以遊玩特徵為導向的程序化內容生成方法

王澤浩
zehaowang.tw@gmail.com

陳品陵
ally53628@gmail.com

林廣柏
npes89033@gmail.com

鄒濬安
ericmina83@gmail.com

戴文凱
wktai@mail.ntust.edu.tw

國立台灣科技大學

ABSTRACT

程序化內容自動生成 (Procedural Content Generation) 在過去就廣泛被應用於遊戲設計領域，其主要目的為增加遊戲內容的隨機性與多樣性。在本文中，我們針對遊戲過程中的遊玩特徵 (gameplay patterns) 進行抽象化，使用程序化生成技術產生帶有意義遊戲關卡內容，藉此消彌或降低因隨機性所產生的不穩定要素，以改善並豐富遊戲體驗。

我們將遊戲關卡的構成劃分為任務 (Missions) 與空間 (Space) 兩種結構後，空間會依照任務結構進行有意義的轉換，接著依照遊玩特徵定義基因演算法 (Genetic Algorithms) 的演化依據。讓玩家在進行遊戲時能夠遵循關卡設計師的劇情脈絡外，亦能夠體驗到有意義且多樣化的遊戲關卡內容。

1 INTRODUCTION

程序化內容自動生成的遊戲開發技術已實施多年，利用此技術開發的經典遊戲中，最早可追朔至由 Michael Toy 和 Glenn Wichman 在 1980 年左右開發的經典電子遊戲《Rogue》。近年來，伴隨著遊戲產業的競爭下，成本的考量更加地重視，而程序化內容自動生成的遊戲開發技術，是為了解決降低開發成本的同時保持遊戲性，甚至創造出更豐富的遊戲性，因此這樣的技術又再度成為熱門的焦點。

在程序化內容自動生成遊戲的研究中，為了採取以演算生成遊戲關卡的策略，將遊戲關卡內容以高階抽象的方式呈現，例如 Joris Dormans 提出的方法是將冒險動作型遊戲的任務 (Missions) 與空間 (Spaces) 轉換成生成語法 (Generative Grammars) 進行運算；Antonios Liapis 則提出方法將策略型遊戲的地圖高階抽象化為地圖草圖 (map sketches)，再將基因演算法調整為兩階段式，進行演化。本論文基於 Joris Dormans 與 Antonios Liapis 各自所提出的方法為出發點，保留了前者追求遊戲進程之順序性，與後者生成穩定且多樣化的遊戲內容，希望藉此程序化生成的遊戲關卡，既符合關卡設計師的需求且具有豐富的遊戲性。

我們採用 Antonios Liapis 所提出的兩階段演化方法。第一階段採用 Joris Dormans 所提出的生成語法及改寫系統生成任務圖，再將任務圖轉換為房型空間，得到尚未包含遊戲物件的遊戲地圖；第二階段則針對動作冒險遊戲提出數項適應性函數，採用基因演算法求得包含遊戲物件的最佳遊戲地圖。以往的相關研究皆以二維遊戲的關卡內容為主，本篇論文則以三維遊戲的關卡內容作為研究對象。

2 RELATED WORKS

我們所參考的文獻主要分為兩個類型，分別為程序化生成任務內容與程序化遊戲物件擺放。

2.1 Mission/Space 框架

Joris Dormans 認為一個完整的關卡需要包含任務與空間二者；需要有一特定的空間佈局，及一系列需要於此空間中被執行的任務。關卡任務代表玩家需要按照任務流程，來依序挑戰才能夠完成該關卡；關卡的空間由其地理佈局所組成，或者由與地圖相似的節點網絡所構成。由於任務與空間之間的交錯混雜，導致關卡設計者最終採取簡單卻有效的策略，也就是讓任務與空間同構。雖然同構在設計上不是唯一的選擇，但對於某些遊戲是非常合適的，特別是一具有線性的關卡設計。而 Joris Dormans 亦提出了一種自動關卡設計的方法，藉由產生一個任務，再利用這個任務去產生適合此任務的空間。

