行政院國家科學委員會專題研究計畫 成果報告

研究混合型的平行運算系統及其應用在三維人物運動、布料模擬、頭髮模擬以及光線追蹤
研究成果報告(精簡版)

計 畫 類 別 ：個別型
計 畫 編 號 ：NSC 99-2221-E-009-143-
執 行 期 間 ：99 年 08 月 01 日至 100 年 07 月 31 日
執 行 單 位 ：國立交通大學資訊工程學系（所）

計 畫 主 持 人：黃世強

計畫參與人員：碩士班研究生-兼任助理人員：劉政旻
碩士班研究生-兼任助理人員：鄭游駿
碩士班研究生-兼任助理人員：葉喬之
碩士班研究生-兼任助理人員：陳蔚恩
碩士班研究生-兼任助理人員：陳奕辰
碩士班研究生-兼任助理人員：方士偉
碩士班研究生-兼任助理人員：盧詩萍
碩士班研究生-兼任助理人員：李幸宇
碩士班研究生-兼任助理人員：洪駿宏
碩士班研究生-兼任助理人員：王阿敏

報 告 附 件 ：出席國際會議研究心得報告及發表論文

處 理 方 式 ：本計畫可公開查詢

中 華 民 國 100 年 10 月 24 日

0 前言

我們建置一個系統,進行三維人物運動、布料模擬、頭髮模擬以及光線追蹤,它們的應用都十分廣泛，可以被應用在不同領域,例如電動和數位電影。多核心平台和繪圖處理器的計算能力不斷提高,我們發展平行運算方法，對頭髮模擬以及光線追蹤都可以達到即時效果。

1 研究目的

我們的研究是關於發展一個平台，進行三維人物運動、布料模擬、頭髮模擬以及光線追蹤。這系統是建立在多核心平台以及具有串流處理器的繪圖處理器上。我們完成三個子系統，它們包括下列三個部分:應用視角速進行光線追蹤,頭髮運動模擬，人物動作和衣服互動系統。

I.我們應用視角方法把物體的基本元素三角形分堆，然後把每一分堆建立 Bounding Volume Hierarchy，對這些 Bounding Volume Hierarhies 進行光線追蹤，整個過程完全在圖形處理器(GPU)上進行。

II.在多核心平台上模擬頭髮運動方面，我們將計算頭髮運動的模擬，利用圖形處理器(GPU)來進行，在 NVIDIA 所開發的 CUDA(Compute Unified Device Architecture)的架構上實現演算法的加速。我們利用 OpenGL 4.0 繪圖流程管線(pipeline)所新增的鑲嵌階段(tessellation stage)，撰寫自定的著色程式(shader)，在 GPU 上動態的產生頭髮的幾何模型，省去 CPU 和 GPU 間要透過 PCI-E bus 互相傳遞的花費，以達成即時的頭髮渲染。

III.在人物動作和衣服互動系統，人物的運動以骨架方式進行，使用碰撞偵測並且予以適當的碰撞反應，應用適應性方法對人物模型和衣服的互動重複調校，處理衣服的自身穿透或衣服穿透角色的情形。

2 文獻探討

我就三方面的研究進行文獻探討，包括光線追蹤、頭髮模擬和人物動作和衣服模擬。

2.1. 光線追蹤

Ernst et al.[1] 提出一個遞歸演算法，在圖形處理器上進行光線追蹤，他們把場景以kd-tree作空間分割,加快運算速度，可是由於當時圖形處理器的功能有限,他們方法的表現難以達到互動效果。Foley et al.[2]基於kd-tree提出一個stackless的方法，改進速度。Horn et al.[3]利用short stack結構，大大減少遍歷的花費，這個方法能夠達到互動的效能。Popov et al.[4]也提出一個stackless的方法，應用uniform grids和kd-trees為主的加速結構上，自此應用圖形處理器進行光線追蹤，受到更多的重視。Gunther et al.[5] 基於共享stack，提出一個BVH-based 的平行封包處理方法，可以對動態場景進行互動的光線追蹤。Zhou et al.[6] 提出一個即時的方法，在圖形處理器上建置kd-trees。Lauterbach et al.[7]提出一個混合型的方法建置SAH-based hierarchies。加速結構對光線追蹤十分重要，大多數的研究都在探討建置這些結構。

2.2 頭髮模擬

頭髮的模擬可以從利用頭髮線段的組合來進行。Rosenblum et al.[1] 提出應用彈簧來模擬頭髮的運動。Anjyo [2]提出利用 projective dynamics，利用串聯的剛體模擬頭髮運動。Bertails et al.[3] 利用繩的運動原理模擬頭髮，這個方法假設頭髮線段是一維的可變形物體，應用彈性原理，一些自然的頭髮運動可以模擬出來，例如扭轉的運動。Hadap[4]利用流體力學模擬的方法去處理頭髮間的互動，也可以處理頭髮和其他物體間的互動情況。Bando et al.[5] 提出鬆散的粒子方法進行頭髮運動模擬。Petrovic et al.[6] 對一個固定的格子，進立一個以 Eulerian 為基礎的頭髮運動模擬。Bertails et al.[7]提出利用頭髮密度的方法計算頭髮間的力，以便處理頭髮間的碰撞。Tariq and Bavoil [8] 更把物體全部都以立體方格表示，進一步加快頭髮間碰撞的處理，當然頭髮間碰撞的計算準確度不高。McAdams et al.[9]發展一個混合的方法，利用 Lagrangian 方法模擬頭髮運動以及利用 Eulerian 的流體方法處理頭髮間的碰撞。

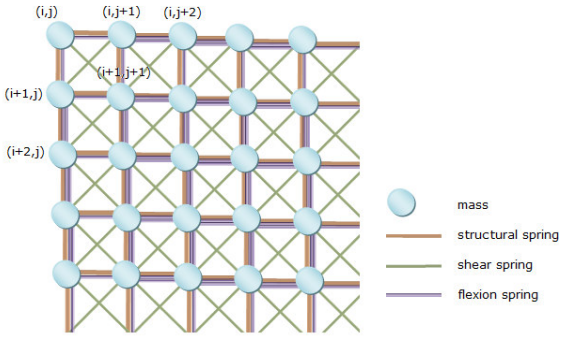
Level-of-detail(LOD)的方法都有應用在頭髮的渲染上，例如[10][11][12]。Kajiya and Kay 提出一個局部的光線散射模型，對頭髮進行渲染。Marschner et al. [24] 進一步利用實驗結果，成功得到更精確的頭髮的光線散射模型，可以渲染逼真的頭髮。

