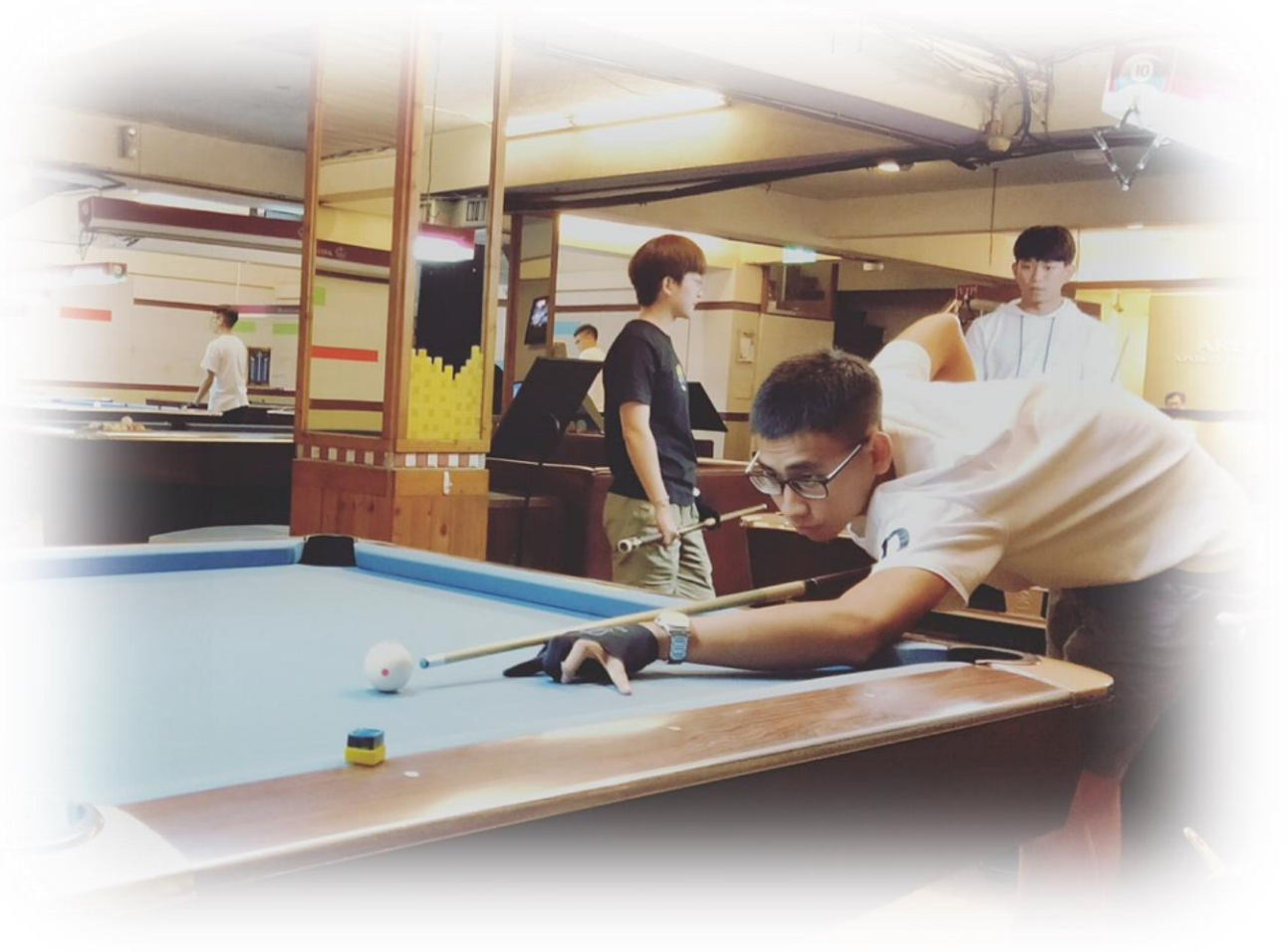
計算機圖學期末報告

撞球模擬器



創作動機

- 熱衷撞球
- 市面撞球遊戲的遊玩經驗

目標

基本要素

遊戲介面

操作設定

碰撞/運動演算

入洞判定

規則設定

基本規則撰寫

計分及勝利條件

犯規判定及作為

對戰

技巧呈現

推/拉/定桿

下塞(左右塞)