舉例來說，關卡設計者透過生成任務的介面來建立任務圖 (mission graph)，玩家必須執行這些任務才能夠完成關卡，接下來將任務轉換為空間，並將任務依序安排至該空間圖 (space graph) 中。設計者接著在地圖添加更細節的內容，直到地圖充滿任務的要素並作為遊戲的關卡。

任務圖注重於任務與玩家的相互關係，表現出玩家距離通關的進度狀況。主要由兩種要件：節點和有向連結線所構成，其中節點再細分為任務、起點與終點；有向連結線再依照兩節點之間的執行先後關係，細分為薄弱條件、強烈條件與抑制。其中，強烈條件或抑制的關係，會導致某些節點無法執行。空間圖直接呈現了關卡的空間結構，且大多數的節點能夠直接表示出玩家目前所在位置。空間圖中的任何節點能透過顏色、字母來表示不同類型。主要亦由兩種要件：節點和連結線所構成。節點細分為場所、鎖和遊戲元素所構成；有向連結線細分為通道、閥、窗、解鎖與上鎖等。

改寫系統 (rewrite system) 由具有左側與右側的規則 (rules) 所組成，能夠將規則中指定的一符號集能夠被另一符號集所取代。改寫規則當中所使用的符號，便是在遊戲中經常會出現一些具有代表性的物件、要素或任務目標等，在字母表 (alphabet) 中定義以抽象化描述遊戲中的週期性結構 (recurrent construction)。改寫系統能夠套用在構成任務的圖形語法 (graph grammars)及構成空間的形狀語法 (shape grammars)，二者能夠獨立生成出結果，不過建議能夠將改寫系統套用在任務圖上，使其能夠產生出空間圖。本文提及之任務圖和空間圖是經過改良後的版本，定義其規則時會有些微上的不同，但更能夠體現出遊戲的關卡結構。

2.2 Map Sketches 與 Segments 的演化

Antonios Liapis 開發了策略型遊戲的抽象化地圖生成工具 - Sentient Sketchbook。在 Sentient Sketchbook 中，遊戲關卡設計師能夠以低分辨率、高階抽象的方式來編輯地圖草圖 (map sketches)，構成地圖的瓦磚類型有資源磚、基地磚、不可通行磚與可通行磚等。典型的戰略型遊戲中，每位玩家都必須從隨機選擇的基地開始採集資源以建構戰鬥單位，並利用這些戰鬥單位摧毀敵方基地以完成遊戲。

當設計師編輯地圖時，該工具能夠測試地圖的可玩性（意旨能夠正常進行遊戲）且量化顯示，如果沒有足夠的基地、資源或可連通的路徑，那麼工具提供的遊玩特徵指標將會提示該地圖為不可遊玩的狀態。而這些遊玩特徵指標分別為資源安全性：距離基地僅一格以內的資源磚數量；安全區域：計算基地與敵方基地間的磚總數；探索性：利用洪水填充演算法，計算從基地至敵方基地時，可通行的磚總數。透過用戶當前編輯的地圖草圖，該工具利用基因演算法進行前述等指標，評估適應性函數 (fitness functions) 以解決約束最佳化 (constrained optimization) 等問題，來產生出更多意想不到的地圖輸出結果。

後續的研究中，Antonios Liapis 將基因演算法調整為兩階段演化，第一階段演化為地圖草圖演化，第二階段為地圖片段 (map segments) 演化。地圖片段的結構類似於地圖草稿，由 NxN 的瓦磚所構成，瓦磚的種類能夠像是空磚、牆、連接處、出口、怪物或寶箱等，其中連接處是為了讓地圖片段彼此能夠接合以填滿地圖。利用地圖草稿所轉化成的初始地圖片段可作為演化用的胚胎 (embryogeny)，於此階段定義的牆、連接處會呈顯穩定狀態，不隨著演化過程而改變，其餘瓦磚有機會由空磚突變為怪物、寶箱或牆，反之亦然。並探討不同的目標函數與胚胎，如何影響的圖片段的最佳解與外觀。