2.3 人物動作和衣服模擬

衣服模擬離不開碰撞偵測，加速碰撞偵測的方法，第一種類型是 BVH(Bounding Volumes Hierarchy)，它的主要目的是減少需要檢查的三角形數量，降低碰撞測試的次數，做法上首先會設計一個 BV 來包住整個物體，接著會將這個 BV 分割成兩個較小的 BV，這兩個 BV 會分別包含著一半的物體，依此類推每次都會包住前一次的範圍的一半，直到 BV 包住的是一個或是數量很少的一些三角形為止，而這其中又可以細分為 AABBtree[1]、OBB tree[2]、K-DOP[3]和 Sphere[4]這四種不同的分割方式。在建立樹狀結構之後，只有與父節點有發生碰撞才要往下檢查左邊或右邊子節點，如此一來只要依序尋訪樹狀結構，就能知道物體間是否產生碰撞，如果有發生碰撞，也能找出發生碰撞的那些三角形；第二種類型是 Spatial partitioning，這類方法的基本概念與 BVH 十分相似，其中的差別在於做法是對空間做切割，先將全部空間分割成幾部分，之後只在有物體存在的空間才會再繼續做分割，同樣會形成樹狀結構，而這其中也可以細分為 octree[5]和 BSP[6]這兩種不同的分割方式；第三種類型是以 image-based 的方法為基礎，其概念是將 3D 物體的網格柵格化(rasterize)到 framebuffer，而 frame buffer 包含 stencil、color 和 depth buffer[7]。衣服的自我碰撞可以應用 regular patch 的方法進行加速[8]。

在角色的動作模擬方面，大部分研究使用的方法有幾類，第一種類型是以 Kinematic 為基礎，而其中又分為給予角色的狀態參數，然後去計算出角色姿勢的 Forward Animation[9]和先給予各個姿勢點的位置，再去計算出各個關節的角度的 Inverse Animation[10]；第二類是以動態為基礎的模擬[11]，這一類型的模擬通常會以物理學的定律為基礎，同時考量了很多環境因素或施加的外力會對角色造成的影響，因此可以表現出更接近真實人物動作的結果；第三個類型是以近年來十分

熱門的動作捕捉系統(motion capture)為基礎[12][13]，將由 motion capture 所得到的資料做為輸入，再依照所想要達成的目的加以做處理，做不同角色動作的合成，或是對角色的動作做修改使其符合所給予的限制條件。由於近年來的角色動作模擬以追求更高的真實性為目標，因此使用的方法常會結合上述提到的幾個類型。

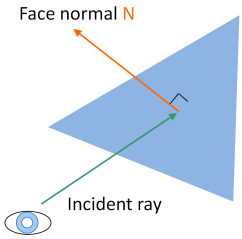


圖一：各質點與彈簧的連接方式示意圖。

在衣服模擬的部分，我們參考了 Provot[14]的做法，每個模型都是由 M 乘 N 個質點所組成，我們可以將其建構成二維矩陣，彼此間的連接方式總共有三種，不同彈簧的連接方式如圖一。第一種為質點[i, j]至質點[i+1, j]和質點[i, j]至質點[i, j+1]，也就是以水平方向和垂直方向連接相鄰的兩個質點，該連接方式的彈簧稱之為 structural springs，負責擴展與壓縮的處理；第二種為質點[i, j]至質點[i+1, j+1]和質點[i+1, j]至質點[i, j+1]，與對角相鄰，稱為 shear springs，有助於維持對角形狀；最後一種則為質點[i, j]至質點[i+2, j]和質點[i, j]至質點[i, j+2]，以水平和垂直方向連接相隔一質點的兩質點，稱為 flexion springs，其作用在於能夠讓布塊彎曲。當沒有 flexion springs 時，布會很容易摺疊，彎曲變形差，不圓滑，很容易有角的出現。如果同時沒有 shear springs 和 flexion springs 彈簧的話，布很容易失去它原來的形狀，所以只有同時具備這三種彈簧，布才能將它的彈性以及形狀確實呈現出來。

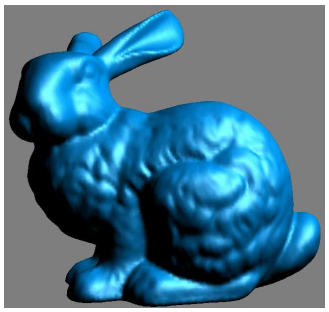
3 研究方法

3.1 光線追蹤

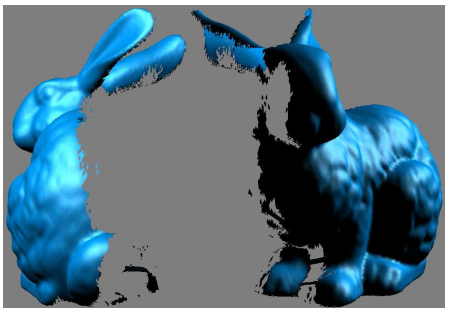


圖二：三角形相對觀察點的方向性，以內積計算。如果 $R \cdot N > 0$，方向性為正。

我們依照物體的三角形相對觀察點的方向性進行分堆，接著對每一分堆重建 Bounding Volume Hierarchies，這個過程是在圖形處理器上進行，我們的BV是 AABBs (axis-aligned bounding volumes)。觀察點包括光源和攝影機。下圖顯示一個模型面向攝影機的三角形分堆。



圖三(a): 面向攝影機的三角形分堆。



圖三(b):從側面觀察的三角形分堆。

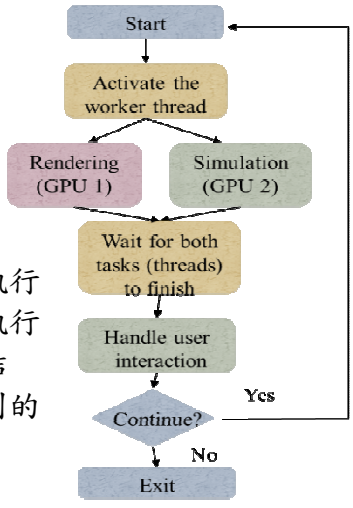
利用方向性分堆方法，可以大大減少遍歷的時間。我們的BVHs重建基於Lauterbach et. al.的方法，它可以快速地在圖形處理器上完成重建過程。光線追蹤的遍歷都是完全依賴這些分堆的BVHs進行。我們的方法支援primary rays, reflection rays and shadow rays。我們方法的流程如下：

Algorithm 1 GPU ray tracing by using view-based BVHs

```
1: if Objects deformed then
2:     Divide triangles of objects based on the camera position and compute C+
3:     construct the BVH for C+
4:     Divide triangles of objects based on the light source and compute L+
5:     construct the BVH for L+
6: else
7:     if Camera moved then
8:         Divide triangles of objects based on the camera position and compute C+
9:         construct the BVH for C+
10:    end if
11:    if Light moved then
12:        Divide triangles of objects based on the light source and compute L+
13:        construct the BVH for L+
14:    end if
15: end if
16:
17: Perform ray tracing with primary rays and shadow rays
```

