以一致的方式規劃自主式數位演員之多種運動能力
A Unified Approach to Planning the Motion of an Autonomous Digital Actor with Multiple Motion Abilities

李岳澤
國立政治大學資訊科學系
g9337@cs.nccu.edu.tw

李蔡彥
國立政治大學資訊科學系
li@nccu.edu.tw

摘要

讓電腦動畫中的數位演員在虛擬場景中自主的規劃路徑並且移動，一直是個富有挑戰性的問題。近年來已有相當多與此主題相關的研究，而數位演員也已能在高低起伏的地形中規劃可行的路徑。本研究的目的是希望數位演員能以更多的運動能力，在更一般化的場景中，以一致化的搜尋方式找出最適合的路徑。本研究為數位演員所擴充的新的能力包含了躍過障礙物的能力以及搬移障礙物的能力。在以往藉由搬移障礙物而抵達目的地的相關研究，都是以先決定欲搬移障礙物的順序，再為數位演員規劃出相對應的移動路徑；而本研究則是將搬移障礙物這項能力與數位演員其他的運動能力同時考量，將各種運動能力以一致的觀點來規劃數位演員的行走路徑。在本研究中我們已實做出一個數位演員的運動規劃系統，其初步實驗的成果將在本論文中呈現。

關鍵詞：運動計畫、數位演員、電腦動畫、搬移障礙物

1. 前言

在 3D 場景中，為自主式數位演員進行路徑規劃，一直都是具有挑戰性的議題。現行的方式大多由使用者透過鍵盤及滑鼠這樣的輸入介面來對數位演員進行操控，讓數位演員在場景中巡訪(Walkthrough)。除了由使用者自行透過操作介面來控制數位演員的移動之外，我們較為關心的是如何能夠透過指定起點與終點的方式，使得數位演員能夠自行規劃一條合理且可行的路徑。近年來，J. Kuffner[8]、Z. Shiller[16]與 Huang[11]各自提出了在較為複雜的場景中規劃路徑的能力，而本研究主要的目的便是將目前現有的人物路徑運動計畫加以延伸，著重於加強數位演員本身的能力，使其能處理更一般化的場景。

數位演員的能力大致可以分為移動(Movement)與操作(Manipulation)。不論是走、跑、跳躍這些能力，只要是能夠讓數位演員改變其目前位置的動作，都屬於移動的範疇；而搬移物品、抓取物品這類的問題則是歸類為操作方面。而在本研究中，我們嘗試將這兩種能力作進一步的結合，使得數位演員能夠在可以移動某些障礙物的情況下抵達目的地。

在過往的研究當中，我們可以發現搬動障礙物本身即是一個複雜的問題[20]。Stilman 及 Kuffner [18]以圖(Graph)來呈現由場景所簡化的狀態空間(statespace)，並且藉此決定障礙物被搬動的順序。當障礙物被搬動的順序決定之後，再由路徑計畫器規劃依序通過這些障礙物的路徑。以此方式所規劃出的結果，可以顯現出數位演員在搬動障礙物方面的能力，但也顯現出這項能力與其他數位演員的運動能力並不平等，總是先決定要依序搬動某些障礙物，再為數位演員規劃通過這些障礙物的路徑。這樣先入為主的規劃方式與一般生活情況並不相符；例如，面對面前的桌椅，我們可能會採取輕推桌椅而通過，也可能稍微側身即可通過，端看哪種行為較不費力。也就是說，在現實生活當中，我們能夠將數位演員所具備的各種能力放在同一座天平上面做考量。而這就是本研究所提出的以統一的觀點，嘗試將所有的運動能力做一致的評估，使數位演員的行為模式與現實生活中我們的思考模式更相近。

本論文接下來的架構如下：第二節我們會對運動計劃及其他相關研究作介紹。第三節為問題描述。在第四節會將一致化的計畫原則做更進一步的說明。第五節與第六節則是介紹我們新增的運動能力的設計。而在第七節介紹我們全域路徑規畫的演算法。第八節為實驗結果、最後為結論與未來延伸。

2. 相關研究

運動計畫的研究是屬於機器人學中幾何推理的範疇，最早起於鋼琴搬運問題[15]，目的是為可以移動的物體規劃出一條避免碰撞的路徑。此領域已有相當多的研究成果，Latombe 的著作[12]中詳細的定義運動計畫和相關演算法。根據[1]的定義，絕大多數的路徑計畫器都可以切成兩個步驟，首先是前處理(Preprocessing)，對輸入的資料做抽象化的資料型態(ADT，Abstract data structure)，接下來則可對這些抽象化的資料作查詢(Query)，並找出路徑。在[8][11]中提到將路徑計畫分兩部份來進行人體的運動計畫；先由一個路徑計畫器根據使用者輸入的

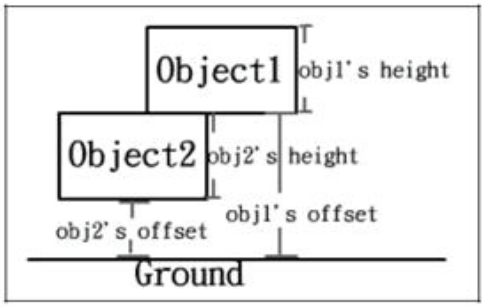


圖 1 物體的 Offset 與 Height 示意圖[10]

