淡江大學資訊工程學系碩士班

碩士論文

指導教授：林其誼 博士

利用改良 A*演算法之智慧停車場路徑規劃與效率優化

Path Planning and Efficiency Optimization in Smart Parking Systems Using an Improved A* Algorithm

研究生：呂家成 撰

中華民國 114 年 6 月

論文名稱：利用改良 A*演算法之智慧停車場路徑規劃與效率優化

頁數：64

校系(所)組別：淡江大學資訊工程學系碩士班

畢業時間及提要別：113 學年度第 2 學期碩士學位論文提要

研究生：呂家成　　指導教授：林其誼 博士

論文提要內容:

隨著都市化進程加速，停車場內的交通壅塞問題日益嚴重，尤其在高負載或突發事件情況下，傳統管理方法無法有效處理即時交通變化。本研究提出了一套適用於停車場環境的動態路徑規劃系統，透過改良式 A* 演算法與動態等待時間機制，有效預測並主動避開潛在的壅塞點，大幅降低車輛的平均行駛與等待時間。此外，本研究設計了動態路徑重規劃機制，能即時處理如車道臨時封閉等突發事件，快速重新規劃替代路徑。

實驗結果顯示，改良式 A* 演算法可將車輛整體平均等待時間降低約 62.02%，並且動態重規劃機制在突發事件情境下之重新規劃成功率達到 100%，證明本研究提出之方法在緩解停車場壅塞問題上具有高度即時性與

可靠性，具備良好的實務應用與推廣潛力。

關鍵字：停車場；改良式 A* 演算法；動態等待時間；動態路徑重規劃；壅塞管理；模擬實驗

Title of Thesis:	Path Planning and Efficiency Optimization in Smart Parking Systems Using an Improved A* Algorithm		Total pages:64
Key word:	parking lot; improved A* algorithm; dynamic waiting time; dynamic path replanning; congestion management; simulation experiments		
Name of Institute:	Master's Program, Department of Computer Science and Information Engineering, Tamkang University		
Graduate date:	06/2025	Degree conferred:	Master
Name of student:	Jia-Cheng Lu 呂家成	Advisor:	Dr. Chi-Yi Lin 林其誼 博士

Abstract:

With the acceleration of urbanization, traffic congestion within parking lots has become increasingly severe. Traditional management methods are inadequate to effectively handle real-time traffic changes, especially under high-load conditions or unforeseen incidents. This study proposes a dynamic path planning system tailored for parking lot environments, integrating an improved A* algorithm with a dynamic waiting time mechanism to proactively predict and avoid potential congestion points, significantly reducing vehicles' average travel and waiting times. Additionally, this study develops a dynamic path replanning mechanism capable of swiftly handling unexpected incidents, such as temporary lane closures, by promptly recalculating alternative routes. Experimental results demonstrate that the improved A* algorithm reduces the average vehicle waiting time by approximately 62.02%, and the dynamic replanning mechanism achieves a 100% success rate in replanning under incident conditions. The results indicate that the proposed methods exhibit high immediacy and reliability in mitigating parking lot congestion issues, highlighting strong potential for practical application and further development.

According to “TKU Personal Information Management Policy Declaration“, the personal information collected on this form is limited to this application only. This form will be destroyed directly over the deadline of reservations.

表單編號：ATRX-Q03-001-FM031-03

目錄

圖目錄

表目錄

第一章 緒論

1.1 研究動機與背景

隨著都市化進程的加速和車輛數量的快速增加，停車場的使用需求與日俱增。尤其在都市中，停車場空間往往有限，車輛進出頻繁，導致停車場內的交通壅塞問題日益嚴重[1][2][3]。傳統停車場管理通常依賴人工引導或靜態的規劃系統，無法有效應對突發狀況和動態壅塞現象[3][4][5]，造成車輛進出效率低、等待時間延長，甚至進一步影響周邊道路的交通流暢度[6]。因此，提升停車場內的車輛行駛效率和改善壅塞狀況，已成為現代交通管理領域的重要議題之一[7][8][9]。

在停車場環境中，車輛等待和壅塞的原因主要來自兩個方面。首先，由於停車位數量有限，車輛在尋找停車位或等待其他車輛正在倒車入庫時，經常會造成後續車輛的停滯[9]。此外，停車場內的道路結構通常狹窄且受限，只能容納一個方向或兩個方向各一個車道，無法進行超車或並行，這樣就算只有少數車輛延誤，也可能導致明顯的壅塞現象[10][11][12]。此類情況若未能妥善處理，不僅會影響停車場的使用效率，甚至可能進一步造成顧

客的不滿和經營者的損失[13]。

另一方面，停車場內偶爾出現的突發事件，如臨時的道路封閉、車輛故障或事故，往往會進一步加劇壅塞情況[14]。此類事件通常難以事先預測，也難以透過靜態的路徑規劃方式來解決。因此，如何在停車場管理系統中導入能夠即時應對突發事件的動態路徑重規劃機制，成為實際營運中必須面對的問題[4][15][16]。

為了解決上述問題，學術界和業界普遍使用各種路徑規劃演算法，如 Dijkstra、A* 等，其中 A* 演算法因為具有良好的效率和靈活性而受到廣泛關注[17]。然而，傳統的 A* 演算法多用於靜態或可預期的交通情況，對於停車場內高度動態且多變的環境，可能無法及時應對車輛壅塞或突發狀況[18][19]。因此，本研究的動機在於如何改進現有的 A* 演算法，結合停車場內即時的動態資訊，建立出能夠有效地緩解壅塞問題，並且能即時處理突發事件的停車場動態路徑規劃系統，從而提升整體交通運行效率，達成實際管理與智慧交通的目標。

1.2 研究目的

根據上述研究背景和動機，本研究旨在停車場環境提出一套完整的動態路徑規劃系統，透過改良傳統的 A* 演算法，並結合動態等待時間和動態路徑重規劃兩個子機制，來有效解決停車場內的壅塞和突發事件問題。具體研究目標如下：

（1） 提升車輛通行效率

目前停車場大多採用靜態路徑規劃或人力引導方式，缺乏即時的壅塞處理能力。因此本研究透過動態等待時間機制對傳統的 A* 演算法進行改良，使得車輛可以提前預測並主動避開壅塞點，降低整體的行駛和等待時間，進一步提升停車場運行效率[5][6][7][8]。

（2） 建立即時壅塞預測機制

因為動態因素的影響，停車場內的壅塞情況經常難以預測，傳統方法難以應對。因此本研究建立動態等待時間預測機制，讓系統在路徑規劃階段就能動態評估潛在壅塞點，協助車輛提前選擇較佳的路徑，從源頭就有效地緩解壅塞[5][6][20]。

（3） 加強突發事件處理能力

停車場內的突發事件（如臨時道路封閉或事故）通常難以即時處理，容易造成大規模的壅塞。因此本研究設計了一種即時動態路徑重規劃機制，確保受影響車輛能夠立即取得替代路徑並順利抵達目標停車位，避免突發事件導致交通壅塞進一步惡化[8][20]。

（4） 建立完善的驗證方法

為了確認本研究提出的動態路徑規劃系統的實際效能，本研究設計了互補且系統化的實驗和驗證流程。透過效能統計模擬和動態事件模擬，針對不同情境進行驗證和分析，呈現具體且量化的結果，完整驗證系統的效用和價值[7][20]。

（5） 提供實際應用和研究基礎

本研究希望提出的方法具備理論創新和實際應用價值，透過完整的實驗驗證和數據分析，為未來的學術研究和產業應用提供紮實且明確的參考基礎，推動停車場管理系統和智慧交通領域的發展[5][7][8]。

1.3 論文架構

本論文共分為五個章節，分別為緒論、技術背景與相關研究、研究方法設計、實驗結果與分析，以及結論與未來研究。各章節的主要內容如下：

第一章 緒論：首先闡述本研究的背景和動機，說明停車場環境中存在的壅塞和突發事件問題，進一步提出本研究的具體目的和研究動機，並簡要描述論文的整體架構。

第二章 技術背景與相關研究：本章介紹路徑規劃相關演算法的基本原理和發展歷程，包括 Dijkstra 演算法、傳統 A* 演算法及其延伸的變體。此外，也探討停車場環境中現有的動態路徑規劃方法，並評估每種方法的優缺點，以突顯本研究方法的創新與價值，作為後續研究方法的理論基礎。

第三章 研究方法設計：本章詳細說明本研究所提出的動態路徑規劃系統，包括停車場環境地圖模型的建立、動態等待時間機制的設計與實現、改良式 A* 演算法的具體做法、突發事件下的動態路徑重規劃機制，以及系統整體運行的多執行緒架構。最後也對系統的設計理念、流程、和實作細節進行詳細說明。

第四章 實驗結果與分析：本章分別針對「效能統計模擬程式」和「動態路徑重規劃模擬程式」進行實驗驗證。透過實際的模擬數據，詳細比較傳統 A* 和改良式 A* 在行駛和等待時間方面的效能差異，並分析了動態路徑重規劃機制在突發事件下的即時性和可靠性，給出了具體的實驗成果和深入的討論。

第五章 結論與未來研究：最後一章總結了本研究的實驗結果，總結研究成果和貢獻，並明確指出未來可以繼續探討和延伸的研究方向，以供後續研究者或業界應用參考。

第二章 背景技術與相關研究

本章主要對本研究所涉及的技術背景進行說明，首先將介紹路徑規劃演算法的基本概念和相關發展，接著再講解傳統的 A* 演算法的基本原理和應用。此外，將針對停車場壅塞管理和動態路徑重規劃的現有研究進行探討，作為本研究的理論基礎和創新基礎。

2.1 路徑規劃演算法概述

路徑規劃（Path Planning）是現代智慧交通系統、自動駕駛和機器人導航等領域的重要技術，其核心目標在於有效地找到從起點到終點的最佳或次佳路徑，藉此提高整體系統的運行效率和可靠性。根據不同的應用情境，最佳路徑的定義可能包含最短路徑、最短時間、最低成本等多種考量因素[19][21][22]。目前常見的路徑規劃演算法主要可以區分為以下幾種類型：

（1）Dijkstra 演算法

Dijkstra 演算法由荷蘭電腦科學家 Edsger Wybe Dijkstra 於 1959 年提出，是一種廣泛使用在圖形理論的最短路徑演算法[23]。此演算法的基本原理是從起點出發，逐步尋找並確認到其