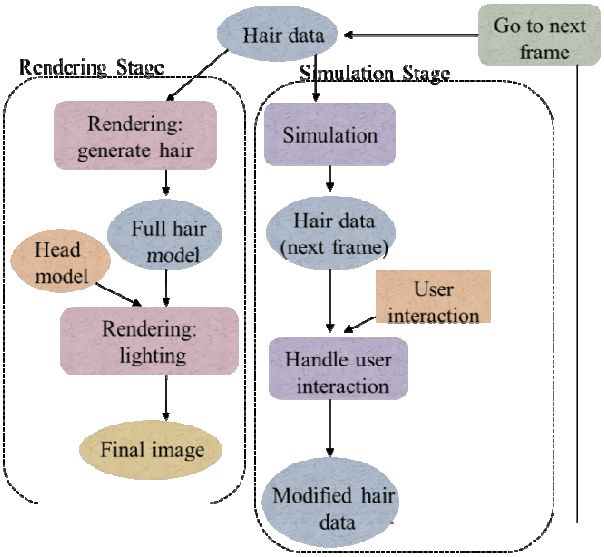
3.2 頭髮模擬

我們利用多執行緒(Multi-threading)的方法來使模擬的程式與渲染的程式同時在兩個 GPU 上執行。在我們的架構中有兩條執行緒，一條是主執行緒、一條是副執行緒。每個在計算每個畫面流程中，由主執行緒去啟動副執行緒，接著兩邊各自執行各自的工作，等待雙方的工作的結束，在回到主執行緒去處理和使用者的互動。最後看收到的指令程式可以繼續跑或結束。



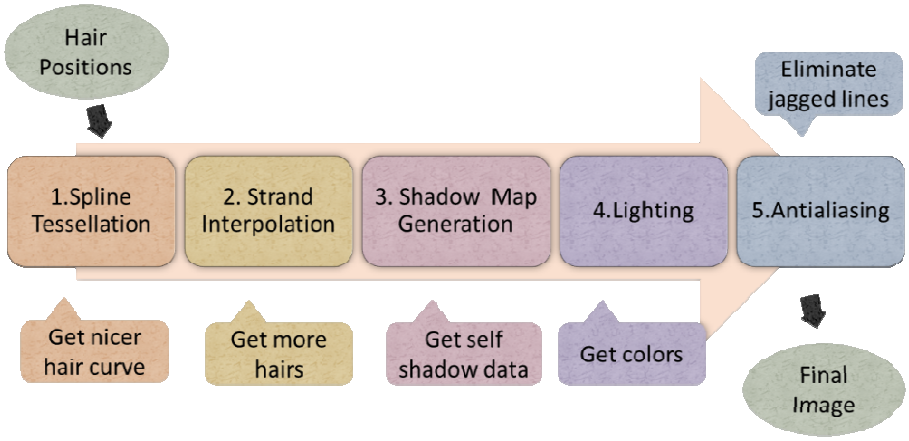
四：流程圖。

這是系統的資料流程圖:輸入的是頭髮資料，線段和控制點。我們把控制點的位置設定為模擬運動的粒子，在模擬的階段會計算下個時間點的位置與速度。在渲染的階段會用輸入資訊產生最後的畫面。



圖五:資料流程圖。

這是渲染階段的流程圖:

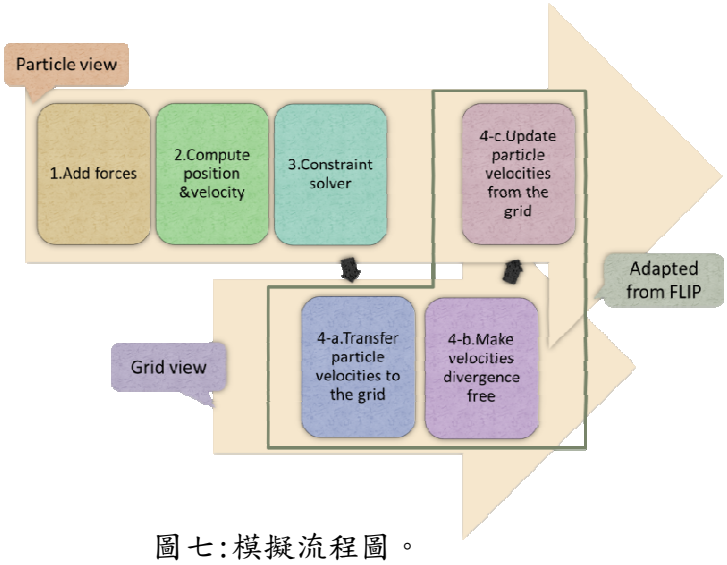


圖六: 渲染階段的流程圖。

第一個步驟是將原本粗糙的線段，按照樣條函數(Spline function)做鑲嵌(Tessellation)，填入更多的線段來使得原本粗糙的曲線更加平滑。

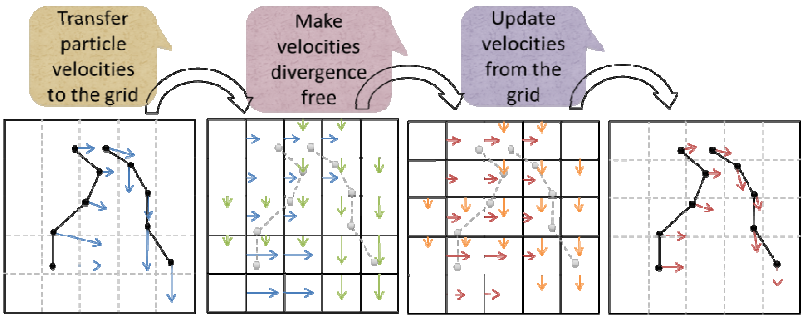
第二個步驟是將第一個步驟的輸出當做輸入，以鑲嵌過後的曲線做內插(Interpolation)產生更多的頭髮線段。前兩個步驟，都是在 OpenGL 鑲嵌階段的著色程式(shader)中實作，並且我們可以根據觀看模型的距離來決定相關的參數，以控制產生的線段多寡。當觀看距離近的時候，我們會採用比較精密的模型，產生更多的線段。而當觀看距離較遠的時候，我們不需要那麼精密的模型，我們便產生較少的線段，使用較粗糙的模型。根據需求而改變使用的模型可以動態的節省繪圖的花費而提高系統的效能，這就是我們在採用的 Level-of-details 方法。第三個步驟是準備計算頭髮陰影(Self-shadow)所需要的資訊。我們所採用的方法是 Deep

Opcatiy Maps，是準備所需的貼圖(Map)會花費兩個回合(Pass)。在打光(Lighting)的步驟，我們可以模擬兩個不同顏色的光澤。最後經過反鋸齒化的步驟就得到結果的圖片。