起點與終點規劃出一條可行的路徑，再由動作產生器(Locomotion Generator)來針對這條路徑產生相對應的動作。[9][10]提出了分層式的場景，縮減了 3D 場景的複雜度，但是延伸了 2D 場景在高度資訊的缺乏。[11]更進一步定義了數位演員在不同運動能力下的可抵達區域，讓數位演員的能力更進一步的延伸，並在場景中加入了不穩定區域(instability)的設計，使得數位演員的路徑選擇有更合理的模擬。

早期關於搬移障礙物的問題，多是將物體沿著指定的路徑移動；如[13][20]提出了以操作者為主體，將目標物體以推或拉的方式搬移到目的地。而[3][18]則以搬移障礙物為數位演員運動能力的一種，目的是為了使數位演員能夠排除障礙物而抵達指定的目的地。在搬動障礙物之前，使用者並不知道要搬移哪些障礙物，以及障礙物將會移動到何處，而是由系統根據使用者前往的目的地，而決定所需搬移的障礙物。由於此類問題的計算複雜度相當高，許多研究將問題限定於 2D 平面，並且只允許數位演員有單純的移動與搬動兩種運動。而在我們的研究當中，我們希望能提供數位演員更多樣化的運動能力，並使數位演員能在分層的場景中移動。

3. 問題描述

在本系統中，場景是由許多大小不等的多邊形所組成，每個多邊形有各自的高度(Height)和離地的高度(Offset)，如圖 1 所示。我們採用[10]中對於分層場景的定義來建構數位演員的工作空間。由於不同物體有不同的離地高度，此工作空間可被分割為多個相連的 Layer。我們在此空作工間為數位演員定義其可抵達的區域，並採用 NF2 的演算法[12]為其建構骨架與虛擬位能場。

我們為數位演員定義數種運動能力，包含正走、側走、搬移障礙物與跳躍，而跳躍又分為躍過溝渠等較為低窪的地勢及能夠翻越阻擋在數位演員面前的障礙物。前者是跳躍過一條水溝，而後者則是能夠翻越一個較為低矮的茶几或矮牆。這些多樣化的運動能力提供了數位演員更多樣化的路徑選擇。我們以用圓柱體來代表數位演員每一種動作所需的涵括容積(Bounding Volume)。而本研究中數位演員對每一種動作能力都預先定義了高度(height)、最大步伐(maximal gait size)、能踩踏的高度(step height)以及每一種運動的涵括容積。

4. 一致化的計畫原則

在本研究中，為擁有多樣化能力的數位演員規劃路徑，是以數位演員可抵達的區域作為路徑搜尋的空間。但是此搜尋空間並非恆久不變。由於本研究中數位演員具備著搬移障礙物的能力，使得數位演員有能力改變場景的狀態，所以我們的搜尋空間也是會跟著變動的。與其他搬移障礙物研究不同的是，我們將搬動障礙物與其他運動能力作統一的考量；也就是說，並不在一開始便預先決定要搬動哪些障礙物，而是在評估每個運動能力每一步的成本，選擇出對於使用者來說花費最小的方式來抵達終點。但是搬動的障礙物每一步都會造成場景的改變，而為了儲存每一步變動過後的場景資訊所需要的空間複雜度是以指數成長的。另外，障礙物的搬移運動，由於障礙物的自由度有 x 軸、y 軸與旋轉三個維度，因此搬移運動相較於數位演員的其他運動選擇所需花費的時間更為龐大許多。所以，我們採取了折衷的方案，使搬移運動從全域路徑規劃器獨立自成一個計畫運算實體（Planning Instance），初始參數包含觸發搬移運動當時的搜尋空間以及要搬移的目標障礙物。當數位演員在全域路徑規劃器中選擇搬動障礙物時，此運算實體會根據給定的時間預算執行固定時間的運算，並回報搬移成功與否，以避免數位演員因為選擇了搬移障礙物而花費太多時間，忽略了其他可能更直覺簡便的路徑。當運算實體在為此障礙物規劃搬移路徑時，數位演員依舊可以在原先的搜尋空間中嘗試其他運動能力；而當此運算實體為障礙物規劃出一條可行的搬移路徑時，搜尋空間才會隨之變動。

在本研究中，搬移障礙物這項運動能力並非數位演員唯一能採取的運動能力，是否搬開障礙物端視數位演員對於路徑品質的取捨。假使能輕巧的從旁邊繞過，就不需要費力搬動障礙物。我們在路徑規劃中，對於障礙物的搬移只是數位演員路徑選擇的其中一種，所以我們選擇採用多重搜尋空間的方式，同時保留被搬動之前的場景的狀態以及經過搬動的場景狀態，由數位演員的偏好或是路徑所花的能量來決定數位演員抵達終點的路徑。

當我們搬移越多的障礙物之後，我們將會有越多不同的搜尋空間，分別記錄著不同的場景狀態，而這些不同的搜尋空間都是同時存在的。數位演員以全域路徑規劃器中搜尋的優先權順序選擇拜訪及擴展的組態。

