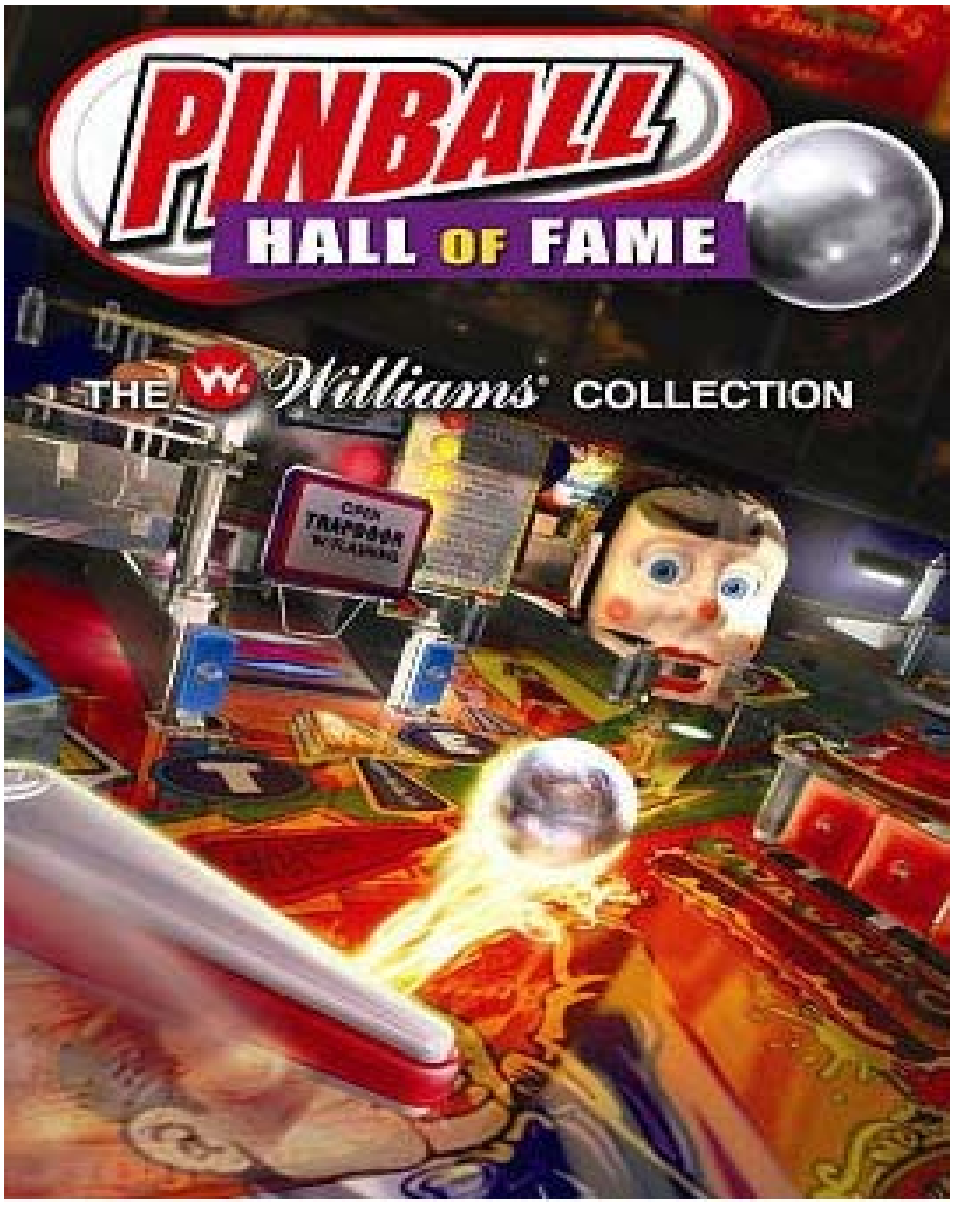
Pinball~遊戲設計



組員: b96902064 莊孟蒔
b96902065 葉治顯
b96902070 郭英樹
b96902077 黃子容

大綱:

一. 動機

二. 學習工具與參考資料

三. 基本物件介紹

四. 難處與解決方法

五. 程式解析與函式介紹

六. 會議記錄

七. 心得感想與未來展望

一. 動機:

從大一至今，學過 C、Java 以及組語，卻還不曾應用於遊戲上，因此，藉著這次機會，選了普遍性最高的彈珠台作為我們的期末報告主題。

二. 學習工具與參考資料

使用 HAM、C 語言與組合語言作為遊戲的開發工具。

資料來源:

