國 立 交 通 大 學

多媒體工程研究所

碩 士 論 文

基於電荷分佈的變形物體連續自身碰撞偵測

Continuous Self-Collision Detection for Deformable Objects Based on Charge Distribution

研 究 生：葉喬之

指導教授：黃世強 教授

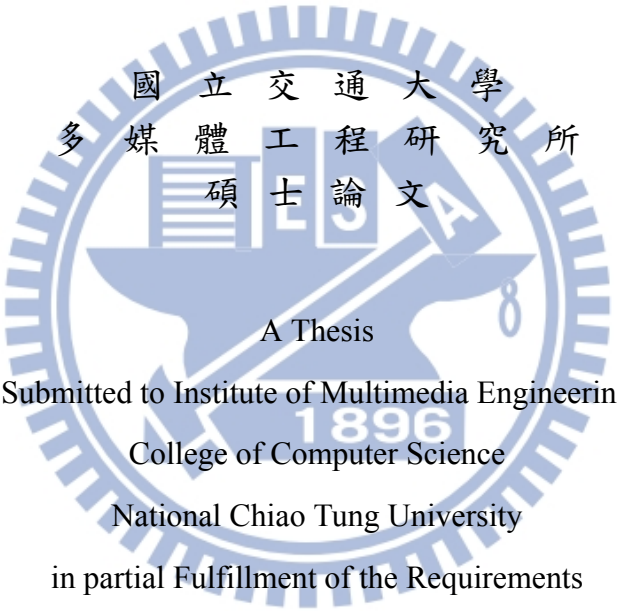
中 華 民 國 一 百 零 一 年 七 月

基於電荷分佈的變形物體連續自身碰撞偵測

Continuous Self-Collision Detection for Deformable Objects Based on Charge Distribution

研　究　生：葉喬之　　Student　：Chiao-Chin Yeh

指 導 教 授：黃世強　　Advisor　：Sai-Keung Wong



國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2012

Hsinchu, Taiwan, Republic of China

中 華 民 國 一 百 零 一 年 七 月

基於電荷分佈的變形物體連續自身碰撞偵測

研究生：葉喬之　　指導教授：黃世強 教授

國 立 交 通 大 學

多 媒 體 工 程 研 究 所

摘要

在這篇論文中，我們提出了一個對封閉物體的連續自身碰撞偵測的新方法，它結合了物體分割和 detector-based 的碰撞偵測。Detector-based 的碰撞偵測是在一個區域中，利用一個區域內部的點來檢查此區域中所有三角形的面向，並以三角形面向的資訊對此區域作自我碰撞偵測的方法。在物體分割部分，我們在前置處理的過程中，利用計算電荷分佈的方法來分析一個物體，電荷的分佈關係能表現出一個物體結構上的強度。一個區域的電荷分佈少，代表它在一個局部區域之中是一個比較凹陷的區塊，我們將它看成是局部結構強度較低的部分，也是物體容易發生形變的部分；反之，一個區域的電荷分佈多，表示是一個局部區域中比較凸的區塊，我們將它看成是局部結構強度較高的部分。我們根據這種物體結構強度的物理特性將物體切割成幾個在模擬過程中較不易發生形變的區域。在模擬的過程中，我們對這些分割好的區域分別作 detector-based 的自我碰撞偵測，區域和區域之間則利用物體間的碰撞偵測。我們利用幾個不同的實驗來對我們的方法和利用 *K*-means 分割物體的方法以及我們實作的 ICCD 方法做比較。實驗的結果展現出，我們的方法整體效能比 *K*-means 穩定，而比 ICCD 提升了 1.88X ～ 2.19X。

Continuous Self-Collision Detection for Deformable Objects Based on Charge Distribution

Student: Chiao-Chin Yeh Advisor: Sai-Keung Wong

National Chiao Tung University

Institute of Multimedia Engineering

Abstract

In this thesis, we propose a new method to accelerate self-collision detection with closed objects. Our method includes object decomposition and detector-based collision detection. In detector-based collision detection, it utilizes a point that inside a region to check all triangles' orientation of this region. And this is a method that performs self-collision detection based on triangles' orientation of this region. On the part of object decomposition, we analyze the physical property of an object by computing its charge distribution in the preprocessing phase. The charge distribution of an object could present strength of structure of the object. A region with less charge means that the region is concave part at the local area. We regard the region as lower structural strength at the local area, and it is easily deformed region of an object. On the other hand, a region with more charge indicates that the region is convex part at the local area. We regard the region as higher structural strength at the local area. We segment an object into several not easily deformed regions based on structural intensity of the object. In the simulation phase, we perform detector-based self-collision detection on these segmented regions and inter-

collision detection between each region. We use some different experiments to compare our method with *K*-means decomposition method and our implementation on ICCD. The experiment results show that our approach is more stable than *K*-means decomposition method. Compared to ICCD, our method improves self-collision detection by a factor of 1.88X $\sim$ 2.19X.

Acknowledgments

At first, I would like to thank my advisor, Dr. Sai-Keung Wong for his supervision in these years. He was very careful and patient in teaching me on my research. Also, he gave me many suggestions that helped me to complete this thesis. In addition to research, he gave me many concepts about the attitude toward life. I would also like to thank my thesis committee members, Dr. I-Chen Lin and Dr. Chun-Cheng Lin, who evaluated my thesis and gave me some suggestions.

I would like to thank all my labmates. They gave me useful comments that helped me a lot. Thanks to all my basketball teammates, we have some memories on the basketball court. And when I was down, they pushed me back up.

Finally, I would like to give many thanks to my family and all my friends for their supports and encouragements when I was disappointed.

Chiao-Chin Yeh

July 2012

Contents

List of Figures

Chapter 1

Introduction

碰撞偵測是一個非常重要的領域而且被廣泛的應用，例如在遊戲、電影、機器人學、以及醫學等，這個領域已經被研究將近三十年，而且傳統的方法也已經到了瓶頸。在這個領域之中，相對於剛性物體的碰撞偵測，軟性物體有更大的機會發生自身碰撞，由於現今電腦硬體設備性能的提升或是人類對視覺品質的要求，物體的複雜度也扶搖直上，在計算的複雜度也相對提高，因此解決軟性物體的自身碰撞是最困難的，並且也是最有研究價值的部分。

碰撞偵測分為非連續性碰撞偵測 (DCD) 和連續性碰撞偵測 (CCD)。在 DCD 中，我們考慮每個 discrete frame 的時間點上三角形之間有沒有發生碰撞。在 CCD 中，影格和影格之間我們會對三角形的位置根據時間的比例做內插，檢查在每個影格之間三角形是否會發生碰撞。相較於 DCD，CCD 會花費大量的時間運算，卻能算出精準的碰撞時間，這對模擬是很重要的，能避免一些物體穿透的問題。

傳統上，大部分的碰撞器 [TCYM09] [VT94] [GS01] [WB05] [SPO10] [TMT10a] [TMT10a]，在前置處理時會先對物體建 bounding volume hierarchies (BVHs)。在 run time 時首先更新 BVHs，並且對 BVHs 作追蹤，在自我碰撞偵測的部分，根據三角形的法向量建立 regular patches，將物體的表面分成幾個區域。在追蹤的過程中，若一個區域的 contour test 通過，才能確定這些區域的三角形是沒有發生碰撞的，contour test 是將一個 regular patch 的輪廓投影到一個平面，並檢查每個投影輪廓的邊在時間區間內是否有重疊發生，若沒有重疊發生表示通過 contour test，而其餘沒有通過 contour test 區域中的三角形都必須繼續追蹤。

在追蹤的過程中，蒐集一些 bounding volume 節點有重疊的三角形對，稱為 potentially colliding pairs，我們將所有蒐集到的 potentially colliding pairs 作 elementary test，找出真正發生碰撞的三角形對和發生碰撞的時間。

同樣地，我們的 detector-based 碰撞偵測方法，在前置處理時也會先對物體建 BVHs 並額外找一個物體內部的點，作為 detector-point。不同的是，在模擬時，物體上的每個三角形都會根據法向量對 detector-point 計算出面向，並根據三角形面向的類型對 BVH 作追蹤。

1.1 Motivation

在碰撞偵測這個領域，大部分的方法都是根據物體表面的法向量將表面分堆或是利用物體的拓撲關係來對碰撞偵測作加速，比較少考慮針對物體在結構上的物理特性作碰撞偵測的加速。我們的想法是：在一個物體中，容易形變的部分就代表它是在一個物體中結構性比較薄弱的部分。Detector-based 碰撞偵測的基本概念是考慮三角形的面向來決定三角形之間有沒有可能發生碰撞。在 detector-based 的碰撞偵測中，我們會先定義一個物體內部的點作為 detector-point。若所有要偵測的三角形都是面對此 detector-point，我們就能直接說這些要偵測的所有三角形彼此之間不會發生碰撞，這對 detector-based 的碰撞偵測來說是最好的情況。我們希望可以將一個物體分成一些不容易發生形變且適合作 detector-based 碰撞偵測的區域，並找到適合的 detector-point，使得每個區域大部分的三角形都是面對該區域的 detector-point。找到物體結構性比較薄弱的部分並且把物體分割成幾個不容易發生形變且適合作 detector-based 碰撞偵測的區域，對這些區域分別作 detector-based 的碰撞偵測能提高 detector-based 碰撞偵測的效率，其餘的就是要考慮每個區域之間發生的碰撞。

1.2 Overview

我們的系統是利用動畫的資訊作為我們測試碰撞偵測的 benchmark，我們的動畫資訊是一個有連續性的物理模擬的過程，它記錄了整個模擬過程中每個

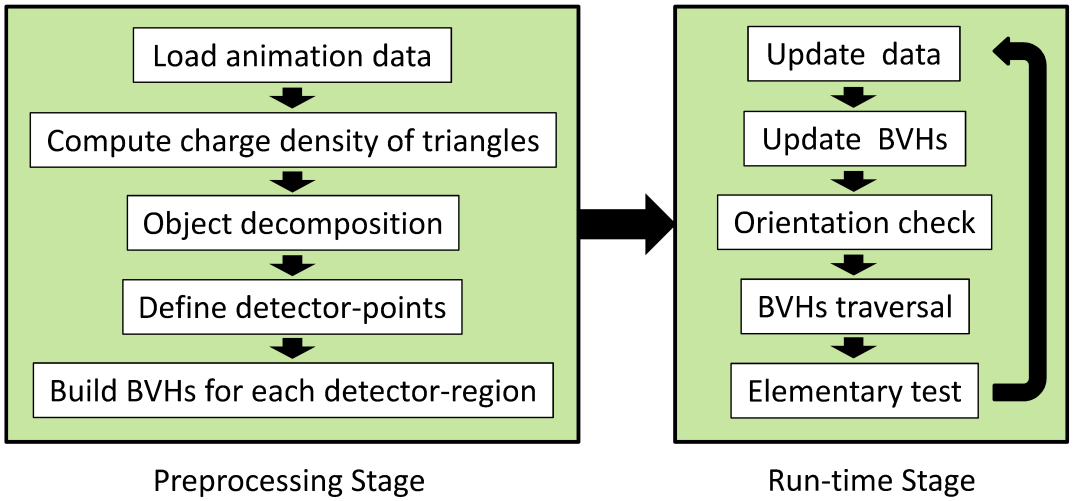


Figure 1.1: 流程圖。

三角形的位置。我們系統的流程主要分成兩大部分，見 Figure 1.1。第一部分是 preprocessing stage，另一部分為 run-time stage。

