以粒子群最佳化為基礎之電腦遊戲角色行為設計與實作

The Game Character Design and Implementation Using Particle Swarm Optimization

徐育良(Y.L.Hsu),[1] 林正基(C.C.Lin),[1] 蔡清欉(C.T.Tsai),[1] 李建樹(J.S.Lee)[2]
1 東海大學資訊工程與科學系　2 虎尾技術學院飛機工程系
1 Dept. of Computer Science and Information Engineering, Tung Hai University
2 Dept. of Aeronautical Engineering, National Huwei Institute of Technology
Tel:(04)23590415　FAX: (04)23591567　E-mail: cttsai@mail.thu.edu.tw

摘要

電腦遊戲擁有豐富娛樂性、高度互動性以及各式媒體整合能力，電腦遊戲已成為人們重要的一項休閒娛樂活動。本文針對遊戲角色之行為設計來作探討，希望能夠創造出具有有效學習能力之遊戲角色。現今角色行為設計普遍存在有角色行為規則訂定與調整耗時、角色行為不具學習性的缺點，本文提出以模糊狀態機實作之規則系統配合機器學習技術來作為角色之行為機制，並且利用粒子群最佳化技術調整影響角色行為權重的值，讓角色具有學習的能力。本文所提出之角色行為設計新方法可以輔助遊戲設計師更容易的為角色訂定行為規則，或是在遊戲進行時讓玩家感受角色具有調整行為之學習能力。

關鍵詞：粒子群最佳化、群體智慧、行為動畫、遊戲人工智慧

Abstract

Computer games have highly interactive ability and can integrate various media. Playing computer games has become people's popular entertainment. The paper proposes a game AI engine that with learning ability. The fuzzy-state machine with a machine learning technique is used as behavioral mechanism. By using particle swarm optimization with several fitness function to dynamically adjust behavior weight has let character with learning ability. The real world game implementation has been applied to our proposed method, such as QUAKE III bot and troll game. Results have shown that the novel mechanism can help AI designer design behavioral rules of characters easier, and can let game characters have in-game learning ability.

Keyword : particle swarm optimization, swarm intelligence, behavioral animation, game AI

一、簡介

隨著生活素質的提升，人們對於休閒娛樂的需求與重視也逐漸轉變，從 1958 年最早的電子遊戲問世以來，根據 NPD Group 的調查，全球電子遊戲市場的產值已於 1999 年超越了電影工業，市場規模達 200 億美元[1]。玩電子遊戲已成為人們重要的一項休閒娛樂活動。

近年來，網路遊戲的出現，更讓玩家感受到與真人的互動更具有挑戰與趣味性， 如此卻也反應了虛擬角色的行為並不能滿足玩家的需求， 此外， 儘管網路遊戲的出現，非玩家控制的角色(none-player character, NPC) 仍有其持續存在的需求與不可替代性[2,3]。NPC 就是由電腦人工智慧(AI)所控制的角色，NPC 電腦遊戲角色設計指的是對其角色進行造型(model) 、動作(motion) 與行為(behavior) 的設計， 可以對應到電腦繪圖領域中的塑模階層式架構(computer graphics modeling hierarchy) [4] ，此架構由下至上代表了對電腦動畫從底層的幾何操作到高層的行為控制。階層架構中最底層的幾何層(geometric) 代表的是電腦圖學領域中所探討的如何利用幾何表示法來描述立體的物體。中高層是利用物理學、運動力學、解剖學等的原理來模擬物體本身與物體在物理環境下的動作以產生擬真的動畫。

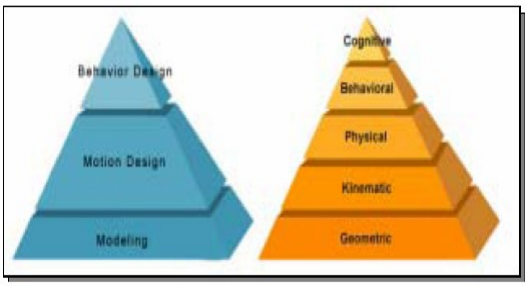


圖 1、遊戲角色設計與電腦繪圖塑模階層式架構的對應

遊戲角色的行為設計又稱作遊戲人工智慧，在電腦動畫又稱作行為動畫(behavioral animation) ，虛擬角色行為設計的方法大致上可以分為三類，以下分別介紹：

(一) 以規則為基礎的系統(rules-based systems)

這種方法利用事先定義好的條列式規則來規定遊戲角色行為， 是一種簡單、容易了解的方法， 可是當角色的行為複雜時， 相對應的規則訂定也困難。根據實作方法的不同，以規則為基礎的系統又可分為有限狀態機(finite-state machines) 、模糊狀態機(fuzzy-state machines)與生產系統(production systems) 。有限狀態機指的是NPC 擁有「有限數量的行為與狀態」。NPC 根據輸入在這些事先設計好的狀態中做切換並輸出相對應的行動，呈現出來的是一種有限的智慧。有限狀態機後來加入模糊邏輯(fuzzy logic) 的概念而成為模糊狀態機，如此可以讓NPC 具有一些不可預測的行為。在模糊狀態機中，一組相同的輸入不見得每次都產生相同的輸出，而是根據隨機選擇或是權重函式來選擇一組輸出中的一個 [5-8]。