HAM Tutorial http://www.aaronrogers.com/ham/index.php

Wiki

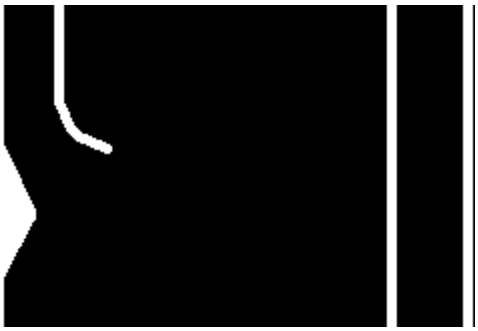
http://www.wikipedia.org/

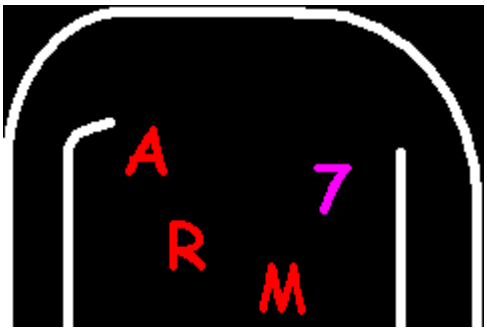
中國遊戲開發論壇

http://bbs.ogdev.net/TopicContent.aspx?BoardID=2&TopicID=6177

三. 基本物件介紹

*彈珠台軌道(背景):

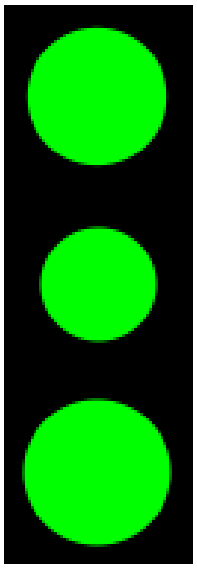




*球:

*打桿:　　*彈簧發射器:　　*緩衝器:

(leftpad)　(rightpad)



四. 難處與解決方法

- 如何使程式、鍵盤與圖形作連結?
- 避免物件間穿透(EX.牆與球間、球與打桿間…)
- 如何處理物件間的關係(會不會碰撞?滾動??)
- 背景或物件的替換(動態物體、背景布幕…)
- 球的碰撞路徑!!!!!!!!!!!!
  (球與軌道、球與緩衝器、球與打桿…)
- 如何處理背景的法向量
- 打桿的施力
- 浮點數除法、開根號(HAM 沒有支援此指令)
- 合適的重力場
- Objectspace 不夠大，無法同時放多物件於同一介面

以上數點會於介紹函式時，說明我們的解決辦法。

而我們花最多時間在處理「**碰撞**」的部分，大致上分為以下數點:

***球與靜態物間:**

背景、緩衝器:

*如何知道每個位置的法向量?

(建表儲存每個位置的法向量值)

*如何判斷球碰撞的點?或同時碰撞數點?

(用 block 將內數點的法向量值取平均值)

*因浮點數誤差而導致物件吃球?

(將數字放大以減少誤差)

*背景間的銜接處，有碰撞失誤?

(將法向量涵蓋範圍擴大)

***球與動態物間:**

彈簧發射器:

*如何流暢地伸、壓縮彈簧?

(調整適當的 frame 更新時間)

*重力的影響下，如何使球靜止於彈簧上?

(用 if 採特例的作法)

打桿:

*當兩者皆動時，如何判別相撞點?

(採預測方式，事先計算下個 frame 的狀態)

*球沿桿子滾動?

(想像成極小的碰撞運動)

*打桿子因打點不同而需施力處理?

(用角速度與力臂去計算施力)

*如何計算打桿上的法向量?

(似背景與緩衝器的作法，建表儲存其的法向量值，但是，多考量了角速度)

五. 程式解析與函式介紹

* wall.h

-wall_hit_check()

處理球與背景的碰撞，利用成是建立好背景法向量 table，接著預測嚇次更新時的位置，將球覆蓋到的面積所有法向量求出平均，這樣順帶可以解決牆角碰撞的問題，接著對球的速度做計算，並且將球貼近牆壁，處理未碰到牆而反彈的問題 。這部分有嘗試過轉成組語，但是因時間關係而失敗了。

* gravity.h

在彈珠台遊戲裡，為了要使彈珠能夠增加遊戲的可玩性，我們增加了重力對彈珠造成的影響。在處理重力的過程中，首先遇到的就是 GBA 不支援浮點數的問題。如果不利用浮點數而以整數的方式處理加速度，勢必會造成速度上的不細緻。

- ball_gravity()

對於以上的問題，我們增加了這個函式。我們的解決方法雖然同樣還是以整數的加法表示速度與加速度。但是我們稍微對於數字上做了一些處理：先以 offset_x, offset_y 表示，將其加入加速度 accel，並且將 offest_x, offset_y 除上 32 (數字越大，速度的表現上會越精細)，所得的商就是我們要的螢幕上顯示出來的速度(ball_vx, ball_vy)，而剩下的餘數不要丟棄，暫時存在 count_x, count_y 裡，否則將會與直接以整數儲存速度無異。

可以這樣想像：offset_x, offset_y 表示的是球真實的速度，而這個真實的速度可以分為能夠在螢幕上察覺的(ball_vx, ball_vy) 與無法在螢幕上察覺的(count_x, count_y) 。往後其他函式對於速度上的處理，也將會以 offset_x, offset_y 為主；處理螢幕顯示的部份則交給 ball_gravity()。