在 preprocessing stage 部分，一開始，讀入我們的 benchmark，計算出物體中，每個面的電荷密度，接著利用電荷密度的結果來將物體作適當的分割，而分割出的每個區域，我們稱為 detector-region，在每個 detector-region 中，我們會指派一個 detector-point，並且建出每個 detector-region 的 BVH。

在 run-time stage，每個影格開始時，我們先更新所有點的資訊，並根據物體中每個三角形新的位置來更新 BVHs，之後每個 detector-region 的三角形都要對該 detector-region 的 detector-point 作 orientation check。orientation check 是利用一個 detector-region 中的三角形的法向量，在一段時間區間內是不是都指向該 detector-region 的 detector-point，作為面向的依據。根據這些三角形的面向來作追蹤，搜集 potentially colliding pairs，最後將過濾所有 potentially colliding pairs，搜集出不會產生重複測試的 pairs 作 elementary test。

1.3 Contribution

我們提出了一個直覺並且新奇的方法加速封閉物體的連續自身碰撞偵測，它的主要方法是在前置處理的過程利用計算電荷密度的方式並結合 cluster-growing

和 cluster-shrinking 的技術，利用物體在結構上的物理特性對物體作分割，預測發生自身碰撞機率較低的區域，並且以 detector-based 為主的方式，對分割後的物體作碰撞偵測，最後展示出我們的實驗結果，以及對傳統的方法和 *K*-means 的分割方法做分析比較。

1.4 Organization

以下各章節的組織如下：第二章我們對碰撞偵測和物體分割的研究作了大略的分類，並加以描述，第三章為計算電荷分佈和物體分割的方法細節介紹。第四章為碰撞偵測的方法細節介紹。第五章描述我們的系統的實作環境和實驗結果，第六章則分析我們的系統並根據方法的一些限制作討論，最後第七章是總結和未來的研究方向。

Chapter 2

Related Work

2.1 Collision Detection

2.1.1 Bounding Volume Hierarchies

現在處理碰撞的方法有非常多種，而 BVHs 的資料結構被廣泛運用，它是利用某些凸多面體將物體包住並建成樹的架構，結合空間的概念將物體分堆，並使用追蹤的方法，能提供最高效能刪去沒有發生碰撞的三角形。最常見的 BVHs 包括 AABBs [VDBVB98] 和 Sphere tree [PG95]。相較於上述兩種 BVHs，OBBs [GLM96]，和 k-DOPs [KHM+98] 比較緊密。k-DOPs 是利用定義好的 $k/2$ 個軸，每個軸的正負兩個方向分別作為 k 面體 k 個面的法向量。對於較緊密的 BVHs 來說，它減少了在追蹤過程中重疊發生的機會，降低 false positve 的機率，進而增加 culling 的效率，但是它們需要更大量的空間和更多的時間來計算 BVHs 的更新。

2.1.2 Normal Cone and Contour Test

自身碰撞偵測加速方面，Volino 提出了 normal cone 和 contour test 的觀念 [VT94]，normal cone 是根據三角形和三角形之間的 normal 關係建立的，在每個 BVH 的節點中，若所有三角形法向量的夾角範圍小於 π，則這個節點存在一個 normal cone，此 normal cone 能將此節點中所有三角形的法向量包住。利用後序的方式檢查每個節點是否有 normal cone 存在，若檢查到一個節點不存在 normal

cone，則此節點的任一子節點所包住的三角形形成 regular patch，物體表面切割成了許多這種 regular patches，每個 regular patch 不會發生自身碰撞。Grinspun 提出了 subdivision-surface [GS01]，這個方法首先將物體表面分成好幾個 patches，並在模擬的過程中，檢查一個 patch 有沒有自身碰撞，若沒有則分別檢查該 patch 的 subpatches 有沒有自身碰撞，並檢查每兩個 subpatches 之間有沒有發生碰撞，利用此遞迴的方式找出發生碰撞的 patches。Contour test 是將 regular patch 的輪廓投影到平面上，檢查輪廓的邊之間有沒有發生重疊的情形，是處理在一些特別情況雖然在一個 bounding volume 節點中有 normal cone 存在卻還是會發生碰撞的現象。Wong [WB05] 和 Tang [TCYM09] 延伸 normal cone 的觀念到連續碰撞偵測。Schvartzman 提出 star-contour [SPO10] 加速 contour test，利用三角形相鄰的關係建立 self-collision test tree (SCTT)，而 SCTT 每個節點的 contour 又會以 contour segment 的方式儲存。每個節點中，AABB 的垂直中線和水平中線定義此節點的 search line，將節點的 contour segment 的所有邊都依照 orientation 將邊分成向上和向下兩類，並檢查 contour segment 所有邊的延伸和 search line 的交點，若向上邊的延伸和 search line 的最低交點 t 高於向下邊的延伸和 search line 的最高交點 b，並且 search line 只和向上的邊只有一個交點，那麼這個節點的 contour 是屬於一個 star-contour，不會發生自身碰撞。

2.1.3 View-Based Collision Detection

View-based 碰撞偵測的直覺想法就是，一些看得到的面不會發生自身碰撞，view-based 碰撞偵測在物體內部找一條 view-line 或是一個 view-point，利用 view-line 或 view-point 對整個物體的三角形作面向的測試，將三角形分成面向和背向的兩種 patches 作自身碰撞偵測 [Che11]。

2.1.4 Duplicate Features

在三角形之間，共享的點或邊可能會造成重複的 elementary tests。因此有一些文獻提出了 assign feature 的方法 [WB06][CTM08]，當一個三角形要做 elementary test 時，只需要做有被 assigned 的 feature，避免重複的 elementary test。Tang 提出

了 procedure representative triangles (PR-Triangles) 的資料庫，它能在 run-time 決定一對 features 要不要執行 elementary test [TCYM09]。而相鄰的三角形之間也有一些 elementary test 是不需要做測試的，Tang 提出了 orphan set 的觀念，在前置處理過程中先將一些相鄰三角形所需要做 elementary test 的 feature pair 儲存到 orphan Set，當一對 features 要執行 elementary test 前，檢查這對 features 需不需要執行 elementary test。

2.1.5 Elementary Test

Elementary test 是決定兩個三角形是否發生碰撞的基本測試，而決定兩個三角形是否發生碰撞，需要執行十五次 elementary tests，其中包括九次 *edge-edge tests* 和六次的 *vertex-face tests*。Elementary test 主要分成兩階段，首先，檢查四個點在一個時間區間 (time interval) 中是不是有共平面的情況發生。若此情況成立，再檢查共平面的時間點，features 之間是否真的發生碰撞。[Pro97] 和 [BFA02] 提出了算出三次方程式的根來找出 features 之間碰撞時間的方法。Non-penetration filter 利用檢查 features 之間共面的條件加速了 elementary test [TMT10a]。

2.1.6 GPU Base

最近幾年有許多平行處理的演算法來改善碰撞偵測的效能。HPCCD [KHH+09] 利用了 CPUs 和 GPUs，它使用多核心 CPUs 來執行追蹤和 culling，並且利用 GPU 來處理 elementary tests。Tang 等人提出了一個利用 front-based decomposition 的演算法 [TMT10b]，考慮了連續影格之間的時間相依性。

2.2 Object Decomposition

物體分割的主要目的是將一個複雜度較高的物體分成許多簡單的區塊，這些較簡單的區塊能被更有效率的應用，而分割的類型大略分成以下三類：

2.2.1 Convex

這一類型的方法是找出物體凹面的部分，將物體分成 convex regions。Convex decomposition 可以分為 exact convex decomposition (ECD) 和 approximate convex decomposition (ACD)，但是 ACD 也會保留重要的 features，而且花費相對少的時間。Liu [LLL10] 根據 Morse theory，計算許多的 Morse functions，而每個 Morse function 都利用 Reeb graph 來找出 Mutex Set 和 Candidate Cut Set，並在 Cut Set 找出哪些 Cut 滿足 Mutex pair 的條件，找出 concave 的部分，最後找出的每個 Cut 根據物體結構來調整，盡量減少分割上的 cost。Lien [LA07] 利用 SL-concavity 測量物體的 concavity，若一個物體的 concavity 大於 τ，就必須分析物體的 knots 和 pocket cut，並從 pocket cut 中找出最適合分割物體的幾個 cuts，將物體分成好幾個 concavity 都小於 τ 的 regions。Katz [KT03] 利用物體建立的 BVH，將物體分成 k 個初始的 patches，算出每個 patch 中面和面之間的距離，利用機率的方式分配每個面是屬於哪一個 patch，而一些 fuzzy 的面則利用 minimum cuts 的方法來決定屬於哪一個 patch，並以 iterative 的方式找出最後的 patch，這個方法避免了一個面被重複分到不同的 patches 中的現象，並且在 patches 的邊界間不會產生鋸齒狀。

2.2.2 Star Shape

若能在一個物體中找到一個點使該點能看到物體表面所有的點，我們稱此物體是 star-shape。如果一個點無法看到物體所有的點，必須找到一些點以便看到物體所有的點，使物體由一些 Star-shape components 所組成。Star-shape decomposition 的做法會根據物體的組成而有所不同，例如 Botsch 提出利用 point set data 來對物體作分割 [Lie07]，對物體的所有點依照 sphere coordinate 的兩個角度去作排序，並利用 sweep 的方法分別檢查這兩個排序的結果，判斷三角形之間有沒有可能被擋住。Ben-Moshe 提出利用物體的 polygons 來對物體作分割 [BMKM05]，主要以 terrain 在 1.5D 的空間中，將 terrain 看成幾個 subterrain，並由 subterrain 之間的點去檢查 subterrain 需不需要 guard，找出最少的 guards 而能看到整個 terrain。

2.2.3 Random Cuts

此類的方法一開始都是偏向以隨機的方式尋找切割的位置，將物體切成許多 components，例如，*K*-means [STK02]，這篇 paper 是利用 *K*-means 的方式將不同的物體分割成好幾個 patchs，並在不同的物體間找出對應的 patch 將物體作變形；hierarchical clustering [GWH01] [KT03]，Garland 利用 dual graph 建立 face clustering，每個 node 為一個 cluster，初始的每個 node 為一個面，若面和面相鄰則相對的 node 與 node 之間就有 edge，利用 Dual Quadric 的方式，算出每個 edge contrast 所需的 cost，並利用 greedy method 的方式，從最小 cost 的 edge 開始作 contrast，而被 merge 成一個 node 的兩個面，就是一個新的 face cluster，並作 iterative 的動作，分出最後的 face clustering。Golovinskiy [GF08] 利用 random segmentation 的方法將物體切成許多 components，產生 partition function，找出每個 component 的邊界。藉由改變參數產生不同的 component，接著利用 partition function 比較哪些邊界的邊出現比率最高，將這些邊界當做切割的依據。