生產系統擁有一個知識資料庫與規則資料庫， 知識庫與規則庫提供角色在做行動時所需的相關資訊，可處理複雜遊戲角色的行為，並訂定與管理複雜的規則[8]。

(二) 以目標為基礎的系統(goal-based systems)

以目標為基礎的系統取代了以規則為基礎的系統需要將角色所有行為模式列舉的方式， 設計人員僅需定義出角色的目標與告知角色達到目標各種可行的方法。在遊戲進行的時候， 遊戲角色動態的決定何種方法才是能夠達到目標的最好解決方案， 此種設計方法的優點在於能夠以較簡單的方式產生出複雜的角色行為。以目標為基礎的方法所設計的角色， 會感受到環境的變化， 也就是會根據環境的變動資訊作為行動的依據。為了讓角色擁用上述的能力， 開發人員必須要給與角色知識、目標、策略， 加上角色本身的感知能力來推斷出達到目標的可行方法。J. Funge[4]將認知心理學(cognitive psychology) 的概念應用到虛擬角色的設計上就是一種以目標為基礎的方法， 他認為認知模型決定角色可以認得什麼、如何取得知識以及如何利用這些資訊來規劃行動。

(三) 學習與適應系統(learning and adaptation systems)

遊戲如同真人交手一樣，相同的策略與行為模式是無法勝任的。對於遊戲設計者來說，正確地應用機器學習技術可以創造出更具有智慧的遊戲角色， 取代以往必須要事先推測玩家所有可能的行動並訂定出相對應的規則。設計具有學習能力的遊戲角色可以分為間接學習(indirect learning)與直接學習(direct learning)。間接學習指的是利用以規則、以目標、統計方法等方式， 紀錄玩家的行動資料並分析玩家的行為模式進而修改電腦角色的行為規則來反應玩家的策略 [9,10]。直接學習指的是利用具有學習概念的演算法讓遊戲角色具有自主的學習能力(active learning) 。通常會將角色的行為以參數化方式表現， 再利用最佳化演算法或回饋式學習(reinforcement learning)，搜尋最佳的參數組合(也就是行為)。類神經網路(neural networks)與基因演算法(genetic algorithms)是目前最被產學界使用的學習演算法[11-13] 。

綜合以上所述， 以規則為基礎的系統是最簡單的一種方法， 容易瞭解、建立與除錯，不過卻也是一種沒有效率的方法，AI設計師不容易能夠設計出一套完美（如：動態、智慧、擬真、合理或具有遊戲性）的行為規則。以目標為基礎的系統則是一種較高階的方法，不用訂定煩雜的行為規則，只要告知角色的目標與達到目標的各種可行方法，不過設計師卻必需要有相當充足的背景知識（包括了認知科學與遊戲本身），才能夠發展出適當並具有智慧的行為。間接學習方法的優點在於，統計資料的收集非常容易，缺點則是設計階段必須小心的決定要收集什麼樣的統計資料，又統計資料要如何與角色的行為對應。直接學習系統（動態決定角色在什麼樣的情況下做出什麼樣的行為）最大的優點在於角色能夠學習與適應環境、能夠即時創造出各式各樣意想不到的行為， 因此可以讓遊戲具有更高的耐玩性，不過行為的不穩定性也造成了測試程序的困難，目前直接學習系統仍難以進行遊戲中的學習(in-game learning) ，也就是遊戲角色藉由與玩家和與虛擬環境的互動，動態與即時的調整角色的行為模式。

此外， 依據我們的觀察與遊戲經驗， 現今遊戲中的角色具有學習的能力相當少見，也就是說 NPC 遊戲角色無法藉由與玩家或是與虛擬環境的互動而改變行動的準則，這是與由玩家所控制的角色之最大不同點。本文以人工智慧領域中的演化計算技術具有解決上述問題的能力，因此本文的目標就是希望利用演化計算技術創造電腦遊戲中的角色，包括(1)讓遊戲人工智慧設計師能夠更方便的創造智慧、擬真與具有學習能力的角色行為，(2)讓遊戲玩家藉由參與角色最佳化的過程，感受到角色行為的成長性、適應性與不可預測性。

二、粒子群最佳化

(一) 群體智慧

在自然界， 生物展現不可思議的社會行

為與高度的智慧， 利如螞蟻群落， 候鳥遷徙與覓食， 魚群逃避補食者。這些行為簡單個體所組成的群落中，沒有領導者更沒有集中式的管理， 靠的只是個體和與其他個體間互動、個體與環境互動的規則[14] 。

在群體智慧的論點中， 強調的並非探究個體的複雜構造， 而是認為複雜的系統、智慧的行為能夠藉由個體間的互動而浮現(emergent) 。和傳統人工智慧方法與集中管理系統的比較，具有自我組織的能力是群體智慧最大的特色，在這樣的系統中錯誤與隨機的行為並不是程式的”臭蟲(bug)“，反而是促使系統更進一步的探索新方法的動力，系統如此將更具有彈性、適應環境、有強韌性，並允許小部份個體的失敗。因此群體智慧研究的重點在於如何設計出個體間適當的互動規則，讓系統浮現出適當並符合需求的整體性行為。蟻群最佳化 (Ant Colony Optimization, ACO)[15] 、文化演算法(cultural algorithms)[16] 與粒子群最佳化[17] ，都是屬於具有群體智慧概念的計算方法。