5. 翻越障礙物的運動能力

相較於[10][11]的研究，我們在本研究為數位演

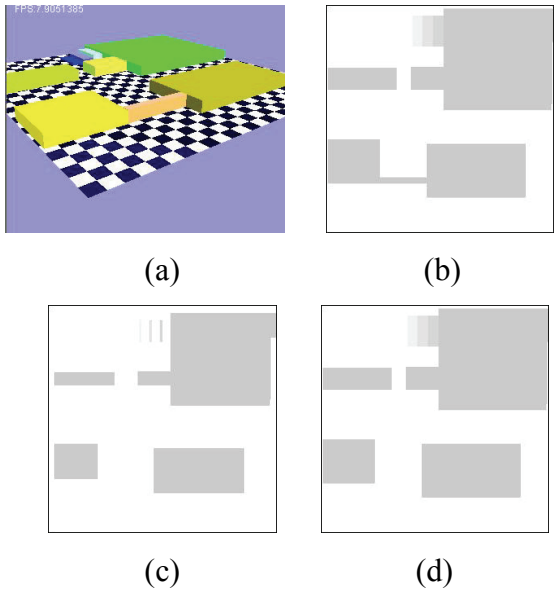


圖 2 對場景進行 Opening 的運算，(a)為場景俯視圖，首先對原場景(b)進行 Erosion 運算得到(c)，再進行 Dilation 運算產生(d)

員新增加了一個跳躍運動能力。擁有這項能力的數位演員，能夠經由手的支撐越過障礙物的阻擋，抵達障礙物的另一端。但是，並不是所有的障礙物都是可以被數位演員所越過的。在我們的定義中，數位演員只能選擇手能夠觸碰到的障礙物，並以手作為支撐躍過障礙物，而障礙物的選擇也限制障礙物的最高高度(H_{max})以及最低高度(H_{min})，數位演員只能翻越高度在最高高度與最低高度之間的障礙物。

除了上面的限制之外，數位演員還有跳躍距離的限制，並不是所有符合上述高度定義的障礙物都可以經由這項能力而躍過，所以我們必須先找出適合數位演員跳躍的區域。我們找出可跳躍的區域的方法是利用電腦視覺(Computer Vision) 中消除雜訊常用的方法 Opening[5]，將一圖形集合 A 與一個 Structure Element B 作 Opening 運算（記為 $\circ$），其定義如下：

$$A \circ B = (A \Theta B) \oplus B$$

Opening 的程序分成兩個步驟：Erosion 與 Dilation。我們所採用的 structure element 為一個以數位演員能躍過障礙物的最大步伐長度為直徑的圓；而圖形集合 A 則是工作空間中符合適當高度差的組態集合。Erosion($A \Theta B$)使場景原先連接在一起的阻隔障礙物分離，使得可以跳躍而通過的區域顯現出來；而 Dilation($A \oplus B$)則將圖形 A 中無法以跳躍通過的區域(障礙物寬度大過於數位演員的跳躍步伐)再度補上。

在圖 2 的例子中，依序對場景進行 Erosion 與 Dilation 運算，場景中適合跳躍的障礙物會在 Opening 的處理當中消失。對數位演員來說，這塊區域將不再是障礙物；而對系統來說，這塊區域並不是消失，而是被標註成是適合跳躍的，以便於我們在全域搜尋時的處理。

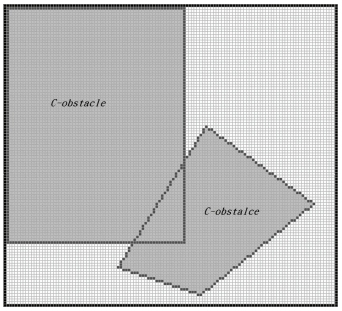


圖 3 在數位演員的組態空間中，此圖標示著兩個障礙物的 C-obstacle 區域。而 C-obstacle 的邊緣，我們定義為數位演員與障礙物的接觸點

6. 搬移障礙物的運動計畫

本研究為數位演員增加了搬動障礙物的能力，允許數位演員能改變工作空間中障礙物的位置組態，使得數位演員的路徑選擇更加多樣化。例如數位演員可以將擋在面前的桌椅搬開，找出一條原先無法通過的路徑。無論是現實生活或是在我們的研究中，並非所有的障礙物都是可以被搬動的。有些障礙物體積過於龐大難以被搬動，有些障礙物重量過重超出數位演員的負荷。而在我們的研究當中，障礙物是否可以被搬動是由使用者預先定義的。

為了避免搬動問題過於複雜，我們再為每一個障礙物分別定義一個旗標，標示著這個障礙物是否可以被搬移，當障礙物的搬移結束之後，則將其標示為不可搬移，以避免重複搬動的問題。當障礙物在搬移的過程中，我們也加入一些限制，例如在搬移的過程中障礙物不能與其他障礙物發生碰撞，並且確保障礙物不會違反物理學的運動。