Chapter 3

3D Mesh Segmentation

為了找出物體容易形變的區域，我們必須分析物體在結構上的物理特性。這個章節敘述我們計算電荷密度的方法以及我們如何對物體分割，這個方法的主要想法是將物體看成是一個帶電的導體，導體的形狀會影響電荷在這個導體表面的分佈。在現實情況中，電荷主要會分佈在物體比較凸的表面，或者是物體的邊緣。而物體凹面的部分僅有少量的電荷存在甚至不帶有任何電荷。通常在切線平面不連續的凹面的區塊，會被視為物體適合被切割的部分，而電荷也幾乎不分佈在此區塊，我們認為這個區塊是物體結構比較薄弱並且容易因外力作用而產生形變的區塊，因此利用電荷分佈來分析整個物體結構的強弱並加以分割。

3.1 Charge Density Computation

我們的物體是由三角形網格組成，我們將它看成是一個在 3D 場景中帶有電荷的導體。電荷的導體的形狀會影響電荷分佈的情況，電荷導體中的每個電荷都是一個可以自由移動的個體，在同性相斥的物體特性下，每個電荷會盡量遠離其他電荷，最後達到平衡。一個物體的表面是由 N 個三角形組成，$T_k,\ k = 1, ..., N$。我們利用在 3D 空間中的這些三角形的相對關係來計算出每個三角形所帶的電荷密度。物體中的一個三角形帶有電荷，此三角形勢必會對其他三角形的帶電荷量有相當程度的影響，每個三角形累積了所有其他三角形對此三角形的電荷影響，藉此計算出每個三角形的帶電比例，並利用總帶電量將每個三角形的帶電量計算

出來。則根據 Wu 等人提出的方法 [WL97]：

令三角形 T_k 的電荷密度是 ρ_k，則

$$V = \sum_{k=1}^{N} \rho_k \int_{T_k} \frac{1}{|r_j - r'|};\ j = 1, ..., N, \tag{3.1}$$

在這裡 V 是一個常數，它的直觀想法是：物體中所有三角形的電荷密度對三角形 T_j 帶電荷密度的影響。

$$Q = \int_S \rho(r)\, dS' \approx \sum_{k=1}^{N} \rho_k S_k, \tag{3.2}$$

Q 為此物體的總帶電量，S_k 為三角形 T_k 的面積。假設 Q 為已知，我們有 $N+1$ 個未知數，ρ_1, ..., ρ_k 和 V，並且有 $N+1$ 條方程式。我們能解此線性系統:

$$A\rho = \phi, \tag{3.3}$$

在這裡

$$\rho = \begin{bmatrix} \rho_1 & \rho_2 & ... & \rho_N & V \end{bmatrix}^T;\ \phi = \begin{bmatrix} 0 & 0 & ... & 0 & Q \end{bmatrix}^T; \tag{3.4}$$

$$A = \begin{bmatrix} A_{11} & A_{12} & . & A_{1N} & -1 \\ A_{21} & A_{22} & . & A_{2N} & -1 \\ . & & & . & \\ . & & & . & \\ . & & & . & \\ A_{N1} & A_{N2} & . & A_{NN} & -1 \\ S_1 & S_2 & . & S_N & 0 \end{bmatrix}, \tag{3.5}$$

在這裡

$$A_{jk} = \int_{T_k} \frac{1}{|r_j - r'|} dS'.\ j, k = 1, 2, ..., N, \tag{3.6}$$

r_j 為三角形 T_j 的質心，r' 為我們所要算電荷密度的位置。

3.2 Surface Sources Distributed on Polygon

計算 A_{jk} 的方法可以參考 [WRG+84]，計算一個表面 T_k 對一個三角形 T_j 所提供電荷的影響，三角形 T_j 的質心為 r。假設三角形 T_k 位於平面 P，Figure 3.1，$\hat{n}$ 為平面 P 的法向量，ρ 是 r 到平面 P 的投影點，而三角形 T_k 的邊分別為 r_i，i = 1, ..., 3，$r_i^{\pm}$ 分別為 r_i 的兩端點。

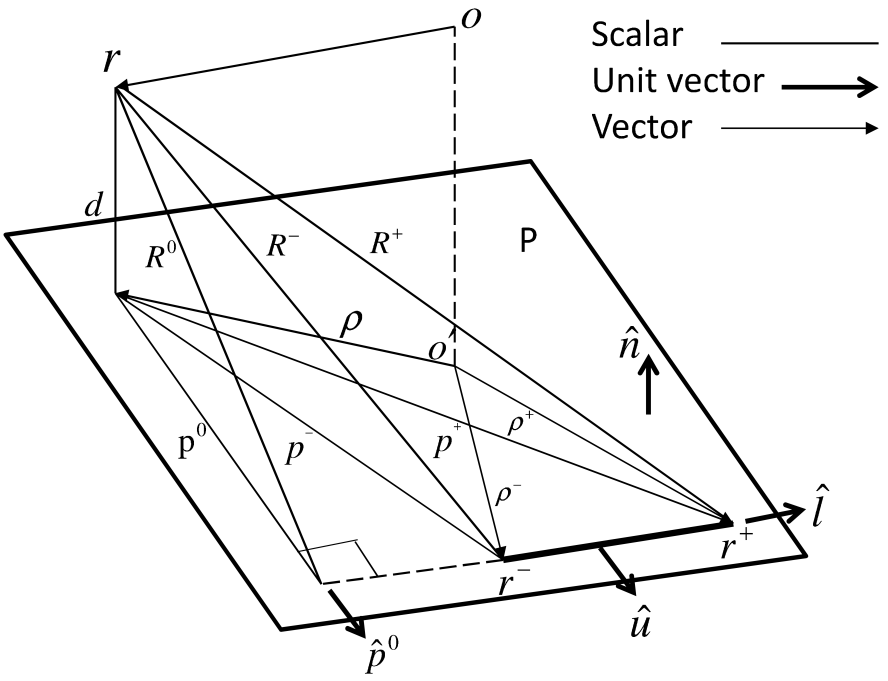


Figure 3.1: 計算電荷密度示意圖。

d 是 r 到平面 P 的距離，

$$d = \hat{n} \cdot \left(r - r_i^{\pm} \right), \tag{3.7}$$

$\hat{l}_i$ 為 r_i 方向的單位向量，

$$\hat{l}_i = \frac{r_i^+ - r_i^-}{\left| r_i^+ - r_i^- \right|}, \tag{3.8}$$

ρ 到 $r^{\pm}$ 的距離為 $\rho^{\pm}$，

$$\rho_i^{\pm} = r_i^{\pm} - \hat{n} \left(\hat{n} \cdot r_i^{\pm} \right), \tag{3.9}$$

$$\hat{l} = \frac{\rho^+ - \rho^-}{|\rho^+ - \rho^-|}, \tag{3.10}$$

$\hat{u}$ 是 $\hat{l}$ 的法向量，

$$\hat{u} = \hat{l} \times \hat{n}, \tag{3.11}$$

純量 $l^{\pm}$、P^0、$P^{\pm}$ 和單位向量 $\hat{P}^0$ 可以分別由 ρ 和 $\rho^{\pm}$ 計算出來，

$$l^{\pm} = \left(\rho^{\pm} - \rho\right) \cdot \hat{l}, \tag{3.12}$$

$$P^0 = \left|\left(\rho^{\pm} - \rho\right) \cdot \hat{u}\right|, \tag{3.13}$$

$$P^{\pm} = \left|\rho^{\pm} - \rho\right| = \sqrt{\left(P^0\right)^2 + \left(l^{\pm}\right)^2}, \tag{3.14}$$

$$\hat{P}^0 = \frac{\left(\rho^{\pm} - \rho\right) - l^{\pm}\hat{l}}{P^0}. \tag{3.15}$$

從 r 到 r_i 的距離為 R_i^0，從 r 到 r_i 兩個端點的距離為 $R_i^{\pm}$，

$$R_i^0 = \sqrt{\left(P_i^0\right)^2 + d^2}\,, R_i^{\pm} = \sqrt{\left(P_i^{\pm}\right)^2 + d^2}, \tag{3.16}$$

則一個表面對一個三角形所提供的電荷影響會和以下公式成比例:

$$\int_S \frac{dS'}{R} = \sum_i \hat{P}_i^0 \cdot \hat{u}_i \left[P_i^0 ln \frac{R_i^+ + l_i^+}{R_i^- + l_i^-} - |d| \times \left(\tan^{-1} \frac{P_i^0 l_i^+}{\left(R_i^0\right)^2 + |d|\, R_i^+} - \tan^{-1} \frac{P_i^0 l_i^-}{\left(R_i^0\right)^2 + |d|\, R_i^-} \right) \right], \tag{3.17}$$

R 為 $|r - r'|$，r' 為 r_i 上的點，我們稱為 source point。

3.3 Object Decomposition

我們的目標是要根據電荷密度將物體分割。我們必須要找出一些三角形，它們能將物體分成不同的區域。在這篇論文中，我們定義如果一些三角形能繞物體一圈，表示它能將物體分割。

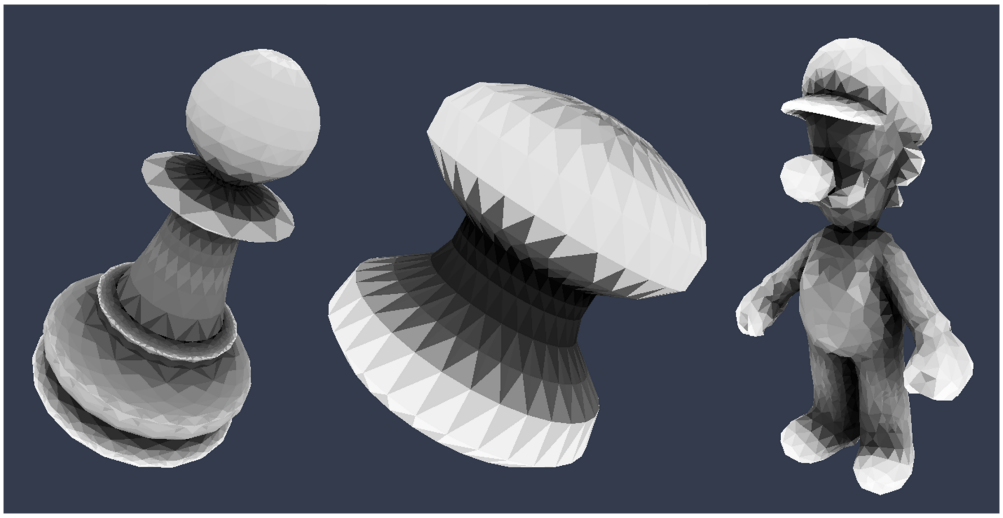


Figure 3.2: Charge distribution: 此圖左、中、右分別為 pawn、dumbbell、Luigi 算出電荷分佈的結果，我們以 gray level 的方式表示電荷的強度，越黑的代表電荷密度越低，越白代表電荷密度越高。