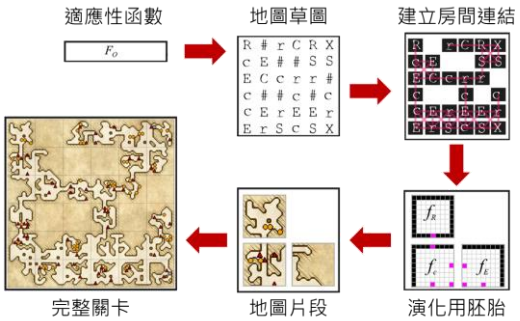

Figure 1: Antonios Liaps 提出的兩階段式關卡演化。

3 METHODOLOGY

這本篇論文中，我們仍保持 Joris Dormans 與 Antonios Liapis 為求遊戲設計過程抽象化與高階化的訴求。我們將「Mission/Space 框架」與「Multi-segment 演化」兩種關卡生成方法結合，保留了前者追求的遊戲進程之順序性，後者帶來穩定且多樣化的遊戲內容，希冀藉此提升整體遊戲體驗、相輔相成。

Figure 2 為系統的整體流程圖。遊戲設計師能夠透過巨觀的觀點來構築遊戲體驗流，將遊玩特徵拆分成多項規則，利用生成語法及改寫系統生成出任務圖。依照任務

圖中對應的終端節點 (terminal nodes) 轉換為事先建構完成的房型空間，並得到尚未包含遊戲物件的遊戲地圖。接下來，針對動作冒險遊戲我們提出了數項評估遊戲性的適應性函數，採用基因演算法的過程，令各房間遵守適應性函數的限制得到符合訴求的最佳解。

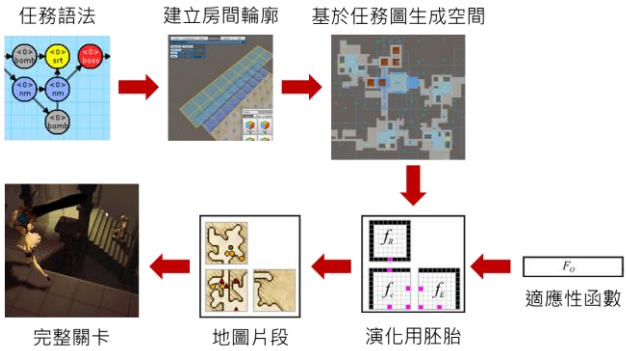

Figure 2: 本論文提出系統之流程圖。

3.1 任務語法

在任務語法的設計階段，我們參考遊戲 Spiral Knight 的關卡地圖，分析其遊戲進程結構，繪製期望的任務圖並將觀察到的遊玩特徵，接著拆分成任務語法規則使用改寫規則產生出近似結構的任務圖。

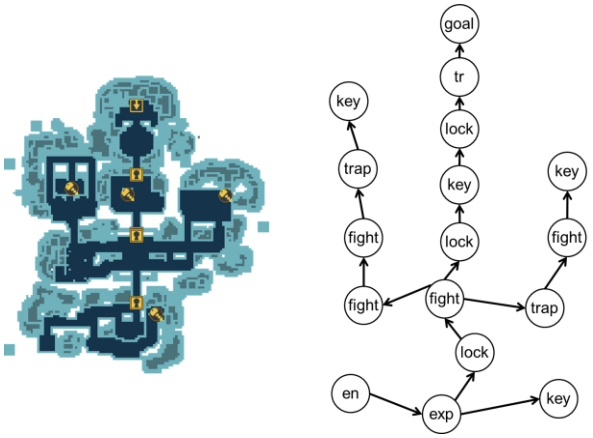

Figure 3: 將 Spiral Knight 遊戲中關卡 Roarsterous Ruins 轉換為任務圖的結構。

3.2 房型建構

Joris Dormans 於文獻中提到為二維空間的範例，我們的實驗環境以三維空間為主。在空間語法中我們將直接構築遊戲的基礎房型，但不設置怪物、寶箱或陷阱足以直接影響遊戲性的遊戲物件，如 **Figure 4**。此外，我們希望空間中的遊戲物件能夠有意義的自動化配置，即在設計空間語法的流程中，忽略絕大部分的遊戲物件配置，直到 3.3 節提出之方法達成。