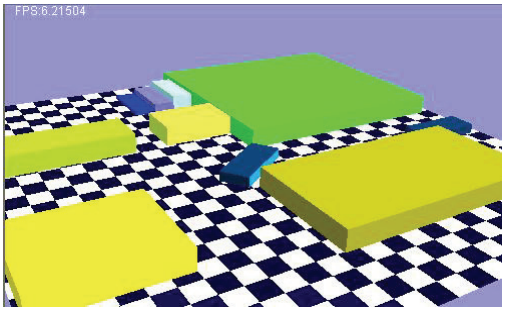
6.1 數位演員開始搬動障礙物的觸發點

當數位演員移動至障礙物周遭的組態，並且可觸碰到障礙物時，我們稱目前數位演員位於搬動此障礙物的觸發點(triggering point)。當數位演員抵達這些被標記為觸發點的組態，數位演員可以對包含此觸發點的障礙物進行搬移的運動。

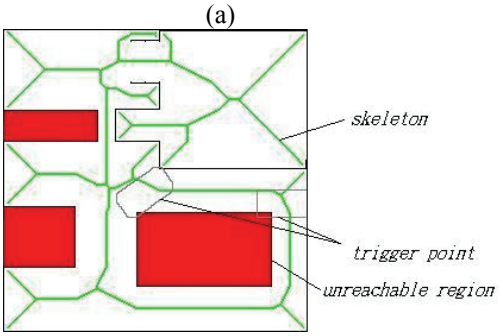
在圖 3 中，我們定義搬移運動的觸發點位於數位演員與障礙物之接觸點。如果此障礙物是不可被搬動的，那麼數位演員移動到黑色的點會發生碰撞；但是假使此障礙物是可以被搬動的，那麼數位演員則會觸發搬動的運動。

6.2 搬動障礙物時的工作空間

一般障礙物搬動的研究都是假設工作空間是一個 2D 的場景，這是因為搬動障礙物本身就是一個



(a)



(b)

圖 2　(a)為場景俯視圖，其中兩個深藍色的物體為可搬動的障礙物，(b)中綠色的線條是根據場景而產生的骨架，並加入了這兩個可搬移障礙物的搬移觸發點

非常複雜的計算，每個障礙物本身都包含著三個維度的運動：分別是對 2D 工作空間的 x 軸的位移、y 軸的位移以及障礙物本身的旋轉。而在多個障礙物的搬動過程中，在每一個障礙物搬動之後都會更新這個工作空間，若是 3D 的工作空間，會使得問題太過於複雜。而本研究採用的分層式場景，是屬於 2.5D 的工作空間，將 3D 場景的問題適度的簡化為一層一層的 2D 空間。但為了進一步簡化我們的問題，在我們的設計中，我們假設障礙物不能進行跨 Layer 的移動；也就是說，可搬移障礙物的移動只能在障礙物所屬的 Layer 中移動。

在搬動障礙物的過程中，我們必須確保障礙物在移動的路徑上皆未與其他障礙物發生碰撞 (collision-free)且障礙物本身有足夠的支撐空間。由於我們的障礙物並不會進行跨 Layer 的移動，我們的障礙物的離地高度也不會改變。我們只針對與此障礙物垂直高度有相交的障礙物做碰撞偵測。確保可搬移障礙物的支撐力量是為了防止障礙物在搬移的過程中，出現不穩定的狀態；例如，將桌上的一本書移動出桌子的範圍。在本研究中，當數位演員將障礙物搬移的時候，要確保障礙物移動不超出下層的支撐範圍。

6.3　搬動障礙物的運動計畫

我們採取的方式是先假設所有可移動障礙物都可以被搬移。在建置可抵達區域(Reachable Region)

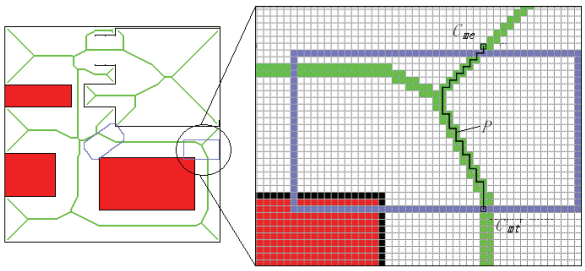


圖 5 當數位演員在全域路徑規劃器中搜尋到搬移觸發點時，為數位演員找出一條預估可通行的路徑

時，我們會暫時忽略此障礙物；也就是說，數位演員暫時不會把可搬移障礙物當成一個障礙物。我們以圖 3 (a)做為範例說明，場景中兩個可搬移的障礙物在我們建構可抵達區域的時候被忽略。如圖 4 **(b)** 所示，我們先在工作空間中建構出可抵達區域（不可到達區域以紅色表示），在根據此可抵達區域建立與 Voronoi Diagram 相類似的骨架(Skeleton)(以綠色表示）。我們進而再將可移動障礙物放回工作空間中，其與自由空間接觸的邊緣，即為可觸發搬移運動的觸發點(Triggering Point)。

在全域路徑規劃器中，當搜尋到可搬移障礙物所屬的觸發點時，即會觸發數位演員搬移障礙物的運動規劃。搬移過後，我們會將此障礙物標記為不可搬移的障礙物，以避免多次重複搬移的問題。不同於其他搬移障礙物的運動計劃的是，本研究是假設在沒有可移動障礙物的情況下，先計畫數位演員的通過路徑。一旦得到通過路徑後，再根據此路徑將可移動障礙物搬離此區域，使其不與路徑上的任一個組態發生碰撞。