（二）粒子群最佳化演算法

粒子群最佳化演算法(以下簡稱 PSO)是由 J. Kennedy 與 R.C. Eberhart 於 1995 年所提出， 是一種具有群體智慧概念，源自於人工生命(artificial life) 與社會心理學的研究[17]。它所關注的是如何利用自然界生物演化的機制，也就是達爾文“物競天擇，適者生存”的演化發展的計算方法。與基因演算法(Genetic algorithm, GA) 相較，PSO 的優勢在於概念簡單、實作容易、參數調整少， 並且在許多的情況下所有的粒子可能更快的收斂於最佳解[18]。

在電腦動畫界，Reynolds[19]僅僅使用簡單的幾條規則就模擬出相當真實的鳥類群聚飛行動畫。後來 Heppner 的鳥類群聚模擬[20]大致上與 Reynolds 的模型類似， 不過卻加入了一個不一樣的特點， 他的鳥群會受到棲息地的吸引， 也就是在模擬的開始， 鳥會逐漸形成群體並且以無特定方向在空中飛行， 直到有一隻飛越了棲息地，並且受到了棲息地的牽引，那麼其它的同伴將同時受到鄰近夥伴與棲息地的影響，逐漸地降落在棲息地， 這二股力量將決定個體是傾向繼續搜尋更佳的棲息地或者是傾向滿足現有的棲息地，這樣的概念讓他們覺得這與搜尋最佳解的問題有關連。Boyd 與 Richerson[21]的個體學習與文化傳承理論也支持了 PSO 的論點，他們認為人們在做決策的過程中會利用到二項重要的資訊， 一項為個體所擁有的經驗。另一項則是他人的經驗， 個體可以知道在他的周圍有誰做過什麼樣的努力，那個人的成果是最好的。

（三）方法

Kennedy 與 Eberhart 發展了一套 PSO 模型用來解決最佳化的問題，在一個最佳化問題的解就像是一隻在空間中飛行的鳥一樣，他們稱作“粒子(particle)” ，在空間中移動的所有粒子都有一個由適應函式(fitness function)所決定的適應值， 另外每個粒子還有一個速度來決定他們移動的方向與距離， 一群粒子靠著追隨本身成功經驗與目前最佳粒子的腳步在解空間中飛行。

在初始階段，PSO 會隨機產生一組粒子（包含位置與速度），然後透過一次次的疊代找尋最佳解， 在每次的疊代過程中每個粒子利用二個“最佳值”來更新自己的速度， 一個是粒子本身所找到的最佳解， 稱為”個體最佳值(pbest)”， 另一個最佳值是由 PSO 所記錄與更新，為全體粒子所找到的最佳解，稱為”全體最佳值(gbest)” 。當然，若一個粒子所能受引響的範圍是區域性的，這裡的最佳解就要改稱為”區域最佳值(lbest)” 。粒子便利用下面這個式子來更新粒子的速度：

$$v_i^{t+1} = wv_i^t + c_1 rand_1 \times (pbest_i - s_i^t) + c_2 rand_2 \times (gbest - s_i^t) \quad (1)$$

其中v_i^k代表粒子i在第k次疊代中的速度；w是權重函式；c_1與c_2是學習因子（通常，c_1=c_2）；$rand$是一個介於 0~1 之間的隨機值；s_i^k代表粒子i在第k次疊代中的位置；$pbest_i$是粒子i的最佳值；$gbest$是全體的最佳值。上述式子所用到的權重函式：

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} \times iter \quad , \quad (2)$$

式子(2)中，$w$max 代表初始權重，$w$min 代表最終權重，$iter$max 是最大的疊代數，$iter$ 則代表目前的疊代次數，w 的值將會隨疊代次數的增加越來越小，這表示 PSO 希望在最佳化開始傾向讓粒子更積極地搜尋更佳的問題解，也就是“探索(exploration)”， 而隨著疊代次數的增加， 粒子則會傾向滿足現有的最佳解，也就是”利用(exploitation) 。每個粒子在第 $k+1$ 次疊代中的位置（粒子在解空間中移動之示意如圖 2），可以簡單的利用下面的式子來更新：

$$s_i^{k+1} = s_i^k + v_i^{k+1} \quad (3)$$

粒子群透過不停的疊代逐漸收斂並找到問題的近似最佳解， 此時大部分的粒子會處在一個相似的狀態並提出類似的問題解。

三、以粒子群最佳化為基礎之遊戲角色行為設計

我們希望解決在規則系統中，AI 設計師如何

有效率的設計出一套完美的行為規則，亦

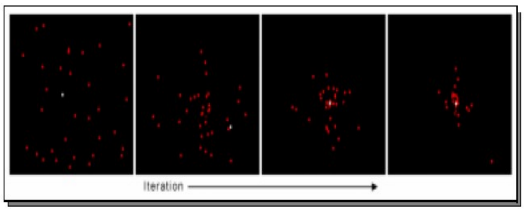


圖 2、粒子群搜尋最佳解過程

即角色在什麼樣的情況下做出什麼樣的行為是比較好的;學習系統中，行為的不穩定性，包含有學習緩慢、產生愚笨無行為能力的角色，而造成測試程序困難的問題; 與難以進行遊戲中學習的問題。如何讓遊戲之角色具有學習能力， 並且具有穩定學習之能力。