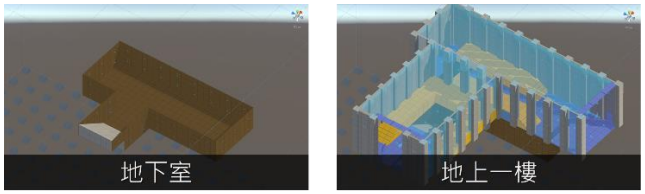

Figure 4: 房間鳥瞰圖。建構房型輪廓，暫不設置遊戲物件。

我們對於空間語法做了修改以利實驗環境建置，在一個關卡 (level) 中包含數個房間容器 (volumes)，每一房間由不定數量的房間塊 (chucks) 組成，且房間塊固定以 9x9x9 個正立方體體素 (voxels) 所構成。為了要從已生成完畢的任務圖再衍生出分支的遊戲空間，在我們創建的房間容器會對應一任務語法之字母表當中一的終端符 (terminal symbols)，這樣的關係稱作為建造指示 (building instruction)；若在改寫規則運行中，且有多項規則同時符合替換的條件時，系統會基於它們的關聯權重 (relative weight) 隨機挑選一個規則。

3.3 地圖片段

延續上一小節的關卡結構，房間視為染色體 (chromosome)；房間中能夠放置遊戲物件的各點座標視為基因 (genes)，而基因的類型有空磚、怪物磚、寶箱磚與陷阱磚等；同一個房間會擁有多種不同遊戲物件配置情況的染色體，這些染色體的集合便是族群 (population)。第一步驟，產生初始父母代族群時，讓全部的基因先預設為空磚，並使各染色體先行隨機突變；第二步驟，透過適應性函數計算各染色體的適應值 (fitnesses)，完整的適應性函數在 3.3.1 小節中說明；第三步驟將會從族群中挑選最優異的兩個父母染色體，高機率進行交配，若無進行交配將會將子代沿用父母代的基因；第四步驟有低機率讓衍生的子代進行突變；第五步驟以新的衍生子代取代舊有的父母代族群；第六步驟會檢查是否達到終止條件，若尚未滿足終止條件，便會回到第二步驟，直到輸出最佳解。

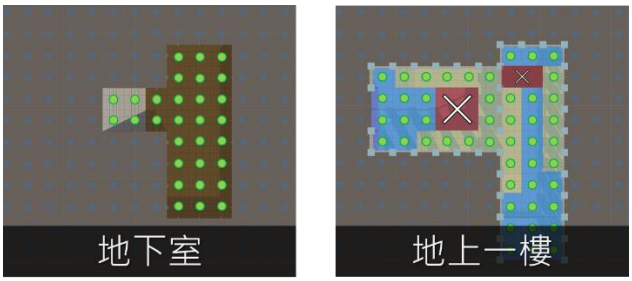

Figure 5: 綠色圓點表示能夠於該房間中放置遊戲物件。

3.3.1 各項適應性函數。 動作冒險遊戲 (A-AVG)、動作角色扮演遊戲 (A-RPG) 等類型遊戲，多可見一些制式化遊戲物件的搭配組合，我們嘗試汲取出多項遊玩特徵並參數化公式，作為評估關卡品質的指標之一。在前處理時，我們使用 A-Star 演算法尋找入口至多個出口的最短路徑，凡經過的座標稱作為空間動線 (MP) 之一，並將其權重值 (mp) 增加一，空間動線為多項指標關鍵性的參考依據。最後的適應性分數會依照各函數得分與其權重值加權後加總。