在計算出每個三角形所帶的電荷密度後，我們先對此物體每個三角形所帶的電荷密度作直方圖均衡化 [L+81]，三角形的總數量為 N 使得每個三角形所帶的電荷密度以線性的方式作累積，提高電荷密度的對比性。我們先用一個陣列 CDF 累積三角形的數量，$CDF[0]$ 為電荷密度等於 0 的三角形數量，$CDF[i]$ 為電荷密度小於等於 i 的三角形數量，$i = 0, ..., 255$，$CDF[255] = N$，以此類推，我們累積所有強度共有多少三角形。並將原始的電荷密度值對應到新的電荷密度值：

$$chargeDensity_{new}[i] = CDF[i]/N. \tag{3.18}$$

我們選擇一個門檻 ($threshold_{cluster}$)，將所有電荷密度小於 $threshold_{cluster}$ 的三角形標記起來。我們對這些被標記的三角形作 cluster，有相連的三角型為同一 cluster。

我們追蹤每個 cluster 的輪廓，若一個 cluster 有兩條輪廓表示其能將物體分割成兩個不同區域，但有時候一個 cluster 幾乎將物體分成兩個區塊，卻因為一些三角形的電荷密度高過 $threshold_{cluster}$，導致沒有辦法在一開始就將物體切割，所以我們必須對無法切割物體的 clusters 作測試，檢查一個 cluster 中的三角形數量是否超過 $threshold_{growing}$，$threshold_{growing}$ 為一個和物體三角形總數成正比的一

個常數，若此 cluster 中的三角形總數大於 $threshold_{growing}$，我們必須對此 cluster 作 cluster-growing。

Cluster-growing 的方法是一個 iterative 的過程，Figure 3.3，首先將一個 cluster 中的所有三角形放入 cuttingList 中，並找出和此 cluster 相鄰的所有三角形存入 growingList。我們利用 growingList，來檢查一個 cluster 是否達到 cluster-growing 的條件，將 growingList 中的三角形分堆，有相鄰的一堆，若只能分成一堆，則將 GrowingList 中的三角形加入 cuttingList 中，繼續 iterative 的過程，若能將 growingList 中的三角形分成兩堆，表示目前 growingList 中的三角形已經能將物體分成兩個區域，因此終止 iterative 的過程。

我們必須在 cuttingList 中，利用 cluster-shrinking 的方法找出確切分割的部分，追蹤 cuttingList 所形成的輪廓，並且由 cuttingList 的三角形中，和輪廓相鄰的三角形開始一一作 shrinking，我們作 shrinking 的依據是，若一個三角形的邊包含了兩條不同輪廓的邊，我們稱此三角形為 cutting triangle，cutting triangle 不能被 shrinking；否則我們將三角形從 cuttingList 中移除，並且更新輪廓，直到 cuttingList 中的所有三角形都是由 cutting triangles 構成。

結束 cluster-shrinking 後所形成的兩條輪廓我們使其平滑化，選擇總長度較小的輪廓當作我們的分割輪廓。

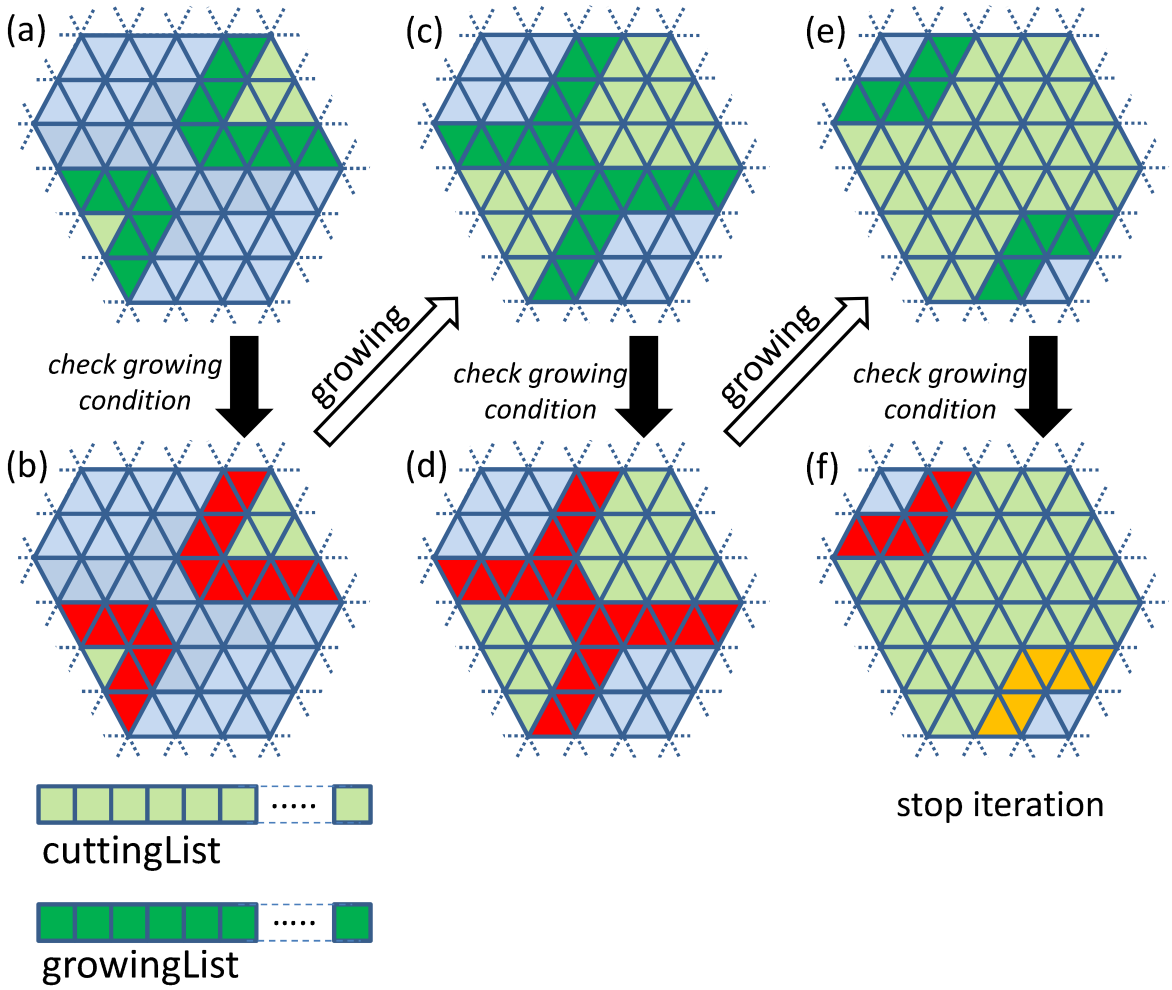


Figure 3.3: Cluster-growing 流程圖: 淺綠色的三角形代表 cuttingList 中的三角形，深綠色的三角形為 growingList 中的三角形，(a) 一 cluster 作 cluster-growing 前，將 cluster 外圍的三角形放入 growingList 中。(b) growingList 中的三角形只能分成紅色的一堆，達到 growing 的條件，繼續 growing，並將 growingList 中的三角形加入 cuttingList 中。(c)(d)(e)(f) 為 (a)(b) 兩個步驟作 iteration，(f) growingList 中的三角形被分成紅色和橘色兩堆，停止 growing。

Figure 3.4: Procedure step of object decomposition: (a) 一開始的 model。(b) 算出每個面的電荷密度，用 gray level 表示。(c) 利用直方圖均衡化增加對比度的結果，用 gray level 表示。(d) 將通過 $threshold_{cluster}$ 的三角形作分堆的結果。(e) 經過 cluster-growing 和 cluster-shrinking 後得到的 contour edges。(f) 物體分割的結果。

Chapter 4

Detector-Based Collision Detection

這個章節我們討論 detector-based 的碰撞偵測，在分割完物體之後，每個獨立的區域即為一個 detector-region，我們在每個 detector-region 中找一個適當的點作為 detector-point，每個 detector-point 管理各自的 detector-region。

4.1 Detector-Points

在物體分割後，我們必須在每個 detector-region 中找出適合的 Detector-point。在模擬過程中，detector-point 在 detector-region 中的位置會影響到 detector-based 碰撞偵測的效能，因此找出較佳的 detector-point 是一個重要的課題。Detector-point 的位置在模擬過程中主要會影響三角形的面向、追蹤的時間和 potentially colliding pair 的數量，進而影響到整個碰撞偵測的時間。在這裡我們所使用的 detector-point 為每個 detector-region 的重心，根據我們的實驗，它有較大的機率對 detector-region 中的所有三角形都是可視的。

4.2 Create BVHs

在 detector-based 的碰撞偵測中，我們對每個 detector-region 分別建立 BVH，它是一個二元樹的資料結構，而每個葉節點的 bounding volume 包著一個三角形。在這裡我們使用的 BVHs 是 *k*-DOPs，*k*-DOPs 是利用定義好的 *k*/2 個軸來決定形成 bounding volume 的凸面體 *k* 個面的法向量，我們使用由上而下的方式將一個

detector-region 建立 BVH。一開始根節點的 BV 會將整個 detector-region 包住，將 detector-region 中三角形的所有點投影到 *k*/2 個軸，算出根節點的 bounding volume 在 *k*/2 個軸的最大值和最小值。並在 *k*/2 個軸中選擇 bounding volume 的最長軸並算出此軸的中心值，將此 bounding volume 中所有三角形的重心投影到最長軸，接著建立左子節點和右子節點，所有重心投影值小於中心值的三角形，用左子節點的 bounding volume 將這些三角形包住；而右子節點的 bounding volume 則將其餘重心投影值大於等於中心值的所有三角形包住，每個節點以遞迴的方式執行，直到每個節點的 bounding volume 中只有一個三角形。

4.3 BVHs Update

每個影格開始時我們會根據我們讀入的動畫資料將每個點的位置做更新，因此我們必須對 BVHs 作更新，我們是利用後序的方式由下而上更新。每個葉節點的 bounding volume 必須包住一個三角形在此影格的三個點和此三角形在下個影格的三個點，每個內部節點 (internal node) 的 bounding volume 則必須包住其左子節點和右子節點的 bounding volume，直到更新完整個 BVHs。