角色行為機制之設計， 我們以模糊狀態機實作之規則系統再加上學習演算法是一種兼具有效率與效果的方法，以模糊狀態機實作之規則系統是目前遊戲業界相當熱門的 AI 實作技術，而學習演算法則是深受遊戲界期待，能夠發展具出有學習能力的遊戲角色之技術。整個系統設計的流程首先由 AI 設計師根據腳本訂定出一組角色所有可以執行的行動，這些角色可執行的所有行動可以使用階層式的方式來歸類， 也就是行動可以歸納為策略、戰術與基礎行動， 或是長期、中期與短期行動這幾種層次。這些行動背後都跟隨著一組行為準則(behavior rules) ，就是什麼樣的條件下會觸發這個行為。每個行動背後的行為準則是由設計人員事先所訂定，而直接學習技術則在動態產生這些行為準則，雖然這種方法可以產生真正的學習行為，不過當遊戲環境複雜、動態的因子過多時，直接學習技術會變得不切實際，不僅無法進行即時的學習、更容易造成行為的無法控制。以規則為基礎之系統則可以利用經驗法則對遊戲角色的行為做基本的行為設計，而具有模糊地帶的行為判斷則可以加入權重的概念，我們認為這是比較實際的作法。

在遊戲的設計中，模糊關連的定義與模糊權重的決定，可以利用簡單的常識、經驗或專業的遊戲知識來設計， 不過當條件相互關係複雜，權重值就無法輕易決定，特別是權重間會相互引響的情況，這往往需要反覆的測試與調整。為了減輕 AI 設計師的負擔，需要一個自動化的方法來縮短權重調整的過程，在這裡我們所用到的就是粒子群最佳化技術。權重的選擇我們可以直接拿歸屬函數中較具關鍵性的葉節點權重，或是更高階的在行為之上再加上一個權重來決定行為展現的頻率。所以影響電腦角色該選擇何種行為的依據包括了二個權重，一個是由各項條件所決定的權重，一個則是可以籍由動態調整代表行為展現頻率的權重。

以圖 3 為例，由 AI 設計師設計出組規則包含 Behavior A, B, C, D，以及觸發執行這些行為與策略的各種 Behavior rule。依據模糊邏輯的概念， 這些影響某行為的狀態會動態的計算出權重（圖 3 中 WeightA-1, WeightB-1…）來決定這個行為或策略是否該被執行（0~1 之間，值越大被執行的機率越高），這是影響行為的第一個權重。接下來系統將這些有限數量之行為的權重編碼，並透過 PSO 最佳化演算的程序找到行為的第二，個權重（圖 3 中之 WeightA-2, WeightB-2…）這些權重的集合即代表了 PSO 公式中的 S：

$$S = (\text{Weight A-2, Weight B-2, Weight C-2, WeightD-2}) \quad (4)$$

PSO 最佳化的程序包括相關參數的設定、隨機初始權重、評估遊戲角色在遊戲中的表現、依據表現結果調整行為權重等(權重調整流程如圖 4)。

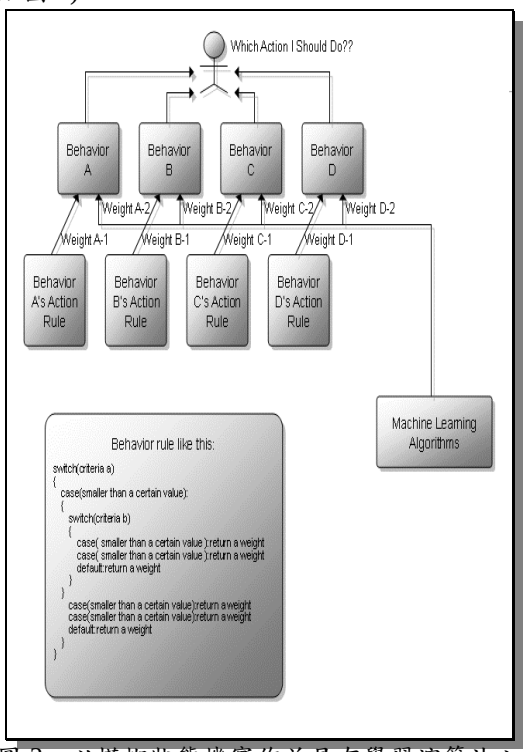


圖 3、以模糊狀態機實作並具有學習演算法之角色行為設計機制

本系統在實作上包含建立角色行為之平台（遊戲環境）、設計角色行為之機制與角色行為之調整。本研究牽涉到遊戲角色行為之設計， 因此需有一個讓角色展現行為的環境，以驗證實作結果， 我們評估， 利用開放程式碼與模擬的方式可以讓我們獲得所要的遊戲環境。

