之雲層。在模型模擬部分：我們提出多邊形取樣法來做雲層外觀形狀之模擬，此法優點為：不必耗費大量記憶體，且能在極短的時間得到我們所需之結果。我們並發現：利用此法，只要我們取得所要模擬之雲的輪廓，我們就能夠做出此種形狀的雲模型，甚至利用手繪的方式，模擬出種種造型有趣的雲。

在成像部分：我們使用體積重建成像技術來成像逼真的雲層圖像。此法可充分利用現今硬體之能力來達到高真實感且快速成圖之效果。我們也提出一個粒子半徑擾動演算法。此演算法雖相當直覺，但效果卻相當良好。我們在每一次成像部分重新給予與原初始之粒子半徑可容許範圍之內新的雲粒子半徑，亦即於每一次重新成像時，以隨機變數的方式重新給予雲粒子半徑。如此，藉由改變雲層中粒子之半徑，我們可以達到動態之視覺效果。此法改進 Harris 所提出的方法，使雲層在固定視點觀察時，其外觀更加生動、真實。

在互動方面：以我們所提方法所做的互動效果優異。實驗結果證實，在中階電腦上，FPS 均大於 10，此結果證明本文之方法足以在中階以上之電腦達到及時互動之效果。

總結本研究：我們所提的方法，同時滿足可塑形狀、即時互動以及動態效果。此外，與 Harris 相較，我們的方法在模型模擬與動態效果上，明顯優於他先前所提的方法。最後，所提方法能實際運用在飛行模擬或電腦遊戲，達到更逼真、流暢的視覺效果。

未來研究可以嘗試自動方式做取樣處理、並利用多解析度(Multiple-Resolution)技巧來顯示成圖後之雲層，加快成像速度來得到更高的 FPS、加強第三維點處理的方式來獲得更好的模擬外觀。

六、參考資料

[1] F. Antonio, “Faster Line Segment Intersection,” *In Graphics Gems III*, D. Kirk Ed, Academic Press, San Diego, pp. 199-202, 1992.

[2] Y. Dobashi, T. Nishita, and T. Okita, “Animation of Clouds Using Cellular Automaton,” *In Proceedings of Computer Graphics and Imaging 1998*, pp. 251-256, 1998.

[3] [3] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, and T. Nishita, “A Simple, Efficient Method for Realistic Animation of Clouds,” *In Proceedings of SIGGRAPH 2000*, pp. 18-28, 2000.

[4] D. S. Ebert, “Procedural Volumetric Cloud Modeling and Animation.” *SIGGRAPH 2000 course notes*, pp. 1-55, 2000.

[5] P. Elinas and W. Sturzlinger, “Real-Time Rendering of 3D Clouds,” *Journal of Graphics Tools*, Vol. 5, No. 4, pp. 33-45, 2000.

[6] G. Y. Gardner, “Visual Simulation of Clouds,” *Computer Graphics*, Vol. 19, No. 3, pp. 297-303, 1985.

[7] E. Haines, “Point in Polygon Strategies,” Graphics Gems IV, P. Heckbert Ed., Academic Press, Boston, pp. 24-46, 1994.

[8] M. J. Harris and A. Lastra, “Real-Time Cloud Rendering,” *Computer Graphics Forum* (Proceedings of Eurographics 2001), Vol. 20, No. 3, pp. 76-84, 2001.

[9] J. Kajiya and B. Von Herzen. “Ray Tracing Volume Densities,” In *Proceedings of SIGGRAPH 1984*, pp. 165-174, 1984.

[10] N. Max, “Efficient Light Propagation for Multiple Anisotropic Volume Scattering,” In *Proceedings of the Fifth Eurographics Workshop on Rendering*, pp. 116-134, 1994.

[11] R. Miyazaki, Y. Dobashi, and T. Nishita, “Simulation of Cumuliform Clouds Based on Computational Fluid Dynamics,” *Eurographics 2002 Short Presentation*, 2002.

[12] D. Overby, Zeki Melek, and John Keyser, “Interactive Physically-Based Cloud Simulation,” In *Proceedings of Pacific Graphics 2002*, pp. 469-470, 2002.

[13] K. Pallister, Generating Procedural Clouds in Real Time on 3D Hardware, Technical Report, Intel Corporation, 2000.

[14] K. Perlin, “An Image Synthesizer,” In Proceedings of SIGGRAPH 1985, pp. 287-296, 1985.

[15] W. Reeves, “Particle Systems- A Technique for Modeling a Class of Fuzzy Objects,” *ACM Transaction on Graphics*, Vol. 2, No. 2, pp. 359-376, 1983.

[16] J. Stam, “Stable Fluids,” In *Proceedings of SIGGRAPH 1999*, pp. 121-128, 1999.