他節點的最短路徑。Dijkstra 演算法能保證在非負權重圖中求得最短路徑，但缺點是計算複雜度比較高，特別是當節點數量龐大時，效率可能會明顯下降。

（2）廣度優先搜尋 (Breadth-First Search，簡稱 BFS)

BFS 是一種基於層次展開的圖搜尋方法，主要用於圖中所有邊的權重相同的情況。此演算法從起始節點開始，逐層往外擴展，能確保找到最短的路徑。

然而，BFS 的計算成本會隨節點數增加而迅速上升，因此在大規模環境中效率有限[24][25]。

（3）深度優先搜尋 (Depth-First Search，簡稱 DFS)

DFS 透過遞迴或堆疊結構，沿著一個方向不斷深入搜尋，直到無法繼續前進，再回溯尋找其他路徑[26]。此方法不保證找到最短路徑，但能快速探索特定路徑的可行性，因此經常被用於某些特定情況或問題求解上，例如迷宮求解或路徑遍歷問題[24]。

（4）A* 演算法

A* 演算法是一種啟發式搜尋方法，1968 年由 Hart、Nilsson 和 Raphael 所提出[17]。其透過一個啟發式函數 (heuristic function)

有效地降低節點的搜尋範圍，使得路徑搜尋的效率明顯高於 Dijkstra 演算法。 A* 演算法已被廣泛應用於導航系統、無人車路徑規劃、遊戲 AI 等多種領域[15]，成為目前最常使用且研究廣泛的路徑規劃演算法之一。

本研究所提出的方法，主要基於 A* 演算法進行改良，以提升其在動態且具有壅塞特徵的停車場環境中的效能和適應性。下一節將針對 A* 演算法的基本原理和現有應用進行更詳細地介紹和討論。

2.2 傳統 A* 演算法之基本原理及應用

A* 演算法是一種廣泛使用且高效率的啟發式路徑搜尋演算法，透過評估節點的成本函數，有效地減少搜尋空間，快速找到從起點到目標節點的最佳路徑或次佳路徑。此演算法相較於其他搜尋方法（如 Dijkstra 或 BFS）的優勢，在於結合了過去成本和未來成本的預測，有效降低了不必要的路徑探索，從而達成快速搜尋的目的[17]。

（1）A* 演算法的運作原理基於以下三個重要函數：

$g(n)$：表示從起點到節點 n 所實際花費的成本。

$h(n)$：稱為啟發式函數（Heuristic Function），表示從節點 n 到目標節點的預估成本，通常以歐幾里得距離、曼哈頓距離等方式估算。

$f(n)$：綜合成本函數，定義為 $g(n)$ 和 $h(n)$ 的和，即為公式：

$$f(n) = g(n) + h(n)$$

在搜尋過程中，A* 演算法會對待擴展的節點依照綜合成本 $f(n)$ 排序，總是選擇成本最低的節點進行探索，並持續更新成本最低的路徑，直到找到目標節點為止。如果啟發式函數 $h(n)$ 滿足可接受條件（即永遠不會高估實際成本），則可以保證 A* 演算法必能找到最短路徑。

（2）A* 演算法的具有下列特點和優勢：

- 高效性：由於使用了啟發式函數，可以快速減少搜尋空間，在大型地圖或複雜環境中特別有效，搜尋效率通常比 Dijkstra 演算法好很多。
- 保證最佳解：在合理選擇啟發式函數的條件下，A* 能夠確保找到最短路徑，並且有嚴謹的理論基礎。
- 靈活性和廣泛的適用性：A* 演算法容易和其他路徑成本

計算機制整合，已廣泛應用於導航系統、智慧交通、自動駕駛、機器人路徑規劃和電子遊戲 AI 等多種領域[15][27]。

（3）A* 演算法的實際應用與侷限性

A* 演算法由於具備高效性和可靠性，目前已廣泛被應用於多個領域：

- 汽車導航系統：為汽車或無人車規劃快速、最短路徑。
- 機器人導航：在有障礙物的環境中迅速找到最短或最佳路徑。
- 遊戲中的 AI 路徑規劃：讓非玩家角色（NPC）能夠即時找到目標位置的最佳路徑，提升遊戲體驗[28]。

然而，儘管傳統 A* 在靜態或變化不大的環境中表現良好，但當應用於高度動態和無法預測的環境（例如即時變動的交通壅塞或突發事件）時，傳統 A* 的效能往往會受到嚴重影響。原因是在於傳統的 A* 無法動態更新路徑成本或迅速調整路徑規劃以應對即時變化，進而可能導致路徑搜尋失敗或找到的路徑不再是最佳路徑。

因此，如何對傳統 A* 演算法進行改良，使其能夠更即時地

處理動態壅塞或突發事件，已成為近年相關研究的重要議題之一[18][29][30]。

2.3 停車場壅塞管理相關研究

當今停車場內的交通壅塞問題在都市越來越嚴重，對於停車場管理的效率、顧客滿意度以及周邊交通的順暢性，都造成極大的負面影響。因此，近年來學術界和業界提出了多種方法試圖解決或緩解停車場內的壅塞問題，以下將針對相關研究進行探討。

2.3.1 傳統靜態管理方法

傳統的停車場管理通常以人工引導或固定的路徑規劃為主，如劃定固定行駛路線、指定停車區域或分配固定停車格。這些方法實施成本較低且操作簡便，但存在明顯的侷限性，無法即時應對突發的交通壅塞狀況或車流變化；也無法動態調整，難以應付不同的車流狀況和突發事件[31]。

2.3.2 動態資訊導引系統

為解決上述問題，近年來動態資訊導引系統的研究與應用逐漸受到重視。此類系統利用感測器或影像辨識技術，即時掌握停車位使用狀況，並將空位資訊回傳給進場車輛，引導駕駛快速找

到空車位。其主要特點包括：

- 提供即時資訊：透過感測器或影像辨識，即時偵測並回傳空車位狀態，並為駕駛規劃最短導引路徑[32]。
- 縮短尋位時間：減少車輛在停車場內的尋位時間，間接緩解場內及周邊道路的壅塞[33]。

然而，儘管此類系統能即時指示空車位，卻尚未整合車輛實際行駛路徑與壅塞熱點的動態規劃與管理機制，因此仍難以從根本上消除因倒車、調頭等行為所造成的交通阻塞[34]。

2.3.3 模擬與演算法輔助管理方法

近期研究中逐漸引入各種模擬工具和路徑規劃演算法來輔助停車場管理。這些方法可以更詳細地分析壅塞的原因，並提供更動態的解決方案：

- 演算法路徑規劃：透過 A*、Dijkstra 等演算法主動規劃車輛的行駛路徑，並預測潛在的壅塞點，降低等待與行駛時間[20]。
- 離散事件模擬：以模擬軟體分析不同的停車和駕駛策略對停車場效率的影響，協助管理單位找到最有效的管理

方案[35]。

然而，大部分研究仍著重於靜態環境或預設的情境，較少考慮停車場內的突發事件（例如臨時車道封閉）對壅塞情況直接造成的影響。

2.3.4 動態重規劃方法

少數研究已開始關注如何處理停車場內的突發事件，提出了動態路徑重規劃方法。此類方法透過即時偵測和重新規劃技術，能夠有效應對突發事件並即時調整車輛行駛路徑，即時偵測突發事件，如車輛故障、意外事故或臨時道路封閉[14]。透過動態路徑規劃演算法快速提供替代路徑，避免進一步壅塞擴大[36]。

雖然這類研究正逐漸受到重視，但目前仍處於初步階段，尚未廣泛整合動態等待時間和路徑重規劃機制，且缺乏足夠的效能驗證和完整的系統實現。

2.3.5 本研究的切入點與創新點

綜合以上相關研究，本研究的切入點和創新點主要有以下幾點：

（1）改良式 A* 結合動態等待時間機制：對傳統的 A* 演算

法進行改良，透過主動預測壅塞並動態調整路徑成本，提升整體通行效率與即時性。

（2）融入動態路徑重規劃機制： 開發了一套可即時偵測突發事件並快速提供替代路徑的機制，解決傳統管理系統無法即時處理突發事件的問題。

第三章　研究方法設計

本章主要介紹本研究提出的停車場車輛動態路徑規劃系統的方法設計。首先，對停車場環境設置和地圖模型進行說明；其次，詳細描述動態等待時間機制和壅塞預測；接著討論動態路徑重規劃機制，最後解釋系統內的多執行緒架構、車輛生成方法、資料結構、策略設計和未來的可擴展性。

3.1 停車場環境與地圖模型

本研究以模擬真實停車場的運行，建構網格狀的停車場地圖模型，用來評估改良式 A* 的效能。地圖模型主要包括以下五種基本網格類型：

- ENTRANCE（入口）：表示車輛進入停車場的起始位置，供車輛進入模擬的停車場環境。
- AISLE（車道）：提供車輛行駛的空間，讓車輛自由移動。
- WALL（牆壁）：不可通行的障礙物，以模擬實體的牆壁結構，車輛規劃路徑時須避開此種網格。
- PARKING_SPACE（停車格）：車輛停放的位置，車

輛只能停留，不能穿越通行。

- VEHICLE（車輛）：正在移動或停放中的車輛，它可以用來標示網格目前的占用狀態。

此外，本研究也特別定義了一種特殊類型的網格：

CLOSED_AISLE（臨時封閉車道）：表示原本可以通行的車道因臨時突發事件而暫時無法通行，CLOSED_AISLE 是從 AISLE 動態轉換的，以觸發動態路徑重規劃功能。

在模擬過程中，最關鍵的判斷為網格的可通行性。WALL、PARKING_SPACE 和 CLOSED_AISLE 是不可通行的網格，演算法在規劃路徑時必須避開這些位置。

3.2 動態等待時間機制

針對停車場內車輛因壅塞導致的等候時間增加的問題，本研究提出了一種動態等待時間預測與管理機制，該機制的核心在於車輛進入停車場時，主動預測並動態量化每輛車沿途可能遇到的潛在壅塞點（以下稱等待點）以及對應的等待時間，從而提高停車場整體行駛效率，有效減少壅塞情況。

傳統 A* 演算法的成本函數定義如下：

$$f(n) = g(n) + h(n)$$

其中，$g(n)$ 是從起點到節點 n 的實際累積成本，$h(n)$ 是從節點 n 到目標節點的預估成本，通常僅使用空間距離估算。本研究針對啟發式函數 $h(n)$ 進行改良，將動態等待時間機制所計算出的潛在等待時間納入考量，使演算法能更有效地處理路徑中的動態壅塞情境。