(一)QUAKE III 之 bot 行為最佳化

id Software 的雷神之鎚 3(QUAKE III:

Arena)[22]是我們所選擇的開放程式碼。QUAKE 是一款第一人稱動作射擊遊戲， 遊戲中玩家在不同場地， 使用不同武器、道具與策略擊倒 NPC 對手（稱作 bot） 或網路玩家以獲得射擊的樂趣。bot 一詞源自 robot，通常指的是第一人稱動作射擊遊戲中由電腦所控制的 AI NPC 角色。我們所要設計的即是 QUAKE 的 bot，圖 5 是 QUAKE 的 AI 運作機制示意圖。

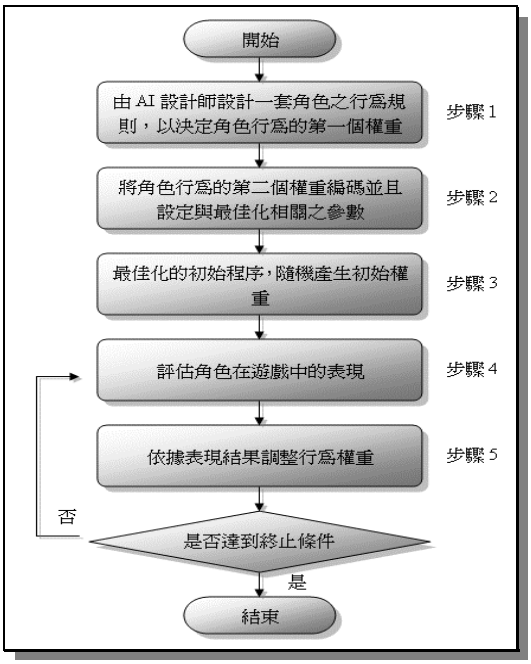


圖 4、角色行為最佳化之系統流程圖

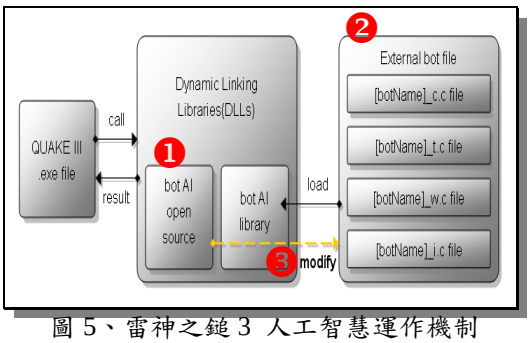


圖 5、雷神之鎚 3 人工智慧運作機制

四、QUAKE III 之 bot 與侏儒巨人(troll)行為最佳化

bot AI 運作的方式為，由 bot AI 開放程式碼(bot AI open source ，即圖 6 中❶與 bot AI 函式庫(bot AI library ，即圖 6 中❸)經由編譯與連結形成動態連結函式庫(DLL)並由遊戲主程式於遊戲期間呼叫。而外部 bot 檔案(external bot files，即圖 6 中❷)則會在每個 bot 加入遊戲之前由 bot AI 函式庫中的程式所負責載入。圖 5 中❶與❷是 QUAKE 所開放能夠讓我們所修改的部份。bot AI 函式庫所負責的是較低階的 AI 工作，包括外部 bot 檔案的載入、bot 的感知與動作展現以及部份的行為控制。bot AI 開放程式碼則負責大部份的 bot 行為控制、長期與短期目標的決定、行為規則的定訂等， 使用到了模糊狀態機的概念， 在長期與短期目標的決定上， 會根據遊戲狀態以決定目標與行為是否符合執行的條件，並且根據外部 bot 檔案中關於 bot 行為特色的權重與一個隨機值來決定是否訂出目標與做出行動。在武器與物品的選擇與取用上， 模糊狀態機會根據狀態與經驗法則動態決定一組權重，並且與外部 bot 檔案中關於 bot 行為特色的權重決定來決定角色的行為。

為了套用最佳化技術， 首先我們必須決定問題解該如何表示， 外部bot 以四個檔案為一組描述一個bot 的各項能力、行為偏好、武器與物品取用之權重等， 這樣的設計最主要是方便AI 設計人員調整與測試角色的行為，不過接近無限的行為權重排列組合是需要長時間嘗試、測試與調整的。此外在外部bot 檔案中包含了有bot 的各項能力描述例如射擊準確度、移動速度、反應時間等， 只要將這些數值簡單的調高（低）就可以產生出超人的角色， 不過這卻無關行為， 因此不會是我們要修改的部份。我們將會在bot AI 開放程式碼中加入適當的模組來評估角色在遊戲中的行為結果， 並且利用粒子群最佳化產生新的權重來修改外部檔案中關於bot 行為描述的部份（圖5中❸之虛線部份）。

[botName]_c.c 這個外部檔案主要是用來描述一個 bot 之各項能力值與行為偏好之權重，在這個檔案中與行為相關的權重包括了有 ，對戰時該採蹲伏(Croucher)、跳躍(Jumper)、該積極進攻(Aggression) 、採取防守(Selfpreservation) 、是否埋伏來狙擊對手(Camper)、是否恃強凌弱(Frag easy target) 、是否採取報復行為(Vengefulness,) 、武器切換的頻率(Weapon jumper) 等等，因此問題解的表示則為：(Croucher, Jumper, Weapon jumper, Aggression, Selfpreservation, Camper, Vengefulness, Frag easy target, ...)