[17] A. Takage, H. Takaora, T. Oshima and Y. Ogota, “Accurate Rendering Technique

Based on Colorimetric Conception," In *Proceedings of SIGGRAPH 1990*, pp. 263-272, 1990.

圖 4.1 實際影像產生雲，左圖為原圖，中圖為取樣圖，右圖為模擬出之形狀。

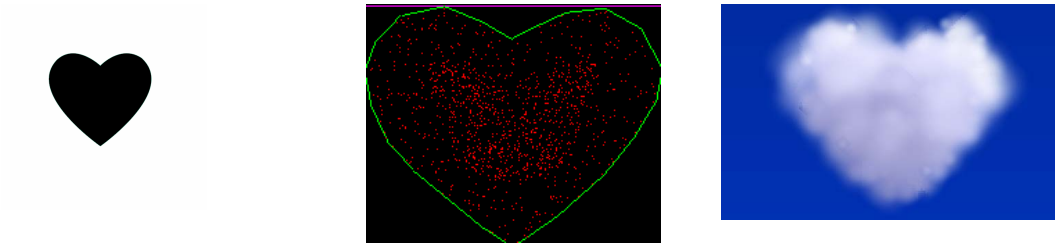

圖 4.2 繪出任意形狀，產生雲模型，左圖為原圖，中圖為取樣圖，右圖為模擬出之雲層。

圖 4.3 視覺經立體化之雲層側面，此圖為雲層場景之側視圖

圖 4.4 此四圖為雲層之四個連續時間之狀態，固定視點達到動態之視覺效果。RMS 值如圖 4.7。

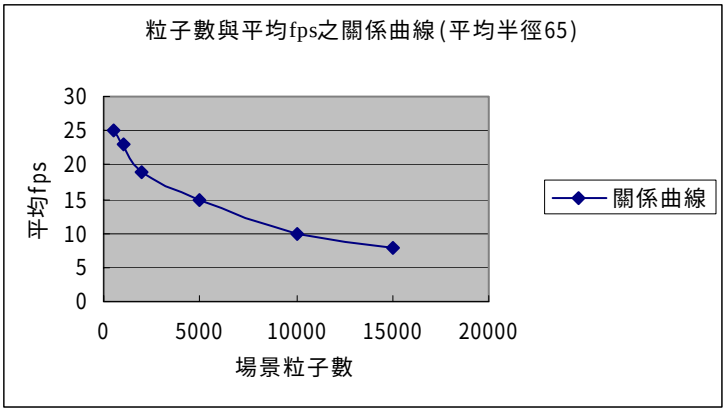

圖 4.5 場景中粒子數與 FPS 關係圖

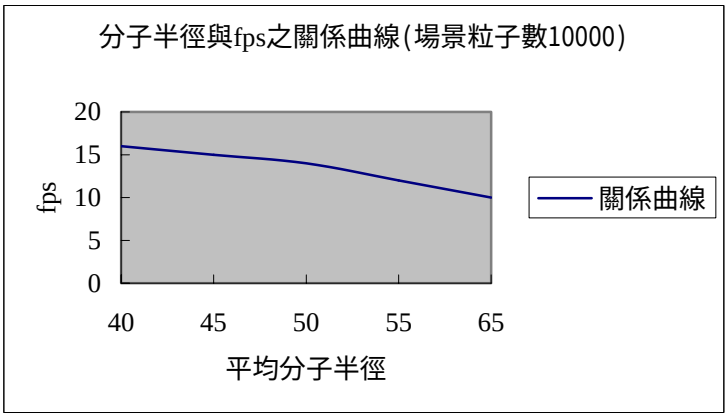

圖 4.6 場景中粒子半徑與 FPS 關係圖

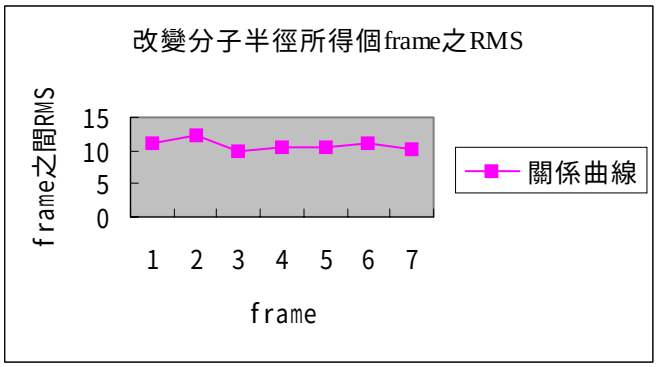

圖 4.7 若每個架框粒子使用動態半徑，各 frame 之間 RMS 恆不為零，此為動態雲層。