4.4 Orientation Check

更新完每個 detector-region 的 BVH，我們會利用每一個 detector-region 的 detector-point (dp) 檢查 detector-region 中所有三角形的 orientation type [Che11]。以 figure 4.1 為例，三角形會在時間區間 $[0, \triangle t]$ 改變其位置，而三角形 $T(t)$ 由三個點 $p_0(t)$、$p_1(t)$、$p_2(t)$ 組成，$\vec{n}(t)$ 是三角形的法向量。藉由三角形任一點和 dp 形成的向量，與 $\vec{n}(t)$ 內積可以計算出在時間區間 $[0, \triangle t]$，每個三角形的 orientation type。下列方程式以 figure 4.1 為例：

$$\vec{n}(t) = (p_1(t) - p_0(t)) \times (p_2(t) - p_0(t)), \tag{4.1}$$

$$orientationValue(t) = (p_0(t) - dp) \cdot \vec{n}(t). \tag{4.2}$$

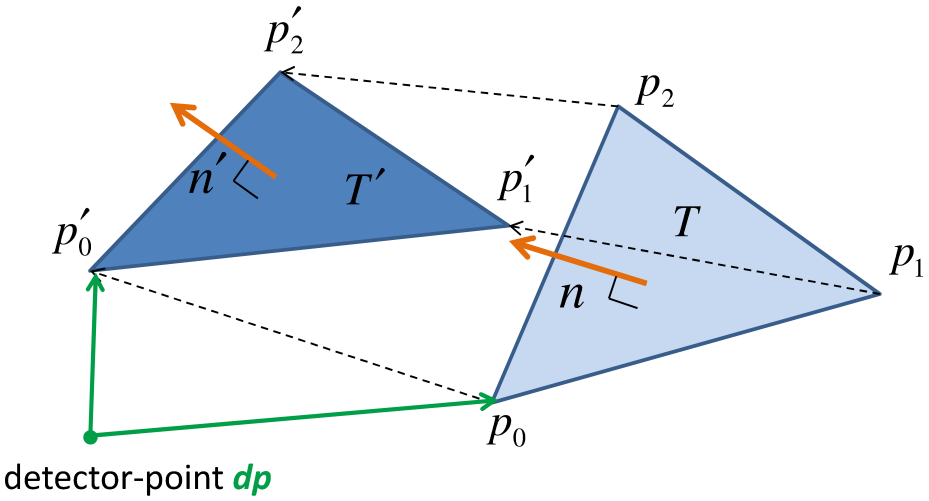


Figure 4.1: Orientation check。

檢查 $orientationValue(t)$ 的正負值，可以算出三角形的 orientation type。方法如下:

$$sign = sign\left(orientationValue\left(t\right)\right). \tag{4.3}$$

若在時間區間 $[0, \triangle t]$ sign > 0，則 $T\left(t\right) \in S^{+}$，若在時間區間 $[0, \triangle t]$ sign < 0，則 $T\left(t\right) \in S^{-}$，否則，$T\left(t\right) \in S^{0}$。

在一個時間區間中，我們將所有三角形 U 分類為 S^{+}、S^{-} 和 S^{0}，S^{+} 是三角形 orientation type 為 positive-oriented 所成的集合，S^{-} 是三角形 orientation type 為 negative-oriented 所成的集合，S^{0} 是三角形 orientation type 為 zero 所成的集合。

4.5 BVHs Traversal

如果同一個 detector-region 中所有的三角形都是屬於 S^{+} 或是 S^{-}，則該 detector-region 沒有自身碰撞。否則，根據 Jordan Curve Theorem，一個 positive-oriented 的三角形和一個 negative-oriented 的三角形對，可能會發生自身碰撞。因此，我們必須針對 (S^{+}, S^{-}) 形成的三角形對以及 (S^{0}, U) 形成的三角形對做追蹤，找出 potentially colliding pairs。

BVH 的每個節點都會有三個 flag，分別是 positive flag、negative flag 和 zero flag，flags 記錄該節點是否包含這三種類型的三角形。如同前面所說，我們將

藉由 BVH 追蹤找出 (S^+, S^-) 的三角形對以及 (S^0, U) 的三角形對，進一步找出 potentially colliding pairs。

基本的 BVH 追蹤由兩個 BVHs 的根節點開始檢查，檢查兩個節點的 bounding volume 是否重疊，沒有重疊就不用繼續檢查子節點，表示物體沒有自身碰撞；如果有重疊再針對兩個節點的四個子節點兩兩檢查，直到葉節點為止，每個葉節點都存在著一個三角形，因此只要兩個節點都檢查到葉節點就能找出 potentially colliding pairs。

我們的做法類似基本的 BVH 追蹤，只是為了找出 (S^+, S^-) 的三角形對，我們會先檢查兩個 BVHs 節點的 flags，如果一個為 positive 一個為 negative，則繼續檢查兩個節點的 bounding volume 是否重疊；不是則結束追蹤。此外如果一個節點的子節點都是 positive 或者都是 negative，就不用檢查此節點和另一個節點的 bounding volume 是否重疊，直接檢查子節點。

同樣的要找出 (S^0, U) 的三角形對，先檢查一個 BVH 節點的 flag 是否為 zero，不需要檢查另一個 BVH 節點的 flag 是哪種類型，若其中一個節點的 flag 是 zero，則檢查兩個節點的 bounding volume 是否重疊，否則結束追蹤。

4.6 Duplicate Features

在做完 BVHs 的追蹤後我們將搜集的 potentially colliding pairs 都存入 PCP Queue 中，PCP Queue 中的 potentially colliding pairs 都是在這個時間區間有可能發生碰撞的三角形對，我們必須再進一步去測試這些三角形對是不是真的會發生碰撞。

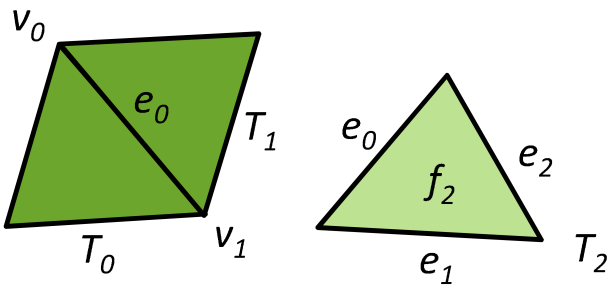


Figure 4.2: 三角形對的重複測試。

共享的 feature 可能會造成重複測試，如 Figure 4.2，舉例來說，假設我們的 PCP Queue 中有 (T_0, T_2)，(T_1, T_2) 這兩個 potentially colliding pairs，(v_0, f_2) 是 (T_0, T_2) 其中一對 feature pair 也是 (T_1, T_2) 的 feature pair，在做 elementary test 時，會重複對這對 features 作測試。相同的情況，(v_1, f_2) 和 (e_0, e_1)、(e_0, e_2)、(e_0, e_3) 也都是這兩對 potentially colliding pairs 所共同擁有的 feature pairs，這些多餘的測試都是不需要的。

我們利用 procedure representative triangles [TCYM09] 來解決 feature 重複測試的問題。在前置處理時，會先將每個點所相鄰的三角形都存入該點的 oneRingList 中，如 Figure 4.3。一組 potentially colliding pairs (T_1, T_2) 要做 vf-test，v 屬於 T_1，

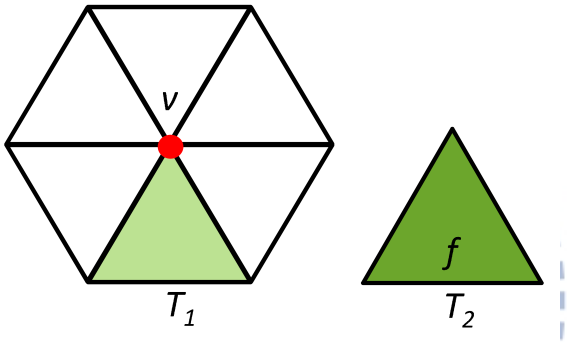


Figure 4.3: Procedure representative triangles: vf-test。

f 屬於 T_2，在 v 的 oneRingList 的三角形中，只有第一個三角形是有被 assigned v 這個 feature，所以我們要做 vf-test 前，必須先檢查 T_1 是不是 v 的 oneRingList 中的第一個，若不是就不需要作 vf-test。在這裡 v 的 oneRingList 的順序是固定的。ee-test 也是類似的情況，和一條邊 e 相鄰的兩個面，只有其中一個被 assigned e 這條邊，一組 potentially colliding pair (T_1, T_2) 要做 ee-test，e_1 屬於 T_1，e_2 屬於 T_2，若 T_1 沒被 assigned e_1 或 T_2 沒有被 assigned e_2，我們就不處理這組 feature pair。

Chapter 5

Implementation and Results

在這個章節我們將對實作的細節部分作敘述，首先對於我們要拿來比較的兩個方法：ICCD 以及利用 *K*-means 分割物體作 detector-based 碰撞偵測的方法敘述實作細節，接下來討論一些其他實作上參數的細節部分，最後展示出我們作的實驗以及實驗的結果。

我們使用 Microsoft Visual Studio 2008 C++ 撰寫我們的系統，執行環境為 Intel(R) Core (TM)2 Quad CPU Q9400 @ 2.66GHz，4GB 的 RAM，32-bit Windows 7 的作業系統。

5.1 ICCD

在這個章節，我們敘述我們是如何實作 ICCD [TCYM09]。總括 ICCD 的流程，在前置處理的過程，先將相鄰三角形對需要作 elementary test 的 feature pairs 存入 orphan set，並對物體建立 BVH。在執行時間，利用 continuous normal cone 將物體表面分成好幾個 regular patches。在追蹤的過程，每個 regular patch 經過 continuous contour test 來決定一個 regular patch 是否發生自身碰撞，若發生碰撞則對此 regular patch 的兩個 sub-patches 繼續作 continuous contour test，遞迴這樣的過程直到找出 potentially colliding pairs；若一個 regular patch 通過 continuous contour test，表示這個 regular patch 沒有自身碰撞發生，結束追蹤。並利用 procedure representative triangles 處理不相鄰的 potentially colliding pairs，orphan set 則處理

相鄰的 potentially colliding pairs。Tang 等人在這篇論文的主要貢獻為: continuous normal cone、continuous contour test 和 orphant set，在我們的系統，相鄰的三角形被視為不會發生碰撞，所以在這篇論文不對 orphan set 作討論，我們主要對 continuous normal cone 和 continuous contour test 敘述實作方法的細節。

5.1.1 Continuous Normal Cone

Normal cone 的觀念是: 給一個連續的表面 S，N 為 S 上任一點的法向量，若能找到一個向量 V，使得 $(N \cdot V) > 0$，我們就將 S，看成一個 regular patch，並且存在一個 normal cone，而向量 V 則為這個 normal cone 的軸。Continuous normal cone 延伸了這個方法，在一個時間區間 $[0, \triangle t]$ 中，我們利用由下而上的方式對一個物體的 BVH 節點檢查該表面是不是有 continuous normal cone 存在，在葉節點的部分，一個三角形的法向量在 $t = 0$ 時為 n_0、在 $t = \triangle t$ 時的法向量為 $n_{\triangle t}$，在這個時間區間中，此三角形的法向量不會超出 n_0 和 $n_{\triangle t}$ 的範圍，我們將 $\frac{n_0+n_{\triangle t}}{2}$ 定義為葉節點 continuous normal cone 的軸，而將 n_0 和 $n_{\triangle t}$ 夾角的一半視為此 continuous normal cone 的夾角 α。而在每個內部節點，我們則利用兩個子節點的 continuous normal cones 來更新此節點的 continuous normal cone。

