國防大學管理學院資訊管理學系碩士班

碩士論文

應用方向式蟻群啟發演算法於動態環境之自走機器人路徑規劃

The Directional Ant Colony Heuristic System for Mobile Robot Path Planning in Dynamic Environment

指導教授：劉豐豪博士

研 究 生：張冠璋　撰

中華民國一〇五年五月

國防大學管理學院碩士學位論文
指導教授推薦書

資訊管理學系 碩士班 張冠璋 君所提之論文

啟發式蟻群演算法結合角度因子
應用於動態環境中機器人路逕之研究規劃 (題目)

係由本人指導撰述，同意提付審查。

指導教授：劉豐豪 (簽名)

(簽名)

(簽名)

105 年 3 月 15 日

國防大學管理學院
學位論文考試委員會審定書

資訊管理學系 碩士班研究生 張冠璋

君，所提論文

(中文題目) 應用方向式蟻群啟發演算法於動態環境之自走機器人路徑規劃

(英文題目) The Directional Ant Colony Heuristic System for Mobile Robot Path Planning in Dynamic Environment

經本委員會審議，符合碩士學位標準，特此證明。

碩士學位論文考試委員會

委員 吳志明

楊毅豪

劉豐豪

指導教授 劉豐豪

共同指導教授

共同指導教授

系主任(所長) 傅振華

中 華 民 國 一〇五 年 五 月 二 日

The Directional Ant Colony Heuristic System for
Mobile Robot Path Planning in Dynamic Environment

By

Chang, Kuan-Chang

A THESIS
SUBMITTED TO THE DEPARTMENT OF
INFORMATION MANAGEMENT
MANAGEMENT COLLEGE
NATIONAL DEFENSE UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF BUSINESS ADMINISTRATION

Approved by：

Jin Yue Wu	Kang Chuan Chang
Tzay-Hao Lin	

Advisor： Tzay-Hao Lin

Chairperson： Fu, Chen-Hua

Date： 02/05/105 (DD/MM/YY)

致謝

「師者，所以傳道、授業、解惑也。」在我讀碩士的兩年間，豐豪老師用言行做了完美的詮釋，而我則在老師的詮釋過程中，學習到了做學問應有的態度和方法。感謝豐豪老師不僅僅是教導我知識，更是為我樹立了典範，在各方面都惠我良多。

其次我要感謝吳庭毓老師及楊顓豪老師在論文審定上給予很多寶貴的意見，特別感謝顓豪老師在演算法上對我的啟蒙，每次的討論都讓我獲益良多。同時我想感謝祥福學長。在學習的路上學長走的比我早也走得遠，在學習的思維及技巧上學長從不吝嗇指導，使我得以在既有的經驗基礎上繼續成長。

最後，我要感謝我的家人及意婷。因為有你們的支持及鼓勵，讓我在這個人生的階段獲得成長，我將以此榮耀獻給你們。

冠璋 謹致於

國防大學管理學院資訊管理學系

中華民國 一〇五 年 五 月

摘要

路徑規劃一直是移動式機器人核心問題之一，而運用啟發式演算法執行路徑規劃，使路境規劃求解效率更好、彈性更高。其中螞蟻演算法因為回饋機制，已被證實在越複雜的環境中，較其他啟發式演算法更具優勢。但由於螞蟻演算法本身的特性，當處於現實環境中容易因忽略朝向目的地前進的有效距離，而造成收斂速度減緩。

因此本研究提出一種基於螞蟻族群系統(Ant Colony System, ACS)演算法的改良式蟻群演算法，稱為方向感的螞蟻族群系統(Directional Ant Colony System, DACS)，將角度因子結合啟發資訊，使其具備方向概念，在動態環境下適應力及存活力比傳統的ACS演算法更有優勢。

關鍵字

螞蟻演算法、路徑規劃、動態環境。

Abstract

Mobile robot path planning is always one of the main problems of Robotics. utilizing heuristic algorithms in mobile robot path planning will have a better efficiency and flexibility. Especially ACS which was being proved that, in more complex environment, it has more advantages than other heuristic algorithms because of it's feedback mechanism. However, because of the characteristics of ACS, it has more possibility to ignore the effective distance toward the destination in realistic. As a result, it will slower the convergence speed.

Therefore, this paper proposed an algorithm based on ant colony system algorithm which named directional ant colony system to improve the weakness. By combining angle factor and heuristic, DACS has the cognition of direction. So it can has better performances and adaptability than traditional ACS algorithm in dynamic environment.

Keywords

Ant colony sestem, Mobile robot path planning, Dynamic environment.

目錄

圖目錄

表目錄

第一章、緒論

1.1 研究背景與動機

自 20 世紀 70 年代起，到現今受注目的工業 4.0，漸漸導入電腦控制自動化機器人來取代人力，而軍事及救災上發展出來的機器人更是功能繁多，如排雷、攻擊、搜救等諸多用途。在科技發展越來越先進的今天，舉凡精細、粗重、繁瑣、危險及任何人體不適合的作業環境大都被機器所取代，機器人已在生活中佔有重要的一席之地。

機器人被製造可模仿人類之行為，重要目的是要代替人類執行某些動作，而其常見的行為流程可以區分為下列 3 個步驟：

1. 即時感知所處環境的情境及限制。

2. 依據感知所獲之資訊，進行任務分析及規劃執行內容。

3. 完成任務，尤其是交由實體執行動作。

而如何讓機器人將獲取的環境資訊進行分析及規劃則是研究的核心內容，至今也有許多的研究成果發表，其中機器人路徑規劃是具代表性的問題之一(De Carvalho, R.N., 1997; Schmidt, G., 1998; Purian, F.K., 2013; Yang Chen, 2014)。

一般來說，移動機器人路徑規劃(朱大奇, 2010)須以演算法克服下列三個問題：

1. 使機器人從起點移動到預定終點。
2. 使用演算法使機器人避開障礙物，並視任務需要將某些必須經過的點納入路徑中，以完成相應的作業。
3. 在完成以上要點的前提及限制下，盡量優化機器人移動軌跡，以提升工作效率及節省時間。

現今研究中的演算法概略可以分成：

1. 模版匹配路徑規劃技術(Template Path Planning, TPP)。
2. 人造力場路徑規劃技術(Artificial Potential Field Path Planning, APFPP)。
3. 地圖建構路徑規劃技術(Map Building Path Planning, MBPP)。
4. 人工智慧路徑規劃技術(Artificial Intelligent Path Planning, AIPP)。

模板匹配路徑規劃法(Christian Hofner, 1994; Schmidt, G., 1998)是將機器人曾走過的路徑放進模板庫，之後機器人在規劃路徑時，會將當前路徑特徵與模板庫裡的數據進行比對，以特徵值找出一條最相近的路徑，再以之為基底修改成當前的路徑。

人造力場路徑規劃法(Robert A. Conn, 1998; Kikuo Fujimura, 1989)就是想像機器人處於一力場中，障礙物對機器人具有斥力，目標點對機器人具有引力，整個力場環境對機器人產生斥引作用，進而讓機器人避開障礙到達目

標位置。

地圖建構路徑規劃法(Avneesh Sud, 2008; Canny J.F., 1988; Takahashi, O., 1989)是利用機器人裝置的感測器去偵測障礙物資訊建構地圖，並分為可行區及不可行區，再依據一定的規則規劃出最佳路徑。

上述三種方法用於探索未知地圖及躲避障礙物時表現得相當優秀，尤其當面臨需要處裡複雜的地形環境時，例如：2016 年高雄美濃大地震為例，既有的路徑在大樓倒塌後已然消失，而災害現場的環境仍不穩定，隨時可能發生崩塌而再次改變地貌。

在完成環境探索後，於某些運用上，如何運用最低的成本，使系統能在兩點間反覆行動就顯得重要，如災後救援物資之往返運送。人工智慧路徑規劃法是適用於此類研究方法之一。

不同於以往傳統數學邏輯或程序性的演算法，人工智慧路徑規劃法不需花大量時間求解不確定、非線性多項式(Non-deterministic Polynomial, NP)問題，轉而模擬生物演化或生存方式，設計出許多不同求解方式的仿生演算法，通常能在可接受的時間內得到最佳解或是較佳解。

目前利用在路徑規劃上常見的啟發式演算法有，A*演算法(A-star Algorithm)、基因演算法(Genetic Algorithm, GA)與螞蟻系統 (Ant System, AS)演算法等。

A*演算法主要是先將環境劃分成大小相等的網格，從所在的網格向四周其他8個網格作移動代價評估，然後選取移動代價最小的網格移動，如此反覆進行最終抵達終點。但是A*演算法在路徑規劃前先將環境畫分網格，其中也包含了環境中不會前往的區域，這將會佔用許多記憶體執行運算，進而影響效率(Dong Jin Seo, 2013)。

GA(John Holland, 1975)主要的構想是源自於達爾文的演化論。達爾文認為生物界中的個體藉由「物競天擇，適者生存」的原則進行演化，生物間互相競爭，適應力良好的物種才得以生存下去；並將本身適合生存的特徵保留給下一代，讓下一代能演化成適應力更好的物種。Holland認為，無論是自然或人造環境，均可將事物依其屬性進行如DNA一樣的編碼(Coding and Representation)，因此GA就將一個字串(String)參數表示染色體(Chromosome)，每一世代(Generation)染色體的集合稱為族群(Population)。應用GA時須先定義一適應力函數(Fitness Function)，以評估染色體在此一問題中的適應力。適應力函數值越高，代表染色體適應力越好，越適合被選取來繁衍下一代，因此GA具有下列幾點特性(林豐澤, 2005)：

1. GA是將參數編碼進行演化運算，而不是使用參數本身做搜尋。
2. GA是高度的平行搜尋，會同時考慮空間的多個點而不是單一點，因此可避免陷入區域的最佳解。

3. GA使用適應力函數做為本體的知識，不需其他繁瑣的數學計算。

4. GA使用機率規則而不是特定規則來導引搜尋方向，因此可適用於不同類型的問題上。

目前GA也被應用在路經規劃的議題上，但由於GA特性使環境越複雜，GA演化出下一代的時間就越久，進而影響求解的速度(Peng .Y, 2006)。

AS 是模仿現實世界中螞蟻覓食行為的演算法，由 Dorigo 在 1996 年提出，對於複雜、資訊不完整的最佳化問題上有良好的表現。隔年 Dorigo 改良了 AS，提出螞蟻族群系統(Ant Colony System, ACS)演算法並用於解決旅行推銷員問題(Traveling Salesman Problem, TSP)。根據 Dorigo 研究顯示，人工螞蟻雖然是模仿真實螞蟻的行為，但卻有以下的不同(Dorigo, 1996)：

1. 人工螞蟻擁有記憶行為：人工螞蟻記得曾走過的路徑或求得的解答，而現實螞蟻沒有。

2. 人工螞蟻並非完全盲目：現實螞蟻無法得知問題全貌或是大致樣貌，僅能隨機或依循費洛蒙前進。

3. 人工螞蟻的行為具有離散性：人工螞蟻的時間點是不連續的。

由於上述人工螞蟻的三項差異，使的 AS 演算法及 ACS 演算法可具有分散計算及正向回饋等優點，可能在求解初期找到不是最優但可接受的解，之後再求得最佳解，使求解過程加速進行。

F. khosravipurian 提到 ACS 演算法與 GA 不同，GA 必須將親代的基因在交配池中經過多次的世代交替，藉此來產生最佳解，機制複雜許多；而 ACS 演算法僅依據相鄰節點距離的長短來進行搜尋，機制簡單。這在複雜程度越高的環境下，ACS 演算法與 GA 在相同迭代次數的條件下，ACS 演算法可以更快速的收斂找到最佳解(Fatemeh K. P., 2013)，因此 ACS 演算法較適合用於複雜度高的環境。

ACS 演算法在進行搜尋的過程是隨機的，雖然有助於跳脫出區域最佳解，但有時候會出現索搜尋的區域並不是目標的區域(Vitorino Ramos, 2002)，如何脫離區域解並快速收斂會是重要課題。因此本研究提出一種基於 ACS 演算法的改良式蟻群演算法，稱為方向感的螞蟻族群系統(Directional Ant Colony System, DACS)，將角度因子結合啟發資訊，使其具備方向概念，在動態環境下適應力及存活力比傳統的 ACS 演算法更有優勢。

1.2 研究限制

本研究所為動態環境，係指在部分已知環境中障礙物發生變化，導致地形地貌發生改變。由於機器人路徑規劃問題包含自動控制、感測元件、通訊系統與電機機械等結構(李孟軒, 2011)，內容涉略廣泛，因此本研究係建立在完整環境資訊已逐步搜集完成，以及系統定位與障礙物迴避皆已有效解決之前提上，僅針對DACS演算法之適應力及生存力進行探討。

1.3 論文架構

本論文架構共分為五章，第一章為緒論，說明研究背景、動機及目的。第二章介紹 AS、ACS 演算法演進及 DACS 演算法設計。第三章描述 DACS 演算法的模擬表現，本研究設計動態環境來驗證 DACS 演算法之生存力，並與 ACS 演算法比較適應力。第四章展示 DACS 演算法應用在自走機器人的實驗結果。最後，第五章為結論與未來研究，討論 ACS 演算法之實用性，並提出未來需要改善之建議。