- gravity.s

組語實作的部份，我們考慮將運作次數上較多的 ball_gravity() 以組語的方式呈現。實作出的程式碼如下：

```
    .file   "gravity.c"
    .text
    .align
    .global gravity
    .type   gravity, %function
gravity:
    STMFD sp!,{r3-r12}
    ldrh    r4, [r0, #0]
    add     r4, r1, r4
    mov     r5, r4, lsr #5
    strb    r5, [r2, #0]
    sub     r4, r4, r5, lsl #5
    strh    r4, [r0, #0]
    @ldrh   r2, [r1, #0]    @ movhi @ * ori
    @strh   r2, [r0, #0]    @ movhi @ * ret
    LDMFD sp!,{r3-r12}
    bx  lr
    .size   gravity, .-gravity
    .ident  "GCC: (GNU) 3.3.2"
```

* score.h

除了彈珠台的架構之外，為了增加遊戲的娛樂性，我們特別加入分數機制。

- add_score(int hit, int x);

這個函式針對各種不同物體的碰撞進行不同的加分。

- add_special();

這個函式針對各種特別的碰撞進行不同的加分。總的來說，停留在特別區域（左側軌道區）會有額外加分、製造出緩衝器的連續碰撞也會有相當的加分。

- update_score();
將分數累計好後，利用 num_Btimap 更新，達到顯示分數更新的效果。

* string.h
彈簧的處理上相對於其他函式較為獨立，其函式也相當的簡單。
- string_start();
在遊戲一開始，按壓 B 鍵時，彈簧 sprite 會根據按壓的時間改變圖片的顯示，達到畫面上壓縮彈簧的效果
- shoot();
放開 B 鍵後，球的速度會根據彈簧壓縮的程度增加。

* Spring_animate(int i)
如果緩衝器與球發生碰撞，播放緩衝器彈跳的動畫。
int i 緩衝器編號

Add_springball(u8 bg, u8 x, u8 y)
在初始遊戲時使用，把緩衝器安置到螢幕上。
緩衝器只有在畫面移至它所在背景時才顯現。
u8 bg 緩衝器所在背景編號
u8 x X 座標
u8 y Y 座標

* Hit_check(int i)
判斷緩衝器是否與球發生碰撞，若發生碰撞，則根據法向量與原速度計算碰撞後的速度。
int i 緩衝器編號
在此使用幾何的方式作判斷，如果球進入緩衝器半徑內則發生碰撞，球與緩衝器間的連心線即為法向量。

* Scroll_bg()

判斷球是否超出畫面上下界，若有，則改變現在所處的區域。

由於 GBA 對 Sprite 佔據記憶體空間的限制，所以在更換區域時必須即時刪減上個區域的 Sprite，並貼上此區域的 Sprite。

* Update_bg()

根據現在所在的區域，貼上相應的背景與物件，如果球掉到最下層背景之下，則判定為 GameOver。

* Control_pad()

控制打桿的按鍵被按下時呼叫，負責改變打桿的角度，並紀錄按鍵按下時的打桿角度，以作角速度運算。

* Rotate_pad()

如果控制打桿的按鍵被按下，播放打桿旋轉的動畫，並呼叫 pad_hit_check()。

* Pad_hit_check()

判斷球是否與打桿碰撞，並以球碰觸到的法向量(查表)與打桿的角速度，改變球的速度。

六. 會議記錄

時間:97/12/21(日)
地點:MSN 開討論室
目的:研討往後聚會時間與思考主題

時間:97/12/26(五)晚上
地點:活大
目的:決定主題與摸索 HAM

時間:98/1/2(五)9:00am~17:00pm
地點:系館地下室
目的:摸索 HAM、分析彈珠台遊戲整體架構、劃出簡易圖檔與分工。

時間:98/1/17(六)9:00am~21:00pm
地點:校內丹堤咖啡
目的:球與緩衝器的碰撞、彈簧發射器球的碰撞及基本的 PPT 架構

時間:98/1/18(日)9:00am~21:00pm
地點:校內丹堤咖啡
目的:球與背景障礙物的碰撞、重力場及函式合併上的問題

時間:98/1/19(一)9:00am~21:00pm
地點:校內丹堤咖啡
目的:打桿與球的碰撞、修改圖檔、組語的函式處理、分數欄、完成 paper 與上台報告的 ppt

七.心得感想與未來展望

1. 心得

這次是第一次和別人一起合作寫一個完整的遊戲程式，主要問題是在於合併程式碼的部分，我們因為都沒有經驗，所以我們是用手動複製貼上的方式，做得並不好，下次希望能改進成類似 plug in 的方式做整合。

2. 未來展望

(1) 完成副函式組語版本，增加效能
(2) 增加浮點數、三角函數的運算
(3) 做出捲軸取代螢幕切換
(4) 減少利用 HAM 的貼圖函式，減少受限