圖七:模擬流程圖。

圖七是模擬階段的流程圖。我們應用流體力學的 FLIP (Fliuid Implicit Particle)計算模型來做頭髮的模擬。流體力學的計算模型主要分成兩大類型:基於網格點(Grid based)的方法和基於粒子(Particle based)的方法。Grid based 又稱 Eulerian approach，Particle based 又稱 Langrangian approach。基於網格點的方法就是流體的計算量，例如速度、密度、壓力都在網格點上計算，而基於粒子的方法就是都在粒子的位置上計算這些計算量。而 FLIP 是一種混合的計算模型，它會先在粒子點上做一些計算，接著將計算量轉換到網格上做另一部分的計算，最後再再轉換回粒子的點上。模擬階段的步驟大致上分為 1.計算作用力 2.計算位置和速度 3.解決系統所設定的限制條件(Constraint) 4.採取自 FLIP 的部分，維持頭髮粒子所佔有的體積。頭髮的長度大致上不太會改變，所以我們在頭髮線段的兩個端點設置距離的限制條件(Distance Constraint)來為持頭髮的長度 3.的部分就是移動粒子的位置來滿足所設定的限制條件，由於一個端點上的粒子可能被設定不只一個限制條線，因此移動一個粒子的位置來滿足一個限制條線的同時，有可能會讓另一個限制條線被破壞。所以我們透過反覆的更新來盡量滿足所有的限制條件。4.的部份就是取自 FLIP 的步驟。



圖八：進行 FLIP 步驟的流程圖。

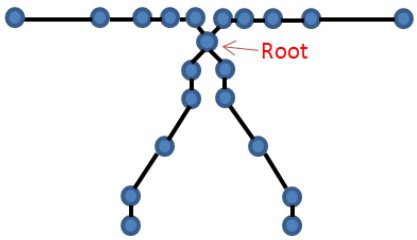
圖八進行 FLIP 步驟的流程圖。先轉換粒子上的速度到網格點上，在網格點上做使速度向量場的散度(Divergence)歸零的計算(在流體的模擬中，這麼做的意義相當於是使流體的體積維持不變)。這一個步驟中，會需要解一組線性系統方程式 Ax = b, A 是一個對稱(Symmetric)且稀疏(sparse)的矩陣, x, b 都是向量，我們是採用 Preconditioned Conjugate Gradient 這個數值方法來在 GPU 上實作。最後再透過散度為零的速度向量場來更新粒子上的速度。

3.3 人物動作和衣服互動系統

我們的角色模型和衣服都是由許多三角形所構成，首先我們會先計算角色的動作，然後在計算衣服的移動，但是衣服的移動會受限於角色的動作，因此需要將衣服所能移動的最大範圍考慮進去。當在計算時，如果在之前的 time step 有一些衣服的三角形落在角色模型上，則該三角形就會跟著角色模型的表面一起移動，但因為構成衣服的三角形間是彼此相互牽制的，所以需要在計算整體衣服的移動時，去修正貼在角色模型上的衣服的三角形，讓其產生滑動的效果。最後則是利用碰撞偵測與反應去處理衣服穿透衣服和衣服穿透角色的情形。

人物角色的動作模擬實作上，首先會由檔案設定人物角色的骨架基本資料，這個骨架的結構是一個樹狀結構，由一個 Root 和許多的子節點所組成，而 Root 和這些子節點代表的就是人物角色的各個關節，這份資料設定的內容包含了 Root 的位置、Root 的旋轉軸、Root 的子節點數量、每一個子節點的位置、子節點的旋轉軸、這一個子節點在運動時會影響的範圍、子節點的角速度和最大的可變動角度。另外在實際運動模擬也會設定各個關節在運動時的角速度與最大變動角度。在設定這些基本資料後，會先繪製出人物角色整體骨架的結構，然後在開始動作模擬的時候，再依照檔案內設定的運動方式改變每一個關節的運動，模擬出人物角色的動作，其中當變動到達最大可變動角度時，控制變動的關節就會將角速度的方向改變為相反方向，使運動可以往反方向繼續下去。

我們的實驗角色的模型建立是一個 Root 有四個子節點，分別代表角色的四肢，其每個子節點的分支分別包含五個節點，骨架模型如圖九。由於我們的系統模型著重在角色的動作與衣服的相對影響形變上，因此沒有很精確的處理角色運動時皮膚表面所產生的變化，故此在真實度上會有些不足。



圖九: 骨架模型

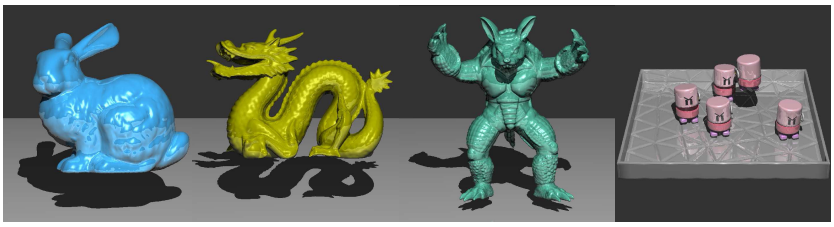
4. 結果與討論

4.1 光線追蹤

我們的實驗都在 Intel(R) Core2 Quad CPU Q9400 @ 2.66GHz 2.67GHz 4 GB main memory 上進行。繪圖處理器是GTX 480 graphics card （2.0 compute capability）。

我們應用視角方法把物體的基本元素三角形分堆，然後把每一分堆建立 Bounding Volume Hierarchy。最後對這些Bounding Volume Hierarhies進行光線

追蹤。整個過程完全在圖形處理器(GPU)上進行。重建BVHs的方法是基於Lauterbach et al.所提出的方法。



圖十：光線追蹤渲染圖。

Objects	Traversal with U				primary rays with C^+	1 light with C^+ and L^+	2 light with C^+, L_0^+ and L_1^+	reflection and 1 light with C^+, C^-, and L_0^+	
	primary rays	1 light	2 light	reflection and 1 light					
Bunny	16.6	30.5	46.1	53	8.6	18.8	31.8	41.7	不包括BVH建置
Dragon	26.4	49.4	76.3	126.9	10.4	22.7	37.9	74.8	
Armadillo	30.7	59.1	95.4	132.2	17.3	39.4	67.5	118.4	
Toys	33.2	48.9	73.6	163.6	20.4	25.2	33.4	122.6	
Bunny	35.3	49.2	64.8	71.7	16.4	34.7	56	65.9	整個過程
Dragon	57.4	80.4	107.3	157.9	26.9	55.4	86.9	124	
Armadillo	134.7	163.1	199.4	236.2	58.4	122.4	192.5	243.5	
Toys	61.9	77.6	102.3	192.3	38.1	48.3	61.8	150.9	