第二章、DACS 演算法之設計與分析

2.1 AS 及 ACS 演算法

本研究所提出的 DACS 演算法改良自 ACS 演算法，故本節先介紹 AS 演算法及其缺點。

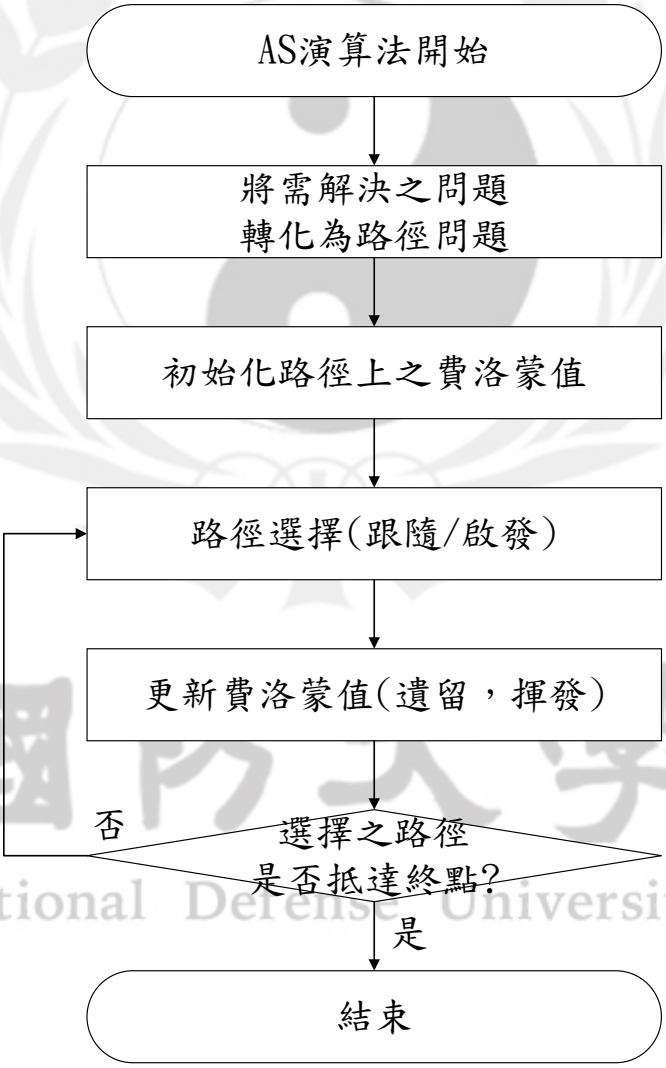


圖 2-1 AS 演算法之流程圖

圖 2-1 為 AS 演算法流程圖，首先將需解決之問題轉化為路徑問題，接

著為避免螞蟻在路徑規劃前期，遺留費洛蒙濃度具有顯著影響，因此會將路徑上的費洛蒙濃初始化為一極小值，隨後進行機率的轉換。依據 Dorigo 在 1996 年提出 AS 演算法之轉換機率數學模型如公式 2-1：

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\times[\eta_{ij}]^{\beta}}{\sum_{k\in allowed_k}[\tau_{ik}]^{\alpha}\times[\eta_{ik}]^{\beta}} & if\ \ j\in J_k(i)\\ 0 & others\end{cases} \tag{2-1}$$

$P_{ij}^{k}(t)$：為路徑(i,j)在時間t時之機率值。

$\tau_{ij}(t)$：為路徑(i,j)在時間t時之費洛蒙值。

α：以費洛蒙值進行跟隨路徑選擇的強度比重參數。

β：以環境參數作為啟發值之探索路徑選擇的強度比重參數。

η_{ij}：為選擇路徑之啟發值，為路徑(i,j)之距離倒數。

$J_k(i)$：位於節點i之螞蟻k尚未拜訪過的節點集合。

AS 利用轉換機率選擇路徑後，會進行費洛蒙更新。費洛蒙更新分成遺留(Laying)及揮發(Evaporating)兩部分，依據公式 2-1 所描述，若此次選擇之路徑為路徑(i,j)，則對路徑(i,j)進行遺留，數學模型如公式 2-2；其餘路徑(i,j)則進行揮發，數學模型如公式 2-3：

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}\dfrac{Q}{L_k} & if\quad \text{路徑 } i,j\in T_k\\ 0 & otherswise\end{cases} \tag{2-2}$$

$\Delta\tau_{ij}^{k}(t)$：為第k隻螞蟻在時間t於路徑(i,j)之遺留費洛蒙值。

T_k：為第k隻螞蟻在本次走訪中所走路徑。

L_k：為第k隻螞蟻在本次走訪中所走路徑的總長度。

Q：為費洛蒙濃度，其值的大小會影響路徑的選擇。

$$\tau_{ij}(t+1) = (1-\sigma)\tau_{ij}(t) \tag{2-3}$$

σ：為費洛蒙揮發係數，$0 < \sigma < 1$ 。

Dorigo 於 1996 年提出的 AS 演算法是模仿自然界的螞蟻如何搜尋食物之演算法，而自然界的螞蟻覓食過程如圖 2-2。首先如圖 2-2(a)A 端為蟻穴，E 端為食物源，$\overline{AE}$構成螞蟻之路徑。螞蟻在前進過程中會分泌沿途分泌一種叫費洛蒙的化學物質，其濃度會隨時間過去而揮發減少，後面的螞蟻則依據費洛蒙濃度選擇路徑，該路徑當時的費洛蒙濃度與被選擇的機率成正相關。由於螞蟻均沿著路徑$\overline{AE}$前進，而費洛蒙遺留與揮發於路徑$\overline{AE}$達成平衡。

如圖 2-2(b)今掉落障礙物(Obstacle)使路徑$\overline{AE}$變成$\overline{AB}$、$\overline{BCD}$、$\overline{BHD}$及$\overline{DE}$四個子路徑，其中$\overline{BCD}$距離＜$\overline{BHD}$距離。由於螞蟻都未曾經過路徑$\overline{BCD}$及路徑$\overline{BHD}$，費洛蒙濃度皆為零，因此螞蟻會啟發選擇$\overline{BCD}$及$\overline{BHD}$之一路徑前進。由於路徑$\overline{BCD}$距離＜路徑$\overline{BHD}$距離，故單位時間內走完路徑$\overline{BCD}$螞蟻的數量高於路徑$\overline{BHD}$，路徑$\overline{BCD}$費洛蒙遺留及揮發後之濃度相對高於

路徑$\overline{BCD}$之費洛蒙濃度。

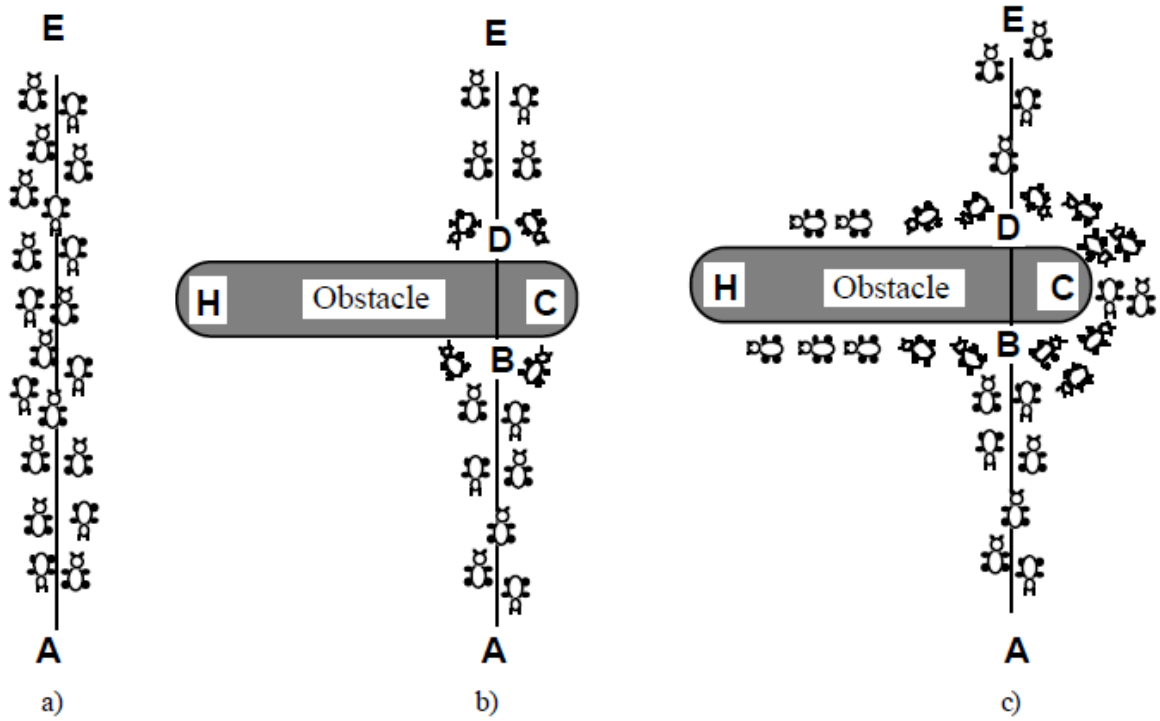


圖 2-2 螞蟻覓食原理圖 (資料來源：Dorigo, 1996)

如圖 2-2(c)路徑$\overline{BCD}$費洛蒙遺留及揮發後之濃度越高，後面的螞蟻選擇路徑$\overline{BCD}$機率越高，相對路徑$\overline{BHD}$因被選擇機率低，加上費洛蒙遺留少並持續揮發，故造成螞蟻跟隨時均選擇路徑$\overline{BCD}$，最終達到新的平衡。

Dorigo將AS與GA演算法以TSP進行比較，結果顯示AS與實際最佳解差異小於3.5%。然而因AS存在搜索時間過長及容易落入區域解的問題，故Dorigo在1997年以AS為基礎，提出ACS演算法，流程圖如圖2-3。

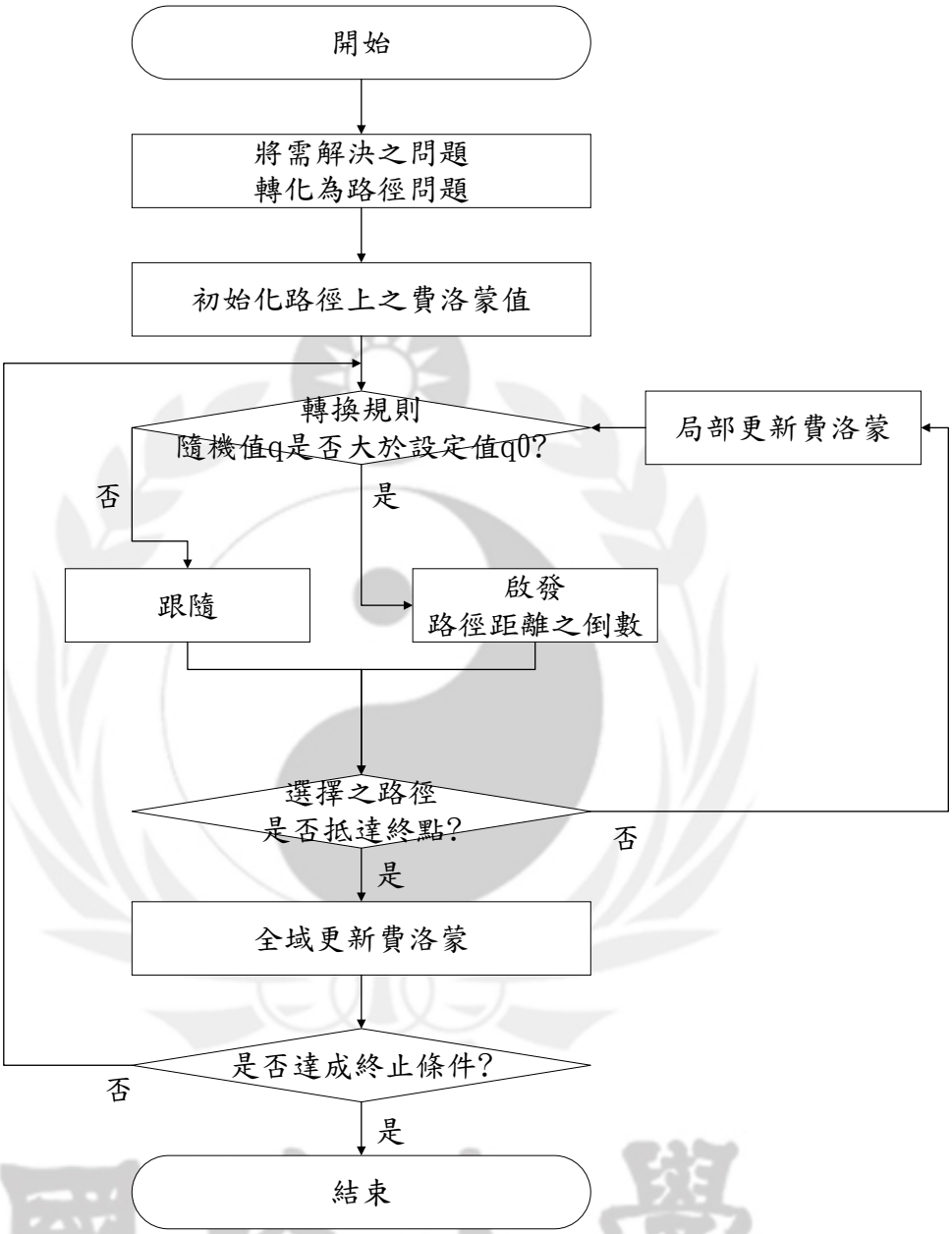


圖 2-3 ACS 演算法流程圖

ACS演算法基於AS做出三點改進：

1. 轉換規則：加快收斂速度，數學模型如公式2-4。

$$J = \begin{cases} \max_{j \in J_k(i)} \left\{ [\tau_{ij}(t)]^{\alpha} \times [\eta_{ij}]^{\beta} \right\} & if \quad q \leq q_0 \\ P_{ij} & otherwise \end{cases} \tag{2-4}$$

q_0：為一個門檻值，，$0 < q_0 < 1$。

q：為隨機亂數值，$0 < q < 1$。

P_{ij}：為AS之路徑選擇。

因為增加轉換規則，故出現$q > q_0$及$q \leq q_0$兩種狀況。當$q \leq q_0$，稱為跟隨，選擇機率最高之路徑，故會加速收斂；當$q > q_0$，則稱為啟發，進行機率轉換，藉此搜尋未曾經過的路徑，避免落入區域解。

2. 局部更新費洛蒙：降低落入區域解機會，數學模型如公式2-5。

$$\tau_{ij}(t+1) = (1-\sigma)\tau_{ij}(t) + \sigma\tau_0 \tag{2-5}$$

τ_0：費洛蒙初始濃度。

$\tau_{ij}(t)$：為此次搜尋路徑之費洛蒙值。

若路徑(i,j)為此次選擇之路徑，則減少該路徑之費洛蒙，以提升螞蟻探索的機制，降低落入區域解之機會。

3. 全域更新費洛蒙：加快收斂速度，數學模型如公式2-6及2-7。

$$\tau_{ij}(k+1) = (1-\sigma)\tau_{ij}(k) + \sigma\Delta\tau_{ij} \quad (2\text{-}6)$$