曲球

跳桿

目標

基本要素

遊戲介面

操作設定

碰撞/運動演算 進行中

入洞判定

規則設定

基本規則撰寫

計分及勝利條件

犯規判定及作為

對戰

技巧呈現

推/拉/定桿

下塞(左右塞)

曲球

跳桿

使用工具

VBO繪圖

Solidwork繪圖

函式建構

型別建構

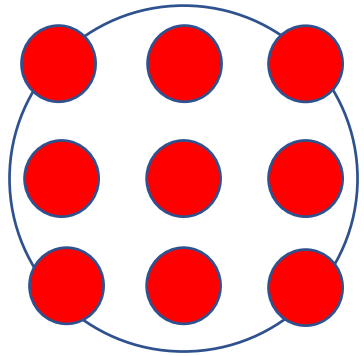
```
    VBOn[i * 9 + 3] = triArray[i].n[0];
    VBOn[i * 9 + 4] = triArray[i].n[1];
    VBOn[i * 9 + 5] = triArray[i].n[2];

    VBOn[i * 9 + 6] = triArray[i].n[0];
    VBOn[i * 9 + 7] = triArray[i].n[1];
    VBOn[i * 9 + 8] = triArray[i].n[2];
}

glGenBuffers(2, VBO_index);

// Vertex
glBindBuffer(GL_ARRAY_BUFFER, VBO_index[0]);
glBufferData(GL_ARRAY_BUFFER, nTriangles * 9 *

Sleep(200);
// Normal
glBindBuffer(GL_ARRAY_BUFFER, VBO_index[1]);
glBufferData(GL_ARRAY_BUFFER, nTriangles * 9 *
```

```
125
126   //調整兩球行為模式(根據其數據、位置關係)
127 ⊞void motion(Ball *a, Ball *b){ ... }
185
186   //判定兩顆球撞擊，若撞擊則回傳1
187 ⊞int Hit(Ball a, Ball b, GLfloat r){ ... }
194
```

```
49
50   //定義球球 x座標.y座標.v速度.W轉速
51 typedef struct{
52       GLfloat x;   //X座標
53       GLfloat z;   //Z座標
54       GLfloat Vx; //X方向速度
55       GLfloat Vz; //Z方向速度
56       GLfloat W_h; //左右塞給的y軸轉速
57       GLfloat W_v; //推拉桿給的x軸轉速
58  }Ball;
59
60
```

操作

方向鍵	左右：瞄球　　上下：視角
Enter鍵	視角控制（俯視／瞄球）
ASWD鍵	擊球點控制
空白鍵	擊球
zx鍵	z：提高力道　　x：降低力道



遭遇瓶頸

碰撞

1.運動方向計算：向量、角度、物理法則

維基百科
自由的百科全書

條目 討論 台灣正體 汉 漢 閱讀 編輯 檢視歷史 搜尋維基百科

高雄@維基聚會定期在每個月的第三個星期六舉辦

首頁
分類索引
特色內容
新聞動態
近期變更
隨機條目

說明
說明
維基社群
方針與指引
互助客棧
知識問答
字詞轉換
IRC即時聊天
聯絡我們
關於維基百科
資助維基百科

列印/匯出
下載為 PDF
可列印版

工具
連結至此的頁面
相關變更
上傳檔案

彈性碰撞 [編輯]

維基百科，自由的百科全書

彈性碰撞是碰撞前後整個系統動能不變的碰撞。彈性碰撞的必要條件是動能沒有轉成其他形式的能量（熱[illegible]），例如原子的碰撞。

目錄 [隱藏]
1 一維碰撞
1.1 性質
1.2 例子
2 應用
3 參見

一維碰撞 [編輯]