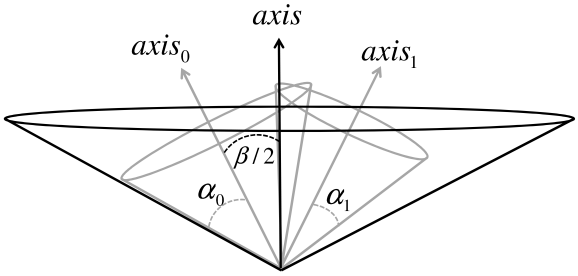


Figure 5.1: 利用兩個子節點的 continuous normal cones，算出一個節點的 continuous normal cone 之示意圖，灰色是兩個子節點的 continuous normal cones，黑色是此節點的 continuous normal cone。

見圖 Figure 5.1，兩個子節點的 continuous normal cones 軸分別為 $axis_0$、$axis_1$，夾角 α 分別為 α_0、α_1，我們將此節點 continuous normal cone 的軸定義為:

$$axis = normalize(axis_0 + axis_1), \tag{5.1}$$

而此 continuous normal cone 的 α 為:

$$\alpha = \beta/2 + max(\alpha_0, \alpha_1), \tag{5.2}$$

β 為 $axis_0$ 和 $axis_1$ 的夾角 [Pro97]。

我們會在每個節點定義一個布林值，作為一個節點是否有 cone 存在的指標，若一個節點的 α 值小於 π，我們就視此節點有 cone 存在，追蹤時若一個節點有 cone 存在，該節點所有的三角形則被視為一個 regular patch。

5.1.2 Continuous Contour Test

我們在前置處理時會先將每個節點的表面輪廓用 linklist 存在節點中，在追蹤時，若一個節點有 cone 存在，則進一步做 continuous contour test。Continuous contour test 首先要先將此節點的輪廓投影到一平面，此平面的法向量為 cone 的軸 V，並且每對投影的邊都要做邊和邊的碰撞測試 *planar-EE* Test，若兩條邊在一個時間區間要發生碰撞，那麼一定會在一個時間點發生一條邊的其中一點和另一條邊共線的情況，因此在 *planar-EE* Test 中，我們可以將其簡化成四次點和邊的碰撞測試 *planar-VE* Test。若四次的 *planar-VE* Test 都沒有發生碰撞，表示作 *planar-EE* Test 測試的兩條邊沒有發生碰撞，若每對邊都沒有發生碰撞，則此節點的整個表面不會發生自身碰撞。

我們也根據 Tang 等人在這篇論文中提到的加速方法對我們實作的 ICCD 作加速，我們將存輪廓邊的 linklist 分成兩段，前段為和父節點共享的邊，後段則為沒有和父節點節點共享的邊，在作追蹤時，若一個節點的父節點已經做過了 continuous contour test，則此節點只需要對沒有和父節點共享的輪廓邊作投影和 *planar-EE* Test。在 *planar-EE* Test 部分，也對每條投影的邊做 bounding volume，若兩條輪廓邊的 bounding volume 有重疊，我們才將此兩條邊作 *planar-EE* Test。

5.2 *K*-means Decomposition

我們測試的物體都是由許多四面體所組成，我們利用所有 nodes 隨機找出 K 個 nodes 當作我們的 initial means，而其他 nodes 從這 K 個 means 中找出最近的

mean 作為 dominate mean，形成 K 個 clusters，並且在這 K 個 clusters 中各自重新找一個最接近重心的 node 作為新的 mean，經過 iterative 的過程，替換這 K 個點，直到每次找到的 K 個 means 是收斂的就結束 iterative 的過程，我們令這 K 個點為我們的 detector-points。此物體的每個面分別對這 K 個 detector-points 計算彼此之間的最小 edge 數作為其距離，並且選擇一個距離自己最近的 detector-point。而一個 detector-point 搜集到的那些面，就是這個 detector-point 的 detector-region。

5.3 Others

BVHs: 我們使用 k-DOPs 作為我們的 BVHs，相較於 AABB，它在 culling 的表現上效果更明顯，而我們使用的 k-DOPs 為 14-DOPs，根據我們的實驗，它在更新和 culling 之間取得了最好的平衡。

Parameters: 在章節 3.3 提到，我們利用 $threshold_{cluster}$ 來決定一些三角形是否會成為 cluster 的門檻，在這裡我們將 $threshold_{cluster}$ 設為 0.1，相當於一個物體的電荷分佈經過直方圖均衡化後，我們取前 10% 電荷密度最小的三角形。

$$threshold_{growing} = TotalTriangles/400. \tag{5.3}$$

Orientation Check: 我們的 detector-point 在時間區間 $[0, \triangle t]$ 並沒有隨著時間的改變而更改位置。

5.4 Results

為了測試我們的方法，我們做了以下三個不同的實驗:

Pawn: 這個實驗是一顆棋子從高處落下產生形變而自身碰撞的過程，此物體包含 3454 個點和 5130 個面。Figure 5.5 上。

Dumbbell: 此實驗是一個類似彈簧的物體從階梯上一層一層掉下來的過程，此物體包含 1249 個點和 1710 個面。Figure 5.5 中。

Luigi: 這個實驗為人物從天而降並且和球發生互動的過程，此物體包含 2433 個點和 4014 個面。Figure 5.5 下。

以上這些實驗都包含了許多影格，我們對這些影格都執行了連續性的碰撞偵測。

method	BVHUP	ORIENTED	CNC & CT	TRA	Contact	#PCPs	TTCD
ours	3.0	0.9	X	1.1	1.8	2281	8.3
$K=1$	3.0	0.8	X	1.5	1.5	1925	8.4
$K=2$	3.0	0.8	X	1.0	1.3	1580	7.6
$K=3$	3.0	0.8	X	1.3	1.4	1725	8.1
$K=4$	3.0	0.8	X	1.6	1.6	1892	8.6
$K=5$	3.0	0.8	X	1.6	1.7	1956	8.6
ICCD	3.1	X	7.3	3.2	2.7	3089	17.1

Figure 5.2: Pawn 的實驗結果，單位為毫秒。

method	BVHUP	ORIENTED	CNC & CT	TRA	Contact	#PCPs	TTCD
ours	1.0	0.3	X	0.2	0.3	339	2.3
$K=1$	1.1	0.3	X	0.1	0.1	132	2.3
$K=2$	1.0	0.3	X	0.2	0.3	296	2.4
$K=3$	1.1	0.3	X	0.4	0.4	374	2.8
$K=4$	1.1	0.3	X	0.5	0.5	427	2.9
ICCD	0.9	X	2.3	0.5	0.5	473	4.5

Figure 5.3: Dumbbell 的實驗結果，單位為毫秒。

method	BVHUP	ORIENTED	CNC & CT	TRA	Contact	#PCPs	TTCD
ours	2.3	0.6	X	0.9	0.7	766	5.9
$K=3$	2.3	0.6	X	1.0	0.8	800	6.0
$K=4$	2.3	0.6	X	1.1	0.8	806	6.1
$K=5$	2.4	0.6	X	1.1	0.8	824	6.2
$K=6$	2.4	0.6	X	1.1	0.8	810	6.3
$K=7$	2.4	0.6	X	1.1	0.9	845	6.3
ICCD	2.4	X	6.5	1.9	1.5	1432	12.9

Figure 5.4: Luigi 的實驗結果，單位為毫秒。

Figure 5.2～ 5.4 中:

- BVHUP 是 BVH 更新的平均時間。
- ORIENTED 是作 orientation check 的平均時間。
- CNC & CT 是作 Continuous normal cone 和 contour test 的平均時間。
- TRA 是追蹤的平均時間。
- CT 是 elementary test 的平均時間。

• TTCD 是一個影格作完整碰撞偵測所花的平均時間。

我們的方法和 *K*-means 的方法中沒有作 continuous normal cone 和 contour test，而 ICCD 中沒有 orientation check，在表格中以 X 表示。

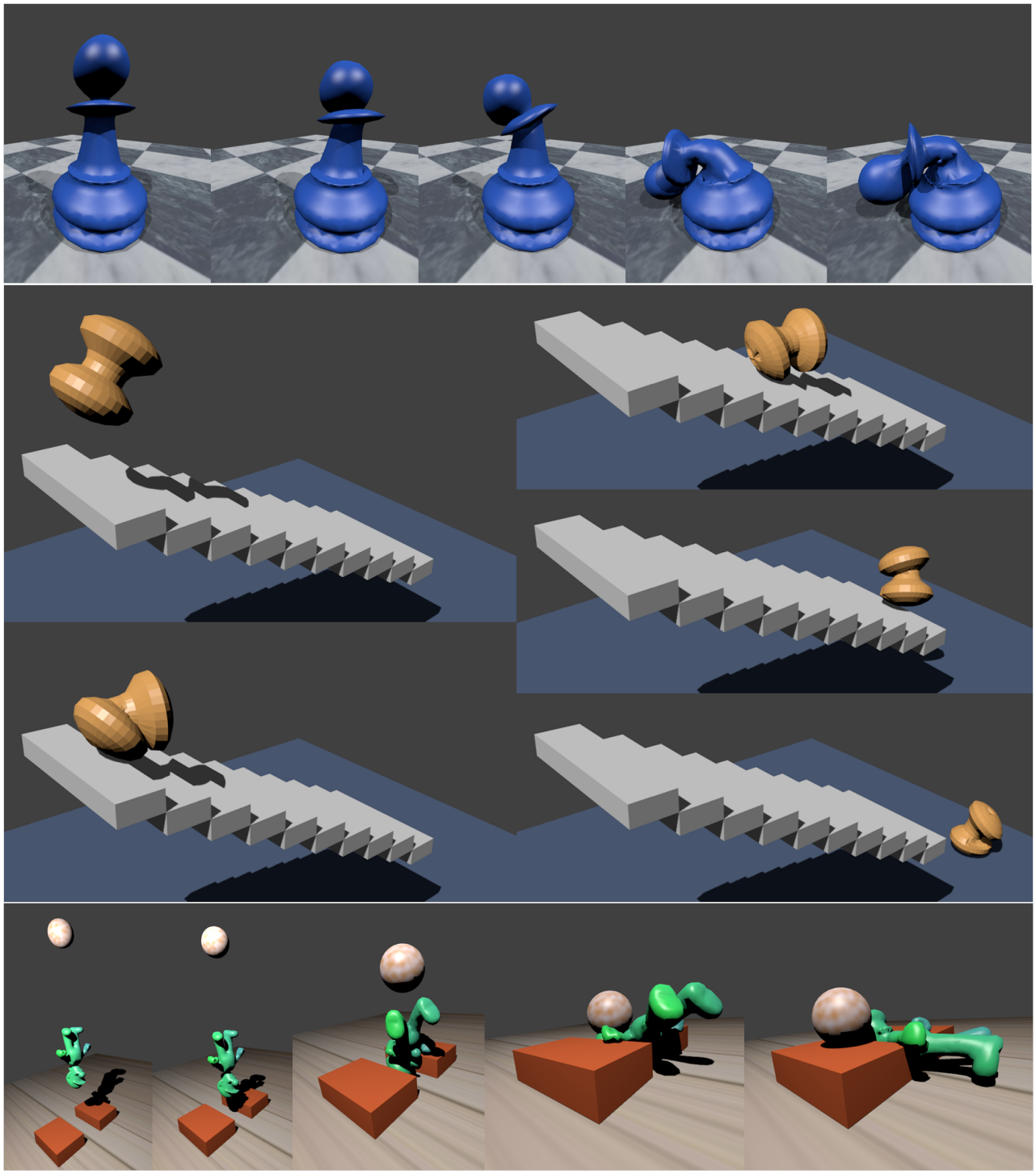


Figure 5.5: 三個實驗的擷取圖。上圖為實驗 Pawn，中圖為 Dumbbell，下圖為 Luigi。

Chapter 6

Analysis and Limitations

在這個章節，我們分析了我們的方法和其他方法在實驗上各方面的表現以及一些我們方法上的限制。

6.1 Analysis

由於 *K*-means 一開始所選的點是隨機的，雖然經過幾次 iterative 的過程，最後挑出的 *K* 個點可能在每次實驗的結果都會不一樣，所以我們在這項實驗數據中，是將五次的實驗結果作平均，若我們的方法能將物體分割成 n 個 detector-regions，我們就分別將 *K* 的值設為 n±1、n±2 和 n 作實驗。以下我們將我們的方法分別跟 *K*-means 和 ICCD 作分析：