$\tau_{ij}(k)$：為此次總路徑之費洛蒙值。

$$\Delta\tau_{ij} = \begin{cases} \frac{Q}{L_k} & if \quad \text{路徑}(i,j) \in T^+ \\ 0 & otherwise \end{cases} \quad (2\text{-}7)$$

T^+：全域最佳解。

綜合公式2-6及2-7，若總路徑(i,j)屬於全域最佳解，則總路徑(i,j)增加費洛蒙。

Dorigo將ACS與GA等演算法，以TSP問題進行比較，結果顯示ACS等於實際最佳解。

2.2 DACS 演算法

Dorigo所提出之ASC演算法及AS演算法於TSP問題表現良好，但AS及ACS演算法均盲目且貪婪的以目前所在位置測得之路徑距離為啟發值，忽略朝向目的地前進的有效距離，造成收斂速度減緩。

如圖2-4，假設有一個二維座標空間，其中有S、T、A、B、C、D及E點，S為機器人所在點，T為終點，A、B、C、D及E點均為可行走之路徑節點，可行走路徑方向以實線表示，虛線為前進方向。若先不考慮費洛蒙值，使用

ACS演算法計算啟發時路徑之機率(公式2-1)，可得結果如表2-1。

表 2-1 ACS 演算法選擇路徑機率

節點	S 與各節點距離	η_{AS}值	ACS 機率值
A(10, 7)	12.2	0.081	13.9%
B(13, 19)	8.6	0.116	19.9%
C(15, 28)	14.87	0.067	11.5%
D(23, 23)	8.54	0.117	20.1%
E(23, 10)	5	0.2	34.6%

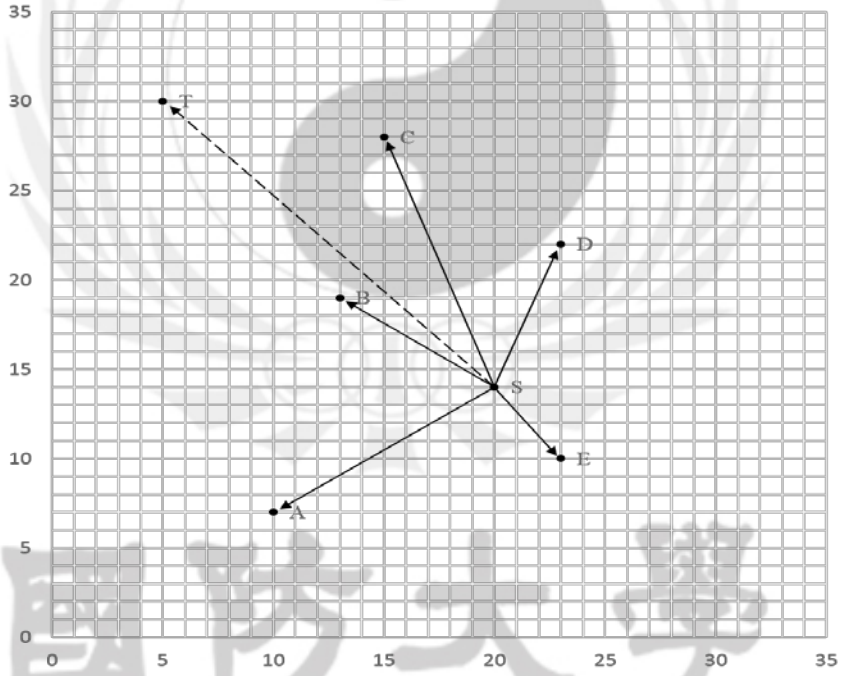


圖 2-4 DACS 選擇節點圖

由表2-1可知E點之η_{AS}值最大，因此在不考慮費洛蒙值之情況下，選擇E點之機率最高，但從圖2-4中卻可發現E點之路徑方向與前進方向相反，如果前進反而加大與目的節點T之距離。因此應用ACS及ACS演算法在現實環

境中的路徑規劃有缺陷。

本研究提出DACS演算法(如圖2-5)，針對ACS及AS演算法之啟發值進行改良，數學模型如公式2-8：

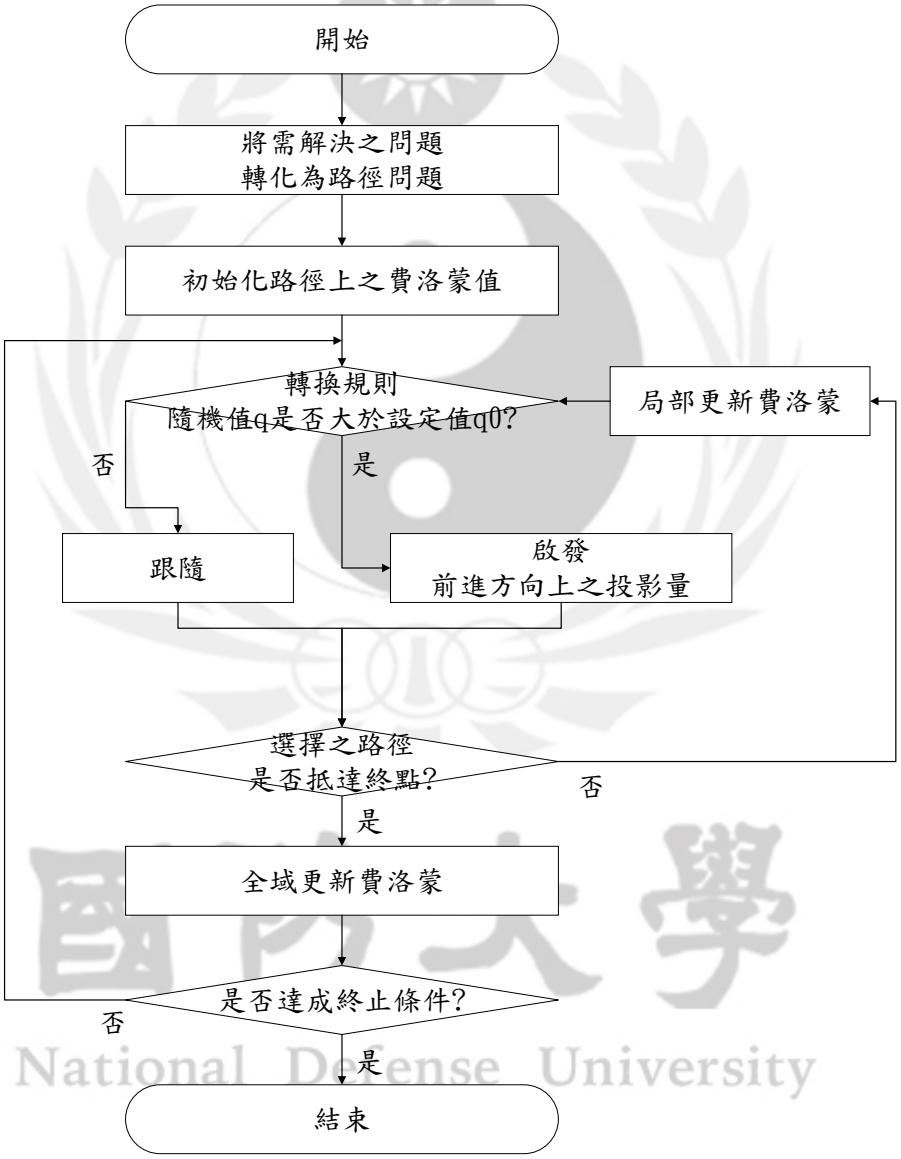


圖 2-5 DACS 演算法流程圖

$$\eta_{ij} = \left(d_{ij}\right)^{\gamma(cos\theta+1)} \tag{2-8}$$

γ：為角度影響啟發資訊的比重參數。
θ：為路徑(i,j)與前進方向之夾角。
d_{ij}：為 i 節點與 j 節點之距離。

本研究目前尚不討論比重設置之合理值，故將γ設為1。

以路徑方向在前進方向上之投影向量做為啟發值，以圖2-4之節點為例，則投影量可以表示為表示如圖2-6，若$\overrightarrow{SC}$與$\overrightarrow{ST}$之夾角為θ，則$\overrightarrow{SC}$在$\overrightarrow{ST}$上之投影量$\vec{k}$為公式2-9。

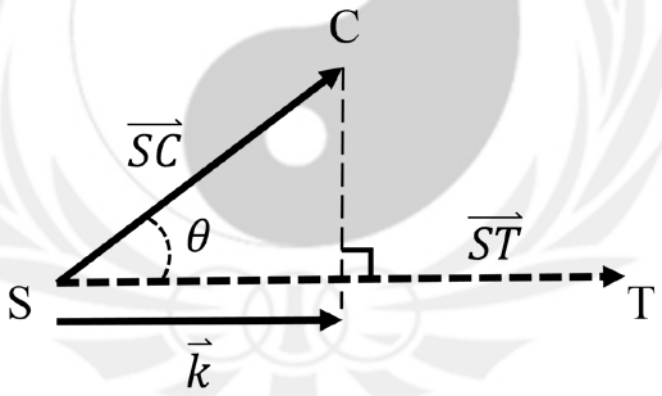


圖 2-6 向量的投影

$$\vec{k} = \left\|\overrightarrow{SC}\right\| \cos\theta \tag{2-9}$$

若將圖 2-4 使用 DACS 演算法計算路徑之機率(公式 2-8)，可得結果如表 2-2。由表 2-2 可知選擇 C 點之機率最高，若對照圖 2-4 可發現該路徑方向在前進方向上之投影量最大，故選擇 C 點較 E 點合理。

表 2-2 DACS 演算法選擇路徑機率

節點	路徑方向與 $\overline{ST}$之夾角	S 與各節點距離	η_{DACS} 值	DACS 機率值
A(10, 7)	81.9°	12.2	17.35	6.2%
B(13, 19)	11.56°	8.6	70.8	24.5%
C(15, 28)	23.4°	14.87	177.09	61.3%
D(23, 23)	63.59°	8.54	22.17	7.7%
E(23, 10)	173.88°	5	1.01	0.3%

第三章、DACS 演算法模擬與結果

3.1 動態模型下 DACS 演算法之表現

本研究以 Thomas Stützle. 在 2002 年開發之 ACSTSP-1.03 開放程式為基礎，配合 DACS 演算法擴充其啟發資訊並實施模擬(如圖 3-1)。

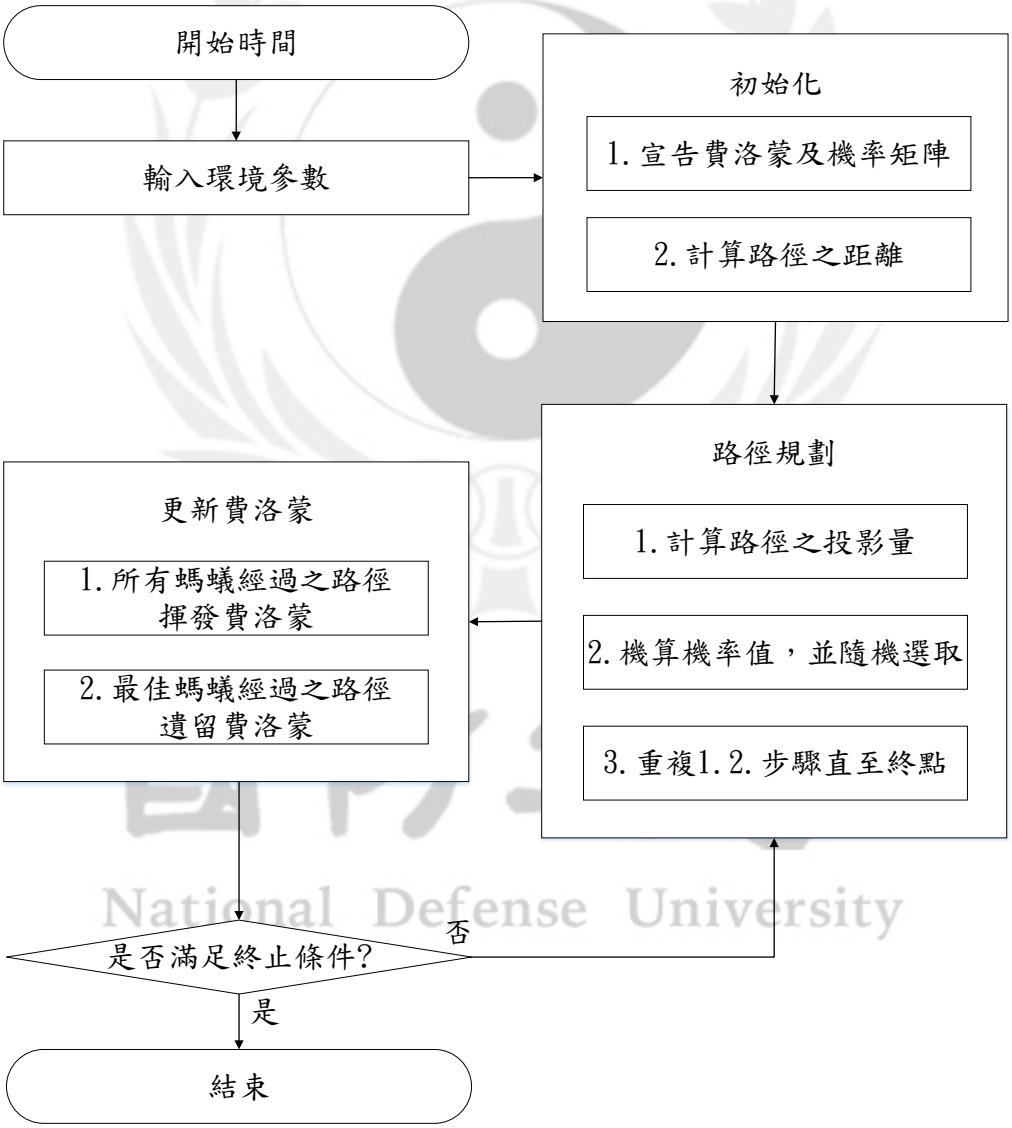


圖 3-1 DACS 模擬流程圖

也為驗證 DACS 演算法是否可隨著動態環境規劃出可行路徑，因此本研究設計於路徑進行中環境產生動態變化之實驗。

機器人於路經規劃初期並不知道環境的全貌，爾後透過自身感測器逐步蒐集完整環境資訊。因此本研究係建立在環境全貌已逐步搜集完成，以及系統定位與障礙物迴避皆已有效解決之前提上。