這些影響行為的權重可以由 0 到 1 之間的實數表示， 將這些問題解編碼之後， 接下來就要決定引響粒子移動方向的適應函數。在適應函數的決定上我們採用下列的公式：

FitnessValue = 2 * num_kills - num_deaths (5)

也就是希望 bot 能夠擊倒越多的對手，並且盡量讓自己不被對手擊倒。適應函數會直接引響 bot 的行為表現， 而適應函數的選擇則要考慮到遊戲性的設計， 我們可以藉由不同的適應函數來設計出具有各種不同特色之電腦對手， 例如聰明、勇猛、膽小或難纏之

電腦對手。

將行為權重編碼與適應函數決定之後，接下來就可以進行最佳化程序， 這裡我們使用到前節所介紹的粒子群最佳化之標準版本。由一個粒子代表一組會影響 bot 行為之權重，將數個 PSO 的 bot(實驗中取三個 Harley)與數個 QUAKE 內建的 bot 放入競技場中互相對戰並且評估它們的對戰結果，與 PSO 的概念直接對應，每個 bot 的表現將會受到最佳 bot 與自身最佳表現的影響， 經由數代的演進之後， 我們會找到較佳的行為權重組合，圖 6 為 bot 行為最佳化之示意圖， 圖 7 則為實驗之相關設定與實驗結果。綠色線條代表整個族群的平均表現， 紅色線條則代表到目前為止的最佳表現， 從本實驗的結果可以發現，我們所產生之 bot 具有因學習而成長之能力，在八人自由對戰模式(free for all)以及地形龐大與複雜的場景(q3dm12: the dredwerk) 中經過二十多代的演進，我們所產生之 bot 普遍具有下列之行為， 對戰時僅量採取跳躍來閃避敵人的射擊、較不積極進攻並且僅可能採取防守以自我保護、僅量利用地形埋伏來狙擊對手、恃強凌弱、不採取報復之行為、武器切換的頻率高等等(場景如圖 8)。

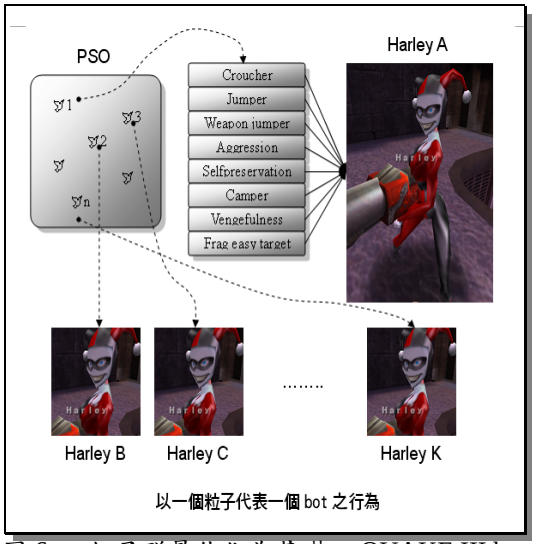


圖 6、以粒子群最佳化為基礎之 QUAKE III bot 之行為最佳化

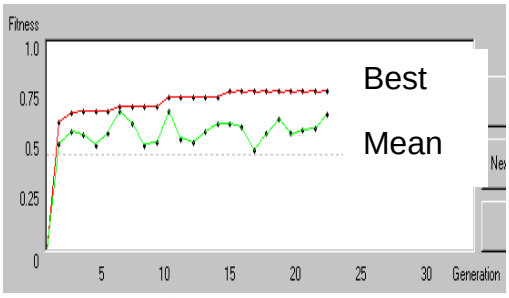


圖 7、以粒子群最佳化為基礎之 QUAKE III bot 之行為最佳化實驗結果

(二)侏儒巨人(troll) 行為最佳化

為了有客觀比較的依據，因此我們進行另外一項實驗。F. D. Laramee[12]在他的研究中展示了如何利用基因演算法發展出角色扮演遊戲(role playing game, RPG) 中遊戲角色的行為－為了偷吃羊而躲避重重障礙的侏儒巨人(troll)，為了演化他所創造的侏儒巨人， 他利用模擬的方式創造了一個遊戲環境， 並且將侏儒巨人放入這個環境中作演化，在模擬中侏儒巨人的目標就是偷更多的羊與殺死越多的冒險者（模擬玩家）並且能僅量的讓自己安然的處在這個世界上。