***K*-means:** 從實驗數據中，Figure 5.3 ～ Figure 5.4，我們可以觀察出，除了 pawn 的實驗中，*K*-means 的方法在有些 *K* 值的整體平均時間稍微比我們的好以外，其餘的實驗都是我們的方法效能稍微優異。但由於 *K*-means 每次執行的結果不是很穩定，在每個實驗中，對不同的 *K* 值，我們分別都作了五次的實驗。在 Figure 6.4 中，我們將五次實驗中整體平均時間表現的最好和最差的情況列出來，我們發現一些最好的情況會比我們的方法在效能上較好，是因為 *K*-means 的方法中也是有一定的機率將物體作較好的分割，但是將 *K*-means 中較差的情況一起平均來看，我們的方法比 *K*-means 整體的效能還好。

在物體切割的方面，Figure 6.1 ～ Figure 6.3 中最左邊的圖為我們的方法切割

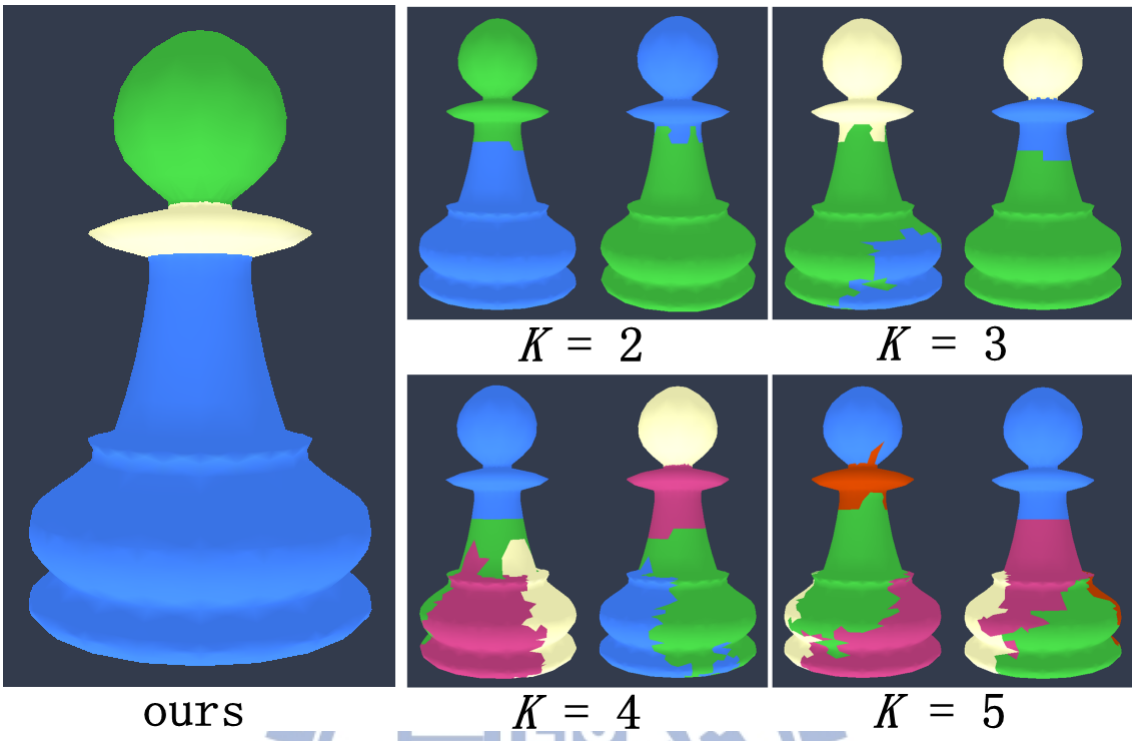


Figure 6.1: Pawn 的分割結果，左邊為利用我們的方法分割的結果，右邊為 *K*-means，不同 *K* 值可能分割出的結果。

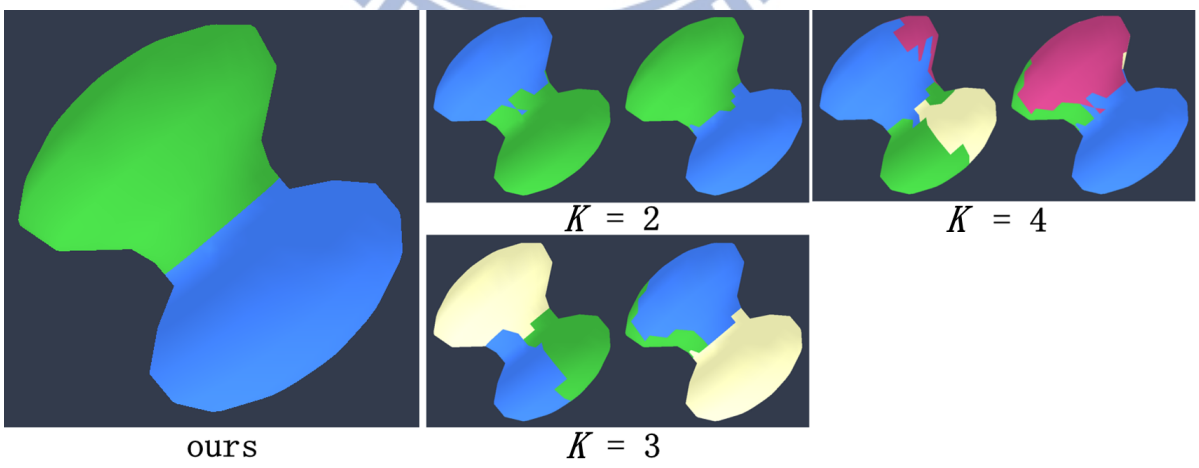


Figure 6.2: Dumbbell 的分割結果，左邊為利用我們的方法分割的結果，右邊為 *K*-means，不同 *K* 值可能分割出的結果。

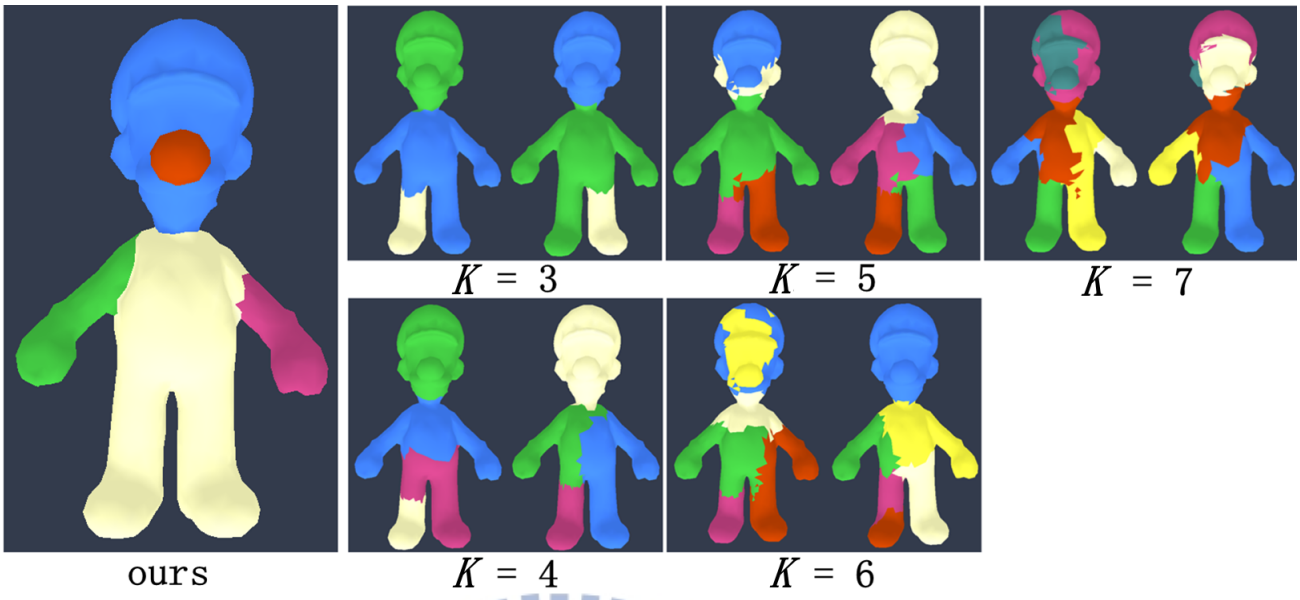


Figure 6.3: Luigi 的分割結果，左邊為利用我們的方法分割的結果，右邊為 *K*-means，不同 *K* 值可能分割出的結果。

的結果，其餘的為 *K*-means 的方法在不同 *K* 值，我們分別隨機選兩個可能的分割結果。相較於 *K*-means 的方法，我們的方法每次對物體切割的結果都相同，所以整體的效能穩定性也較高。我們也可以發現 *K*-means 在分割上有可能會產生部分 detector-region 被另一個 detector-region 包住的情形，這種情況會增加追蹤時，不同的 detector-region 之間的 potentially colliding pairs，降低效能。*K*-means 在每個實驗所使用的最佳 *K* 值不盡相同，用我們的方法可以解決尋找最佳 *K* 值的問題。

K	Pawn	Dumbbell	Luigi
K = 1	8.3~8.4	2.1~2.4	X
K = 2	7.5~7.8	2.2~2.5	X
K = 3	8.0~8.1	2.5~3.1	5.8~6.0
K = 4	8.5~8.7	2.7~3.3	6.1~6.2
K = 5	8.4~9.1	X	5.8~6.5
K = 6	X	X	5.9~6.6
K = 7	X	X	6.2~6.5