如前所述，本研究是以統一的方式評量各種數位演員所擁有的多樣化能力，將搬移運動建立獨立於全域路徑規劃器之外的運算實體(instance)。此運算實體所做的工作，即是規劃障礙物如何移出我們規劃的預估路徑。我們將此搬移運動的處理分為三個步驟：預估數位演員路徑、建立障礙物的組態空間與搜尋搬移路徑的運算實體。以下我們將對此做進一步說明。

- **預估數位演員通過障礙物的路徑**

所謂「預估路徑」指的是指數位演員在沒有可搬移障礙物的阻擋下，數位演員會採取的路徑。而這段路徑 P 是從數位演員在全域路徑規劃器中遇到了觸發點 C_{mt} 起，穿越了欲搬移的可搬移障礙物(O_m)的區域，抵達一個不與任何障礙物碰撞的 C_{me} 組態結束。

為了簡化預估路徑的找尋，我們以數位演員自觸發點 C_{mt} 為起點，沿著 4.1 節中建立的骨架搜尋，直到搜尋到第一個不與任何障礙物發生碰撞的組態 C_{me}，如圖 5 所示。若是 C_{mt} 不在骨架上，則先連

```
STABLE_BFP()
1.  install q_i in T;
2.  INSERT(q_i,OPEN); mark q_i visited;
3.  SUCCESS <- false;
4.  while not EMPTY(OPEN) and not SUCCESS do
5.     q <-FIRST(OPEN);
6.     if q.state is Manip then
7.     Manip_Success<-q.Instance (t)
8.        if   Manip_Success    then
9.           q.searchspace= Create_Searchspace
       (q.Instance);
10.          q.state<-Non-Manip ;
11.       INSERT(q ,OPEN);
12.    else
13.    for every neighbor q' of q in the grid do
14.       if q'    is trigger point
15.          q'.Instance <-Establish_Instance();
16.          q'.state <- Manip ;
17.          Manip_Success <- false;
18.          mark q' is visited;
19.       if q' is stable
20.          mark q' is visited;
21.       if LEGAL( q',q) then
22.          install q' in T with a pointer toward q
23.          INSERT( q' ,OPEN);
24.       if q'=q_g then SUCCESS<-true;
25. if SUCCESS then
26.    return the backtracked feasible path
27. else return failure;
```

圖 6 為數位演員規劃路徑的 STABLE_BFP 演算

往骨架上水平距離與垂直距離相差最少的點，再連往 C_{me}。而這段從 C_{mt} 連至 C_{me} 的路徑，就是我們所求的路徑 P。

- **依據路徑建立欲搬移物體的組態空間**

在我們找到數位演員的預估路徑 P 之後，接著要將欲搬移的障礙物 O_m 搬移出 P 的區域，使得 O_m 不會在任何時間點與 P 上的數位演員發生碰撞。較直接的方法是在 O_m 搬出的過程中，持續與 P 上的每個數位演員可能的組態作碰撞偵測，根據障礙物的自由度，搬移運動的過程是一個以 O_m 為計畫主體，從目前的組態做三個維度的搜尋(x 軸、y 軸的平移與旋轉)。而在每一步搬移搜尋過程中，O_m 在新組態上必須與 P 的所有組態作碰撞偵測，以得知 O_m 是否已搬移出預估路徑 P，但這會使得這個搬移運動的時間複雜度升高。

為了簡化 O_m 搬出 P 的運算，我們採取的方法是為 O_m 建立一個組態空間 S，此組態空間明確紀錄所有 O_m 會與 P 上的數位演員組態發生碰撞的每一個組態。建立的方式是為 O_m 建立針對數位演員所計算的 *C-obstacle* 區域，再將此 *C-obstacle* 區域沿著預估路徑 P 填入 S 中。這些被填入的區域便代表若 O_m 位於此則會與預估路徑 P 發生碰撞，而當我們執行搬移運動運算時，只需要對此組態空間進行查表的工作即可得知是否有碰撞發生。

- **運算實體**

當數位演員依據路徑搜尋的優先權順序選擇了要搬動此障礙物後，我們由 O_m 的起始組態開始進行搜尋。我們為搬動障礙物的動作建立一個獨立的運算實體，而此操作實體會為 O_m 搜尋最短的搬移路徑。由於障礙物的自由度為三，因此此運算實體中的搜尋問題是在一個三維的空間中進行。此操作實體會開始找尋 O_m 在 S 中不會發生碰撞的組態，並確保 O_m 在移動路徑中的組態都不會與場景中其他障礙物發生碰撞，而且 O_m 的重心在路徑中都獲得適當的支撐。若是在 t 的時間內找到此一組態，此搜尋即結束，並且結束此一操作實體；但若是此搜尋執行超過了 t 的時間，此操作實體將會中止並保存，等待下一次數位演員再度依序優先權順序選擇要操作此障礙物的時候再度執行。因此，決定搬動某一障礙物的決定，並非在開始嘗試搬動障礙物時，就已經不能再改變了。反之，搬移運動的搜尋可以跟其他動作能力的搜尋一起進行。這種方式便是我們所稱的以「一致的方式」規劃數位演員的運動。