當車輛剛進入停車場時，系統立即為其目標停車位相鄰的唯一車道位置，設定動態等待時間（waitTime），這個位置稱為等待點。此等待點的動態等待時間主要由以下三個因素組成：

（1）車輛在車位前進行倒車入庫作業所需的固定時間（本研究假設為 10 秒）。

（2）車輛從目前位置到達等待點所需的行駛時間（假設車輛每秒移動一格）。

（3）車輛行駛路徑中，因前方其他車輛造成的額外壅塞時間。

例如，車輛於進入停車場時，距離指定的等待點有 15 格路程，系統首先將該距離換算為 15 秒的行駛時間，然後加上固定的

10 秒倒車時間，這意味著該等待點的初始等待時間為 25 秒。隨著車輛每秒地向前移動，等待時間會根據車輛實際行駛距離動態縮短，真實反映行駛狀態和潛在壅塞情況。

在改良後的 A* 演算法中，啟發式函數 $h(n)$ 被重新定義為：

$$h(n) = dist(n, goal) + delay(n)$$

其中，$dist(n, goal)$ 表示節點 n 到目標節點的空間距離（本研究採用曼哈頓距離）；$delay(n)$ 則是節點 n 所受到的潛在等待時間延遲，計算方式如下：

如果車輛在行駛過程中遇到其他車輛先前已設定的等待點時，系統將計算目前行駛中車輛所受到該等待點潛在的額外等待時間影響，計算公式如下：

remain = waitTime - baseG

其中各個參數的定義為：

- remain：表示「目前行駛中的車輛」抵達該等待點時可能額外需要等待的秒數。
- waitTime：表示「先前已經設定該等待點的其他車輛」目前尚未使用完畢的剩餘等待秒數。

- baseG：表示「目前行駛中車輛」從當前位置到達該等待點所需的預估行駛秒數。

以下透過一個實際範例更具體地說明上述公式的運作方式：

假設車輛 A 已經設定了一個等待點，目前該等待點的剩餘等待時間 (waitTime) 為 20 秒。此時車輛 B 從入口處進入停車場，並預估從目前位置抵達該等待點需耗時 15 秒 (baseG)。系統使用上述公式計算出車輛 B 的額外等待時間：

$$remain = waitTime - baseG = 20\ 秒 - 15\ 秒 = 5\ 秒$$

由於 remain 值為 5 秒，表示車輛 B 抵達該等待點時，車輛 A 仍在進行倒車入庫，因此車輛 B 需額外等待 5 秒。此 5 秒將會被計入車輛 B 路徑規劃時的額外等待成本中，亦即納入 $delay(n)$。

然而，若使用上述公式所計算出的 remain 值超過 10 秒，例如等待點的剩餘等待時間 (waitTime) 為 30 秒，而車輛 B 預估到達該點的時間 (baseG) 為 15 秒：

$$remain = waitTime - baseG = 30\ 秒 - 15\ 秒 = 15\ 秒$$

因為 remain 值（15 秒）超過了 10 秒，代表原先設定此等待點的車輛 A 實際上尚未抵達等待點，仍然在前往等待點的途中，並未真正開始倒車入庫動作。換句話說，此時該等待點並沒有立即產生壅塞影響，因此系統便將 remain 視為 0 秒，不會將這段過多的等待時間計入後續的路徑成本中。

此外，本研究特別引進了 occupiedBy 機制，來解決兩輛車輛同時為同一等待點設定等待時間的衝突情況。所謂「兩輛車輛同時為同一等待點設定等待時間」，指的是當兩車於各自的路徑規劃階段，皆判斷出其將於不同時刻抵達同一個等待點，且都認為該等待點將會因自己倒車入庫而產生一定的等待時間。然而，實際上該等待點在特定時間內僅能有一輛車輛倒車進入停車格，若兩車皆認為自身會造成此等待點的等待，可能導致後續車輛路徑規劃時的等待時間產生頻繁變化，造成系統的不穩定與誤判。

為了避免上述情況，系統引進了「最晚離開」的原則，也就是所有欲佔用同一等待點的車輛中，將倒車入庫結束時刻最晚的一輛車輛設為該等待點的主要佔用車輛 (occupiedBy)，並以這輛車輛的倒車時間做為等待點的標準等待時間。其他車輛則會忽略自身對此等待點的設定，避免重複設定等待時間，從而穩定後續

的路徑規劃。

以下提供一個實際範例說明 occupiedBy 機制的運作方式：

假設停車場內車輛 A 與 B 分別要停入兩個面對面的車格，由於本系統會在車輛進場時為「目標停車格前方的車道」設定等待點，因此其等待點會落在同一格車道上，其共同等待點為位置 P：

- 車輛 A 預估會在第 30 秒完成倒車入庫並離開等待點 P。
- 車輛 B 預估會在第 35 秒完成倒車入庫並離開等待點 P。

在一般情況下，若無 occupiedBy 機制，等待點 P 將在兩輛車各自的路徑規劃中被重複設定為 30 秒與 35 秒，導致後續經過此等待點的其他車輛，無法正確判斷實際等待時間。

透過 occupiedBy 機制，系統會選擇上述兩輛車輛中「最晚離開」等待點的車輛 B（即第 35 秒才離開等待點）作為等待點 P 的主要佔用車輛（occupiedBy）。此時等待點 P 的等待時間將被設定為以車輛 B 為基準的時間點（即 35 秒），而車輛 A 所設定的等待時間則被系統忽略。

如此一來，後續其他車輛在經過等待點 P 時，便會有一個固

定且穩定的等待時間（以最晚離開者 B 車為基準），避免路徑規劃時因等待時間頻繁變動而產生混亂或誤判。

透過上述提出的動態等待時間預測和管理機制，系統可以準確識別和計算每輛車行徑中的潛在等待點，並在車輛路徑規劃階段主動考慮潛在壅塞情況，幫助車輛自動選擇更好的路徑，從而有效緩解停車場整體的交通擁擠狀況。以下為車輛進入停車場後的流程（見圖 1）。

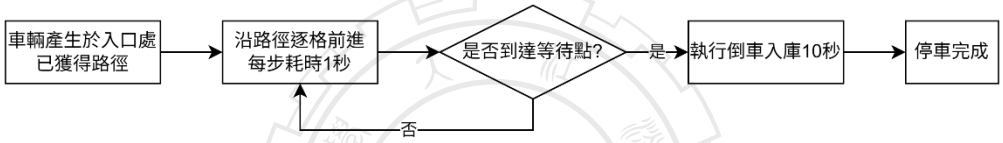


圖 1 車輛停車流程圖

3.3 壅塞預測和路徑選擇

為了進一步說明上述動態等待時間機制實際上是如何執行並避免壅塞的，使用一個具體的實際場景來解釋。當第二輛車進入停車場時，路徑規劃演算法會主動偵測並檢查沿途是否有其他車輛預先設定的等待點，並計算其可能受到的實際等待影響。

例如，當第二輛車規劃前往自身停車位的路徑時，若此路徑

上會經過第一輛車已設定的等待點，且此等待點距離第二輛車目前的位置為 15 格，而第一輛車在該等待點所設定的等待時間（waitTime）為 20 秒時，系統將計算第二輛車受到該等待點影響的潛在等待時間（remain 值）。此計算方式為等待點剩餘等待時間（20 秒）減去第二輛車抵達該等待點所需的行駛時間（15 秒），因此得出第二輛車屆時需要額外等待 5 秒才能通過此等待點。隨後，路徑規劃演算法將這額外的 5 秒等待時間納入第二輛車的路徑成本（cost）之中，並根據更新後的成本評估是否需選擇其他替代路徑，以主動避開可能造成壅塞的區域。

透過上述壅塞預測與動態等待成本調整，本研究所提出的改良式 A* 演算法之核心在於，透過動態調整傳統 A* 演算法中啟發式函數（heuristic function）$h(n)$ 的計算方式，將潛在壅塞的等待時間納入路徑成本考量，以達到預測並避開潛在交通壅塞點的目的。具體實作步驟詳述如下（見圖 2 壅塞預測和路徑選擇流程圖）。

1. 初始化步驟

系統首先將起點節點加入一個稱為「open list」的待探索節點集合中。此時設定起點節點的實際成本 g 值為

0，並計算綜合成本函數 f 值，其中 $f = g + h$，這裡的 h 是估算起點到目標點的成本預測值（以曼哈頓距離計算）。

2. open list 非空判斷

當 open list 中仍存在節點時，演算法將持續執行；若 open list 為空，則表示找不到路徑，演算法終止並回報失敗。

3. 節點選取（current 節點）

從 open list 中選取 f 值最小的節點作為目前要處理的節點，稱此節點為 current。

4. 判斷節點是否為終點

若 current 節點即為終點，則演算法成功，透過回溯節點資訊，重建並回傳完整的最佳路徑。若 current 節點非終點，則繼續下一步的擴展處理。

5. 擴展 current 節點

將 current 節點的鄰居節點逐一進行成本評估與計算（鄰居即 current 節點周圍可移動的一格）。

6. 鄰居節點成本計算

對每個鄰居節點 n，計算其「基礎成本（基礎

cost)」，其公式為：

$$基礎成本 = current.g + 移動代價$$

其中 current.g 是指從起點移動到目前節點 current 所累積的實際成本（移動步數）。「移動代價」是指從節點 current 移動至鄰居節點 n 所需的成本，本研究中每次移動代價統一設定為 1（代表車輛每次移動一格所需時間為 1 秒）。

7. 動態等待時間（waitTime）判斷

檢查鄰居節點 n 是否存在「等待時間（waitTime）」，即檢查該節點是否已被其他車輛設定為「等待點」。若該節點確實存在等待時間，代表前方有車輛預計會在此節點停留並進行倒車或其他停頓行為，可能導致壅塞。若該節點不存在等待時間，則此節點的實際成本 newCost 即為基礎成本。若該節點存在等待時間，則需計算「剩餘阻塞時間」，並將其加入基礎成本之中，使公式為：

$$newCost = 基礎成本 + 剩餘阻塞時間$$

此步驟即為改良式 A* 演算法的關鍵改良點，因為傳

統 A* 演算法未考慮壅塞所造成的額外等待時間。

8. 節點成本比較與更新

若上述計算出的新成本值（newCost）比原先紀錄在該鄰居節點的 g 值更小，則更新該鄰居節點的 g 值為 newCost，並且重新計算該鄰居節點的 f 值（f = newCost + h），然後將該鄰居節點加入 open list 中進行後續探索。若新成本值並未比原有的 g 值更低，則表示已有其他更好的路徑抵達此鄰居節點，因此無須更新該節點。