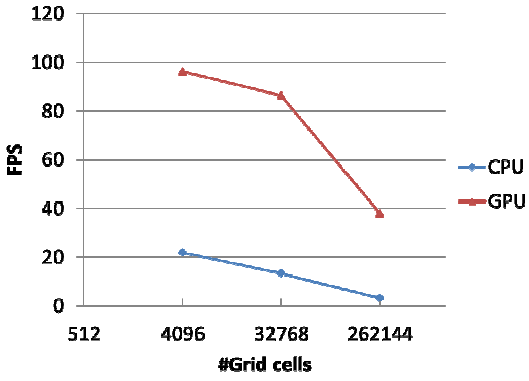
表一:光線追蹤時間數據(msec)。

在表一中,左四列是 Traversal with U 為 2009 年國外的方法.右四列為我們的方法。在有 Reflection and 1 Light,對於整個過程,他們的方法分別是 71.7, 157.9, 236.2 和 192.3,我們方法是 65.9,124,243.5 和 150.9。在 2Lights 時,他們的方法分別是 64.8,107.3,199.4 和 102.3,我們方法是 56,86.9,192.5,61.8。我們的方法比過往的方法快了大概 15%以上。

4.2 頭髮運動模擬

程式執行環境。GPU 1:負責跑模擬的 GPU，是 NVIDIA GeForce9600GT 是 2008 年二月推出桌上型電腦中階的顯示卡產品，有 64 個核心。GPU 2:負責做渲染的 GPU，是 NVIDIA GeForce GTX480 是 2010 年三月推出桌上型電腦高階的顯示卡產品，有 480 個核心。CPU: 跑模擬的對照數據，Intel i7-930 是 2010 年第一季(1st Quarter)推出的，有 4 核心 8 執行緒。作業系統是 Windows XP (Service Pack 3)。

模擬結果。這是 CPU 和 GPU 跑模擬的效能數據比較。效能是以每秒鐘畫面的更新率(FPS: Frames per second)為單位。

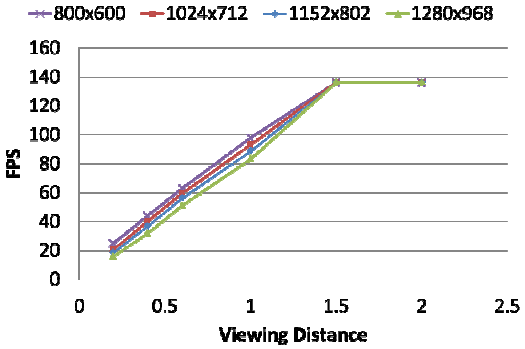


#Grid cells vs. FPS	16^3	32^3	64^3
CPU	21.8	13.3	3.19
GPU	96.2	86.5	37.7
Speedup	4.4x	6.5x	11.8x

表二：對於不同 GridCells 的 FPS。

我們改變網格的大小來做為控制變數。由表中的數據我們可以看出當網格越大的時候，GPU/CPU 的效能比值就越高。而圖中，我們的 x 座標是採用對數刻度。在 GPU 需要比較平行化。

渲染結果：這是我們做 LOD(Level-of-details)後，FPS（frame per seconds）與觀看距離的數據。



圖表:LOD

Viewing Distance vs. FPS	0.2	0.4	0.6	1.0	1.5	3.0
800x600	25	44	63	98	136.1	136.1
1024x712	20.6	40.2	59.6	93.2	136.1	136.1
1152x802	18.3	36.9	56.1	88.7	136.1	136.1
1280x968	15.9	31.9	50.9	83.2	136.1	136.1

表三：FPS

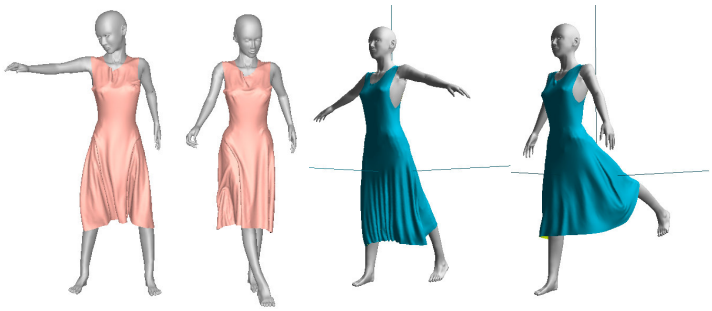
由於渲染所花費的時間，與所要處理的像素(Pixel)數目有很大的關係，所以我們在四種不同的畫面解析度下做量測。我們可以觀察到確實在同樣的觀測距離之下，解析度越高的畫面，所對應的 FPS 是比較低的。而當觀看距離大於某個值之後，FPS 不再提昇的原因，是因為處理每個畫面的流程是包括渲染與模擬，而這兩個部分是在兩個執行緒中一起執行，即使渲染的部份先跑完，我們也必須等到模擬的部分跑完才會換到下一個畫面。因為當渲染的效能提高到某個程度，整體系統效能的瓶頸變成模擬的部分時，渲染部分的效能提昇就沒有辦法反應在 FPS 上了。也就是圖表中 FPS 維持不變的、線段水平的部份。下面就是一段人物甩頭動作的截圖。

圖十一：甩頭動作的截圖。

4.3 人物動作和衣服互動系統

4.3.1 實驗結果

首先，我們執行載入角色模型與衣服模型的系統，在這之中可以設定兩者在模擬時的初始位置，修改衣服的大小，並且為此一衣服模型加入彈簧，使其可以在模擬時產生形變。這個系統可以直接模擬可形變物體與不可形變之剛體的互動，藉由給予不同的外力讓物體形變，所以我們可以利用此系統預覽衣服受環境力後所產生的變化。



圖十二：人物和衣服截圖

若想要讓角色可以動作，我們可以在上述系統中將衣服和角色初始設定完成，然後進行儲存動作，系統會自動產生一個資料夾，其中包含了角色與衣服的相關資料。接著在該資料夾中額外添增一個資料檔，用以設定角色的動作，像是角色的每個關節所在位置以及運動的角速度等，然後就可執行我們的角色動作與衣服形變的模擬系統，在執行前必須先修改系統中設定檔的讀取路徑，選擇我們想要模擬的物件的所在資料夾，再開啟程式執行模擬的動作，這個模擬程式會同時執行碰撞的偵測與碰撞後的反應，然後我們可以使用系統中預設的功能，產生每個 frame 的所有點的位置與速度的記錄資料。由於同時執行模擬與顯像會讓整個程式運行較緩慢，因此我們可以在模擬完成後執行顯像系統，導入上述模擬程式所產生的數據，由於不再需要繁瑣的計算，因此畫面的顯像流暢許多。

我們的角色連續動作設定內容是，角色會從初始的大字形開始動作，緩緩的將雙手放下，當旋轉接近 90 度後，會在施以反方向的速度運動，另一方面，角色也會慢慢的將一隻腳緩慢舉起，直至 45 度後，同樣會施以反方向的速度繼續運動。而衣服在受重力的影響下，會從剛開始與角色有一小段距離的間隔，慢慢的下降貼至皮膚表面