$$f_{all} = \sum_{i=1}^{M} (f_i \times w_i) \tag{1}$$

3.3.1.1 死角點 (Neglected)。 由於房與房之間的牆壁阻隔，使得敵人能夠埋伏於入口附近之死角處，出奇不意地對玩家展開攻擊。為了體現出這種現象，我們將敵人 (E_i) 與主要動線上各點 (MP_i)，兩端點連線之對角線所構成的立方體，立方體所涵蓋各座標點 (N_j) 至該對角線的距離為 d_k，隨著距離增加影響程度會衰減；vis_k 為該點的可視情形，若有不可視的座標存在便會提高適應值。隨著動線的順序演進，影響程度逐漸衰減。

$$f_{neg} = \frac{1}{\sum_{i=1}^{M} i} \times \left(\sum_{j=1}^{M} \left(\frac{-1}{N_j} \times \left(\sum_{k=1}^{N_j} \frac{1}{d_k} \times vis_k \right) \right) \times (M - j + 1) \right) \tag{2}$$

3.3.1.2 阻攔點 (Block)。 敵人會專注於阻擋玩家繼續前進，迫使玩家與其發生衝突。MP 為加總所有空間動線權重 mp_i；E 為加總所有敵人於空間之動線權重 (e_j)，倘敵人並未落在動線上，權重則為 0。為了達到隨著動線上的敵人愈多，對此適應性函數的影響力愈低，則取以 MP 為底 E 的對數。

$$f_{blk} = \log_{MP} E, MP = \sum_{i=1}^{N} mp_i, E = \sum_{j=1}^{M} e_j \tag{3}$$

3.3.1.3 攔截點 (Intercept)。 與阻攔點近似，但敵人會被配置於動線附近非動線上，以快速追擊玩家為目的。各敵人 (E_i) 越接近空間動線各點 (MP_j) 時影響愈大，且動線權重 (mp_j) 亦會影響加權程度。

$$f_{itc} = \frac{1}{M \times N} \sum_{i=1}^{N} \sum_{j=1}^{M} \left(\frac{1}{dist(E_i, MP_j)} \times mp_j \right), E_i \neq MP_j \tag{4}$$

3.3.1.4 巡邏點 (Patrol)。 為確保各敵人擁有足夠的空間能夠進行移動。利用 $Cover_i$ 計算敵人 (E_i) 在指定半徑 (r) 內，能夠行動的座標點數量比例，$count(E_i, r)$ 代表指定半徑內敵人可以行動的座標點數量；$plane(E_i, r)$ 代表以敵人為中心的指定半徑內，平面上所有座標數量（包含不可通行的牆壁等類型）。將 $count(E_i, r)$ 和 $plane(E_i, r)$ 的比例作為可以行動的座標數量比例。另外，本次實驗為三維空間，因此有機會出現可行走的數量大於平面上所有座標數量，在此對兩者比較大小，取最大值作為所有座標數量，以確保 $Cover_i$ 的數值介於 0 至 1 之間。

$$Cover_i = \frac{count(E_i, r)}{MAX(plane(E_i, r), count(E_i, r))} \tag{5}$$

最後，為了達到隨著敵人數量愈多，對此適應性函數的影響力愈低，針對不同敵人之可以行動的座標數量比例，分配不同權重值，再加總成為所需的適應性函數值。

$$f_{plt} = \sum_{i=1}^{N} (d_i \times Cover_i), d_i = \begin{cases} \frac{1}{2^i}, if\ i \neq N \\ \frac{1}{2^{i-1}}, if\ i = N \end{cases} \tag{6}$$

於本次實驗中，我們採用 $r = 3$ 作為實驗範例，該數值可由遊戲設計師決定。

3.3.1.5 守衛點 (Guard)。 為體現出敵人會保衛寶箱 (T) 與出口 (E) 的現象，計算敵人 (E_i) 與關鍵性較高遊戲物件 (O_j) 之間的距離，倘若距離愈近則帶來的影響力愈大。

$$f_{grd} = \frac{1}{M \times N} \sum_{i=1}^{N} \sum_{j=1}^{M} \frac{1}{dist(E_i, O_j)}, O_j \in \{Treasure, Exit\} \tag{7}$$