9. 重複執行

演算法持續重複上述步驟（步驟 2～步驟 8），直到 open list 為空（無法找到路徑）或找到目標節點（成功）。

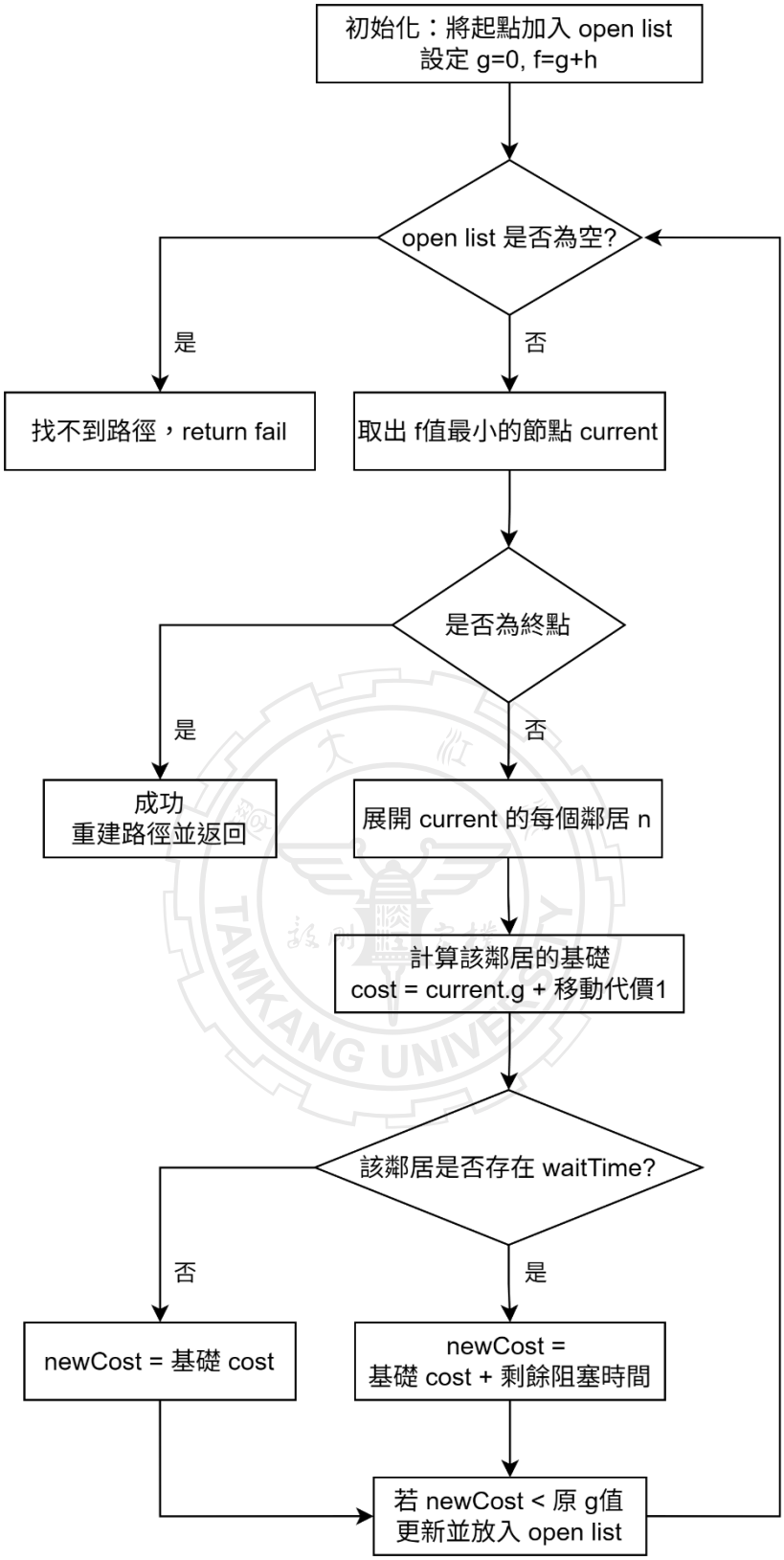


圖 2 壅塞預測和路徑選擇流程圖

3.4 動態路徑重規劃機制

在停車場的實際執行環境中，經常會發生突發事件，例如車道因故障或突發事件臨時封閉，導致原有的路徑規劃無法繼續有效執行。為了處理這種動態且不可預測的情況，本研究設計了一種動態路徑重規劃機制，以提高停車場車輛通行的效率和可靠性。該機制透過即時偵測場內的障礙或意外事件，主動重新規劃可能受影響的車輛路徑，避免壅塞情況加劇。

3.4.1 障礙物偵測和事件觸發機制

為了有效表示臨時封閉的車道，本研究設計了一種特殊的網格類型，即 CLOSED_AISLE，來標記目前無法通行的區域。一旦系統中某個網格的狀態改變為 CLOSED_AISLE，事件觸發機制就會立即啟動，將系統中的原子變數 eventTriggered 設定為 true，從而通知停車場中正在行駛的所有車輛，目前某個區域已經無法通行。另外，封閉的網格位置 (closedCellRow, closedCellCol) 也將同步儲存在全域變數中，以便所有正在行駛的車輛可以快速確定其目前路徑是否需要重新規劃。

3.4.2 判斷受影響的車輛並重新規劃路徑

當事件觸發時，每輛正在移動的車輛都會立即偵測 eventTriggered 是否已被設定為 true，如果是，車輛將進一步檢查其原先規劃路徑是否經過封閉區域。一旦發現自己的路徑受到影響，該車輛會立即中斷現有的移動過程，並將自身的當前位置、尚未行駛的剩餘路徑、以及原始的目的地座標等資訊，完整地記錄在特定的資料結構 affectedVehicles 容器中，以便系統稍後進行統一的動態路徑重規劃。建立此容器的目的是集中處理受影響的車輛，便於後續規劃流程的高效執行。

3.4.3 車輛重新規劃路徑的優先排序策略

為了提高整體重新規劃路徑的效率和系統的執行順暢性，本研究提出了一套基於「剩餘路徑長度」的優先排序策略，即首先處理距離目的地較近的車輛。這種作法的邏輯是，將距離目的地最近的車輛優先重新規劃，快速到達目的地，有助於盡快釋放車道空間，進一步減少其他車輛的壅塞等待時間。

此外，為了貼近真實的駕駛情境，避免不合理的車輛駕駛行為，系統在第一次重新規劃路徑時，會禁止車輛立即迴轉回到之前的位置 (noGoCell)，以避免車輛出現不合理的迴轉行為。若第一次重新規劃失敗，系統才會允許車輛進行迴轉，以提供更大的路徑規劃自由度，確保車輛能夠盡可能順利地到達目標位置。這項設計能夠有效限制車輛的不合理行為，又能在必要時提供靈活性，反映了實際停車場管理的現實需求。本研究提出的動態路徑重規劃機制流程如圖 3 所示。

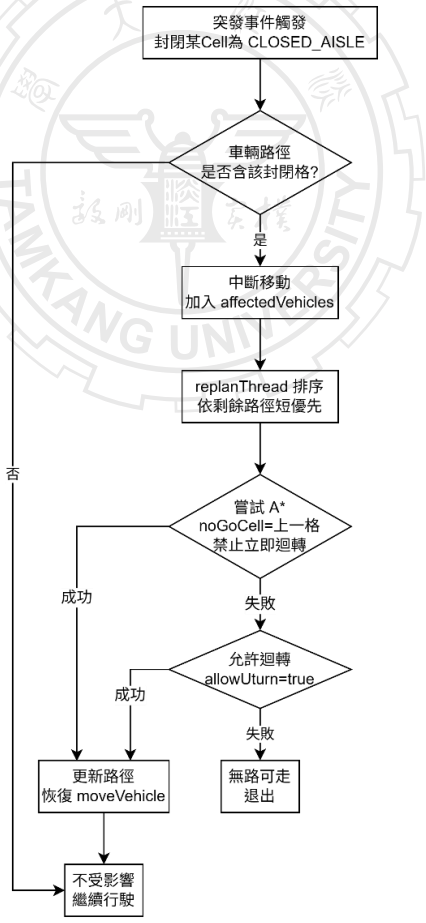


圖 3 動態路徑重規劃機制流程圖

3.5 系統實現和執行機制

為了完整呈現本研究提出的停車場動態管理系統的執行機制和內部設計，本節將簡要說明模擬車輛生成方法、多執行緒架構設計、系統中使用的重要資料結構，並提供路徑規劃時的特殊策略等，幫助讀者充分理解系統的設計邏輯和實作細節。

3.5.1 車輛生成和進場模式

本系統採用隨機產生的方式來模擬車輛進入停車場的過程。每輛車輛生成時，從停車位現有的空車位中隨機選取目標停車位，將車輛的起始位置固定在停車場入口（ENTRANCE），並透過隨機數字決定車輛生成的時間間隔，以模擬真實情況下車輛進入停車場的不規則性，從而證明了本研究所提出的動態路徑規劃演算法的適應性和可靠性。

3.5.2 多執行緒和同步機制

為了提高模擬系統的執行效率和真實感，本研究採用多執行緒的方式來同時處理多輛車的路徑規劃和移動控制。在多執行緒架構下，每輛車都擁有獨立的執行緒，因此其路徑規劃和行駛行為可以同步進行，不會相互阻塞或等待。這種設計有助於避免當

任何一輛車的邏輯運算時間較長時，其他車輛必須等待相同的執行序列而無法同步前進的情況，從而提高整體模擬效能和真實感。

然而，多個執行緒同時存取共享資源（如停車場狀態、等待時間記錄）可能會導致資源競爭和資料一致性問題。因此，本研究使用標準的同步機制，例如 mutex 和 std::atomic 變數來嚴格控制對共享資源的存取，以確保系統的穩定性和資訊的正確性。例如，原子變數 eventTriggered 用於通知突發事件的發生，mutex 用於鎖定共享結構（如停車格狀態和等待時間設定），確保多個執行緒並行執行時資料的一致性。