4.3.2 討論

我們的方法主要是不斷計算角色和衣服的動作，然後將超出限制的值做修正，如同前面提出的方法，一開始要先計算角色的動作，第二步驟是計算衣服的動作，然後讓落在角色上的衣服的三角形跟著角色模型的表面一起移動，接著再一次地計算衣服的動作，再讓貼在角色身上的衣服的三角形滑動。最後再處理碰撞部分，讓衣服不要有穿透現象。如此繁瑣的步驟，一個 time step 可能不夠，所以我們需要重複執行，如此一來就會多耗費一些時間在計算上，但相對的會得到較好的結果。

另外，我們在即時環境中模擬衣服和物體的碰撞時，偶爾會發生一些破圖的現象，原因是因為衣服的三角形和物體的三角形貼在一起，所以我們只看到了物體的三角形，誤以為衣服破了洞。解決方法之一是讓衣服和物體保持一定距離，不要讓兩者完全緊黏在一起，這個方法雖然可以避免破圖現象產生，但是因為會非常的不自然，而且和實際情況有所衝突，所以我們暫不採用。

此外如上述所提到的，如果角色運動速度過快，會產生衣服破圖的現象，因此給予角色適當的運動速度會改善許多，但是偶爾仍會發生一小部分的破圖現象，原因在於構成衣服的三角形可能過大，移動時不小心產生錯位現象，縮小三角形的大小是一個簡單的解決方法。

5 參考文獻

5.1 光線追蹤

[1] M. Ernst, C. Vogelgsang, and G. Greiner. Stack implementation on programmable graphics hardware. In Vision, modeling, and visualization 2004: proceedings, November 16-18, 2004, Standford, USA,page 255, 2004.
[2] T. Foley and J. Sugerman. KD-tree acceleration structures for a GPU raytracer. In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware, page 22. ACM, 2005.
[3] D. Horn, J. Sugerman, M. Houston, and P. Hanrahan.Interactive kd tree GPU raytracing. In Proceedings ofthe 2007 symposium on Interactive 3D graphics and games, page 174. ACM, 2007.
[4] S. Popov, J. G¨unther, H. Seidel, and P. Slusallek. Stackless kd-tree traversal for high performance GPU ray tracing. In Computer Graphics Forum, volume 26, pages 415–424, 2007.
[5] S. Guntury and P. J. Narayanan. Ray tracing dynamic scenes with shadows on the gpu. In Eurographics Symposium on Parallel Graphics and Visualization 2010, pages 27–34, 2010.
[6] K. Zhou, Q. Hou, R. Wang, and B. Guo. Real-time kd-tree construction on graphics hardware. In ACM SIGGRAPH Asia 2008 papers, page 126. ACM, 2008.
[7] C. Lauterbach, M. Garland, S. Sengupta, D. Luebke, and D. Manocha. Fast BVH construction on GPUs. In Computer Graphics Forum, volume 28, pages 375–384,2009.

5.2 頭髮模擬

[1] R. E. Rosenblum, W. E. Carlson, and E. Tripp, “Simulating the structure and dynamics of human hair: modelling, rendering and animation,” *The Journal of Visualization and Computer Animation*, vol. 2, no. 4, pp. 141–148, 1991.
[2] K. Anjyo, Y. Usami, and T. Kurihara, “A simple method for extracting the natural beauty of hair,” in *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, 1992, pp. 111–120.
[3] F. Bertails, B. Audoly, M.-P. Cani, B. Querleux, F. Leroy, and J.-L. L´evˆeque, “Super-helices for predicting the dynamics of natural hair,” in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 1180–1187.
[4] S. Hadap and N. Magnenat-Thalmann, “Modeling dynamic hair as a continuum,” *Computer Graphics Forum*, vol. 20, no. 3, pp. 329–338, 2001.
[5] Y. Bando, B.-Y. Chen, and T. Nishita, “Animating hair with loosely connected particles,” *Computer Graphics Forum*, vol. 22, no. 3, pp. 411–418, 2003.
[6] L. Petrovic, M. Henne, and J. Anderson, “Volumetric Methods for Simulation and Rendering of Hair,” Pixar Animation Studios, Tech. Rep., 2005.
[7] F. Bertails, T.-Y. Kim, M.-P. Cani, and U. Neumann, “Adaptive wisp tree: a multiresolution control structure for simulating dynamic clustering in hair motion,” in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2003, pp. 207–213.
[8] S. Tariq and L. Bavoil, “Real time hair simulation and rendering on the gpu,” in *ACM SIGGRAPH 2008 talks*, 2008, pp. 37:1–37:1.
[9] A. McAdams, A. Selle, K. Ward, E. Sifakis, and J. Teran, “Detail preserving continuum simulation of straight hair,” in *ACM SIGGRAPH*, 2009, pp. 62:1–62:6.

[10] K. Ward, M. C. Lin, J. Lee, S. Fisher, and D. Macri, “Modeling hair using level-of-detail representations,” *International Conference on Computer Animation and Social Agents*, p. 41, 2003.
[11] K. Ward and M. C. Lin, “Adaptive grouping and subdivision for simulating hair dynamics,” *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, pp. 234–243, 2003.
[12] T.-Y. Kim and U. Neumann, “Interactive multiresolution hair modeling and editing,” *ACM Trans. Graph.*, vol. 21, pp. 620–629, 2002.
[13] J. T. Kajiya and T. L. Kay, “Rendering fur with three dimensional textures,” in *Proceedings of the 16th annual conference on computer graphics and interactive techniques*, 1989, pp. 271–280.
[14] S. R. Marschner, H. W. Jensen, M. Cammarano, S. Worley, and P. Hanrahan, “Light scattering from human hair fibers,” in *ACM SIGGRAPH 2003 Papers*, 2003, pp. 780–791.

5.3 人物動作和衣服模擬

[1] G. Van Den Bergen, G. Van, et al. Efficient collision detection of complex deformable models using aabb trees. In J. Graphics Tools, 1998.
[2] S. Gottschalk, M. Lin, and D. Manocha. Obbtree: A hierarchical structure for rapid interference detection. In Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pages 171-180. ACM, 1996.
[3] J. Klosowski, M. Held, J. Mitchell, H. Sowizral, and K. Zikan. E_cient collision detection using bounding volume hierarchies of k-dops. Visualization and Computer Graphics, IEEE Transactions on, 4(1):21-36, 1998.
[4] I. Palmer and R. Grimsdale. Collision detection for animation using sphere-trees. In Computer Graphics Forum, volume 14, pages 105-116, 1995.
[5] S. Bandi and D. Thalmann. An adaptive spatial subdivision of the object space for fast collision detection of animated rigid bodies. In Computer Graphics Forum, volume 14, pages 259-270, 1995.
[6] S. Melax. Dynamic plane shifting bsp traversal. In Graphics interface, pages 213-220, 2000.
[7] T. Vassilev, B. Spanlang, and Y. Chrysanthou. Fast cloth animation on walking avatars. In Computer Graphics Forum, volume 20, pages 260-267, 2001.
[8] W. Wong and G. Baciu. Dynamic interaction between deformable surfaces and nonsmooth objects. IEEE Transactions on Visualization and Computer Graphics, pages 329-340, 2005.
[9] A. Watt and M. Watt. Forward vs inverse kinematics in computer animation. Advanced animation and rendering techniques, pages 371-384.
[10] K. Grochow, S. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. In ACM Transactions on Graphics (TOG), Vol. 23, pp. 522-531. ACM, 2004.
[11] Z. Andrew. Physically based motion transformation. 1999.
[12] S. Ishigaki, T. White, V. Zordan, and C. Liu. Performance-based control interface for character animation. ACM Transactions on Graphics (TOG), 28(3):1-8, 2009.
[13] U. Muico, Y. Lee, J. Popovi_c, and Z. Popovi_c. Contact-aware nonlinear control of dynamic characters. ACM Transactions on Graphics (TOG), 28(3):1-9, 2009.
[14] X. Provot. Deformation constraints in a mass-spring model to describe rigid cloth behaviour. In Graphics interface, 1995.