動能守恆：

$$\frac{1}{2}m_1v_1^2+\frac{1}{2}m_2v_2^2=\frac{1}{2}m_1v_1'^2+\frac{1}{2}m_2v_2'^2$$

動量守恆：

$$m_1v_1+m_2v_2=m_1v_1'+m_2v_2'$$

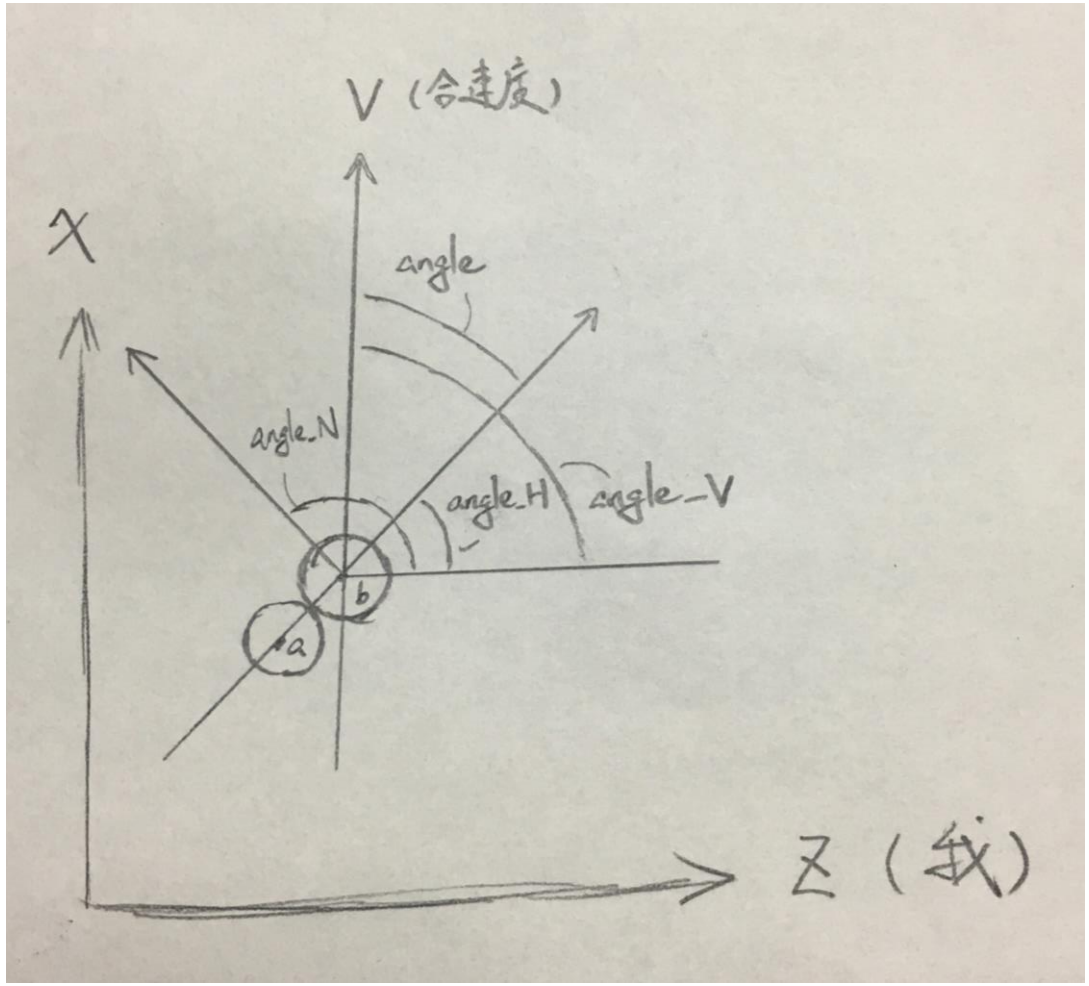
通過$v_1'-v_2'=v_2-v_1$ 解得兩個物件碰撞後速度：

$$v_1'=\frac{v_1(m_1-m_2)+2m_2v_2}{m_1+m_2}$$

和

遭遇瓶頸

1.運動方向計算：向量、角度、物理法則




```
128 void motion(Ball *a, Ball *b){
129
130     GLfloat X, Z, V, angle_hit, angle_V, angle_N, angle;
131
132     //因為球質量相等，動量守恆 -> 撞擊前後的 x.z方向球速總和相同
133     //並我暫且假設兩球撞擊後速度夾角為90度
134
135
136
137     //首先計算兩方向的速度和
138     X = a->Vx + b->Vx;
139     Z = a->Vz + b->Vz;
140     V = sqrt(X*X + Z*Z);//此為合速度值
141
142
143     angle_hit = atan2((b->x - a->x) , (b->z - a->z));//此為撞擊角度(a -> b)