當我們為 O_m 成功地搬移了目標障礙物後，我們並不會改變數位演員目前的搜尋空間，而是為數位演員建立一個新的搜尋空間。此新的搜尋空間會將 O_m 搬移到我們為它規劃的新位置，並將 O_m 所佔據的 *C-obstacle* 區域標記為不可抵達的區域。

最後，我們將觸發此搬移運動的觸發點 C_{mt} 加入全域路徑規劃器的優先權佇列中，但不同的是 C_{mt} 所在的空間是我們新建立的搜尋空間。

7. 全域路徑規劃

在我們的數位演員運動計畫器中，由使用者輸入的資訊包含數位演員的各種運動能力、數位演員在運動學上的描述與限制或是數位演員對於路徑的偏好，以及場景中各個障礙物的資訊。我們將以上資訊進行前置處理，以建構數位演員的碰撞區域以及可抵達的區域。而使用者只需輸入數位演員的起始組態(q_{init})與終點組態(q_{goal})，全域運動計畫器便會規劃出一條符合所有限制與偏好的路徑。

我們以圖 6 中所示的 STABLE_BFP 演算法為數位演員規劃路徑。本系統在搜尋路徑時使用的是類似 Best-First Search 的演算法。我們首先將起始組態放入用來儲存候選組態的 OPEN 中，利用 FIRST 這個函數從 OPEN 中取出目前最好的組態(由虛擬位能場的值與數位演員所採取的運動能量做為衡量的標準)。我們會確認此組態是否為搬動狀態(Manip)，也就是此組態是否正在進行搬動障礙物的運動。若是，則給予此計算實體一固定時間 t，以執行搬移運動的運算，無論成功或失敗均將此組態

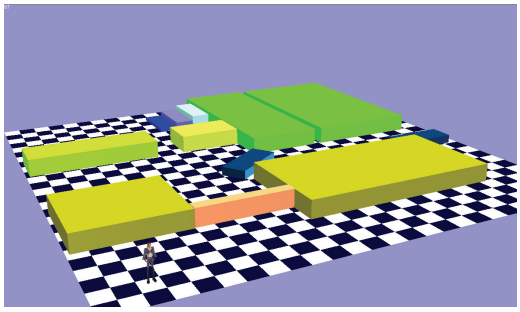
```
LEGAL(q',q)
1.  STABLE( q',q)
2.  if q' is visited or forbidden then
3.     return false;
4.  if q' is gap and    q'.gapcnt>N then
5.     return false;
6.  if q' is border and q'.bordercnt>M then
7.     return false;
8.  if q' is stable and q'.ingap then
9.     check distance between gap_begin and
    gap_end;
10.    if the distance is large than gait size, then
11.    return false;
12. if q' is jumpable and q'.jumpcnt>J then
13.    return false
14. return true;

STABLE(q',q)
1.  if q' is border then
2.     q'.bordercnt = q'.bordercnt +1;
3.  else if q' is gap then
4.     if q'.ingap    then
5.        q'.gapcnt= q'.gapcnt +1;
6.     else
7.        q'.gapcnt =1;
8.        q'.ingap =true;
9.        gap_begin=q' ;
10. else if q' is stable and q.ingap then
11.    gap_end =q';
12. else if q' is stable and q is jumpable then
13.    q'.jumpcnt =0;
14. else if q' is jumpable then
15.    if q is jumpable and q'.jumpcnt>0 then
16.       q'.jumpcnt = q'.jumpcnt +1;
17.    else
18.       check height difference between q' and q;
19.       if the height difference allows jump then
20.          q'.jumpcnt =1;
```

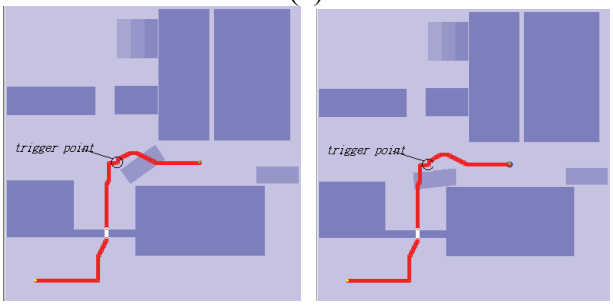
圖 7　STABLE 與 LEGAL 的檢查

存回 OPEN 中。如果搬移運動成功，則將此組態標記為非搬動狀態(Non-manip)，並為其建立一個新的搜尋空間。若此組態為非搬動狀態，我們依序拜訪此組態所有鄰接組態。若抵達搬移運動的觸發點，我們為其建立搬移操作實體並將此組態標記為搬動狀態。若是此組態於 instability 圖中標記為 stable，則我們將此組態標記為已拜訪，再由 LEGAL 檢查驗證此組態是否符合穩定合法狀態的條件。