6 計畫成果自評

主持人和他的指導碩士學生總共發表了八篇論文, 其中一篇論文是最佳論文。論文列表如下:

1. Yu-Chun Cheng and **Sai-Keung Wong**, GPU Ray Tracing with Dynamic Scenes Using Reduced View-based BVHs, *Computer Graphics Workshop*, Taiwan, 2011.
2. Yu-Chun Cheng, **Sai-Keung Wong**, and Chun-Hung Hung, Hybrid View-based Approach for Continuous Self-Collision Detection Using CPUs and GPUs, *Computer Graphics Workshop*, Taiwan, 2011. (**Best Paper Award**)
3. Shih-Ping Lu, Tzung-Min Wang, and **Sai-Keung Wong**, Dressed Character Animation, *Computer Graphics Workshop*, Taiwan, 2011.
4. **黃世強**, 劉家銘, 何單期, 楊筱筑, 莊榮宏, 對稱感知的形體對應, *Computer Graphics Workshop*, Taiwan, 2011.
5. Cheng-Min Liu, **Sai-Keung Wong**, and Shing-Yeu Lii, Simulation of Inextensible Cloth Embedded with Rigid Objects, *Computer Graphics Workshop*, Taiwan, 2011. (poster)
6. Yi-Chen Chen, Chiao-Chih Yeh, Shih-Wei Fang, Kuan-Chen Lin, and **Sai-Keung Wong**, Modelling Fish Behaviours with Interaction in A Large Environment, in *Proceedings of the 24th IPPR Conference on Computer Vision, Graphics, and Image Processing*, 2011.
7. Yi-Chen Chen and **Sai-Keung Wong**, Animating Complex Trees with Collision Handling, in *Proceedings of 24th IPPR Conference on Computer Vision, Graphics, and Image Processing*, 2011.
8. Wei-En Chen, **Sai-Keung Wong** and Shih-Ping Lu, Real Time Simulation and Rendering of Hair on Graphics Hardware, *National Computer Symposium (全國計算機會議 NCS), Workshop on Image Processing, Computer Graphics, and Multimedia Technologies*, 2011, accepted.

學生論文：

1. 鄭游駿，View-based Continuous Self-Collision Detection with Graphics Hardware，碩士論文，國立交通大學。
2. 劉政旻，Simulation of Inextensible Cloth Embedded with Rigid Bodies，碩士論文，國立交通大學。
3. 陳蔚恩，基於圖形處理器的即時頭髮渲染與模擬，碩士論文，國立交通大學。

人物運動、衣服模擬、頭髮模擬以及光線追蹤都有廣泛的應用，在多核心平台上進行平行運算，可以加快模擬和渲染程序的速度，能夠在比較短的時間內完成模擬和渲染程序，縮短製作時間，不論在電影、遊戲及衣服設計都有十分重要意義。

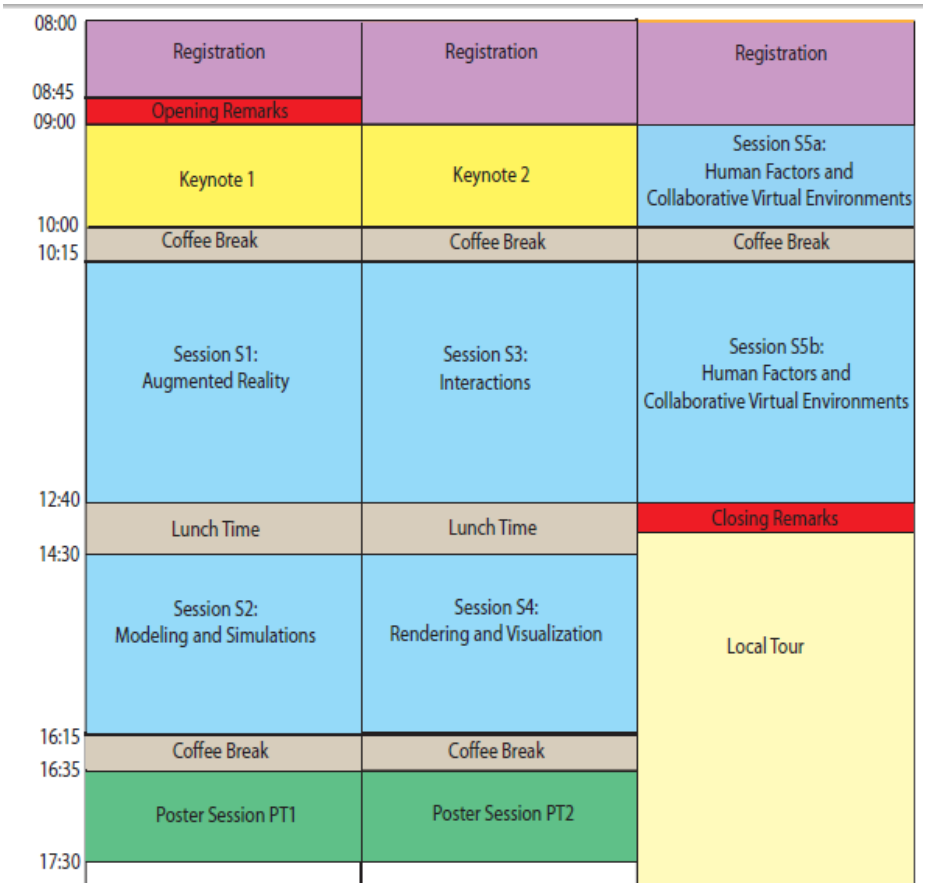
附件四

國科會補助專題研究計畫項下出席國際學術會議心得報告

日期：100 年 10 月 22 日

計畫編號	NSC 99 - 2221 - E - 009 - 143 -		
計畫名稱	研究混合型的平行運算系統及其應用在三維人物運動、布料模擬、頭髮模擬以及光線追蹤		
出國人員姓名	黃世強	服務機構及職稱	國立交通大學 助理教授
會議時間	99 年 11 月 22 日 至 99 年 11 月 24 日	會議地點	香港
會議名稱	(中文) (英文) The ACM Symposium on Virtual Reality Software and Technology		
發表論文題目	(中文) (英文) **Robust Continuous Collision Detection for Deformable Objects**		