Figure 6.4: *K*-means TTCD: 根據我們對於 *K*-means 的實驗，TTCD 介於的區間。

ICCD: 我們的方法和 ICCD 主要差別在於我們的方法使用 orientation check 來對三角形的面向作測試，將三角形根據面向作分堆；而 ICCD 則是利用 continuous normal cone 來將三角形分成幾個 regular patches，並利用 continuous contour test 對

每個 regular patch 的輪廓作測試。在分堆的技巧上，假設利用我們的方法和 ICCD 的方法分別對一個球體作分堆，以我們的方法，我們能將 detector-point 找在球體的內部，因為球體的所有面的面向對 detector-point 都是一致的，所以整個球體的面會被我們分為同一堆，面與面之間不會發生碰撞；ICCD 則會根據 BVH 的結構，利用後序的方式檢查每個節點所包含的 normal cone 夾角的範圍 α，當一個節點的 α 值大於等於 π 時，該節點的每個子節點所包含的三角形即為一個 regular patch。所以一個球體以 ICCD 的方式作 regular patches，最好的情況下，最少要將一個球體分成八塊 regular patches。而 contour test 還需要個別將 regular patches 的所有輪廓的邊作投影，並且檢查那些投影的邊在一個時間區間內有沒有發生重疊。根據我們的實驗結果，contour test 在時間上的花費相當可觀。分堆的數量大小也影響了我們的方法和 ICCD 在實驗上的表現，在我們執行堆與堆之間的追蹤時，在兩堆交界附近的三角形有很大的機會成為 potentially colliding pairs，所以在追蹤和計算 contact time 上，我們的表現也比較好。整體來看，我們的方法比我們自己實作 ICCD 的方法效能大約提升了 1.88X ～ 2.19X。

6.2 Limitations

我們的研究有一些限制。我們的方法必須算出每個面的電荷密度，見 Eq. (3.5)，A 是一個稠密矩陣，無法利用一些特別的資料結構來節省此矩陣的空間。A 的空間複雜度為 $O(N^2)$，假設物體有 N 個面，我們就必須利用 $(N+1)^2$ 的空間來儲存 A 這個矩陣。我們使用 ALGLIB 3.4.0 函式庫 [ALG11] 來解線性系統，ALGLIB 3.4.0 中二維陣列必須藉由傳入一維陣列來設定每個元素的值，而我們 C++ 實作的環境中一維陣列的大小不能超過 16383，因此我們能處理面的最大數目為 16383。

在物體分割方面，物體必須要有明顯凹陷的一整塊區域，我們才能將此區塊視為要做分割的部分。在這方面雖然我們有使用 cluster-growing 方式來加強分割的效果，但還是會有不能分割的情況發生。例如 Figure 3.4(c)，Luigi 大腿的內側是一個非常明顯的凹面，不過因為大腿外側是非常平滑的，因此就算利用 cluster-growing 的處理也無法適當的將腿部分割出來。

Detector-based 的碰撞偵測只適合在物體本身沒有穿透的情況，原因是我們不會對 orientation type 相同的三角形作追蹤，若在一個 cluster-region 中，有兩個相同 orientation type 的三角形彼此穿透，我們在追蹤時沒有辦法將它們找出來，因此在三角形有穿透的情況下，我們無法保證能找出所有發生碰撞的三角形。

Chapter 7

Conclusion and Future Work

7.1 Conclusion

在這篇論文中，對於物體的自身碰撞偵測，我們提出了利用電荷分佈的物理特性將物體分割，並且利用 detector-based 來作物體的碰撞偵測，最後將我們的結果分別和 *K*-means 以及我們實作的 ICCD 來作比較。實驗的結果顯示了在大部分的狀況下我們的方法比 *K*-means 的方法效能還好，整體來看也穩定許多，並且加速了物體的連續自身碰撞偵測。

7.2 Future Work

我們的方法還有很多的發展空間，以下為我們未來的研究方向。我們希望能針對章節 6.2中提到的幾項限制作改善：

我們可以將擁有大量三角形的物體作簡化，並且運用 feature 對應，將原始物體和簡化後物體作對應，使我們計算出電荷分佈的結果對應到大量三角形的原始物體上，並利用原始物體對應的電荷分佈來作物體分割和 detector-based 的碰撞偵測。

我們希望更加深入的考慮物體的物理特性，例如由四面體組成的 mass-spring 物體，(1) 我們可以藉由考慮所有彈簧的分佈或彈性係數來改變物體的電荷分佈。(2) 或是將物體在模擬的過程中，分析其形變的過程，將容易形變的部分記錄下

來，作為我們分割的依據。上述兩個方法使系統有機會能對一些在物體表面上平滑卻容易產生形變的區域作更好的分割。

對於在模擬過程中物體的穿透，我們想針對那些沒有偵測到碰撞的三角形對，在它們在一開始發生碰撞時就將它們記錄下來，並持續對它們作追蹤。但是必須要想一個機制結束這些額外追蹤的三角形對，否則當需要額外追蹤的三角形對越來越多，系統的負荷就越大。

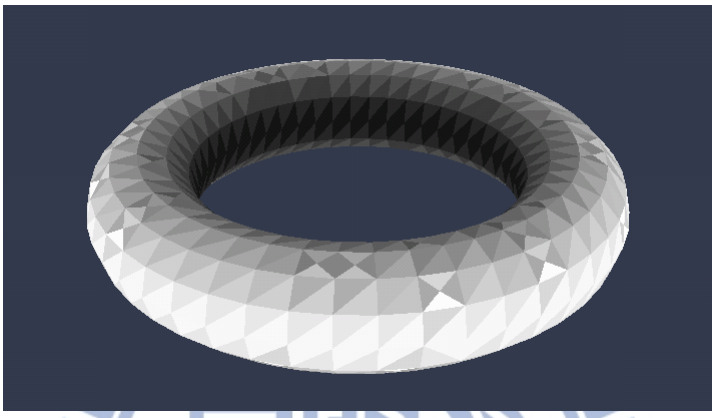


Figure 7.1: Torus 算出電荷分佈的結果，電荷主要分佈在 Torus 外圈。

另外，torus 類型的物體，電荷會分佈在圓環的外圈，如圖 Figure 7.1。由於此類型的物體在結構上也沒有明顯能分割的區域，就算利用 cluster-growing 的方式也不能將其作適當的分割，我們必須找出其他分割此類物體的方法。

最後，現今的很多演算法都使用了多核心 CPUs 和 GPUs 作加速，因此我們也希望利用結合 CPUs 和 GPUs 平行處理的方式來提高我們的系統效能。

Bibliography

[ALG11] Alglib 3.4.0. 2011.

[BFA02] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 594--603, 2002.

[BMKM05] B. Ben-Moshe, M.J. Katz, and J.S.B. Mitchell. A constant-factor approximation algorithm for optimal terrain guarding. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 515--524. Society for Industrial and Applied Mathematics, 2005.

[Che11] Y.C. Cheng. View-based continuous self-collision detection with graphics hardware. *Mater Thesis, National Chiao Tung University, Tiawan, R.O.C*, 2011.

[CTM08] S. Curtis, R. Tamstorf, and D. Manocha. Fast collision detection for deformable models using representative-triangles. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 61--69. ACM, 2008.

[GF08] Aleksey Golovinskiy and Thomas Funkhouser. Randomized cuts for 3D mesh analysis. *ACM Transactions on Graphics (Proc. SIGGRAPH ASIA)*, 27(5), December 2008.

[GLM96] S. Gottschalk, M.C. Lin, and D. Manocha. Obbtree: A hierarchical structure for rapid interference detection. In *Proceedings of the 23rd annual con-*

ference on Computer graphics and interactive techniques, pages 171--180. ACM, 1996.

[GS01] E. Grinspun and P. Schroder. Normal bounds for subdivision-surface interference detection. In *Proceedings in Visualization*, pages 333--570, 2001.

[GWH01] M. Garland, A. Willmott, and P.S. Heckbert. Hierarchical face clustering on polygonal surfaces. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 49--58. ACM, 2001.

[KHH+09] D. Kim, J.P. Heo, J. Huh, J. Kim, and S. Yoon. Hpccd: Hybrid parallel continuous collision detection using cpus and gpus. In *Computer Graphics Forum*, volume 28, pages 1791--1800, 2009.

[KHM+98] J.T. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):21--36, 1998.

[KT03] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 954--961, 2003.

[L+81] S.B. Laughlin et al. A simple coding procedure enhances a neuron's information capacity. *Z. Naturforsch*, 36(910-912):51, 1981.

[LA07] J.M. Lien and N.M. Amato. Approximate convex decomposition of polyhedra. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 121--131, 2007.

[Lie07] Jyh-Ming Lien. Approximate star-shaped decomposition of point set data. In *SPBG'07*, pages 73--80, 2007.

[LLL10] H. Liu, W. Liu, and L.J. Latecki. Convex shape decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010*, pages 97--104, 2010.

[PG95] I.J. Palmer and R.L. Grimsdale. Collision detection for animation using sphere-trees. In *Computer Graphics Forum*, volume 14, pages 105--116, 1995.

[Pro97] X. Provot. Collision and self-collision handling in cloth model dedicated to design garments. In *Graphics interface*, 1997.

[SPO10] S.C. Schvartzman, Á.G. Pérez, and M.A. Otaduy. Star-contours for efficient hierarchical self-collision detection. *ACM Transactions on Graphics (TOG)*, 29(4):80, 2010.

[STK02] S. Shlafman, A. Tal, and S. Katz. Metamorphosis of polyhedral surfaces using decomposition. In *Computer Graphics Forum*, volume 21, pages 219--228, 2002.

[TCYM09] M. Tang, S. Curtis, S.E. Yoon, and D. Manocha. Iccd: Interactive continuous collision detection between deformable models using connectivity-based culling. *IEEE Transactions on Visualization and Computer Graphics*, 15(4):544--557, 2009.

[TMT10a] M. Tang, D. Manocha, and R. Tong. Fast continuous collision detection using deforming non-penetration filters. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, pages 7--13, 2010.

[TMT10b] M. Tang, D. Manocha, and R. Tong. Mccd: Multi-core collision detection between deformable models using front-based decomposition. *Graphical Models*, 72(2):7--23, 2010.

[VDBVB98] G. Van Den Bergen, G. Van, and D. Bergen. Efficient collision detection of complex deformable models using aabb trees. In *J. Graphics Tools*, 1998.

[VT94] P. Volino and N.M. Thalmann. Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. In *Computer Graphics Forum*, volume 13, pages 155--166, 1994.

[WB05] W.S.K. Wong and G. Baciu. Dynamic interaction between deformable surfaces and nonsmooth objects. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):329--340, 2005.

[WB06] W.S.K. Wong and G. Baciu. A randomized marking scheme for continuous collision detection in simulation of deformable surfaces. In *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pages 181--188, 2006.

[WL97] K. Wu and M.D. Levine. 3d part segmentation using simulated electrical charge distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1223--1235, 1997.

[WRG+84] D. Wilton, S. Rao, AW Glisson, D. Schaubert, O. Al-Bundak, and C. Butler. Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains. *IEEE Transactions on Antennas and Propagation*, 32(3):276--281, 1984.