3.5.3 實驗用地圖的選擇理由

本系統所提出的改良式 A* 方法理論上可以應用於任何大小的網格狀地圖。但為了凸顯演算法在壅塞情況下的優勢，後續的實驗測試將使用較小尺寸的地圖。原因在於，在較小型停車場環境當中，車輛集中特性較為明顯，容易造成壅塞，對本研究而言更容易觀察到傳統 A* 和我們所提出改良式 A* 之間的效能差異。更詳細的實驗設定和分析結果，將於第四章進一步解釋。

第四章　實驗結果與分析

4.1 實驗設計

為了驗證本研究提出的改良式 A* 和動態路徑重規劃機制的有效性，本研究建立了兩個獨立的停車場模擬系統，分別測試和分析不同的研究目標和情境。

4.1.1 實驗目標

本研究透過兩個獨立模擬程式的多層次測試，進行完整驗證：

（1）改良式 A* 與動態等待時間機制的效能比較

在無突發事件的標準模擬場景下，比較傳統 A* 與本研究提出的改良式 A* 在壅塞情況下的平均行駛時間和等待時間。透過特別關注高負載狀態（例如最後 10 輛入場車輛）進行比較，可以分析改良式 A* 是否能夠有效預測和避開潛在的壅塞，從而縮短車輛整體的行駛時間和等待時間。

（2）動態路徑重規劃機制的可行性和即時性驗證

透過另一個獨立的模擬程式，在模擬過程中刻意臨時封閉車

道，檢測車輛是否能夠即時偵測到這個突發事件並有效重新規劃路徑。本實驗並不著重詳細的數據統計（例如平均行駛時間），而是確認車輛是否能在極短的時間內完成重新規劃並避開封閉區域，證明系統面對突發事件時的反應速度和可靠性。

（3）整個系統對停車場壅塞問題的貢獻評估

結合以上兩套模擬實驗的結果，評估本研究所提出的動態路徑重規劃機制是否能有效解決停車場常見的壅塞問題。如果實驗結果顯示改良式 A* 能夠有效減少車輛在高壅塞狀況下的平均耗時，並且動態路徑重規劃機制能夠即時處理突發事件，那麼就證明了本研究在更廣泛交通場景中應用的潛力。

透過這些多方面的驗證，本研究可以深入探討提出方法的運作效能和實際應用可行性，並評估在停車場實際管理中的潛在效益。接下來將進一步介紹實驗環境設定和具體實驗步驟，並於後續結果分析章節呈現具體結果。

4.1.2 實驗環境設定

為了清楚呈現本研究所使用之模擬環境，本研究針對兩個模擬系統訂定以下共同設定：車道採用雙向各一車道配置，僅能雙

向行駛而無超車空間；停車格分布於場地四邊及中央，以模擬真實多樣化佈局；所有車輛均自固定入口（ENTRANCE）隨機生成並分配目標車位，以確保模擬變化性；系統執行於 Intel® Core™ i7-12700K（12 核心、20 執行緒）、32 GB DDR5-5200 RAM、C++17/GCC 13.2.0、Windows 10 平台上。其後，兩套模擬在環境細節上有所差異：效能統計模擬採用 13×12 格（見圖 4 停車場 13x12 地圖示意圖）小尺幅地圖、固定產生 20 輛車輛（前 10 輛為低壅塞、後 10 輛為高壅塞），並於完全相同之車輛進場順序、入場時間及目標車位設定下，分別運行傳統 A* 與改良式 A*，以確保比較公平；動態路徑重規劃模擬則在 17×24 格接近真實規模場地下，於整個模擬期間持續隨機產生車輛並分配停車位，並隨機封閉部分車道（AISLE → CLOSED_AISLE），重點觀察系統是否能及時偵測突發事件並成功為受影響車輛執行動態路徑重規劃，使其順利完成停車。

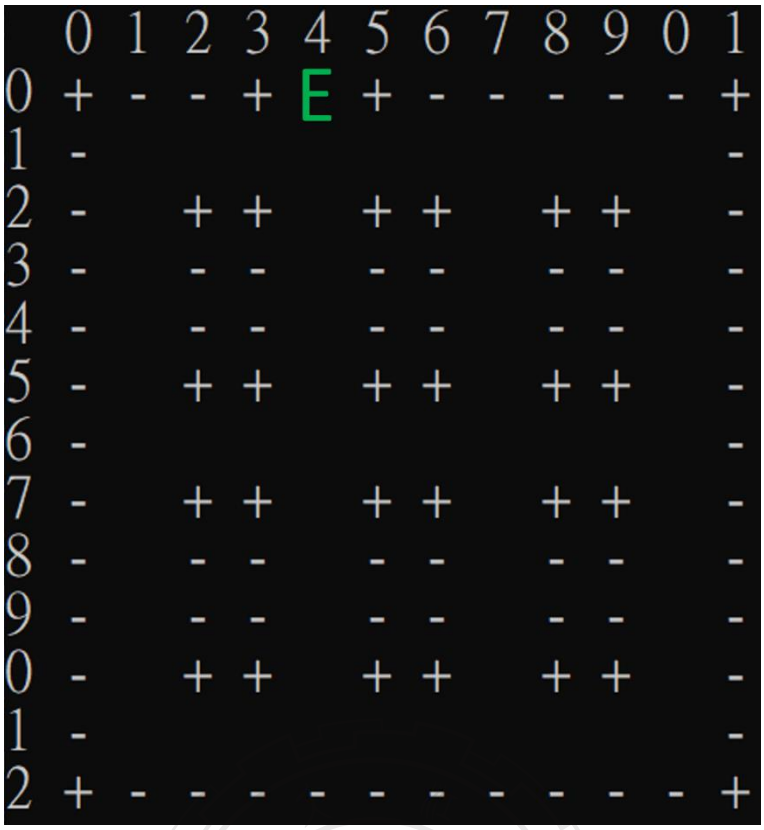


圖 4 停車場 13x12 地圖示意圖

4.2 實驗方法及步驟

為了實現上述實驗目標，本研究分別針對「效能統計模擬程式」和「動態路徑重規劃模擬程式」進行了獨立的實驗測試。兩個實驗的具體執行步驟描述如下：

4.2.1 效能統計模擬程式實驗步驟

本模擬主要評估傳統 A* 和改良式 A* 在停車場內壅塞情境下的效能差異，步驟包括：

（1）初始設定

啟動模擬環境，載入停車場地圖（尺寸：17×24）。系統隨機產生 20 輛車，並隨機分配每車輛的目標停車位，每個停車位在單次模擬中只能被選擇一次，以確保不會重複佔用。為每輛車隨機分配進入停車場的時間順序，使其模擬真實車輛分別進入停車場的情況，確保每次模擬都是完全隨機的。

（2）進行第一次模擬（傳統 A*）

根據步驟 1 中已固定的車輛設定，使用傳統 A* 逐一規劃每輛車的路徑。系統模擬所有車輛都在自己的執行緒中，每秒移動一步（格子），並持續計算自己的行駛時間和等待時間。模擬持續進行，直到所有車輛都順利停放完畢，紀錄所有車輛各別從進入停車場到停入目標停車位的行駛時間和等待時間，即結束該次模擬。

（3）進行第二次模擬（改良式 A*）

使用完全相同的車輛設定（包含進場順序和目標停車位），再使用改良式 A* 逐一進行路徑規劃。以同樣的方式模擬車輛逐秒移動，直到所有車輛完成停放。記錄每輛車的行駛時間和等待時間。

（4）重複執行並資料收集

整個實驗流程（步驟 1～步驟 3）透過外部的 shell 腳本持續重複執行，直到手動中斷為止。記錄每次模擬的所有車輛平均行駛時間和等待時間。每次都紀錄「前 10 輛」和「後 10 輛」車輛的平均行駛時間和等待時間，觀察不同負載階段下的效能差異。

每一組完整實驗（20 輛車輛都完成模擬傳統 A* 和改良式 A*）的行駛和等待時間資料都會完整紀錄在外部檔案中，後續再進行平均數和極端值的統計分析。整個實驗流程如圖 5 所示。

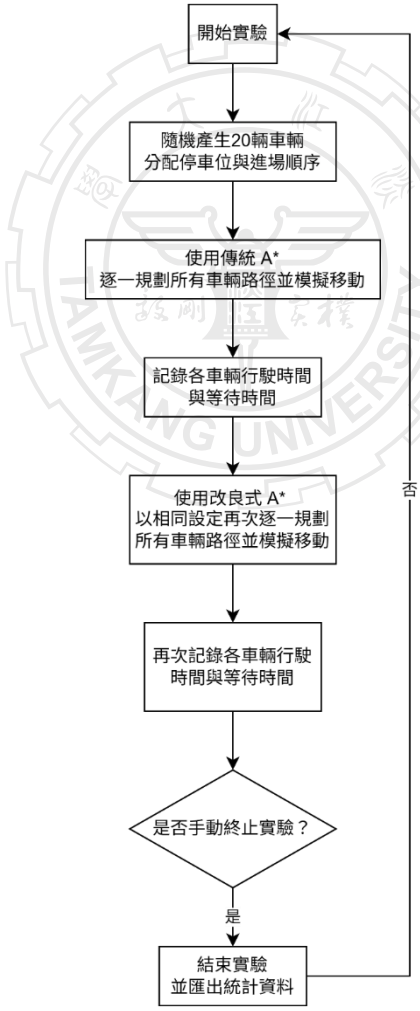
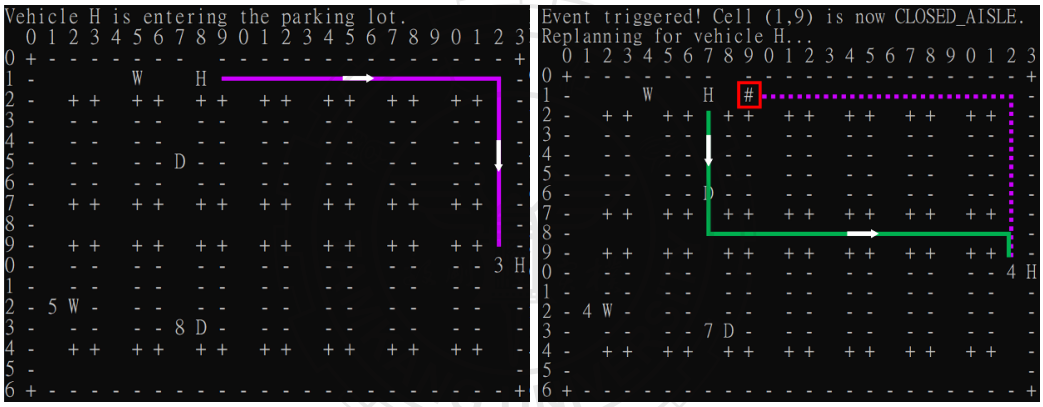