144     angle_V = atan2(X , Z);
145     angle = angle_V - angle_hit;
146     if (angle_V >= angle_hit){
147         angle_N = angle_hit + 90 / 3.1415926 * 180;
148     }
149     else{ angle_N = angle_hit - 90 / 3.1415926 * 180; }
```

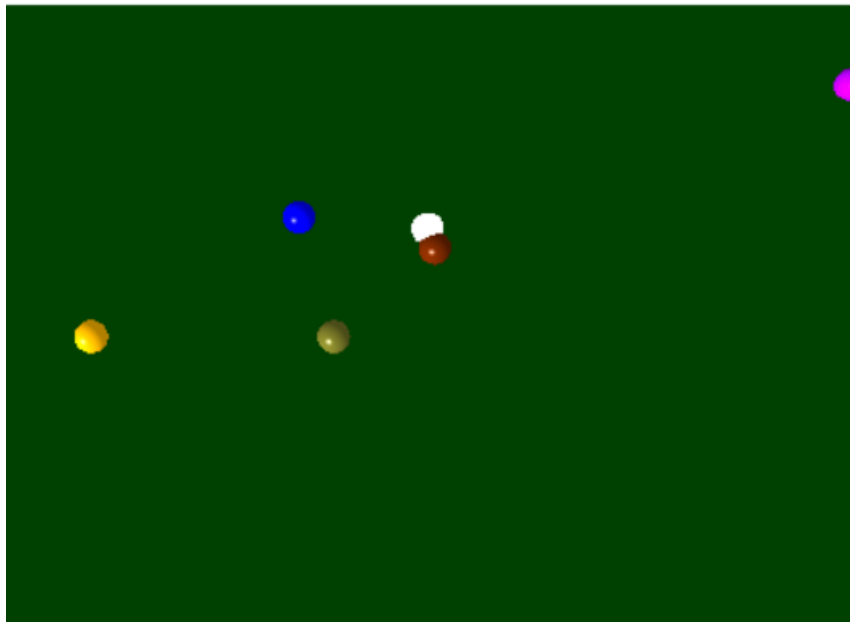
```
186         b->Vx = abs(V*cos(angle))*sin(angle_V);
187         b->Vz = abs(V*cos(angle))*cos(angle_V);
188         a->Vx = abs(V*sin(angle))*sin(angle_N);
189         a->Vz = abs(V*sin(angle))*cos(angle_N);
190
191         return;
192 }
```

遭遇瓶頸

碰撞

1.運動方向計算：向量、角度、物理法則

2.撞擊判定：黏住、穿透、消失不見



CODE展示

未來目標

1.完成前述目標項目

2.美術加強(球桌實體化、球道建構、球體貼圖)

3.優化程式(函式化、精簡化、運算法改良)

4.結合自身撞球經驗及球感，
將遊戲體驗盡可能改良至仿真

感謝聆聽