

Computer Graphics Final Project

Sketch System for Animation -- Teddy System

B05902005 蕭乙蓁

B05902038 林詩芩

B05902118 陳盈如

Repository link: <https://github.com/s123unny/sketch3D>

1. Introduction

實作教授上課時所提及的Teddy System，並加上一些自身想法。利用Python和WebGL將2D線條轉換成3D模型，以及新增shading、動畫等其他效果，亦運用使用者介面給予使用者更多繪製3D模型的功能。

2. Challenge

- System design

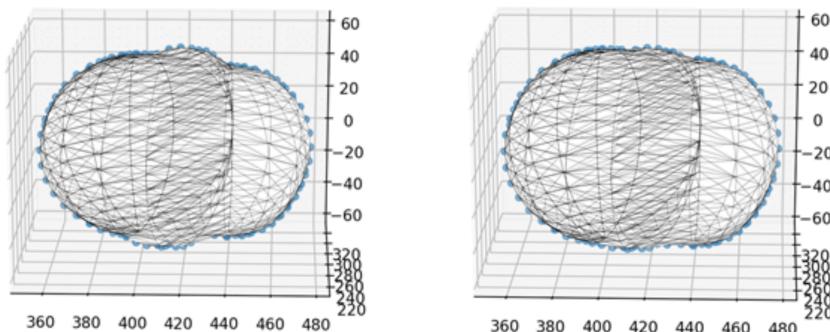
我們打算利用作業一的webgl來呈現我們運算出來的模型，但同時還要支援另一個畫布給使用者畫畫。所以我們在頁面的顯示上採用overlay的屬性，使提供畫畫的畫布覆蓋在webgl的畫布上。讓使用者能夠畫完平面的圖形後，在同一個視窗中就顯示出立體的模型。

- Triangulation

起初我們打算利用Delaunay triangulation後，從中找到分佈較不均勻的部分，直接插入點在多邊形中，最後用力學彈簧的概念，拉開成立體模型。但這會使得我們的模型z軸非常不平滑，因為受力點不均勻的關係。所以我們研究之後，決定將內部的點連接形成骨架，以解決拉成立體時高度呈非連貫的變化。

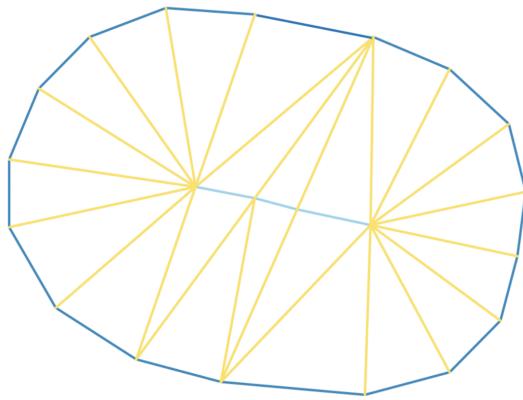
- Elevate Spine

一開始，我們打算按照論文的算法，依該骨架到外圈的距離作為其z軸座標，然而成果卻像是(圖一之左圖)，非常地凹凸不平，中間的骨架特別突出。後來，經過一番嘗試，我們使用將骨架的高度與前後取加權平均的方法，順利地讓表面趨於平整(圖一之右圖)。



圖一

我們推斷其原因為：因多邊形切割時，做了扇形特殊處理(Algorithm之多邊形切割會詳述)，所以只要有和扇形相連的骨架，與外部連接的短邊會變多，單純取平均就會讓短邊的權重提高太多，使其與其他骨架的高度相差甚遠。



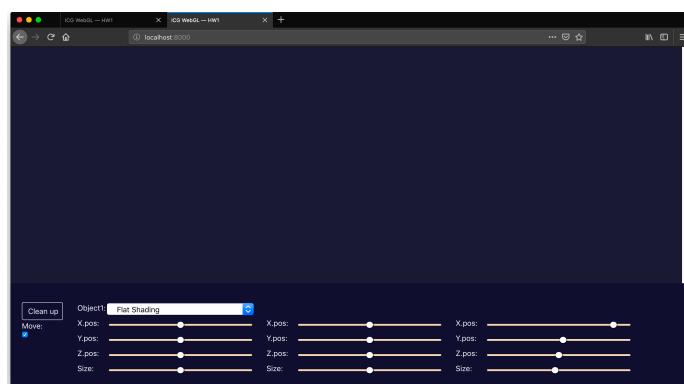
圖二. 骨架四點之原始高度為48, 56, 57, 52，經轉換，高度變為51, 55, 56, 54，分布較平均

3. System Implementation

我們的系統分為兩部分，一個是使用者操作畫圖的介面，另一個是轉換成立體模型後呈現的介面。藉由python的HTTPserver，我們建立一個簡易的網站當作操作介面及資料傳輸介面。當使用者畫完一個圖形後，將圖形的點座標傳送給後端server，server運算出立體模型後，再傳給前端用webgl的方式加上光線及顏色。