圖 5 效能統計模擬程式流程

4.2.2 動態路徑重規劃模擬程式實驗步驟

為驗證動態路徑重規劃機制之實務效能，本研究進行了模擬測試，展示了動態調整後之車輛移動路徑（如圖 6 所示），圖 6(a)顯示突發事件發生前之初始規劃路徑情況，此時車輛皆按原訂路徑正常行駛；而圖 6(b)則呈現突發事件發生後，動態路徑重規劃機制啟動後的結果，顯示各車輛依照新路徑行進，有效地避開臨時封閉區域，降低了停車場內的交通壅塞。



(a) (b)

圖 6 動態路徑重規劃前後之模擬示意圖（a）重規劃前；(b）重規劃後

實驗步驟包括：

（1）初始設定

載入尺寸為 17x24 之真實規模停車場地圖。停車場入口位置是固定的，每輛車的目標停車位是隨機決定的，每格停車位在單

次模擬中只能被選擇一次，以確保不會重複佔用。

（2）車輛模擬與路徑執行

車輛在停車場入口持續生成，並立即透過改良式 A* 規劃初始路徑。在模擬過程中，所有車輛繼續在自己的執行緒中，每秒移動一格，直到順利停在目標位置。

（3）突發事件觸發

在模擬過程中，特定車道在隨機時刻封閉，將該格子型態由 AISLE 動態轉換為 CLOSED_AISLE，並啟動事件觸發機制（eventTriggered），以模擬停車場真實的突發事件情況。

（4）動態路徑重規劃啟動

當受事件影響的車輛偵測到 eventTriggered 時，立即中斷原來的路徑，加入 affectedVehicles 容器中，等待動態路徑重規劃執行緒（replanThread）處理。

（5）重新規劃與重新移動

系統依照「最短剩餘路徑者優先」的原則，逐一為受影響的車輛執行動態路徑重規劃。在首次進行重新規劃時，系統會讓車輛禁止迴轉，若初次重新規劃失敗，則允許車輛迴轉，透過返回

原路來有效取得新的替代路徑。成功規劃路徑後，車輛繼續向目標停車位移動。

（6）終止條件

實驗是由手動控制的，直接觀察多次的動態路徑重規劃情境且確認該機制有效。本模擬程式的完整流程見圖 7。

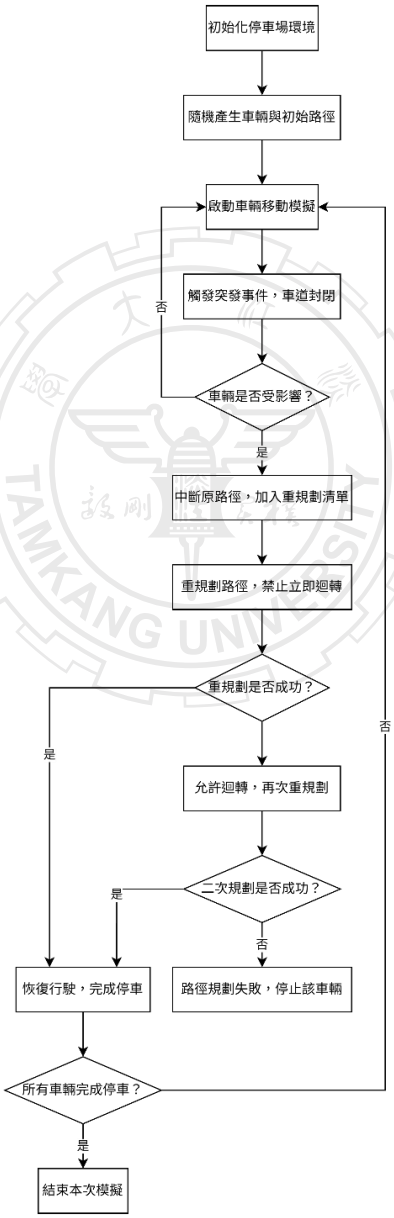


圖 7 動態路徑重規劃模擬程式流程圖

4.3 實驗結果與分析

4.3.1 統計數據結果與分析

本節透過「效能統計模擬程式」所蒐集的實際模擬數據，對傳統 A* 和本研究提出的改良式 A* 的平均行駛時間和等待時間進行評估並比較，以驗證改良式 A* 在壅塞情況下對停車場整體運行效能的改善效果。

（1）數據來源與極端值處理方式

本研究透過外部 shell 腳本重複執行效能統計模擬，初始資料共計產生 54,573 筆車輛行駛和等待時間數據。

（2）由於該模擬隨機分配停車場給車輛，因此在某些極端情況下，數據可能會出現「極大值」或「極小值」的行駛或等待時間，這些極端值可能會扭曲整體的平均值和統計結果，無法真實反映正常情況下演算法的優勢。以下針對極大值與極小值這兩種極端情形進行說明：

極大值（行駛或等待時間異常高）：行駛時間（秒）39.5、50.1 等，等待時間甚至高達 27.4 秒，這些數據的等待或行駛時間通常顯著超出一般壅塞情形的合理範圍（例如，等待超過十秒甚

至二十秒），多半發生於大量車輛集中分配至特定入口附近車位或單一路徑時，導致局部區域完全停滯，無法真實代表演算法在正常情況下的效能。圖 8 即為本研究某次模擬實驗中的極端案例分布示意圖，標記圓點中的數字為車輛進場順序，可見此回合中車輛的停車格明顯集中於地圖左中區域與單一車道上，造成極為嚴重的交通壅塞情況。這種極端情境在真實停車場內雖罕見，但於隨機模擬下偶爾仍可能發生。

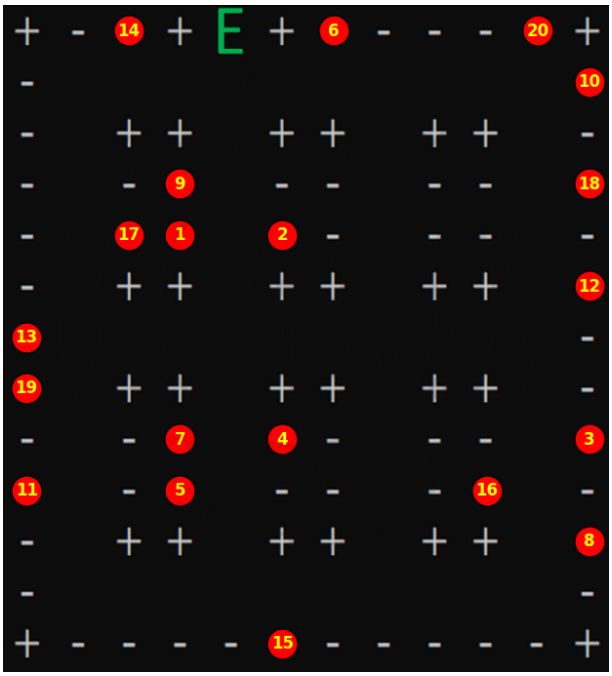


圖 8 某次模擬中之極端壅塞案例車輛停車位置與進場順序示意圖

極小值（行駛或等待時間異常低）：行駛時間（秒）：19.2、19.6、19.7 等，等待時間皆為 0 秒。這些數據完全沒有發生壅塞或等待，通常是車輛高度均勻分散，恰巧完全沒有壅塞情境，導致數據無法真實地反映停車場內壅塞環境下的演算法表現。

上述數據說明去除這些極端值是為了確保數據能更真實地反映一般情況下改良式 A* 演算法的效能。這些極端情況在現實中罕見且對本研究探討壅塞改善的目的無直接價值，因此排除後更能準確地評估系統的一般效能。

基於上述原因，本研究使用四分位距（Interquartile Range, IQR）法移除上述過大或過小數值[37] 。IQR 定義為資料的第三四分位數（Q3）與第一四分位數（Q1）之差（即 IQR = Q3 − Q1），並據此設定界限為：

$$\text{下界(Lower Bound)} = Q1 - 1.5 \times IQR,$$

$$\text{上界(Upper Bound)} = Q3 + 1.5 \times IQR$$

透過此方法進行處理後，原始 54,573 筆資料去除了 10,849 筆（約 19.88%）極端值，剩餘 43,724 筆資料，顯示極端值並非少數，也再次驗證去除極端值的必要性。為了提升後續資料處理和效率分析，再從該資料中隨機抽取 5,000 筆作為本節分析的基礎數據。

表 1 分別呈現使用傳統 A* 與改良式 A* 演算法的車輛平均行駛和等待時間的比較結果。

表 1 車輛平均行駛和等待時間比較

車輛平均行駛時間比較（秒）			
車輛群組	傳統 A* 平均行駛時間	改良式 A* 平均行駛時間	行駛時間改善率
前 10 輛	21.989	20.79	5.10%
後 10 輛	24.491	21.832	10.02%
整體平均	23.24	21.311	8.30%
車輛平均等待時間比較（秒）			
車輛群組	傳統 A* 平均等待時間	改良式 A* 平均等待時間	等待時間改善率
前 10 輛	2.293	0.836	63.53%
後 10 輛	4.908	1.899	61.31%
整體平均	3.6	1.368	62.02%

表 2 為 IQR 處理前的原始數據，其中包含極端值，例如 RunID 13895 和 24096 在『傳統 A*後 10 輛車的行駛時間』欄位數值明顯高於其他數據，因此視為極端值。經過 IQR 處理後的數據如表 3 所示，該 RunID 已被移除，以確保分析結果的可靠性。

為避免表格過寬，以下各欄位名稱皆以簡寫呈現，說明如下：「傳 A*」指傳統 A* 演算法、「改 A*」指改良式 A*；「前 10」與「後 10」分別代表入場順序前 10 輛與後 10 輛車輛；「行」為平均行駛時間（秒）、「待」為平均等待時間（秒）。