圖 8、自由對戰及地形龐大與複雜的場景

在這個實驗，我們採用的就是 F. D. Laramee 所創造的遊戲環境作為角色展現行為的平台， 利用粒子群最佳化演算法來演化侏儒巨人的行為。在模擬的遊戲環境中， 侏儒巨人是棲息在一個 30x30 方格大小的世界中(如圖 9)，每一個方格擁有下列任一元素：安全區域、陷阱、綿羊、騎士與人類碉堡， 每個不同的元素都擁有不同的特徵。侏儒巨人靠著與這些地圖上的元素互動來達成它的目的。為了讓侏儒巨人的行為不會過度的適應於某項環境， 因此每一次的測試都會在三種不同的環境中進行（充滿綿羊、充滿騎士與生物稀疏的世界），並對這些結果平均來評估侏儒巨人的適應值。

不同於 QUAKE 實驗， 在演化完美之侏儒巨人實驗中我們利用一個粒子群來代表一個侏儒巨人之行為， 以一個粒子代表此侏儒巨人的一次嘗試， 將此侏儒巨人放入模擬環境中與其它的個體互動， 也就是說每一代的侏儒巨人會選擇 n 次不同的方法來嘗試， 每一次的嘗試會得到一個適應值， 最好的嘗試結果會成為下一代中各個嘗試的參考， 每一次的嘗試也會受到之前幾代中相對應嘗試中最好一次的引響， 過了幾代之後侏儒巨人將會找到符合遊戲目標的較好策略，這與 PSO 的概念是間接的對應。

為了達到前述侏儒巨人的目標， 在模擬中的每個回合侏儒巨人能夠作出五項決策與行動， 包括了有： 吃羊、殺死冒險者、逃離危險、療傷與探索這個世界。每項決策被選擇的機率會取決與當時遊戲世界之條件， 舉個例子來說， 隨著侏儒巨人的飢餓程度越高(一

段時間沒有吃羊了)，決定追捕迷失小羊的機會就會越高。當侏儒巨人受傷了，它會選擇到安全的地方而不會在人類碉堡附近來進行療傷。每項行動背後都連結著一個決策函式， 因此如果侏儒巨人決定進行吃羊這個行動， 它會看看週遭是否有落單的羊可以讓它在一此的回合中被補獲，若沒有，那它就會先進行探索世界的這個行為。

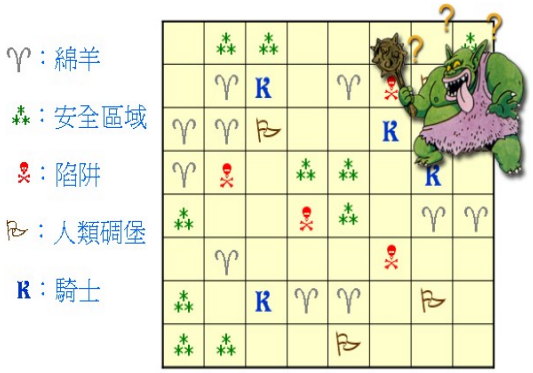


圖 9、侏儒巨人棲息世界

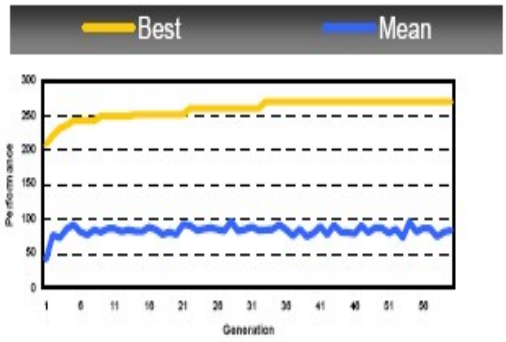
每項決策被選擇的機率除了取決與當時遊戲世界之條件之外還會乘上一個權重， 這個權重就是最佳化演算法中所要動態決定的值， 為一個 0~1 之間的實數， 每項決策都會分別與一個權重相連結， 因此在這裡問題解的表示則分別為， 吃羊、療傷、擊倒騎士、逃跑、與探索行為的優先順序：
(Eating priority, Healing priority, Killing priority, Fleeing priority, Exploring priority)

在適應函數的決定上， 也就是對於侏儒巨人行為的評估上， 我們仍然採用 F. D. Laramee 的設定， 他希望他的侏儒巨人能夠展現出令人類厭惡與頭疼的行為， 因此對於偷越多羊與殺死越多騎士的侏儒巨人會給它更高的分數， 此外也希望侏儒巨人在這個世界上能夠活的越久以及受越少的傷害，下面列出這個適應函數：

FitnessValue = 8* KnightKilled + 10 * SheepEaten + 1.5 * TimeAlive– Time-Captive – 2.5 * DamageTaken (6)

問題解編碼與適應函數決定之後， 接下來就可以進行最佳化程序， 這裡我們使用到前面所介紹的粒子群最佳化之標準版本，以及與之前實驗類似的概念。圖 10 為實驗之結果。我們發現實驗的結果不管在任何方面都比 F. D. Laramee 利用基因演算法演化遊戲角色行為之結果來的較為優異， 以下對實驗所產生之角色行為分別在學習效率與行為特色上與 F. D. Laramee 的研究作比較：