模擬環境對照組以一個靜態地圖(如圖 3-2)做為所蒐集之環境全貌。由 DACS 演算法規劃之結果如圖 3-3。

本研究設計於對照組執行路徑規劃至第二次迭代時，使環境分別產生節點移動、節點增加、節點消失及目標變換等變化，藉此模擬動態環境。因實際情況中，感測器測量距離能力有限，故於實驗中設兩節點距離若超出 10 unit 者無法偵測，將判定為不可行走之路徑。測試檔內部各個節點均以平面座標表示之，並分別標上n_0~n_{31}等編號以利識別，若兩節點間螞蟻經過的次數越多則線段越粗，螞蟻經過的次數越少，則線段越細。$n_m(x,y)$括弧中 x, y 為平面座標 x 及 y 值，$p\{n_{m1},n_{m2},\dots,n_{mn}\}$代表一路徑，$d\{n_{m1},n_{m2}\}$代表$n_{m1}$及$n_{m2}$之直線距離。

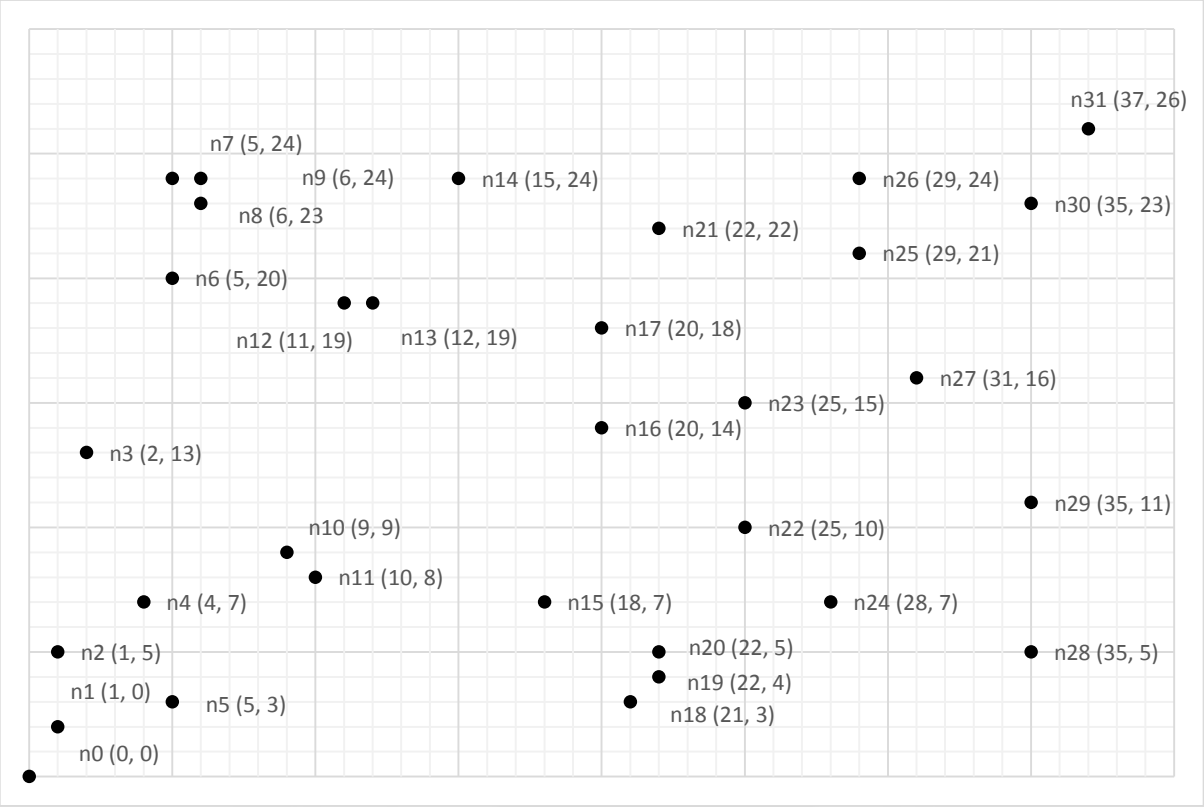


圖 3-2 對照組之地圖測試檔

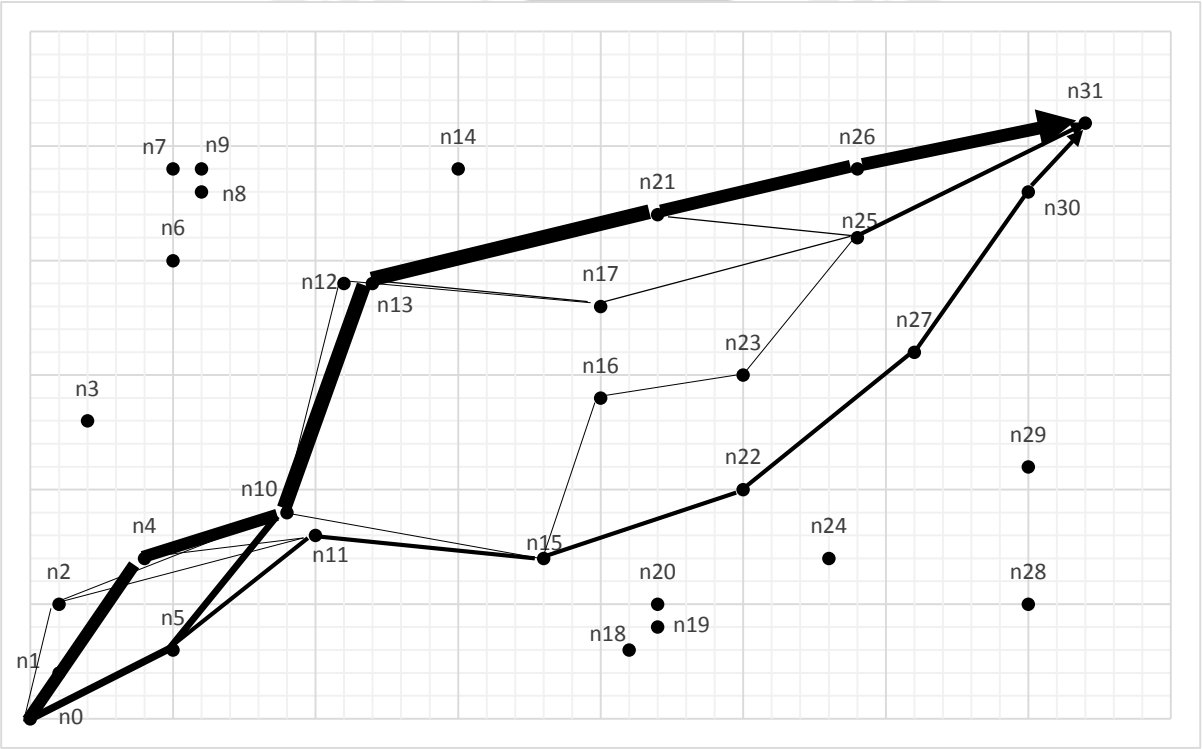


圖 3-3 DACS 演算法於對照組之路徑規劃結果

模擬一：節點移動

此實驗欲證實地圖中的節點若發生移動，則 DACS 演算法仍可正常運作。實驗中起點為節點$n_0(0,0)$，終點為節點$n_{31}(37,26)$。如圖 3-3， $d\{n_{10}, n_{16}\}$為 12.8 unit、$d\{n_{11}, n_{16}\}$為 11.66 unit，兩者路徑長度均超出 10 unit 檢測範圍。今將節點$n_{16}(20,14)$移至$n_{16'}(18,13)$後如圖 3-4，$d\{n_{10}, n_{16'}\}$變為 9.84 unit、$d\{n_{11}, n_{16'}\}$為 9.43 unit，按照實驗設定均判定為可行路徑。

實驗結果，圖 3-3 中行經節點n_{10}及節點n_{11}時，因為判定為不可行路徑而不經過節點n_{16}，改走其他路徑。圖 3-4 則因$n_{16'}$的位置是前進方向上距離最短、機率提升的捷徑，使的螞蟻經過節點$n_{16'}$次數增加。

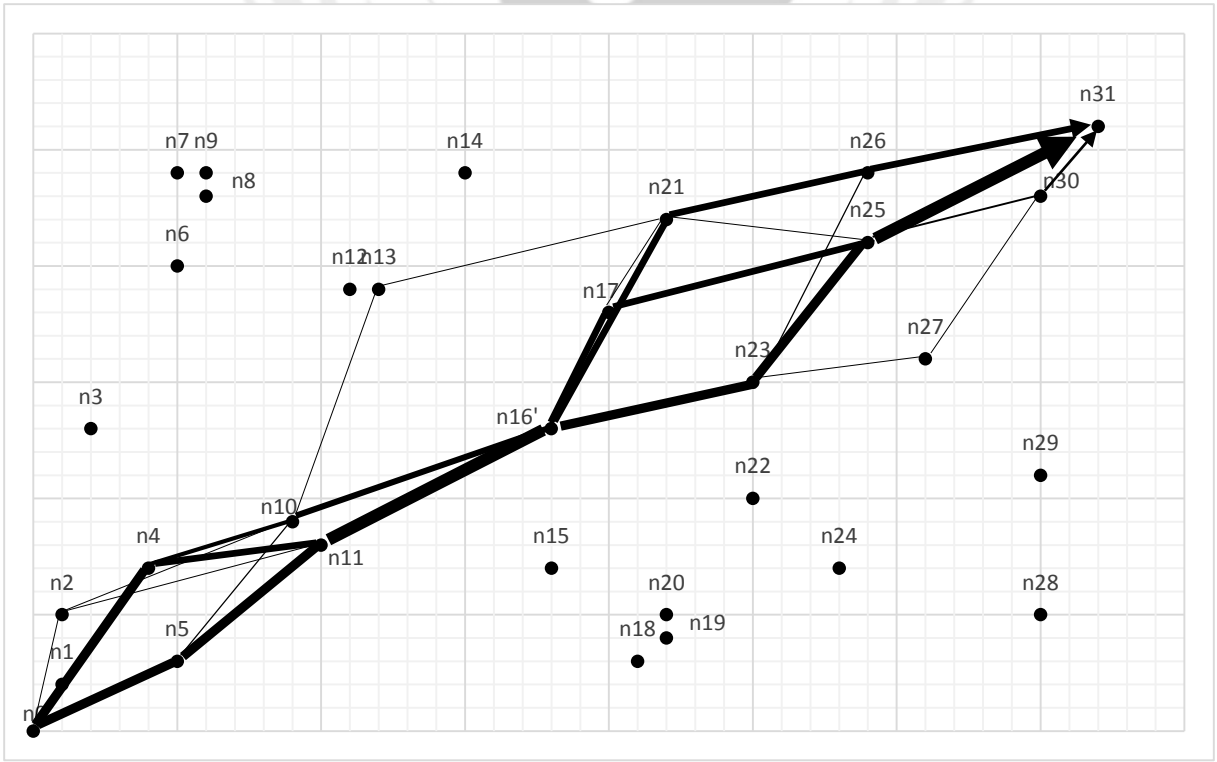


圖 3-4 DACS 演算法於節點移動之路徑規劃結果

模擬二：節點增加

此實驗為驗證若環境變動增加新節點，DACS 演算法是否仍可有效規劃路徑前進，因此在行經路徑中圖 3-5 中節點n_{11}及節點n_{16}之間加入一新節點$n_{32}(15, 11)$。實驗中起點為節點$n_0(0, 0)$，終點為節點$n_{31}(37, 26)$。如圖 3-3， $d\{n_{10}, n_{16}\}$為 12.8 unit、$d\{n_{11}, n_{16}\}$為 11.66 unit，兩者路徑均超出 10 unit，加入新節點後如圖 3-5，使$p\{n_{10}, n_{32}, n_{16}\}$、$p\{n_{10}, n_{32}, n_{17}\}$、$p\{n_{11}, n_{32}, n_{16}\}$及$p\{n_{11}, n_{32}, n_{17}\}$四條路徑成為可行路徑。

實驗結果如圖 3-5 所示，由於經過節點n_{32}之總距離比經過節點n_{13}之總距離要短，故原本經過節點n_{13}的次數減少，經由節點n_{32}的次數增加。