3.3.1.6 至高點 (Dominated)。 當玩家可能所在動線上之位置 (MP_j) 與敵人的位置 (E_i) 具有高低差時，敵人便適合採取遠程攻擊；為了提供玩家思考對付遠程敵人的緩衝時間，將敵人配置於動線末端附近是較好的選擇，j 隨著動線的順序演進，影響程度逐漸增幅。

$$f_{dom} = \sum_{i=1}^{N} \sum_{j=1}^{M} \left(\frac{1}{dist(E_i, MP_j)} \times mp_j \times j \times high(E_i, MP_j) \right) \tag{8}$$

3.3.1.7 支援點 (Support)。敵人 (E_i, E_j) 之間擁有一定程度的護援關係，當敵人彼此的距離愈低其影響程度越大，同時該敵人 (E_i) 必須遠離動線 (MP_k)。

$$f_{sup} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\frac{1}{dist(E_i,E_j)} + \frac{1}{M}\sum_{k=1}^{M} dist(E_i, MP_k)\right) \quad (9)$$

4 RESULTS AND DISCUSSION

為了探討前述方法是否符合需求目標，將進行以下實驗：4.1 小節中，針對所挑選的房間採用不同權重的適應性函數，會如何影響房間內的遊戲物件之配置結果。4.2 小節中，觀察染色體中的基因數量對於演化過程之影響。**Error! Reference source not found.**小節，進行評估不同的交配策略下，何者優劣優勢之情形。

進行本實驗時，在一次世代的演化過程，有 80% 機率父母代間會進行兩點交配 (two-point crossover)；10% 機率衍生子代會進行突變，染色體個體中有 5% 至 20% 的基因數量會轉換成其它的物件種類。

4.1 演化結果與其品質

我們運行了 100 次實驗，每次實驗演化 100 個世代，其使用的族群含有 100 組染色體個體。在 **Figure 6** 將各演化之世代取最佳染色體得分，計算其平均值以及標準差，並獨立運行兩項適應性函數，分別給予其權重為 1，其餘為 0 所得之數據。從該圖中觀察出守衛點於前 10 世代左右已達收斂；阻擾點之標準差隨著不同實驗的交配、突變情形不同，結果有著顯著差異。但二者演化的趨勢仍與世代數量有著正相關，直至收斂趨緩。

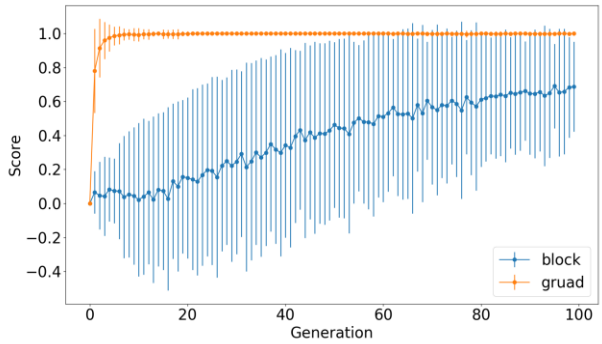

Figure 6: 守衛點與阻擾點之適應性函數於各世代得分情形。

Figure 7 中展示了部分適應性函數的最佳佈局，其中紅圓點為敵人、黃圓點為寶箱、灰圓點為陷阱，下方端點為入口、兩側端點為出口。在左圖當中的守衛點權重為 1，其餘適應性函數為 0 的收斂情形，能夠觀察到高得分的佈局中，寶箱附近必伴隨著敵方單位守護著。而右圖的阻擋點權重為 1，其餘為 0，觀察到將敵人放置在入口通往兩條出口的必經道路的得分較高。