表 2 未做 IQR 過濾的樣本數據

RunID	傳A* 前10行	傳A* 後10行	傳A* 前10待	傳A* 後10待	改A* 前10行	改A* 後10行	改A* 前10待	改A* 後10待
3013	21.9	24	2.6	4.6	20	20.3	0.1	0.9
4439	20.6	22.6	0.3	4.2	20.6	22.8	0.3	4
7561	20.3	23.4	0.1	3.4	20.3	21.2	0.1	1.2
13895	24	31.7	6	11.9	19.1	31.5	1.1	11.7
14018	18.3	23.8	0	5.2	18.3	23.2	0	4.2
19635	24.1	21.8	3.7	3.3	23	20.6	2.4	2.1
24096	21.6	33.7	2.7	14.7	19.8	29.9	0.9	10.9
24813	20	25.3	1	5.3	19.8	22.3	0.8	2.3
32098	24.8	23.4	3.5	4.2	22.2	21.6	0.1	1.8
34273	21.4	23.3	0.6	4.2	21.4	24.1	0.6	5
38902	19.4	26.6	0.6	5.3	19.8	23.2	1	1.5
40288	22	25.3	4	5.1	22.3	22.1	4.3	1.9
42440	21.3	27.1	1.8	7.7	20.8	23.4	0.7	3.8
44770	22.5	23.2	2.1	3.3	22.2	22.4	1.4	1.7
47919	21.9	21.3	2.3	3.5	21.3	18.5	1.5	0.7

表 3 做完 IQR 過濾的樣本數據

RunID	傳A* 前10行	傳A* 後10行	傳A* 前10待	傳A* 後10待	改A* 前10行	改A* 後10行	改A* 前10待	改A* 後10待
3013	21.9	24	2.6	4.6	20	20.3	0.1	0.9
4439	20.6	22.6	0.3	4.2	20.6	22.8	0.3	4
7561	20.3	23.4	0.1	3.4	20.3	21.2	0.1	1.2
14018	18.3	23.8	0	5.2	18.3	23.2	0	4.2
19635	24.1	21.8	3.7	3.3	23	20.6	2.4	2.1
24813	20	25.3	1	5.3	19.8	22.3	0.8	2.3
32098	24.8	23.4	3.5	4.2	22.2	21.6	0.1	1.8
34273	21.4	23.3	0.6	4.2	21.4	24.1	0.6	5
38902	19.4	26.6	0.6	5.3	19.8	23.2	1	1.5
42440	21.3	27.1	1.8	7.7	20.8	23.4	0.7	3.8
44770	22.5	23.2	2.1	3.3	22.2	22.4	1.4	1.7
47919	21.9	21.3	2.3	3.5	21.3	18.5	1.5	0.7

4.3.2 分析與討論

根據上述數據可得以下幾點結論：

整體平均行駛時間從傳統 A* 的 23.240 秒降至改良式 A* 的

21.311 秒，整體改善幅度為 8.30%。特別是在「後 10 輛車」較嚴重的壅塞情況下，改善幅度達到 10.02%，這清楚地展現改良式 A* 可以在壅塞發生前預測並避開潛在壅塞點，顯著降低行駛時間。

整體平均等待時間從傳統 A* 的 3.600 秒降至改良式 A* 的 1.368 秒，改善幅度達 62.02%，展現出非常顯著的減少幅度。

由於等待時間更直接反映了車輛在壅塞時的阻塞狀況，因此其改善率也更明顯，凸顯了本研究所提出的動態等待時間機制的顯著效益。

等待時間的改善遠高於行駛時間的改善，主要原因為：

等待時間本身反映出車輛因壅塞而完全停滯的時間，本研究提出的改良式 A* 可以提前避開這些造成停滯的高風險阻塞點，因此對於完全停滯等待的情況改善特別顯著。

相較之下，行駛時間除了受等待時間影響外，還包含車輛自由行駛的時間，因此整體的改善幅度比起等待時間略低。

綜合上述結果，本研究透過效能統計模擬實驗，明顯地驗證了改良式 A* 演算法在減緩停車場內壅塞情況的顯著優勢，尤其

在等待時間方面的改善極為突出，驗證了本研究的動態等待時間機制的有效性和實際應用價值。

4.3.3 動態路徑重規劃機制驗證

（1） 實驗設計之驗證重點

本節著重動態路徑重規劃機制在停車場內突發事件時的實際表現，驗證過程的主要觀察點包括：

- 系統是否能即時偵測突發事件並立即啟動路徑重規劃。
- 車輛是否能成功找到替代路徑並順利到達原始目標停車位。
- 路徑重新規劃過程中是否正確顯示動態等待時間（waitTime），並正確應用本研究提出的動態等待時間機制。

由於此驗證主要強調突發事件下重新規劃的成功率和機制即時性，因此沒有特別蒐集行駛時間的詳細統計數據，而是專注於觀察動態路徑重規劃功能的可靠性和可行性。

（2）首次規劃禁止迴轉的原因

在本研究第一次重新規劃時，禁止車輛直接迴轉返回原路，

目的是為了更真實地模擬實際停車場情況。在現實環境中，停車場內車道狹窄且不易迴轉，因此首次規劃希望車輛盡可能選擇其他替代路徑。

但如果突發事件恰好封閉了該車輛前方唯一的通道（例如狹長車道的唯一出口），則首次規劃可能無法找到合適的替代路徑，必須透過第二次規劃允許迴轉時，才可以成功取得替代路線。

（3）實驗驗證結果

實驗中觀察到的動態路徑重規劃具有極高的穩定性和即時性：

- 所有受影響的車輛都可以即時偵測到突發事件並啟動路徑重規劃。
- 車輛都能夠成功取得替代路徑，正確地重新計算並顯示等待時間 (waitTime)。
- 在實驗中，第二次允許迴轉後的路徑重規劃成功率達到 100%，沒有出現二次重新規劃仍失敗的情況。

為補充實驗結果的具體情形，本研究擷取了模擬過程中兩輛車輛在遭遇突發事件後進行路徑重規劃的實際畫面。

圖 9 示範了一輛車在路徑前方遭遇突發封閉區域（紅框處）後，立即啟動路徑重規劃，成功繞行封鎖區域，並順利抵達目標停車格的情形（綠線為新路徑）。

圖 10 則展示另一輛車在同樣遭遇突發事件時，首次重規劃失敗，系統隨即允許該車輛進行方向迴轉，最終在第二次路徑重規劃後成功找到可行替代路徑，展現本研究設計的重規劃演算法在即時反應與穩定性上的優越表現。

在圖 9 與圖 10 中，畫面中央的 A 表示目前正在進行動態路徑重規劃的車輛，而位於停車格內的 A 則表示該車輛所被分配的目標停車位。每個停車格前方的車道上會顯示一個數字，該數字為等待點上的 waitTime 值，代表該位置當前尚需等待的秒數（車輛倒車入庫時的剩餘時間）。紅色方框 # 表示因突發事件（如臨時封閉）而無法通行的區塊；紫色虛線為原本規劃的行駛路徑，而綠色實線為車輛啟動重規劃後成功找到的替代新路徑。

（4）動態路徑重規劃機制的討論與意義

透過上述驗證結果，本研究所提出的動態路徑重規劃機制證明了在突發事件下具有高度即時性和可靠性，能夠有效避免因車道臨時封閉所造成的停車場內嚴重壅塞或停滯問題。

與效能統計模擬程式強調壅塞情境下行駛時間和等待時間的統計分析不同，本節的驗證注重在實際情況中更具不確定性的突發事件，強調的是機制的即時反應能力和重新規劃的成功率。這兩項實驗各自專注在不同面向，但整體互補性高，都共同支持本研究的總體目標：提升停車場環境內車輛的通行效率和可靠性，提供完整且具有實際意義的體系。

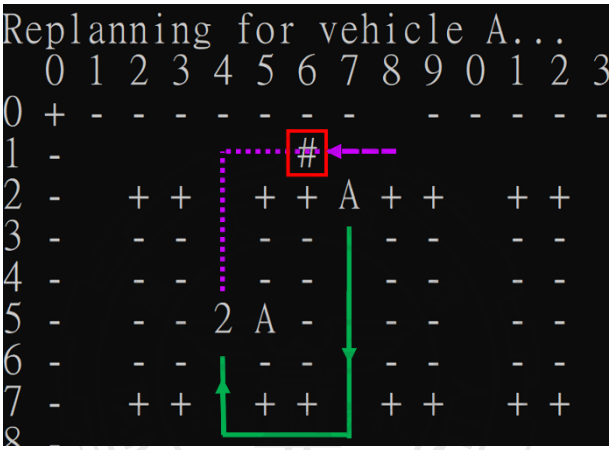


圖 9 遭遇突發事件後之首次路徑重規劃成功案例

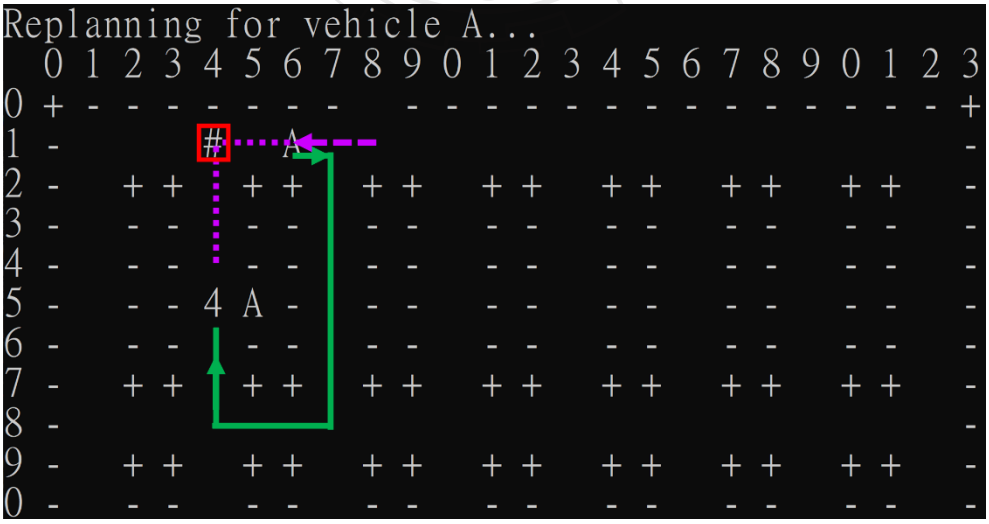


圖 10 首次重規劃失敗後允許迴轉之成功重規劃案例

4.4 本章小結

本章透過一系列模擬實驗和數據分析，完整地驗證了本研究所提出的改良式 A* 演算法以及動態路徑重規劃機制的實際效能和應用可行性。透過不同實驗設計和驗證方式，針對實際停車場內常見的壅塞情況和突發事件，本研究提供了具體且具有實際意義的結果和討論。