我們根據[10]中對不穩定區域的定義，更進一步將不穩定區域區分為 border(在樓梯邊緣的不穩定狀態)、gap(在跳躍越過低窪地型時的不穩定狀態)以及 jumpable(在翻越障礙物時的不穩定狀態)。STABLE 與 LEGAL 的檢查函式如圖 7。當一個組態尚未被拜訪過、無碰撞(collision-free)、且為暫時穩定狀態，則我們稱這個組態為*合法組態*。所謂暫



(a)



(b)　(c)

圖 8 路徑規劃結果範例。(a)為數位演員所處的場景，(b)為規劃出的路徑，(c)為搬移後的場景。

時穩定狀態指的是數位演員在 border、gap、jumpable 這些不穩定區域停留還沒有超過限定的時間的狀態。我們在這些區域中以計數器檢查數位演員是否在此區域停留超過限定的時間。而在 LEGAL 檢查中的 M、N、J 分別是根據數位演員的能力而定出在 border、gap、jumpable 這些不穩定區域所能停留的最長時間。當該組態通過 LEGAL 的檢驗後，我們會將此合法組態放入 OPEN 中並繼續搜尋，直到從 OPEN 取出的組態與終點目標組態相同為止。

8. 實驗結果

本系統實作上都是使用 Java 語言，實驗的平台為 Intel 1.66G，1024MB RAM 的個人電腦，所有 map 的解析度皆為 256*256。

圖 8 為一個搜尋結果的範例。使用者指定位於左下角的起點以及位於場景中央的終點，由系統為數位演員規劃出的路徑如圖中紅色線條所示。根據圖 8 的場景，將 3D 場景劃分為分層式場景所花費的時間為 31ms，建構可抵達區域花費 63ms。為數位演員依據 NF2 演算法建構骨架以及虛擬位能場所花費的時間為 78ms。

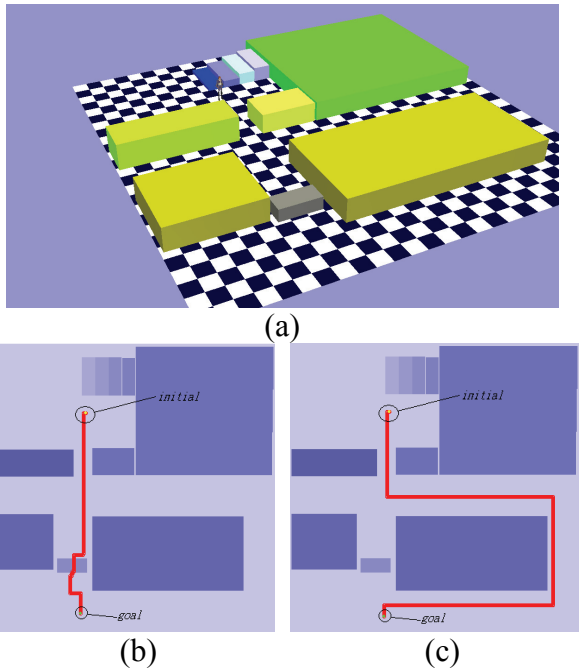
此搜尋所得的路徑及數位演員所採用的動作如下。數位演員先翻越過前方的低牆，當全域路徑規劃器搜尋到可搬移障礙物的觸發點時，會為此障礙物規劃搬移的路徑。圖 8(c)就是障礙物經過搬移之後的位置。在搬移過後的新場景中繼續搜尋而找到終點。而當數位演員搜尋到可搬移障礙物的觸發點時，會根據預估路徑而建立一個為了計算搬移運動

的計算實體。在此範例中，搬移此障礙物的預估路徑長度為 65 個節點，建立此操作實體的工作空間則花了 219ms。這段路徑搜尋所花費的總時間為 360ms，尚能滿足線上即時計算的要求。

圖 9 為在另一個實驗場景中搜尋的範例，在此場景的前置處理中，轉為分層式場景花費 31ms，建構可抵達區域花費 47ms，而建構 NF2 骨架以及虛擬位能場則花費 109ms。

在圖 9(b)的設計中，我們假設數位演員對於選擇一般行走與搬移障礙物的運動有相同的傾向與喜好，於是數位演員採取了距離較短的方式，將擋在前方的障礙物搬移而抵達目的地。總搜尋時間為 250ms，建立搬移障礙物計算實體的工作空間花費了 156ms。

而在圖 9(c)中，我們假設數位演員較傾向於一般行走而多過於搬移障礙物，也就是說對數位演員而言，搬移障礙物所花費的能量是遠高於一般行走運動的，於是數位演員採取了繞遠路的路徑以抵達目的地。總搜尋時間 235ms，建立搬移障礙物計算實體的工作空間花費了 156ms。由於我們對各種運動所消耗的能量以一致的方式評估，所以數位演員可以根據不同的使用者偏好來選擇對數位演員來說花費最低能量的路徑。



(a) (b) (c)