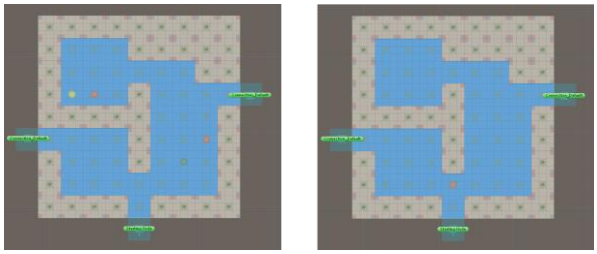

Figure 7: 左圖為守衛點指標；右圖為阻擾點指標。

接下來調整不同的染色體個體數量，觀察對於適應性函數的影響情形。在 **Figure 8** 中，增加 50 組、200 組的染色體個體數量進行比較。在單一阻擋點適應性函數權重的分配下，平均的運算時間分別為 1,500 毫秒、3,500 毫秒、20,000 毫秒等。

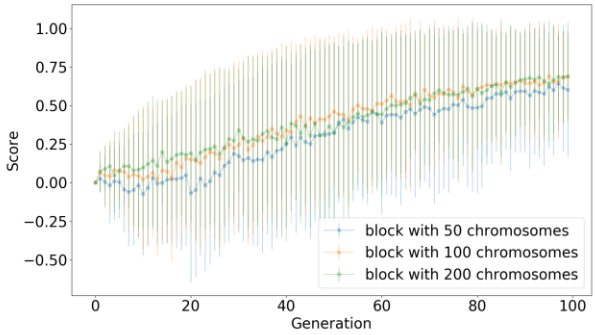

Figure 8: 不同染色體數量在阻擾點指標的表現情形。

4.2 房型規模之比較

在本小節中，依 **Figure 9** 分別以三種房型進行演化表現上的比較，在不同房型間擁有相同的主要動線（以綠色表示之），此外，不同處為剩餘的空瓦磚數量，代表三者所構成染色體個體的基因數量上的差異。

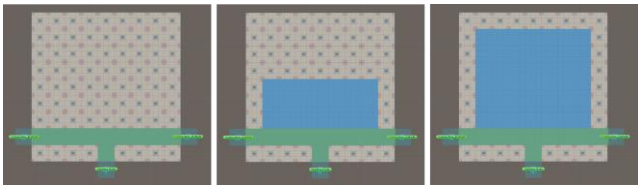

Figure 9: 三種擁有相同動線，但不同規模的房型。

與主要動線相關的適應性函數有死角點、阻攔點、攔截點、巡邏點…等，以下將採用守衛點作為實驗參考對象。由 **Figure 10** 得知，在小房型中，約莫在第 40 世代左右達到收斂；在中房型與大房型中，二者之平均值不分軒輊，但在第 70 世代開始，中房型的標準差逐漸少於大房型之標準差。

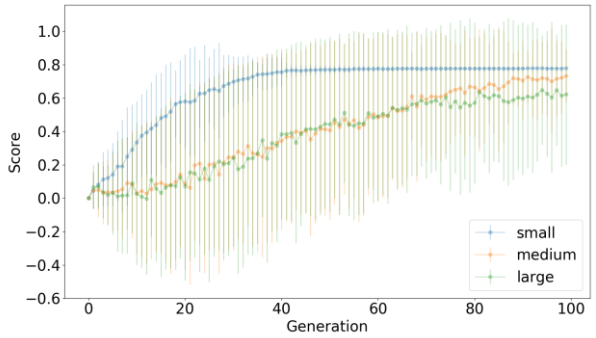

Figure 10: 不同房型規模對於與動線相關之函數表現狀況。

4.3 評估交配策略優勢

於 **Figure 11** 中我們觀察出，採用交配策略可以使子代延續上世代良好的基因，並搭配突變讓更好的基因留給子代。若不採用交配策略，不但無法延續上世代良好的基因，還因為只有突變，而有機率使得分高的染色體突變成得分低的染色體，造成演化成高得分的過程緩慢。

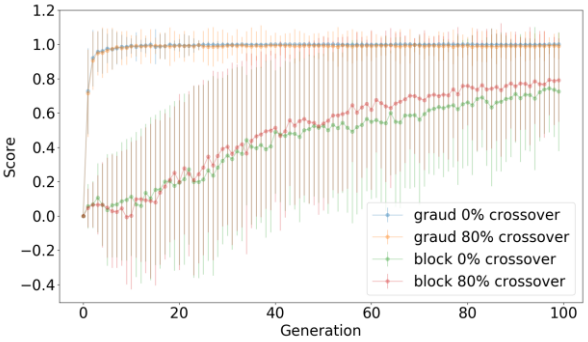

Figure 11: 守衛點與。

5 CONCLUSIONS

首先，使用生成語法產生關卡的任務圖，利用改寫規則將其轉換為遊戲空間的雛形。接著，制定系列能夠體現特定遊玩特徵的適應性函數（例如，寶箱必須附近敵人的保護），藉由體素結構的房型進行基因演算法求得最佳配置，輔以抽象化的動線作為評估的指標。我們的實驗表明，驗證即使描述空間的精確度降低之情形下，仍確保演化結果具有一定品質。

試舉實驗設計當中一例，守衛點的適應性函數，當中沒有涵蓋寶箱與敵人之間的最短通行距離，可能發生二者的直線距離符合門檻，事實上卻被一道不可通行的牆壁所阻隔。欲更加具現出特定的遊玩特徵，在設計適應性函數時並須多加著墨不同面向。

在某些情況下，關卡設計師會希望多種不同的適應性函數指標，彼此間會獨立體現其遊玩特徵，但是在目前的函數設計上較難實現該現象。

藉此研究成果，希冀提供遊戲開發者不同面向遊戲設計模式，以及加速關卡開發效率。

6 ACKNOWLEDGMENTS

感謝蔡建毅先生與其團隊對於本研究給予相當大的支援與建議，以及感謝相關計劃成員 Ardiawan Bagus Harisa、黃小峰、葉定豪三位同學提供程式編程等協助。

REFERENCES

[1] Dormans, J. (2010, June). Adventures in level design: generating missions and spaces for action adventure games. In *Proceedings of the 2010 workshop on procedural content generation in games* (p. 1). ACM.

[2] Dormans, J. (2011, June). Level design as model transformation: a strategy for automated content generation. In *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games* (p. 2). ACM.

[3] Dormans, J. (2012, May). Generating Emergent Physics for Action-Adventure Games. In *PCG@ FDG* (pp. 9-1).

[4] Karavolos, D., Liapis, A., & Yannakakis, G. N. (2016, September). Evolving missions to create game spaces. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on* (pp. 1-8). IEEE.

[5] Liapis, A. (2017). Multi-segment Evolution of Dungeon Game Levels.

[6] Karavolos, D., Bouwer, A., & Bidarra, R. (2015). Mixed-Initiative Design of Game Levels: Integrating Mission and Space into Level Generation. In *FDG*.

[7] Liapis, A., Yannakakis, G. N., & Togelius, J. (2015). Constrained novelty search: A study on game content generation. *Evolutionary computation*, 23(1), 101-129.

[8] Liapis, A., Yannakakis, G. N., & Togelius, J. (2013, April). Generating map sketches for strategy games. In *European Conference on the Applications of Evolutionary Computation* (pp. 264-273). Springer Berlin Heidelberg.

[9] Font, J. M., Izquierdo, R., Manrique, D., & Togelius, J. (2016, March). Constrained Level Generation Through Grammar-Based Evolutionary Algorithms. In *European Conference on the Applications of Evolutionary Computation* (pp. 558-573). Springer International Publishing.

[10] Lanzi, P. L., Loiacono, D., & Stucchi, R. (2014, August). Evolving maps for match balancing in first person shooters. In *Computational Intelligence and Games (CIG), 2014 IEEE Conference on* (pp. 1-8). IEEE.

[11] Zook, A., & Riedl, M. O. (2014, July). Automatic Game Design via

Mechanic Generation. In *AAAI* (pp. 530-537).

[12] Dormans, J. (2012). *Engineering emergence: applied theory for game design.* Creative Commons.

[13] Neil, K. (2015). *Game Design Tools: Can They Improve Game Design Practice?* (Doctoral dissertation, Conservatoire national des arts et metiers-CNAM).