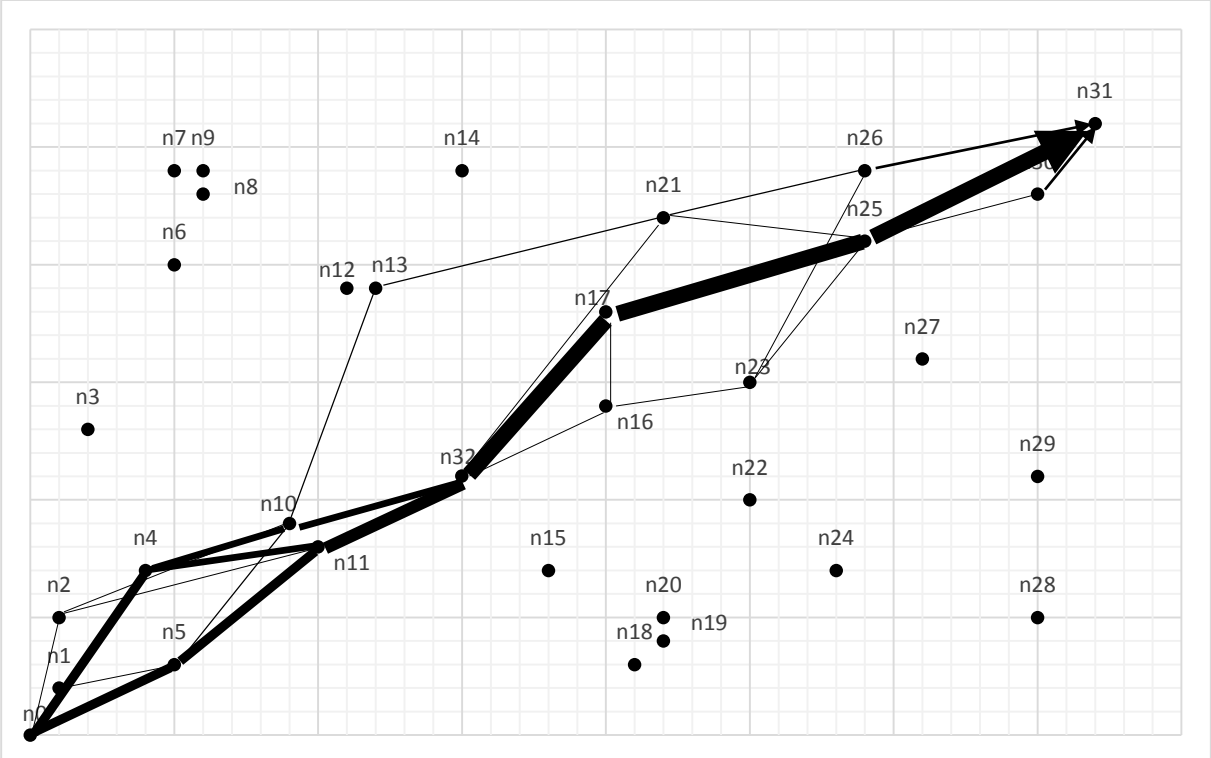


圖 3-5 DACS 演算法於節點增加之路徑規劃結果

模擬三：節點消失

此實驗欲證明若地圖中原本可行路徑消失，DACS 演算法仍可規劃出路徑前進。如圖 3-3 原地圖中起點為節點$n_0(0, 0)$，終點為節點$n_{31}(37, 26)$，大部分經過的路徑為 $p\{n_0, n_4, n_{10}, n_{13}, n_{21}, n_{26}, n_{31}\}$，今將節點$n_{12}$、$n_{13}$及$n_{15}$去掉如圖 3-6，則 $d\{n_{11}, n_6\}$、$d\{n_{11}, n_7\}$、$d\{n_{11}, n_8\}$、$d\{n_{11}, n_9\}$、$d\{n_{11}, n_{14}\}$、$d\{n_{11}, n_{16}\}$、$d\{n_{11}, n_{17}\}$、$d\{n_{11}, n_{18}\}$、$d\{n_{11}, n_{19}\}$、$d\{n_{11}, n_{20}\}$都會超出 10 unit，原本可行之路徑也就消失。實驗結果顯示，原本經過節點n_{13}及節點n_{15}之次數，因為該節點的消失，所以次數全部加到節點n_3上，路徑也變為 $p\{n_0, n_4, n_{10}, n_3, n_6, n_9, n_{14}, n_{21}, n_{26}, n_{31}\}$，繞過了不可行區域。

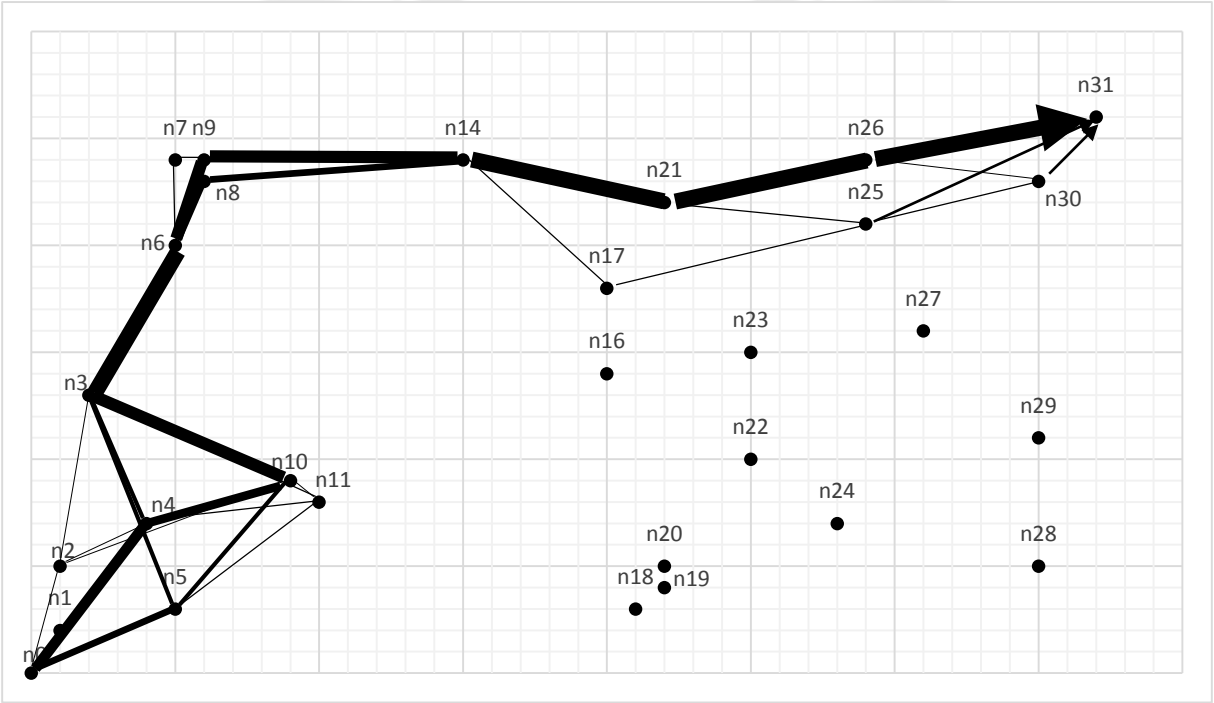


圖 3-6 DACS 演算法於節點消失之路徑規劃結果

模擬四：目標變換

此實驗為驗證若 DACS 演算法若遇到目標變換之情況，仍可規劃出可行路徑。圖 3-3 中起點為節點$n_0(0, 0)$，終點為節點$n_{31}(37, 26)$。圖 3-7 則將終點改為節點$n_{28}(35, 5)$。實驗結果如圖 3-7 所示，由於終點變為節點n_{28}，前進方向變換，故主要前進路徑變為 $p\{n_0,\ n_5,\ n_{11},\ n_{15},\ n_{24},\ n_{28}\}$，證明 DACS 演算法即便在目標變換情況下，仍可有效規劃路徑。

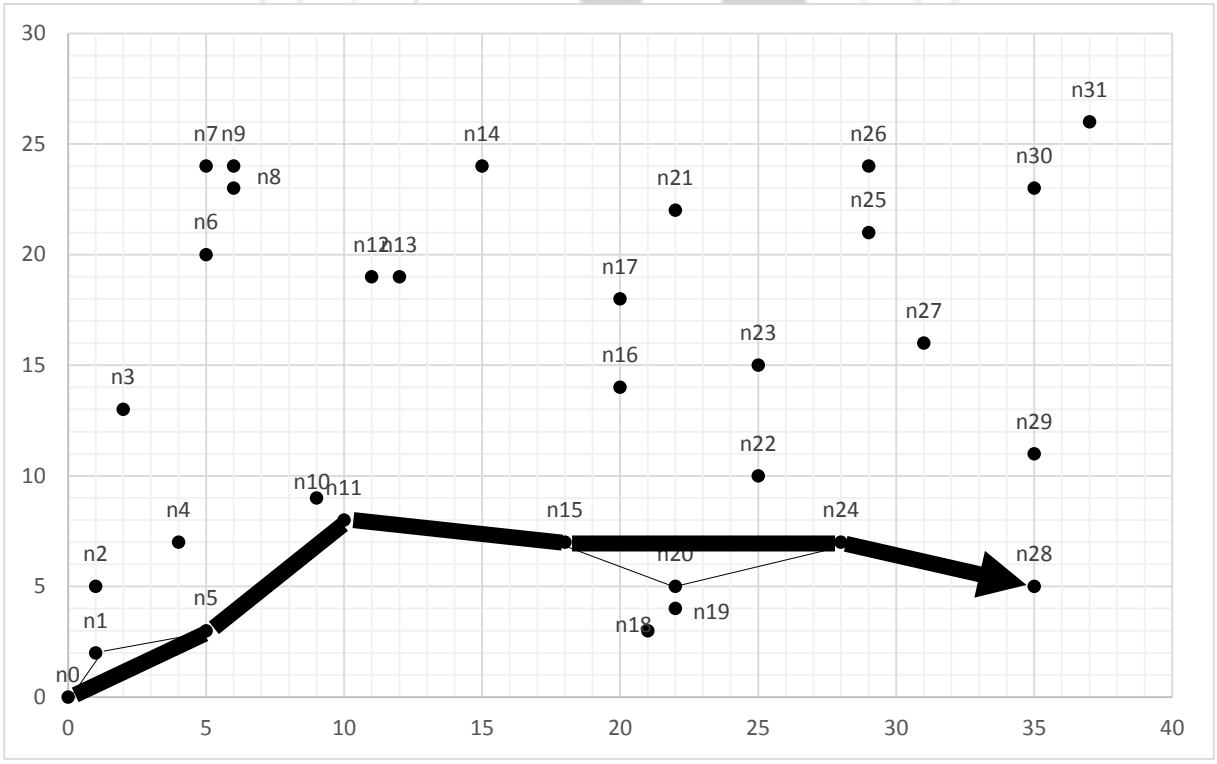


圖 3-7 DACS 演算法於目標變換之路徑規劃結果

從上述實驗可知 DACS 演算法在四種變動的環境下，皆能因應環境變化而做出反應，並且找到最佳解。值得一提的是 DACS 演算法屬於啟發式

演算法，在設計上較傳統數學邏輯演算法具有跳脫區域解之能力，以圖 3-4 為例，最佳路徑為 $p\{n_0, n_5, n_{11}, n_{32}, n_{17}, n_{25}, n_{31}\}$，但仍會有 $p\{n_0, n_1, n_5, n_{10}, n_{13}, n_{21}, n_{26}, n_{31}\}$或是 $p\{n_0, n_2, n_{11}, n_{16}, n_{23}, n_{25}, n_{30}, n_{31}\}$ 等其他可能路徑出現，使 DACS 演算法有機會可以嘗試不同的解，避免落入區域解。

3.2 DACS 演算法與 ACS 演算法之模擬比較

此實驗設計一個在相同的環境條件下，分別執行 DACS 演算法及 ACS 演算法並分析其結果，輸入參數值如下(表 3-1)：

表 3-1 實驗參數表(資料來源：Romain Hendrickx, 2012)

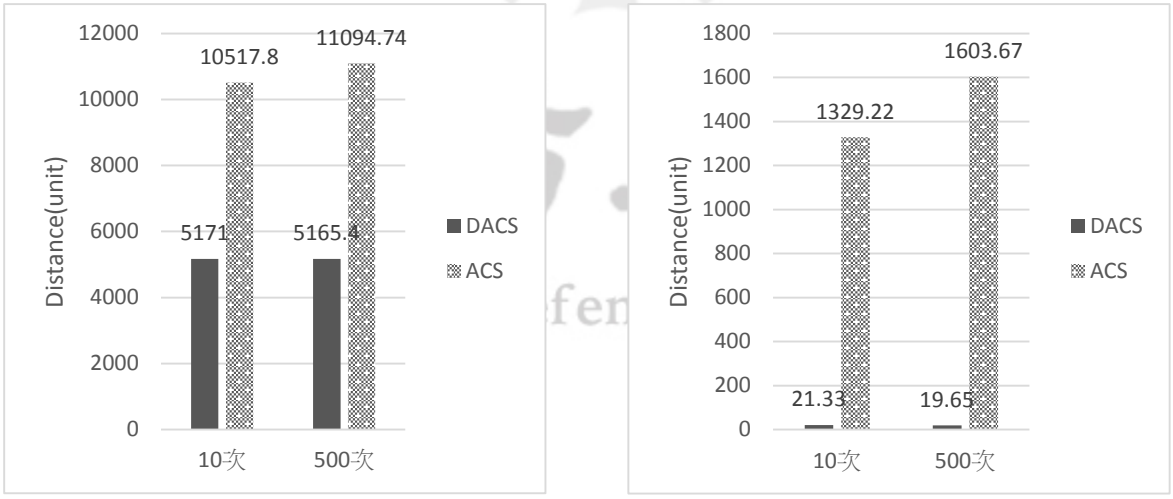
Nmae	**Value**	**Denotation**
max_tries	10/500	number of independent trials
max_time	10	maximum time for each trial
tapfile	lin318.tsp	inputfile
n_ants	10	number of ants
alpha	1	alpha (influence of pheromone trails)
beta	2	beta (influence of heuristic information)
rho	0.5	pheromone trail evaporation

max_tries 為執行程式次數，實驗中執行 10 次及執行 500 次的狀況；max_time 為執行 1 次程式的最大時間，實驗中設定為 10 秒；tapfile 為載入的測試檔 lin318.tsp，包含 318 個節點；n_ants 為執行搜索的螞蟻數量，設

定 10 隻；alpha 為費洛蒙的比重值，實驗中設為 1；beta 為啟發資訊的比重值，實驗中設為 2。rho 為費洛蒙的揮發值，實驗中設為 0.5。

平均路徑總長：

從圖 3-8 (a)、(b)中顯示執行 10 次時 DACS 演算法平均長度為 5171unit，標準差為 21.33unit；ACS 演算法平均長度為 10517.8unit，標準差為 1329.22unit。DACS 演算法平均長度比 ACS 演算法縮短 51%，標準差低了 1307.89unit。執行 500 次時 DASC 演算法平均長度為 5165.4unit，標準差為 19.65unit；ACS 演算法平均長度為 11094.74unit，標準差為 1603.67unit。DACS 演算法平均長度比 ACS 演算法縮短 53.5%，標準差低了 1584.02unit。實驗數據說明 DACS 演算法所規劃的路徑比 ACS 演算法可以更快抵達終點，且規劃之結果相當平均，長度較 ACS 演算法短。