圖 9 為另一路徑規劃結果範例。(a)為數位演員所處的場景，(b) (c)分別為根據不同權重所規劃的路徑。

9. 結論與未來延伸

本研究的目標是在虛擬環境的場景中指定終點與起點，希望系統能為數位演員自動產生一段自主移動的路徑。我們期望能夠在複雜的場景之中，為數位演員增加更多克服場景障礙的能力。目前本研究已為分層場景中的數位演員，加入跳躍以及搬移障礙物兩項運動能力，並套用在全域路徑規劃器中。我們將搬移障礙物的處理獨立為一個計算實體，使其可以將搬移搜尋的過程切割為數個階段執行，也成功地使數位演員在搬移運動與其他運動的選擇上更為靈活有彈性。

由實驗結果可知，雖然我們已為數位演員計畫在路徑中的各種運動能力，但是搬移障礙物所花費的時間，仍佔據總搜尋時間很大的比例。此成本遠高於數位演員規劃其他運動能力。針對搬移障礙物本身的規劃做進一步的改良，也將是我們未來繼續研究的方向。

致謝

本論文作者感謝國科會計畫（計畫編號：NSC94-2213-E-004-006）的補助。

參考文獻

[1] J. Barraquand, L. Kavraki, J.C. Latombe T.Y. Li, and P. Raghavan, “A Random Sampling Scheme for Path Planning,” *Intl. J. of Robotics Research,* 16(6):759-774, Dec. 1997.

[2] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, Second Edition, Chapter 7: Voronoi Diagrams, 2000.

[3] P. C. Chen and Y. K. Hwang, “Practical path planning among movable obstacles,” in *Proc. of IEEE Intl. Conf. on Robotic Automation*, pp. 444-449, 1991.

[4] P. F. Chen, and T. Y. Li, “Generating Humanoid Lower-Body Motions with Real-Time Planning,” in *Proceeding of 2002 Computer Graphics Workshop*, Taiwan, 2002.

[5] E. D. Demaine, M. L. Demaine, J. O'Rourke, “PushPush and Push-1 are NP-hard in 2D,” in *Proc. of the 12th Annual Canadian Conference on Computational Geometry*, pp. 211-219, August 2000.

[6] R. C. Gonzale and R. E. Woods, *Digital Image Processing*, Second Edition, Prentice Hall, pp.224-244, 1985.

[7] M. Lau and J. Kuffner. “Behavior Planning for Character Animation,” in *Proc. of ACM SIGGRAPH / Eurographics Symp. on Computer Animation*, Los Angeles, CA, 2005.

[8] J. Kuffner, “Goal-Directed Navigation for Animated Characters Using Real-time Path Planning and Control” in *Proc. of CAPTECH’98 Workshop on Modeling and Motion capture Techniques for Virtual Environments*, Springer-Verlag, 1998.

[9] T.Y. Li and P.Z. Huang, “Motion Planning for a

Humanoid Walking in a 3D Space," in *Proc. of the 2001 National Computer Symp.*, Taipei, Taiwan, 2001.

[10] T.Y. Li, and P.Z. Huang, "Planning Humanoid Gross Motions on a Layered Scene," in *Proc. of 2002 Computer Graphics Workshop*, Taiwan, 2002.

[11] T.Y. Li and P.Z. Huang, "Planning Humanoid Motions with Striding Ability in a Virtual Environment," in *Proc. of the 2004 IEEE Intl. Conference on Robotics & Automation*, pp. 3195–3200, 2004.

[12] J. Latombe, *Robot Motion Planning,* Kluwer Academic Publishers, 1991.

[13] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability, and planning," *Int. Journal of Robotics Research*, 15(6):533–556, 1996.

[14] M.T. Mason. *Mechanics of Robotic Manipulation*. MIT Press, 2001.

[15] J. Schwartz, M. Sharir, "On the 'piano movers' problem II, General techniques for computing topological properties of real algebraic manifolds," *Advances in Applied Mathematics*, 4:298--351, 1983

[16] Z. Shiller, K. Yamane, Y. Nakamura, "Planning Motion Patterns of Human Figures Using a Multi-Layered Grid and the Dynamics Filter" in *Proc. of 2001 IEEE Intl. Conf. on Robotics and Automation,* pp.1-8, May 2001.

[17] S. Y. Shin and T. Kunii, "Pseudo Dynamic Keyframe Animation with Motion Blending on the Configuration Space of a Moving Mechanism," in *Proc. Of Pacific Graphics*, August 1995.

[18] M. Stilman and J. Kuffner, "Navigation Among Movable Obstacles: Real-time Reasoning in Complex Environments," in *Proc. IEEE Int. Conf. on Humanoid Robotics (Humanoids'04),* 2004.

[19] A. Witkin, and Z. Popovic, "Motion Warping," in *Proc. of ACM SIGGRAPH*, 1995.

[20] G. Wilfong, "Motion Planning in the Presence of Movable Obstacles," in *Proc. of ACM Symp. Computational* Geometry, pp. 279–288, 1988

[21] H. C. Sun and N. M. Dimitris, "Automating Gait Generation," in *Proc. of ACM SIGGRAPH*, 2001.

[22] S. K. Chung and J. K. Hahn, "Animation of Human Walking in Virtual Environments," in *Proc. of Computer Animation Conf.*, 1999.

[23] A. Bruderlin and T. W. Calvert, "Goal-Directed, Dynamic Animation of Human Walking," in *Proc. of ACM SIGGRAPH*, 1989