一、參加會議經過



這個會議維持三日，第一日在上午有 Keynote speech, 接下來是兩場的 Full paper presentation, 然後是 poster-presentation，第二日在上午有 Keynote speech, 接下來是兩場的 Full paper presentation, 然後是 poster-presentation。最後是晚宴. 第三日在上午有兩場的 Full paper presentation. 最後散會。

二、與會心得

在會議期間，能夠和一些國際的知名學者討論研究上的問題, 獲益良多。一些演講者在演講的時候十分生動. 都是十分值得我們學習，同時也可以請教主辦單位，如何籌備國際會議，在可見的未來, 當自己要籌備同樣的國際會議時，有一定的幫助。

三、考察參觀活動(無是項活動者略)

無。

四、建議

沒有。

五、攜回資料名稱及內容

Proceedings of the conference，內容主要是在會議發表的論文。

六、其他, 沒有。

論文摘要:

Continuous collision detection improves the computation of the contact information for interacting objects in dynamic virtual environments. The computation cost is relatively high in the phase of the elementary test processing. In virtual environments, such as crowds in large urban models, there is a large portion of feature pairs that do not collide but the computation is relatively of high cost. In this paper, we propose a robust approach for solving the scalability of the collision detection problem by applying four distinct phases. First, k-DOPs are used for culling non-proximal triangles. Second, the feature assignment scheme is used for minimizing the number of potentially colliding feature pairs. Third, an intrinsic filter is employed for filtering non-coplanar feature pairs. Forth, we use a direct method for computing the contact time that is more efficient than the numerical Interval Newton method. We have implemented our system and have compared its performance with the most recently developed approaches. Six benchmarks were evaluated and the complexity of the models was up to 1.5M triangles. The experimental results show that our method improves the performance for the elementary tests.

國科會補助計畫衍生研發成果推廣資料表

日期:2011/07/20

國科會補助計畫	計畫名稱: 研究混合型的平行運算系統及其應用在三維人物運動、布料模擬、頭髮模擬以及光線追蹤
	計畫主持人: 黃世強
	計畫編號: 99-2221-E-009-143- 學門領域: 計算機圖學

無研發成果推廣資料

99 年度專題研究計畫研究成果彙整表

計畫主持人：黃世強	計畫編號：99-2221-E-009-143-
計畫名稱：研究混合型的平行運算系統及其應用在三維人物運動、布料模擬、頭髮模擬以及光線追蹤	

成果項目			量化			單位	備註（質化說明：如數個計畫共同成果、成果列為該期刊之封面故事...等）
			實際已達成數（被接受或已發表）	預期總達成數(含實際已達成數)	本計畫實際貢獻百分比		
國內	論文著作	期刊論文	0	0	100%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	8	0	100%		其中一份論文是會議最佳論文。
		專書	0	0	100%		
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（本國籍）	碩士生	10	0	100%	人次	
		博士生	0	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		
國外	論文著作	期刊論文	0	1	100%	篇	已投稿兩份期刊論文。
		研究報告/技術報告	0	0	100%		
		研討會論文	0	0	100%		
		專書	0	0	100%	章/本	
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（外國籍）	碩士生	4	0	100%	人次	
		博士生	0	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		

其他成果 (無法以量化表達之成果如辦理學術活動、獲得獎項、重要國際合作、研究成果國際影響力及其他協助產業技術發展之具體效益事項等，請以文字敘述填列。)	無。

	成果項目	量化	名稱或內容性質簡述
科教處計畫加填項目	測驗工具(含質性與量性)	0	
	課程/模組	0	
	電腦及網路系統或工具	0	
	教材	0	
	舉辦之活動/競賽	0	
	研討會/工作坊	0	
	電子報、網站	0	
	計畫成果推廣之參與（閱聽）人數	0	

國科會補助專題研究計畫成果報告自評表

請就研究內容與原計畫相符程度、達成預期目標情況、研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）、是否適合在學術期刊發表或申請專利、主要發現或其他有關價值等，作一綜合評估。

1. 請就研究內容與原計畫相符程度、達成預期目標情況作一綜合評估

■達成目標

□未達成目標（請說明，以 100 字為限）

□實驗失敗

□因故實驗中斷

□其他原因

說明：

2. 研究成果在學術期刊發表或申請專利等情形：

論文：■已發表 □未發表之文稿 □撰寫中 □無

專利：□已獲得 □申請中 ■無

技轉：□已技轉 □洽談中 ■無

其他：(以 100 字為限)

3. 請依學術成就、技術創新、社會影響等方面，評估研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）(以 500 字為限)

我們的研究是關於發展一個平台, 進行三維人物運動、布料模擬、頭髮模擬以及光線追蹤。這系統是建立在多核心平台以及具有串流處理器的繪圖處理器上。我們提出的方法對這方面的研究，十分有助益。

學術成就: 主持人和他的指導碩士學生總共發表了八篇論文, 其中一篇論文是最佳論文。

在技術創新方面, 我們完成三個系統:

1. 在圖像處理器上應用視角速進行光線追蹤, 我們應用視角方法把物體的基本元素三角形分堆,然後把每一分堆建立 Bounding Volume Hierarchy。最後對這些 Bounding Volume Hierarhies 進行光線追蹤。整個過程完全在圖形處理器(GPU)上進行。

2. 在多核心平台上模擬頭髮運動, 我們將計算頭髮運動的模擬流程，利用圖形處理器(GPU)來做加速，在 NVIDIA 所開發的 CUDA(Compute Unified Device Architecture)的架構上實現演算法的加速。我們利用 OpenGL 4.0 繪圖流程管線(pipeline)所新增的鑲嵌階段(tessellation stage)，撰寫自定的著色程式(shader)，在 GPU 上動態的產生頭髮的幾何模型，省去 CPU 和 GPU 間要透過 PCI-E bus 互相傳遞的花費，以達成即時的頭髮渲染。

3. 在人物動作和衣服互動系統方面, 人物的運動以骨架方式進行，使用碰撞偵測並且予

以適當的碰撞反應，應用適應性方法對人物模型和衣服的互動重複調校，處理衣服的自身穿透或衣服穿透角色的情形。

應用價值: 人物的運動, 衣服和頭髮模擬以及光線追蹤都有廣泛的應用，是十分值得研究的項目。在多核心平台上進行平行運算，能夠在比較短的時間內完成運算，可以加快模擬的速度，縮短製作時間，不論在電影、遊戲及衣服設計都有十分重要意義。同時，由於模擬的速度加快，增加使用者的工作效率，對社會的進步有所助益。