4. User Interface

從後端讀取座標點的資料，再利用WebGL呈現出3D模型，給予三個光源做出四種shading樣式。介面上增設下拉式選單，讓使用者能夠自行操控想要呈現何種shading；增設數個滑桿，分別可以控制所有3D模型的位置、大小、xyz方向的旋轉，讓使用者可以能夠自行選擇觀看角度及模型精細程度；增設兩個按鈕，分別是清除畫布鍵以及動畫開始鍵，避免畫布上過多線條影響使用者體驗，啟動動畫後，使用者畫得3D模型便會依照程式進行移動。



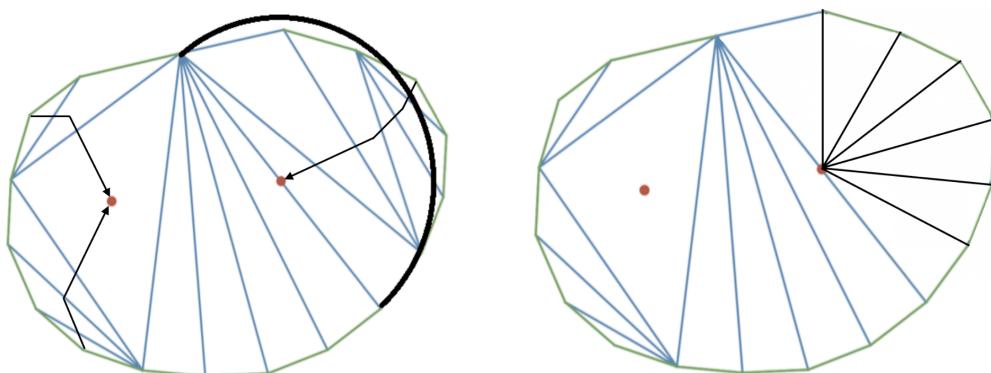
圖三

5. Algorithm

- 多邊形切割

首先，我們會從使用者的輸入拿到由一組點構成的凸多邊形，我們要進行切割、產生骨架，以利後續轉成立體模型。我們的目標是生成一組骨架，使所有邊上的點連接到骨架，且中間不交叉。

第一步是運算Delaunay triangulation。接著，我們將這些三角形分成三種類型：1.ear: 有兩個外圍邊的三角形 2.inner: 沒有任何外圍邊的三角形 3.other: 其他。我們接著對所有的ear三角形作下列的動作。先將ear三角形的內部邊當作直徑，檢查剩下的點是否在半圓內，若是在其中，我們就移動到ear三角形的共邊三角形，將新的內部邊定義為新的直徑，檢查剛才的所有點是否都在半圓中。我們會一直重複這個步驟直到有任何的點不在半圓內或是碰到inner三角形，如圖四。

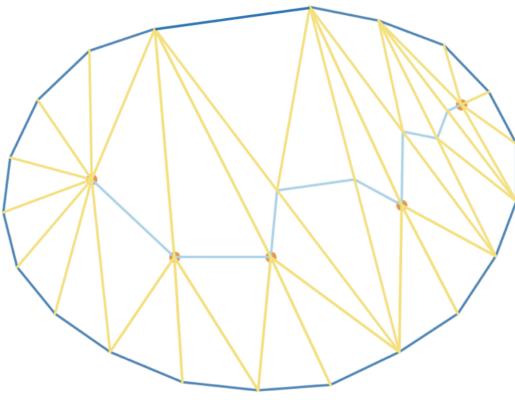


圖四. 左邊是碰到inner三角形，右邊是有點位於半圓外

圖五. 將外圍的點與圓心連接

若碰到第一種終止條件，也就是有點不在半圓中，則將當下為直徑的圓心點加入為之後生成骨架的起點。若是第二種終止條件，碰到三邊在內部的三角形，那就將此三角形的重心設為骨架起點。另外，剛才一路檢查過的三角形的頂點們則與此新增加的點連接，形成扇形，如圖五。當完成上述的步驟後，我們就獲得骨架中各個分支的起點。

接下來，我們要將這些點連起來做成我們的骨架。連接的方式與上面類似，一樣是找共邊的三角形，但每當找到新的共邊三角形，將此三角形的另一個內部邊的中點加入骨架，並將三角形的點連交到此點，重複一樣動作直到碰到另一個骨架起點。如此一來，我們的骨架就完成了。