在行為產生之效率上，同樣以 300 個候選解同時在解空間中作移動並且經過 100 次疊代，在 Intel Pentium 4 2.0 的電腦上，整個模擬與行為產生的過程，基因演算法所需的時間為 690.2 秒，而粒子群最佳化演算法所花費的時間則為 530.9 秒。在行為展現的成效上我們以統計圖表來說明，Best 代表曾經出現過之最佳角色行為，Mean 則代表整個族群的平均表現。由以上的統計圖表我們可以發現， 粒子群最佳化演算法能夠較快的收斂於較好的結果。在角色行為表現之特色上，到了第 60 次的疊代我們的結果倒是相似，只是特色更加的強化。我們所產生之侏儒巨人到演化後期都具有類似下列之行為： 不懂得自我保護， 也就是都不長命； 在有限的生命中最常做的事就是拚命的偷羊吃； 僅可能的不與騎士發生衝突； 並不太積極的從事療傷與探索世界之行為； 為了避免落入陷阱中，僅量的不用進入安全區域。



(a)

(b)

圖 10、最佳化效能 (a)以基因演算法效能表現, (b)以粒子群最佳化最佳化的效能表現

五、結論

為了解決“遊戲角色行為訂定與調整困難”，”角色學習的不穩定性”與”難以進行遊戲中學習”之問題，本文提出以模糊狀態機實作的規則系統配合機器學習之技術來作為角色之行為機制，並且利用粒子群最佳化技術與不同之適應函式動態調整行為權重的值，讓角色具有學習的能力。經由適當的應用能夠解決電腦遊戲領域中關於角色行為設計上的

部分問題，並有良好的成果。這樣的方法可以輔助AI 設計師在設計階段更容易的為角色訂定行為規則， 或是在遊戲進行時讓玩家感受角色具有調整行為之學習能力

六、參考文獻

[1] NPD Group, http://www.npd.com

[2] S. Cass, "Mind Games: To beat the competition, video games are getting smarter, " IEEE SPECTRUM, http://www.spectrum.ieee.org/WEBONLY/publicfeature/dec02/mind.html, December 2002.

[3] S. Rabin, "AI Game Programming Wisdom: Preface, " *AI Game Programming Wisdom*, Charles River Media Press, pp. xi-xiv, 2002.

[4] J. Funge, X. Tu, and D. Terzopoulos, "Cognitive Modeling: Knowledge, Reasoning and Planning for Intelligent Characters, " *Proc. ACM SIGGRAPH'95*, pp. 47-54, 1995.

[5] T. Alexander, "An Optimized Fuzzy Logic rchitecture for Decision-Making, " *AI Game Programming Wisdom*, Charles River Media Press, pp. 367-374, 2002.

[6] M.Zarozinski, "Imploding Combinatorial Explosion in a Fuzzy System, " *Game Programming Gems 2*, Charles River Media Press, pp. 342-350, 2001.

[7] M. McCuskey, "Fuzzy Logic for Video Games," Game Programming Gems , Charles River Media Press, pp. 319-329, 2000.

[8] J. E. Laird and M. Lent, "Interactive Computer Games: Human-Level AI's Killer Application," *Proc. Nat'l Conf. A.I., AAAI Press*, pp. 1171-1178, 2000

[9] T. Alexander, "GoCap: Game Observation Capture," *AI Game Programming Wisdom*, Charles River Media Press, pp. 579-585, 2002.

[10] R. Evans, "Varieties of Learning, " *AIGame Programming Wisdom*, Charles River Media Press, pp. 567-578, 2002.

[11] T. E. Revello, "Generating War Game Strategies Using A Genetic Algorithm," *Proc. IEEE Evolutionary Computation'02*, pp. 1086-1091, 2002.

[12] F. D. Laramee, "Genetic Algorithms: Evolving the Perfect Troll," *AI Game Programming Wisdom*, Charles River Media Press, pp. 629-639, 2002.

[13] J. Manslow, "Using a Neural Network in a Game: A Concrete Example, " *Game Programming Gems 2*, Charles River Media Press, pp. 351-358, 2001.

[14] J. Kennedy, R. C. Eberhart, *Swarm Intelligence*, Morgan Kaufmann Press, 2001.

[15] M. Dorigo, "Optimization, Learning and Natural Algorithms," Ph.D.Thesis, Politecnico di Milano, Italy, 1992.

[16] R. G. Reynolds, William Sverdlik, "Problem Solving Using Cultural Algorithms, " I*nternational Conference on Evolutionary Computation*, pp. 645-650, 1994.

[17] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *Proc. IEEE International Conference on Neural Networks*, Vol. IV, pp. 1942-1948, 1995.

[18] X. Hu, "Particle Swarm Optimization Tutorials, " http://web.ics.purdue.edu/~hux/tutorials.shtml, 2002.

[19] C. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *Proc. ACM SIGGRAPH'87*, pp. 25-34, 1987.

[20] F. Heppner and U. Grenander, "A stochastic nonlinear model for coordinated bird flocks," In S. Krasner, Ed., The Ubiquity of Chaos. AAAS Publications, 1990.

[21] R. Boyd and P. J. Richerson, Culture and the Evolutionary Process, Chicago, The University of Chicago Press, 1985

[22] Quake III: Arena, id Software, www.idsoftware.com.