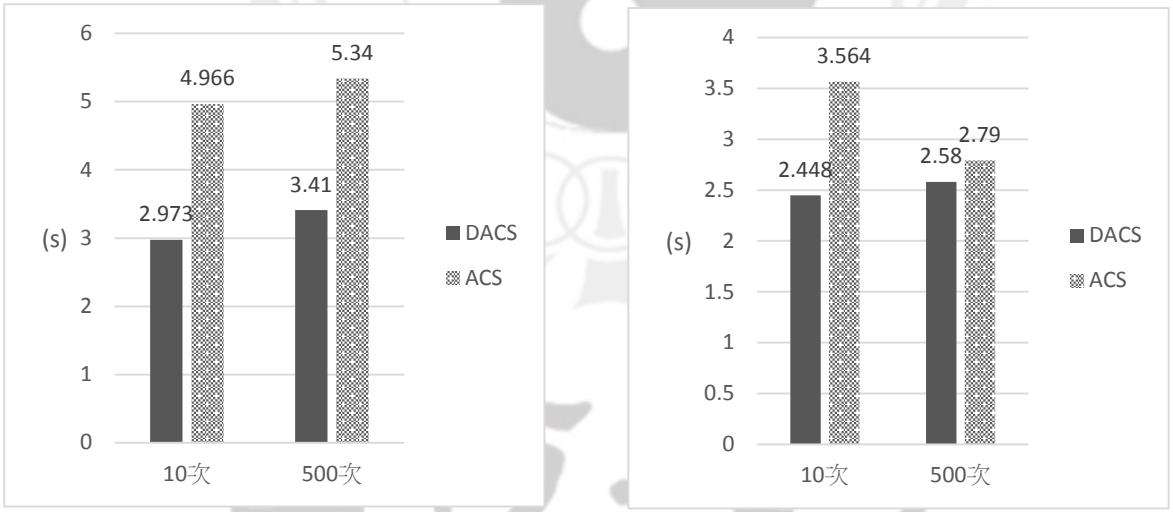


(a) (b)

圖 3-8 (a)DACS 及 ACS 平均路徑長(b)DACS 及 ACS 平均路徑長之標準差

路徑規劃時效性：

從圖 3-9 (a)、(b)可看出執行 10 次時 DACS 演算法找到最佳解的平均時間為 2.973 秒，標準差為 2.448 秒；ACS 演算法平均時間為 4.966 秒，標準差為 3.564 秒。DACS 演算法找到最佳解的平均時間比 ACS 演算法少 41.2%，標準差低了 0.591 秒。執行 500 次時 DACS 演算法找到最佳解的平均時間為 3.41 秒，標準差為 2.58 秒；ACS 演算法平均時間為 5.34 秒，標準差為 2.79 秒。DACS 演算法找到最佳解的平均時間比 ACS 演算法少 37.2%，標準差低了 0.21 秒，說明 DASC 演算法收斂速度較佳。



(a) (b)

圖 3-9(a)DACS 及 ACS 找最佳解之平均時間(b) DACS 及 ACS 找最佳解平均時間之標準差

透過上述實驗可以得知，DACS 演算法演算法將角度因子結合啟發資

訊，不僅加快演算法的收斂速度，尤其經過大量測試後仍可維持求得解之品質，證明本研究所方法之可行性。

第四章、DACS 演算法實作與自走機器人整合

前兩章已介紹 DACS 演算法之概念及電腦模擬之可行性，本章將介紹如何將 DACS 演算法實際運作於機器人。

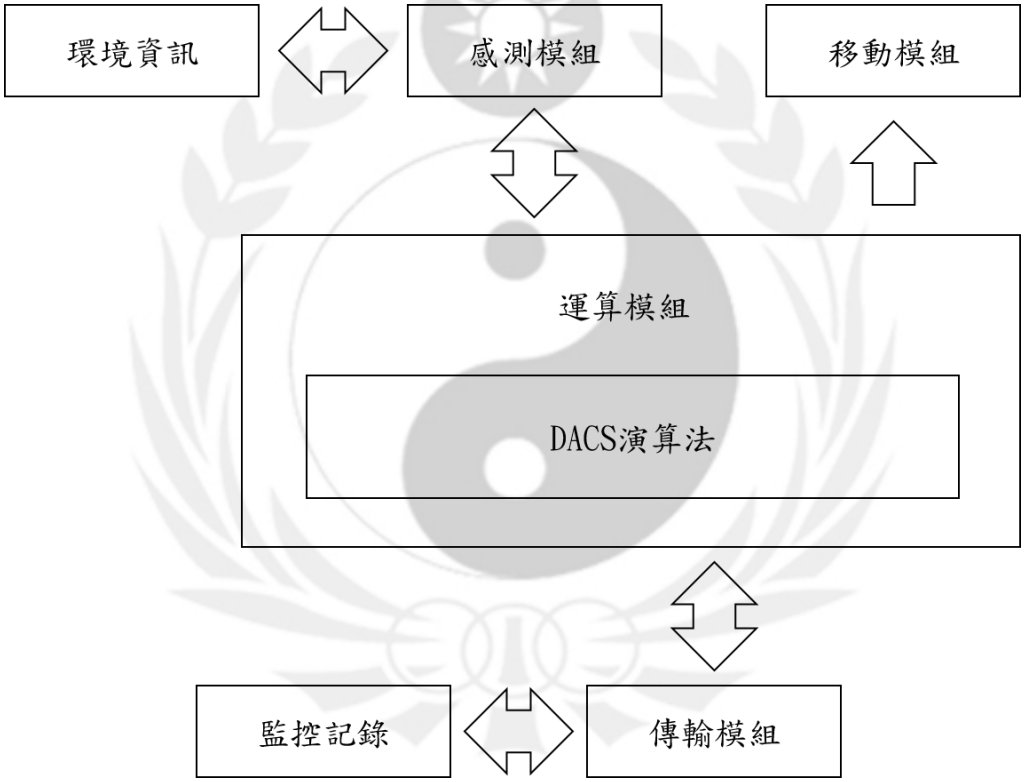


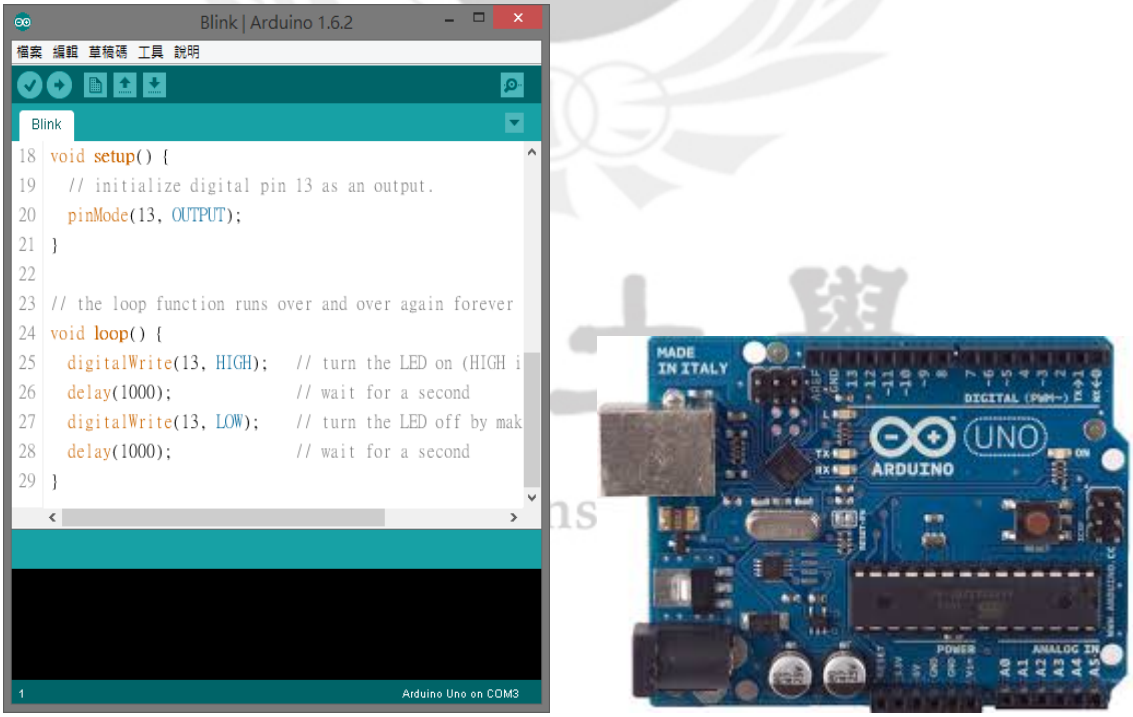
圖 4-1 機器人功能方塊圖

依據 DACS 演算法之模擬方塊圖(圖 3-1)，可以得知機器人需裝載感測模組以獲取環境資訊，並將環境資訊交由運算及控制平台進行運算，並將 DACS 演算法運算之結果，由移動模組進行輸出及傳輸模組進行監控及記錄(如圖 4-1)。

4.1 運用模組之硬體

運算與控制平台：

Arduino 是在 2005 年(艾迪諾，2014)由米蘭互動設計學院的教授 David Cuartielles 和 Massimo Banzi 所設計，原始構想是希望能替學生及藝術家們，找到一種便宜且簡單易懂的微電腦裝置，依此開發出基於開放原始碼發展出來的 I/O 介面如圖 4-2(a)。主要採用 Atmel 公司的 AVR 系列微處理器。它的硬體很簡潔，沒有什麼特殊的設計，除了開放式硬體架構之外，重點在於它提供了一個基於 C/C++的程式語言來控制 Arduino，讓使用者可以輕易使用 Arduino 語言與 Flash 或 Processing...等軟體，作出互動作品。



(a) (b)

圖 4-2 (a)Arduino IDE 開發環境 (b)Arduino Uno

Arduino 已發展出許多開發板，如 Arduino Uno、Arduino Due、Arduino Mega SDK、LilyPad Arduino、Arduino Nano…等等，雖然不同開發板適用於不同用途，但其基本功能都大同小異。由於 Arduino Uno 開發板(如圖 4-3(b))取得容易、泛用性高，故本研究選擇 Arduino Uno 開發板作為運算與控制平台主體。

Arduino Uno如圖4-2 (b)的微控器使用ATmega328(趙英傑，2014)(徐德發，2012)，具有13 個數位輸入/輸出埠，其中Pin 3、5、6、9、10、11具有PWM輸出功能，Pin 2、3具有中斷輸入功能，Pin 0、1 為一組串列通訊埠(0 = Rx, 1 = Tx)；Pin 13具有LED輸出功能。Arduino Uno還有6個類比輸入埠(A0 ~ A5)。Arduino Uno使用16MHz的振盪器，並具有128K的記憶體。至於下載程式的介面，則是使用ATmega8U2晶片來處理USB通訊(B-Type接頭)，並可支援多平台作業系統windows/Linux/Mac的開發環境。電源方面使用2.1mm的Power Jack接頭，可輸入6~20V 的DC電源，官方則建議使用7~12V的DC電源。

感測模組：

本研究採用超音波感測器HC-SR04(如圖4-3)進行距離之偵測，HC-SR04具有Transmitter端及Receiver端，工作電壓為5*V*，量測距離為2*cm*~400*cm*，精

度為0.3*cm*，感應角度為15°，各腳位連接如表4-1。

圖 4-3 HC-RS04 超音波感測器

表 4-1 HC-RS04 腳位表

HC-SR04腳位	Arduino Uno 腳位
Vcc	+ 5 V
Trig	GPIO
Echo	GPIO
GND	GND

其工作原理是利用Transmitter端發射40*kMz*的聲波，碰到物體後反射回來由Receiver端接收。其量測距離之公式如下：

$$D = V \times t / 2 \tag{4-1}$$

D：為所量測之距離，單位：m。

V：聲音於空氣介質傳播之速率，單位：m/s。

t：為從發射到接收之時間，單位：s。

本研究採用伺服馬達SG-90(如圖4-4)結合HC-SR04進行特定角度之量測，SG-90的尺寸為$23 \times 12.2 \times 29mm$，工作電壓為$4.8V$，轉矩為$1.8kg/cm(4.8V)$，轉速為$0.1s/15°(4.8V)$，脈衝寬度為500~2400$\mu s$，死頻帶寬度為10 μs，各腳位連接如表4-2。

圖 4-4 SG-90 伺服馬達

表 4-2 SG-90 腳位表

SG-90腳位	Arduino Uno 腳位
Vcc	+ 5 V
GND	GND
Signal	GPIO

其工作原理是利用脈衝控制轉動的角度(如圖4-5)，利用高、低電壓時間週期來進行編碼，若在一個時間周期中，50%的時間輸出高電位 5V，則平均電壓可視為2.5V，轉動角度為90°；若 10% 的時間輸出高電位 5V，則平均電壓可視為 0.5V，轉動角度為18°。

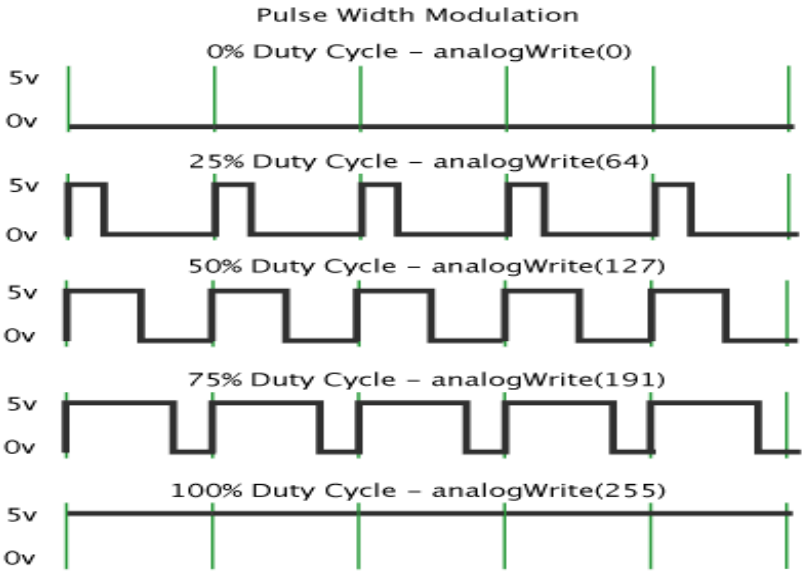


圖 4-5 脈衝寬度調製圖

為使機器人在行走過程中得以校正方向，因此本研究在移動機器人上加裝三軸陀螺儀/加速感測模組GY-521(如圖4-6)。GY-521之核心晶片為

圖 4-6 GY-521 三軸陀螺儀/加速度感測模組

MPU-6050由Invensense公司開發，其整合了三軸陀螺儀及三軸加速器，運用I^2C通信協定進行數據傳輸，工作電壓為3~5V，其角速度感測範圍為±250°、

$\pm 500°$、$\pm 1000°$及$\pm 2000°/s(dps)$，加速器感測範圍為±2g、±4g、±8g及±16g。本研究僅運用GY-521之三軸陀螺儀功能進行角度運算，各腳位連接如表4-3。

表 4-3 GY-521 腳位表

SG-90腳位	Arduino Uno 腳位
Vcc	+ 5 V
GND	GND
SCL	A5
SDA	A4

移動模組：

本研究採用L298N馬達控制電路版(如圖4-7)驅動兩顆直流馬達。L298N 是由 SGS Thomas公司生產的一種IC，可在高電壓(46*V*)、大電流(4*A*)環境下工作，內部具有兩組H橋式電路，可工用於繼電器、電磁閥、步進馬達和直流馬達。L298N馬達控制電路版有ENA、ENB、IN1~4共六個腳位，其中三隻腳位可以控制一顆直流馬達，真值表如表4-4，藉由兩個直流馬達正反轉達到移動目的。

圖 4-7 L298N 馬達控制電路版

表 4-4 L298N 腳位真值表

ENA	IN1	IN2	功能
HIGH	HIGH	LOW	馬達正轉
HIGH	LOW	HIGH	馬達反轉
HIGH	IN1 = IN2	IN1 = IN2	馬達快速停止
LOW	ingored	ingored	馬達慢速停止

傳輸模組：

本研究使用之傳輸工具為藍牙模組HC-05(如圖4-8)，其目的在於將機器人所經過之路徑傳輸至遠端筆記型電腦。HC-05採用Cambridge Silicon Radio公司製作的BC417143晶片，使用藍牙2.1通訊協定，發射功率3*dBm*，通訊距離8*m*~10*m*，各腳位連接如表4-4。須注意的是，Arduino內建的序列阜監控窗會占用Pin 1、2腳位，因此使用HC-05時必須使用其他腳位，以免發生衝突。

圖 4-8 HC-05 藍牙模組

表 4-5 HC-05 腳位表

HC-05腳位	Arduino Uno 腳位
Vcc	+ 5 V
GND	GND
TXD	GPIO
RXD	GPIO

結合上述模組，本研究製作一架具兩輪之移動機器人(如圖 4-9)

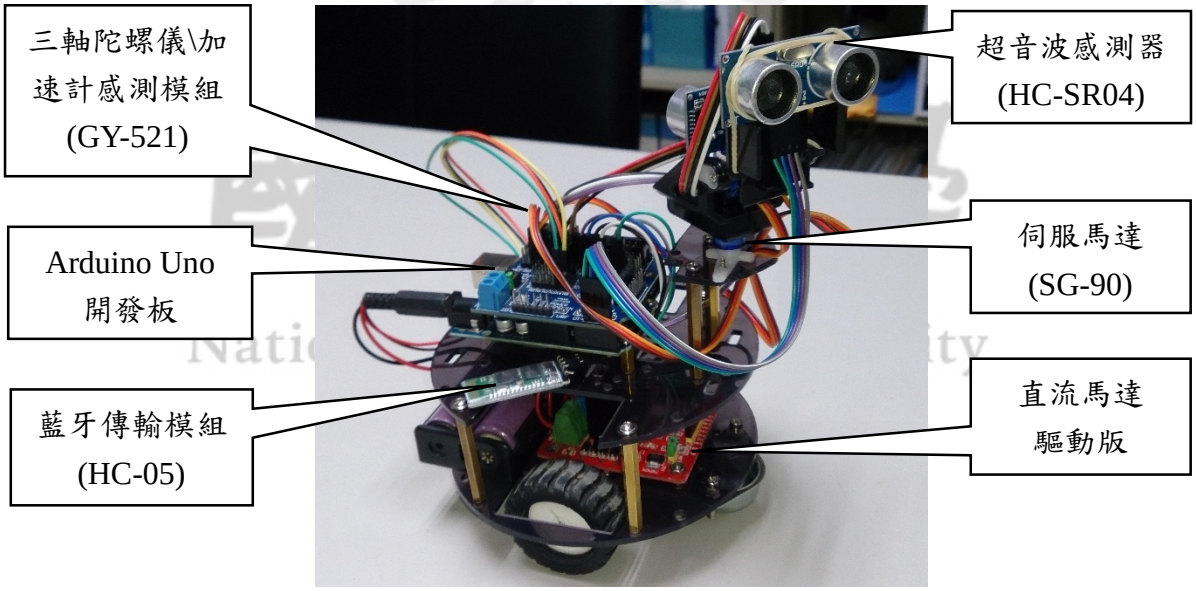


圖 4-9 機器人結構圖

4.2 演算法實作

將DACS演算法模擬流程圖(圖3-1)結合機器人功能方塊圖(圖4-1)可以得到機器人功能程式圖(圖4-10)。圖3-1中輸入環境參數之步驟，即為機器人獲取環境資訊的功能，以此為開端進行路徑規劃。

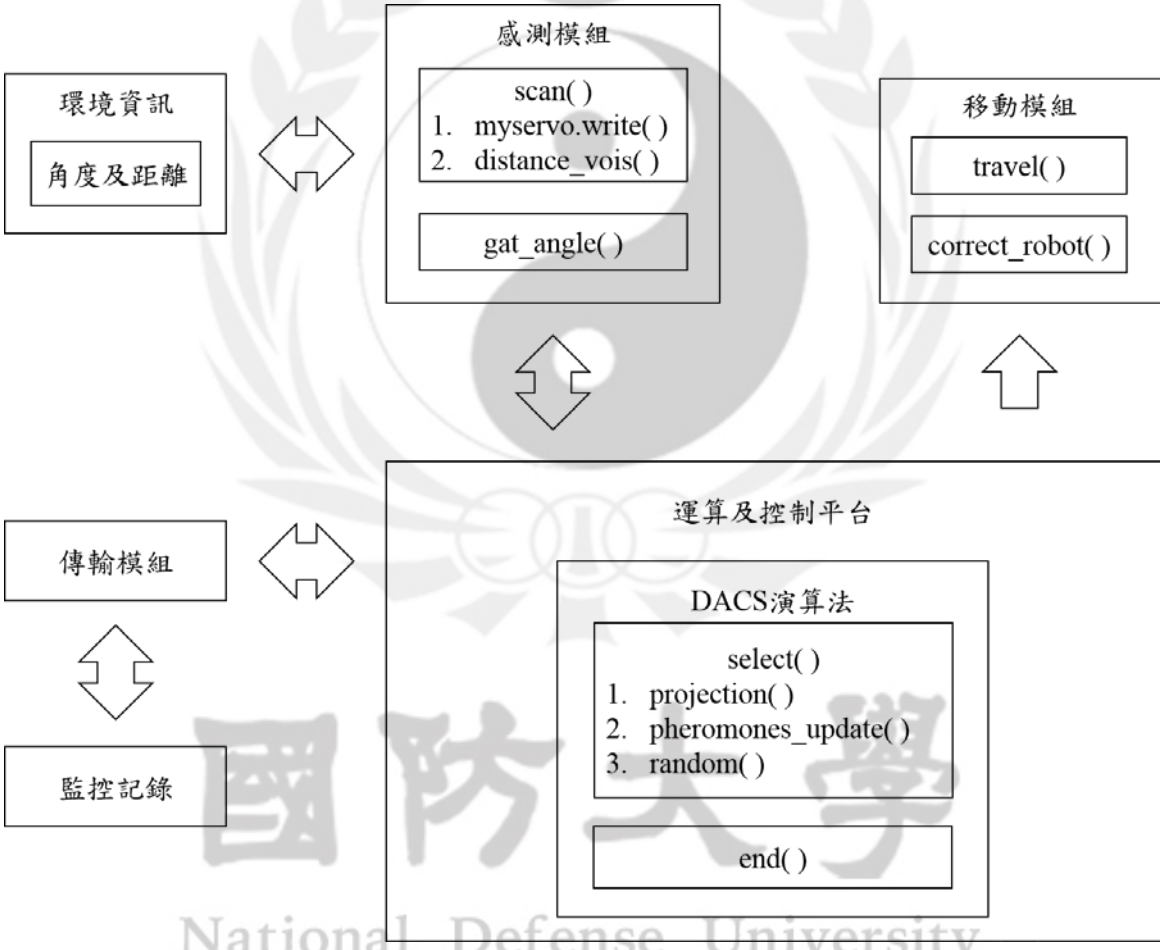


圖 4-10 機器人功能程式方塊圖

進行獲取環境資訊的步驟，本研究稱為Scan( )。函式Scan( )中包含了兩個子函式myservo.write( )及distance_vois( )。myservo.write( )會依據所輸入的

數值，驅動伺服馬達轉動至指定的角度；distance_vois( )則是利用超音波感測器感測該角度之距離，因此透過函式Scan( )可以得知環境中的角度及距離資訊，此結果將成為啟發值進行後續運用。執行Scan( )的程式條列如下：

Function：Scan( )

```
1       for ( ) {
2             myservo.write( angle )
3             distance_angle = distance_vois( )
4       }
```

取得環境資訊後，送至運算及控制平台函式Select( )運算。函式Select( )中包含了projection( )、pheromones_update( )及random( )三個子函式。projection( )是計算到下一節點的向量在前進方向上的投影量，再將此結果當成啟發值，pheromones_update( )是計算費洛蒙值及更新費洛蒙，藉由啟發值及費洛蒙值可得選擇該節點之機率值。random( )為Arduino的程式庫內建之函式，利用Arduino Uno 開發板未連接的任意類比腳位之雜訊來當亂數種子，此函數用於產生一隨機亂數，藉此來選擇下一路徑。

執行Select( )的程式條列如下：

Function：Select( )

```
1       heuristic = projection( )
2       pheromones = pheromones_update( )
3       probability = pow(pheromones, alpha) * pow(heuristic, beta)
4       randomSeed( analogRead(2) )
5             randNumber = random(probability)
6       return randNumber
```

執行projection()的程式條列如下：

```
Function：projection( )
1       p  = pow( distance, angle);
2       retun  p
```

執行pheromones_update( )的程式條列如下：

```
Function：pheromones_update( )
1       pheromones = pheromones * ( 1 + evaporating _ value )
2       if ( have gone though )
3             then pheromones = pheromones * ( 1 + laying _ value )
```

運算及控制平台得出之結果，傳送到移動模組的函式travel( )執行。由於機器人是利用兩顆直流馬達正反轉進行移動，而馬達運轉時產生的誤差，將導致機器人前進方向上的偏轉，因此本研究利用三軸陀螺儀感測誤差值，由函式get_angle( )獲取誤差值，之後再交由移動模組的函式correct_robot( )修正。

執行Correct_robot( )的程式條列如下：

```
Function：Correct_robot( get_angle( ) )
1       if ( get_angle( ) ＞ 0)
2             then turn right
3       else
4             then turn left
```

最後再由函式 end( )判斷是否達到終止條件，若是，則結束本次任

務；若否，則重複執行上述動作。

4.3 實驗與結果

本節將DACS演算法實際運用於機器人上，驗證該演算法於現實環境之適應力與存活力。

機器人之適應力：

適應力實驗所用之靜態地圖為一個5×5方格地圖(如圖4-11)，每一方格為30×30公分，並於右上角放置白紙標示為終點，起點則置於左下角位置。

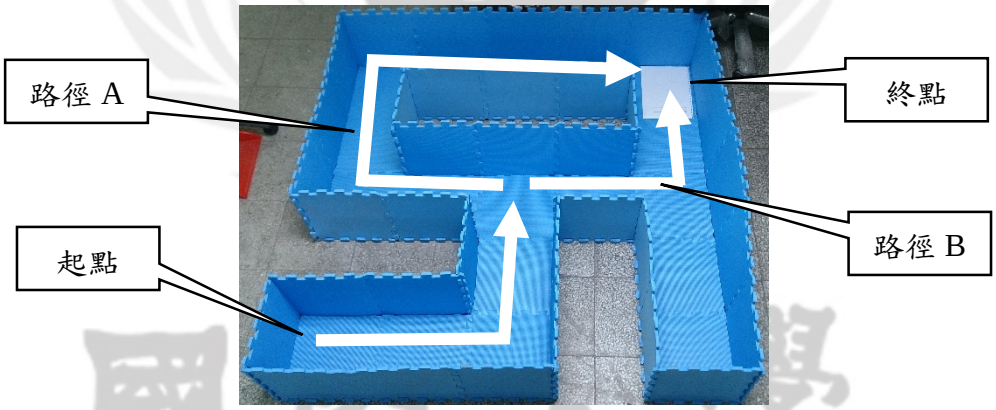


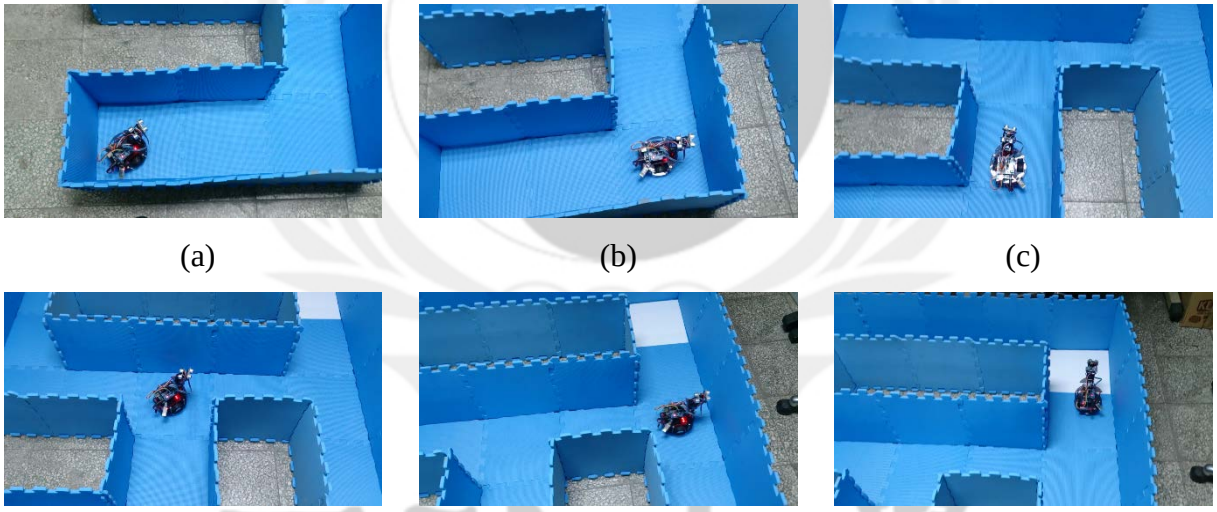
圖 4-11 靜態地圖

地圖中有兩條不同長度的路徑A及路徑B，並於實驗中讓機器人從起點至終點連續行走5次，以驗證機器人經過多次路徑規劃後，可否因為前次經驗而縮短抵達終點時間，實驗數據如表4-6，實際行走狀況如圖4-12。