首先，在「效能統計模擬」實驗中，經過嚴謹的極端值處理和統計分析後，證明改良式 A* 演算法相較於傳統 A* 在平均行駛時間和等待時間都有明顯地改善，尤其等待時間的改善幅度高達 62.02%，凸顯本研究提出的動態等待時間機制在緩解壅塞情況的成效。此外，透過分組比較分析（前 10 輛、後 10 輛），亦證實了改良式 A* 演算法於高負載狀態下的效益更為明顯，實際應用潛力更高。

其次，「動態路徑重規劃模擬」實驗進一步驗證本研究在突發事件時的系統反應能力和機制可靠性。實驗證明本系統能夠迅速偵測停車場內突發事件，並啟動動態路徑重規劃，所有受影響的車輛都能夠順利找到替代路徑並正確抵達原目標停車位，顯示動態重規劃機制具有高度的穩定性和即時性，且重規劃成功率為

100%。此外，本研究針對首次路徑規劃禁止迴轉的策略進行討論，證明該設計符合真實停車場環境需求，並有效提高規劃的實際性和適用性。

綜合上述，本章透過統計效能分析與動態路徑重規劃驗證，成功證明了本研究所提方法在停車場環境中顯著改善效果和實際價值。下一章將針對整體研究成果進行總結，並提出未來可能的研究方向和建議。

第五章　結論與未來研究

5.1 研究成果總結

本研究提出了一套適用於停車場環境的動態路徑規劃系統，透過改良式 A* 演算法和動態等待時間機制，有效地緩解了停車場內因壅塞所產生的額外等待時間和行駛效率問題。此外，本研究也針對停車場內突發事件設計了動態路徑重規劃機制，進一步提升系統在各種情境下的即時性和可靠性。本研究成果可分為以下幾類：

5.1.1 改良式 A* 演算法的效能驗證成果

本研究透過「效能統計模擬程式」，透過大量的隨機化測試來比較傳統 A* 與改良式 A* 演算法的效能，得出了以下明顯的成效：

在平均行駛時間方面：

改良式 A* 將車輛的整體平均行駛時間從 23.240 秒 縮短至 21.311 秒，改善幅度為 8.30%。

特別是在停車場進入高度壅塞階段的「後 10 輛車」情境

下，改善幅度更高達 10.02%，證明了本研究提出的動態等待時間機制能有效識別並提前避開潛在壅塞點。

在平均等待時間方面：

傳統 A* 的整體平均等待時間為 3.600 秒，改良式 A* 則僅有 1.368 秒，等待時間的改善幅度高達 62.02%。

由於等待時間更直接反映出車輛因壅塞而完全停滯的情況，本研究的改良式 A* 透過動態等待時間的預測和壅塞點迴避機制，有效地大幅減少了此類停滯情況。

整體而言，改良式 A* 在降低等待時間上的成效顯著，凸顯了本研究提出的動態等待時間機制在實際上的價值與效益。

5.1.2 動態路徑重規劃機制驗證成果

在另一套「動態路徑重規劃模擬程式」中，本研究針對突發事件（如臨時封閉車道）可能造成的影響，進行了機制的驗證，具體成果如下：

系統能夠在車道突然封閉時即時偵測到該事件，並立即啟動動態路徑重新規劃，展現了高度的即時性。

所有受影響的車輛都可以成功取得替代路徑，並順利抵達原

始設定的目標停車位，並在重規劃路徑過程中正確地計算並顯示等待時間 (waitTime)，驗證了動態等待時間機制的穩定性。

為符合實際環境，首次路徑重新規劃時禁止車輛立即迴轉；但在首次規劃無法取得替代路徑的情況下，放寬限制允許迴轉，實驗中並未觀察到二次規劃仍失敗的情況，重規劃的成功率達到 100%。

動態重規劃機制的成功驗證，進一步證實了本研究系統在實際停車場環境中面臨突發事件時的實用性和可靠性，具備高度應用價值。

5.1.3 整體系統對壅塞問題的貢獻

綜合上述兩個實驗，本研究的動態路徑規劃系統透過兩個互補的驗證方法，成功證明：

改良式 A* 與動態等待時間機制能顯著降低日常停車場中車輛的行駛和等待時間，有效緩解停車場常見的壅塞問題。

動態路徑重規劃機制則能可靠地處理突發事件，確保車輛在發生意外狀況時仍可順利通行。

5.2 研究貢獻

本研究針對停車場常見的壅塞和突發事件問題，提出了一種結合改良式 A*、動態等待時間機制和動態路徑重規劃機制的系統。透過詳細的實驗和驗證，本研究的貢獻具體如下：

5.2.1 改良式 A* 與動態等待時間機制的整合

本研究成功將傳統的 A* 進行改良，透過動態等待時間機制，使車輛能夠提前預測並有效避開停車場內潛在的壅塞點。此整合的貢獻包括：

即時壅塞預測：本系統在路徑規劃階段就可以預測壅塞並動態調整路徑成本，讓車輛提前避免潛在的交通堵塞點，大幅提升了車輛通行效率。

等待時間顯著改善：透過實驗驗證，平均等待時間降低幅度高達 62.02%，顯示本研究提出的方法在緩解壅塞問題上具極大的實際效益。

具備實際應用潛力：改良式 A* 不僅提高了模擬環境的運行效率，更具備實際停車場管和與智慧交通領域的廣泛應用潛力。

5.2.2 突發事件下的動態路徑重規劃機制

針對停車場內的突發事件（例如車道封閉），本研究提出了高效的動態路徑重規劃機制，驗證成果顯示該機制具備高度的即時性和可靠性。主要貢獻為：

- 即時事件處理能力：系統能在突發事件發生後即時啟動路徑重新規劃功能，確保停車場內的交通秩序迅速回復正常，防止壅塞情況進一步擴大。
- 高度的成功率和可靠性：實驗驗證中重規劃成功率達到100%，展現出系統對於突發事件的卓越反應能力和強大的實用性。
- 適合現實情況的設計策略：首次規劃禁止迴轉，第二次則允許迴轉的策略，更貼合現實停車場的限制，能更有效地處理狹窄空間內的交通問題。

5.2.3 完整且互補的實驗驗證方法

本研究設計兩種模擬程式，分別針對日常壅塞情境和突發事件情況進行驗證。這種完整且互補的實驗設計提供了對系統全方位的驗證成果，其貢獻包括：

- 雙重驗證機制：透過效能統計和動態事件處理的驗證，充分展示本系統在不同狀況下的綜合能力，使研究成果更具說服力和完整性。
- 明確的驗證範例與成果：提供了具體且量化的成果和數據，能清楚展示出所提出的系統在停車場環境中的實際效益。

5.2.4 實際價值和未來推廣性

整體而言，本研究所提出的系統不僅有顯著的學術價值，亦具備高度的實際價值和應用潛力：

- 本系統能有效降低停車場內車輛的等待和行駛時間，直接提升用戶體驗並緩解交通壅塞。
- 可進一步延伸至一般交通路網或其他場域（如物流中心或工業園區），具有高度的推廣性和適用性，能有效改善各種環境下的交通管理問題。

綜合以上所述，本研究的動態路徑規劃系統結合了演算法創新和系統設計之優勢，成功證實其對停車場管理的實際價值，並提供了完整且可供未來研究延伸的理論和實際基礎。

參考文獻

[1] D. C. Shoup, "The High Cost of Free Parking." *Chicago, IL, USA: Planners Press*, 2005.

[2] Y. Sun, "Research on strategies to address urban parking issues in China," *Frontiers in Humanities and Social Sciences*, vol. 4, no. 1, pp. 128–133, 2024.

[3] M. R. Islam *et al.*, "Smart parking management system to reduce congestion in urban area," in *Proc. 2nd Int. Conf. Electr., Control Instrum. Eng. (ICECIE)*, Kuala Lumpur, Malaysia, Aug. 2020, pp. 1–6.

[4] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 354–363, Jun. 2005.

[5] J.-L. Lupien *et al.*, "Entropy-based dynamic programming for efficient vehicle parking," *arXiv*, arXiv:2411.17014, Nov. 2024. [Online].

[6] H. Taghipour, A. B. Parsa, and A. K. Mohammadian, "A dynamic approach to predict travel time in real time using data-driven techniques and comprehensive data sources," *Transportation Engineering*, vol. 2, p. 100025, 2020.

[7] B. Adabala and Z. Ajanović, "A multi-heuristic search-based motion planning for automated parking," *arXiv*, arXiv:2307.07857, Jul. 2023.

[8] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[9] G. F. Newell, "A simplified theory of kinematic waves in highway traffic—Part I: General theory," *Transportation Research B*, vol. 27, no. 4, pp. 281–287, 1993.

[10] D. C. Gazis and R. Herman, "The moving and 'phantom' bottlenecks," *Transportation Science*, vol. 26, no. 3, pp. 223–229, 1992.

[11] N. Fulman and I. Benenson, "Approximation of search times for on-street parking based on supply and demand," Geosimulation Lab, Tel Aviv Univ., 2024.

[12] T. Fabusuyi and R. C. Hampshire, "Rethinking performance-based parking pricing: A case study of SFpark," Transp. Res. Inst., Univ. Michigan, 2024.

[13] C. Nugent, "Americans' addiction to parking lots is bad for the climate. California wants to end it," *TIME*, Sep. 28 2022. [Online]. Available: https://time.com/6217873/parking-lots-climate-change-california/. [Accessed: May 2 2025].

[14] A. Stentz, "The Focussed D* algorithm for real-time replanning," in *Proc. 14th Int. Joint Conf. Artif. Intell. (IJCAI)*, Montr éal, QC, Canada, Aug. 1995, pp. 1652 – 1659.

[15] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, San Diego, CA, USA, 1994, pp. 3310 – 3317.

[16] S. Koenig and M. Likhachev, "D* Lite," in *Proc. AAAI Conf. