為什麼會想做這個library呢？  
主要是因為不滿原書的寫法，不夠靈活快速，只限於2D座標系，所以才會想要自己寫一個。

以下是預計SGAL 1.0 會有的功能

100000(十萬)個炮彈不畫出來只更新，release下fps可以在350以上，  
我的電腦比較好跑出來的數據是

time = 1.866 ms (535.691 fps, 0.0019sec ,11.200% cpu of each frame)，但這不包含畫出來與新增炮彈，所以可能還要再調整。  
The following is the function would be expected to SGAL 1.0

計算部份的程式庫是跨平台的  
Calculation part is cross-platform library

使用3D座標系

Use 3D coordinate.

實作部份分為彈道與子彈行為兩種，可以自由組裝

Implement part of the bullet trajectory and Bullet behavior.

It can free assembly of two kinds.

可以讀檔和存檔。

Can read files and save files.

有一個簡單的程式可以測試碰撞,彈道,子彈行為與效能，使用directx10, 11。

Have easy program can test Collision, Trajectory, Bullet behavior and Performance by use direct 10, 11.

如果有人有任何建議我會考慮加在後續的版本中。

If anyone have any good ideas for me, maybe will appear follow-up version.

|  |  |
| --- | --- |
| Trajectory part | 彈道部份 |
| Direction way | 方向彈 |
| Set direction then shooting. | 選擇方向後發射 |
| N way | 規則分散彈 |
| Set direction, n, Radiation angle then shooting. | 選擇方向，子彈數量，放射角再發射 |
| Random way | 隨機彈 |
| shooting. | 直接發射 |
| Grapeshot | 散彈 |
| Set direction, n, Radiation angle then shooting. | 選擇方向，子彈數量，放射角再發射 |
| Circle n way | 圓形發散彈 |
| Set start direction, up, n, radius then shooting. | 選擇起始角度，向上方向，子彈數量，發散半徑再發射 |
| Radiation angle Random way | 有限角隨機彈 |
| Set direction, Radiation angle then shooting. | 選擇方向，放射角最大值再發射 |
| Tail Missile | 長尾飛彈 |
| Set head bullet and tail bullet, tail length | 選擇彈頭與尾巴，尾巴長度再發射 |

|  |  |
| --- | --- |
| Bullet behavior part | 子彈行為部份 |
| Straight | 直線 |
| Set velocity | 速度 |
| Constant acceleration | 等加速度 |
| Set acceleration, velocity | 起始速度、加速度 |
| B-spline | B-spline決定方向 |
| Control point, velocity of coefficient, ending way | 控制點，內插點的間距大小，結束的方式 |
| Basic function | 簡單函數 |
| Use std::string to description | 用字串表示數學函數 |
| Lua function | Lua函數飛行 |
| Use lua script to compute next position | 用lua腳本計算下一個位置 |
| Circular motion | 圓周運行 |
| Set Bullet and radius, velocity of angle | 設定要繞的子彈、半徑、角速度 |
| Prospective heart | 準心 |
| Set target and Tracking speed | 設定目標跟追蹤速度 |
| Limited angle Chase | 有限角追踨彈 |
| Set angle, velocity, acceleration | 設角度、速度、加速度 |
| Division Missile | 分裂彈 |
| Set Trajectory, Division time | 設定彈道、分裂時間 |
| Ending time | 設定結束時間 |
| If time over then ending bullet | 時間一到就子彈爆炸 |
| Near-detection ending | 接近彈 |
| Set minimum distance and if distance near so near then bomb | 設定距離最小值，距離小於最小值就爆炸 |

碰撞部份

Collision part

|  |  |
| --- | --- |
| 2D part | 2D 部份 |
| Triangle | 三角形 |
| Rectangle | 矩形 |
| Point | 點 |
| Circle | 圓 |

使用gc來配置少數無法控制的記憶體，  
與boost::pool加入記憶體配置，

使用lua與工廠方法產生物件。  
亂數部份使用boost:: mt19937產生。

射擊關卡使用lua描述。

預設使用2個thread做彈道與碰撞計算，  
所以每次彈道產生子彈時，會把該次所有子彈分到一個manager，  
總共會有2個thread，每次把子彈分到最閒的manager，  
更新時，  
一．主執行緒更新自機敵機坐標  
二．主執行緒把可能被擊中的目標(少數)做複製到各個thread避免同步  
三．主執行緒sleep，多執行緒子彈座標更新與碰撞判斷  
四．主執行緒整理結果

五．主執行緒畫出來

本程式使用多執行緒加速座標更新與計算，使用boost.thread來做跨平台的支援

使用100000(十萬)直線炮彈在debug中測試

Thread:1

時間 fps 時間(秒) 與60分之1秒的比例

time = 18.676 ms (53.544079 fps, 0.0187 use 112.057%)

Thread:2

time = 11.802 ms (84.727211 fps, 0.0118 use 70.816%)

Thread:3

time = 9.463 ms (105.669815 fps, 0.0095 use 56.781%)

Thread:4

time = 8.752 ms (114.253365 fps, 0.0088 use 52.515%)

使用1000000(一百萬)直線炮彈在release中測試

Thread:1

time = 10.863 ms (92.053375 fps, 0.0109 use 65.180%)

Thread:2

time = 6.661 ms (150.112686 fps, 0.0067 use 39.970%)

Thread:3

time = 5.210 ms (191.928360 fps, 0.0052 use 31.262%)

Thread:4

time = 4.517 ms (221.382874 fps, 0.0045 use 27.102%)

發現超過兩個以上的thread由於要同步的關係，速度增加不大，需要改寫同步部份才有可能再加速。