表 4-6 靜態地圖測試次數所耗費時間

行走次數	耗費時間	偵測次數	碰撞障礙物次數
1	47秒	8	2
2	42秒	8	0
3	32秒	5	1
4	26秒	4	0
5	25秒	4	0

從表4-5可以看出機器人隨著行走次數增加，越趨於適應環境，因此所耗費的時間就越短，證明機器人具有依循前次經驗之能力得以適應環境。



(a) (b) (c)

(d) (e) (f)

圖 4-12 機器人於靜態地圖行走狀況

機器人之存活力：

存活力實驗所建置之動態地圖為一個5×5方格地圖(如圖4-13)，每一方格為30×30公分，並於右上角放置一張白紙標示為終點，起點則置於左下角位置。

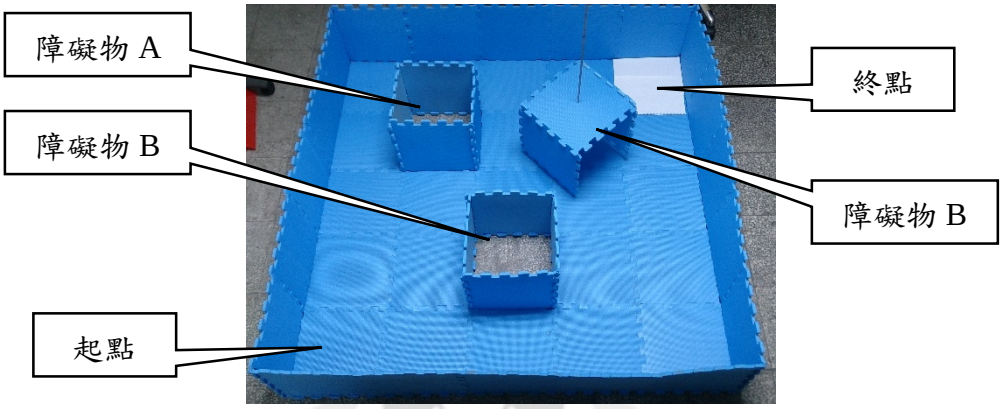
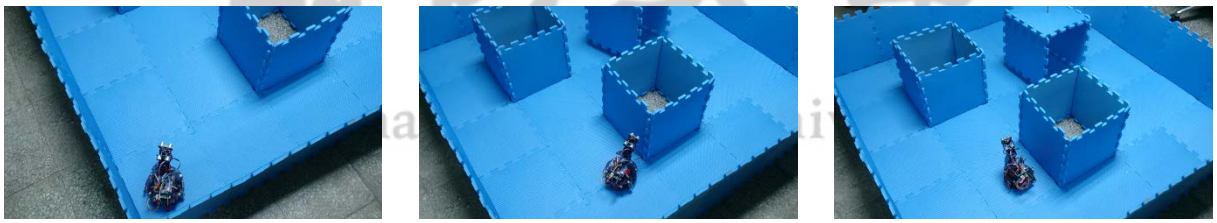


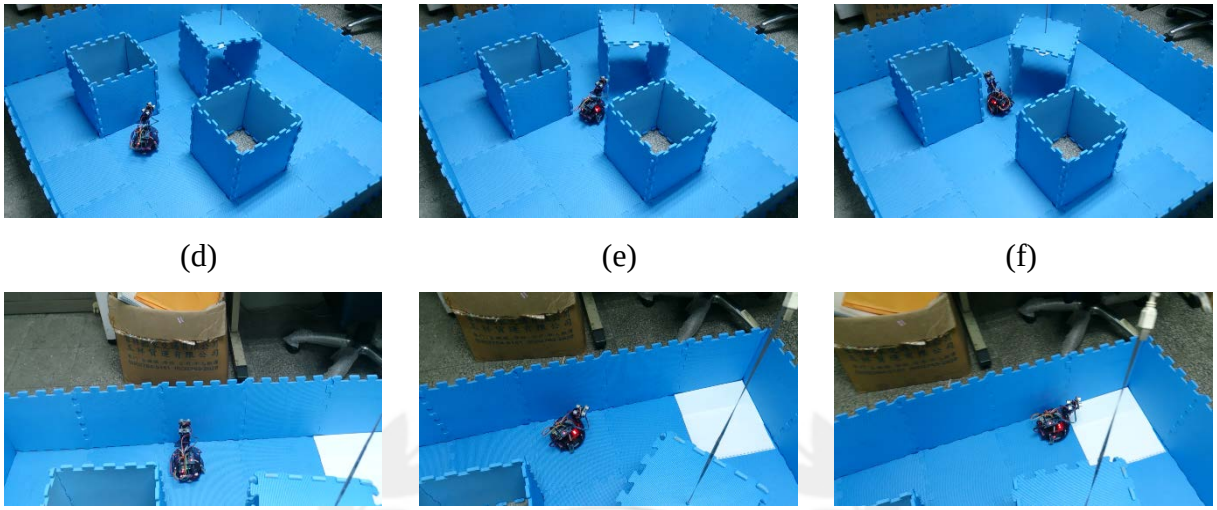
圖 4-13 動態地圖

地圖中共有障礙物A、B及C，其中障礙物A及B為固定式正方體障礙物；障礙物C則為一懸吊式冂字型動態障礙物，會隨著空氣的流動產生不定向的左右旋轉，藉此營造動態環境。圖4-12為機器人於動態環境實際行走之狀況。

從圖4-14(a)機器人開始前進，繞過障礙物B後(圖4-14(b))往障礙物A前進，在圖4-14(e)時障礙物C正好轉至不能通行的那一面，因此機器人選擇從障礙物A及障礙物C之間穿過(圖4-14(f))，最後繞過障礙物C到達終點。由存活力實驗可知，DACS在動態環境裡可以針對即時環境狀況作出反應。



(a) (b) (c)



(d) (e) (f)

(g) (h) (i)

圖 4-14 機器人於動態地圖行走狀況

本研究以伺服馬達結合超音波感測器獲取環境中之角度與距離，然而伺服馬達在各個角度均停留進行超音波偵測，完成360°偵測的時間約要2秒，而這段時間內機器人必須處於靜止，以保持獲取之環境資訊的正確性。若在這段時間內有外力作用於機器人，則機器人毫無反應能力。因此，若能縮短機器人獲取環境資訊的時間，就能提升機器人反應能力，進而增加存活力。

第五章、結論與未來展望

5.1 結論

螞蟻演算法比其他啟發式演算法更適合運用在複雜的環境中，但是螞蟻演算法在進行搜尋的過程是隨機的，雖然有助於跳脫出區域最佳解，但有時候會出現索搜尋的區域並不是目標的區域，因此本研究提出 DACS 演算法，改良 ACS 演算法僅依據兩相鄰節點間之距離作為判斷啟發資訊依據，將角度因子加進啟發資訊，使得演算法具有方向概念，在動態環境下適應力及存活力比傳統的 ACS 演算法更有優勢。

在模擬方面，本研究設計動態實驗利用節點移動、消失、增加及目標變換來模擬現實環境的動態變化，實驗結果顯示 DACS 演算法不僅能依照環境的變動調整規劃的路徑，也顯示具有跳出區域解的能力，展現了在動態環境下的存活力。本研究亦將 DACS 演算法與 ACS 演算法進行 500 次模擬實驗，結果顯示 DACS 演算法較 ACS 演算法之平均最佳路徑距離短 46.5%、平均迭代次數少 28.5%、平均時間少 63.8%，明顯較 ACS 具適應力優勢。

5.2 未來研究

從本研究實驗可以得知，僅將角度因子結合啟發資訊，就可以加速螞蟻演算法的收斂且維持解的品質，但現實環境中還有許多因子會影響路徑的

規劃，如機器人所剩餘的電量、所處環境的溫溼度、路徑的坡度等等，假使能考慮多種環境因素，則 DACS 演算法則會越趨於完善。

另於系統實作方面，因本研究係建立在完整環境資訊已逐步搜集完成，以及系統定位與障礙物迴避皆已有效解決之前提上，故所提之演算法需結合適當定位及障礙物迴避演算法始能有效運作。在未來研究上可在定位及障礙物迴避演算法上精進，以期提升整體效能。

【參考文獻】

中文部分

朱大奇、顏明重(2010)。移動機器人路徑規劃技術綜述，**控制與決策**，25:7，961-967。

李孟軒(2011)。**輪型機器人之路徑追蹤與避障**(碩士論文)，國立中央大學電機工程研究所。

林豐澤(2005)。演化式計算下篇：基因演算法以及三種應用實例，**智慧科技與應用統計學報**，3:1，9-56。

徐德發(譯)(2012)。**Arduino 錦囊妙計**(第2版) (原作者：Michael Maegolis)，台北：碁峯。

趙英傑(2014)。**超圖解Arduino互動設計入門**(第2版)，台北：旗標。

艾迪諾(2014)。**Arduino全能微處理機實習**，台北：全華。

英文部分

Avneesh Sud, Erik A., Sean C., Ming L. and Dinesh M.. (2008). Real-time Path Planning in Dynamic Virtual Environment Using Multiagent Navigation Graphs. *IEEE Transactions on Visualization and Computer Graphics*, (14:3), pp.526-538.

Canny J. F., 1988, *The Complexity of Robot Motion Planning*. Boston, MIT Press.

Christian Hofner and Gunther S.. (1994). Path Planning and Guidance Techniques for an Autonomous Mobile Robot. *IEEE Intelligent Robots and Systems*, (1), pp.610-617.

De Carvalho R.N., Vidal H.A., Vieira P. and Ribeiro M.I.. (1997). Complete Coverage Path Planning and Guidance for Cleaning Robots. *IEEE International Conference Industry Electrontics. Guimaraes*, (2), pp.677-682.

Dong Jin Seo and Jongwoo K.. (2013). Development of Autonomous Navigation System for an Indoor Service Robot Application. *IEEE Proceedings of 13th International Conference on Control Automation and Systems*, pp.204-206.

Dorigo M., Maniezzo V. and Colorni A.. (1996). The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, (26:1), pp.29-41.

Dorigo M. and Gambardella L.M.. (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, (1:1), pp.53-66.

Fatemeh K. P., Fardad F. and Reza S. N.. (2013). Comparing the Performance of Genetic Algorithm and Ant Colony Optimization Algorithm for Mobile Robot Path Planning in the Dynamic Environments with Different Complexities. *Journal of Academic and Applied Studies (JAAS) Int. Association for Academies, Canada, (3:2)*, pp.29-44.

John Holland. (1975). *Adaptation in Natural and Artificial Systems*, Adaptation in natural and artificial systems MIT Press Cambridge, MA, USA.

Kikuo Fujimura and Hanan S.. (1989). A Hierarchical Strategy for Path Planning Among Moving Obstacles. *IEEE Transactions on Robotic and Automation*, (5:1), pp.61-69.

Peng .Y and W. Wei. (2006). A New Trajectory Planning Method of Redundant Manipulator Based on Adaptive Simulated Annealing Genetic Algorithm. *IEEE International Conference on Computational Intelligence and Security*, (1), pp.262-265.

Purian F.K. and Sadeghian E.. (2013). Mobile Robots Path Planning Using Ant Colony Optimization and Fuzzy Logic Algorithms in Unknown Dynamic

Environments. *Proceedings of International Conference on Control, Automation, Robotics and Embedded Systems (CARE)*, pp.1-6.

Robert A. Conn and Moshe K.. (1998). Robot Motion Planning on *N*-Dimensional Star Worlds Among Moving Obstacles. *IEEE Transactions on Robotic Automation*, (14:2), pp.320-325.

Schmidt G. and Hofner C.. (1998). An Advaced Planning and Navigation Approach for Autonomous Cleaning Robot Operation. *IEEE International Conference on Intelligent Robots System*, (2), pp.1230-1235.

Takahashi O. and Schilling R. J.. (1989). Motion Planning in a Plane Using Generalized Voronoi Diagrams. *IEEE Transactions on Robotics and Automation*, (5:2), pp.143-150.

Vitorino Ramos and Juan J. M.. (2002). Self-Organized Stigmergic Document Maps: Environment as a Mechanism for Context Learning. *1st Spanish Conference on Evolutionary and Bio-Inspired Algorithms*, pp. 284-293.

Yang Chen, Lei C., Huaiyu W. and Yanhua Y.. (2014). Receding Horizon Control for Mobile Robot Path Planning in Unknown Dynamic. *Proceedings of 26th Chinese Control and Decision Conference*, pp.1505-1509.

網路部分

大兵萊恩一路直前 (http://gogoprivateryan.blogspot.tw/2014/07/mpu-6050-google.html) [visited in 2016/01/21]

Arduino (available online at http://www.arduino.cc/) [visited in 2015/12/13]

Romain Hendrickx and Leonardo Bezerra "About Ant Colony Optimization", March 2012 (available online at http://www.aco-metaheuristic.org/about.html). [visited in 2015/09/07]

Gyroscopes and Accelerometers on a Chip (http://www.geekmomprojects.com/gyroscopes-and-accelerometers-on-a-chip/) [visited in 2016/01/21]