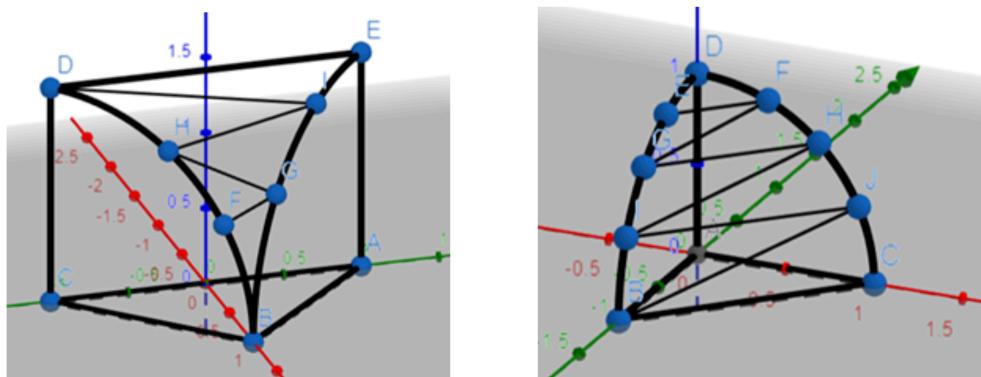


圖六. 骨架完成圖，淺藍色線條為骨架，黃色線條為外圍點到骨架的連線，橘色點是骨架分支的起點。

- 建立3D模型

- (1) 計算每個骨架點，與其連到的外部點的平均距離，記為它的提高值。算好全部點的提高值後，再讓每個骨架點的提高值，與和其前後相連骨架點的提高值做加權平均，即為各骨架點的z軸座標。
- (2) 將每個提高的骨架點，和原本與其相連的外部點，用一條1/4橢圓的邊連起來。
- (3) 接著是較複雜的一步，要將相鄰的橢圓邊，用三角形縫合。

首先，有兩種三角形縫合的情況，分別為圖七之左圖與右圖。

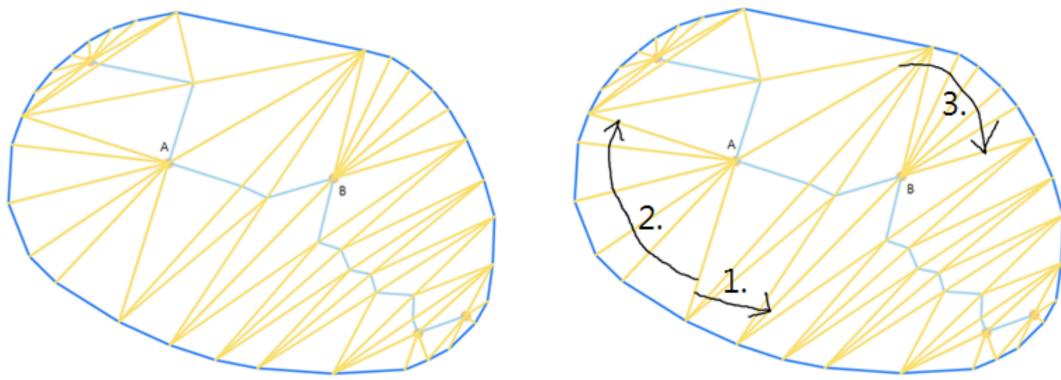


圖七

左圖的情況，設我們現在確認了B點與C點相連，我們就要檢查C點的下一點(也就是A點)，是否與B點相連。是的話，就要用三角形縫合BDE。右圖的情況，設我們現在確認了B點與A點相連，我們就要檢查B點的下一點(也就是C點)，是否與A點相連。是的話，就要用三角形縫合BCD。

一般狀況下，我們可以從每一條骨架的第一個點，任意取一個與他相鄰的點做為起始點，分別依順逆時針繞行一次，即可縫合該條骨架範圍的所有三角形。然而需要特別注意的是，當骨架的兩端都是內部時(圖八之AB折線)，選取任意起點繞完兩個方向

後，仍會有另外一邊還沒進行縫合，所以要再進行一次搜索，選取還沒繞過的點，再進行一次雙方向繞行縫合。

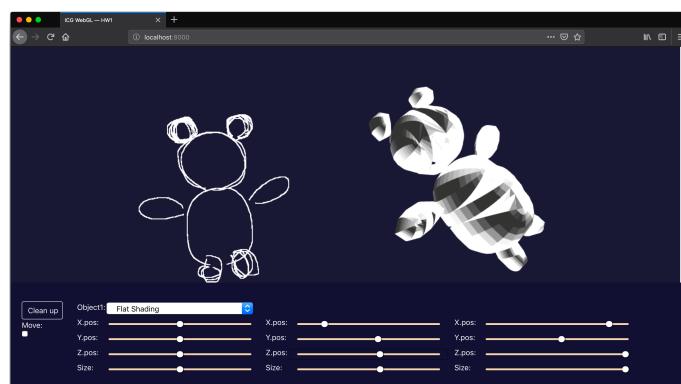


圖八

- (4) 複製所有頂點和三角形的邊到z軸負向。
- (5) 計算每個頂點的法向量(頂點接觸的每個面的法向量平均)。如此一來，就完成了根據使用者繪製的多邊形所製成的3D模型了。

6. Conclusion and Future work

我們成功實作出了Teddy System最核心的功能，但仍有部分空間可以再作進一步地加強。後端部分希望未來能夠支援立體的非凸多邊形，利用使用者多次作圖的線條，偵測新繪製的線條位置是否已有模型產生，若是，則可知此位置使用者希望呈現凹陷或是中空的效果；若否，則維持原本的演算法繪製立體模型。前端則希望增設其他能夠豐富使用者體驗的功能，舉例：提供使用者選取欲著色的模型位置以及顏色、能刪除單一已繪製的立體模型。



圖九

7. Teamwork

蕭乙蓁：系統架設、多邊形切割

林詩苓：建構3D模型所需之各要件(vertex positions, vertex indices, vertex normals)

陳盈如：使用者介面、WebGL應用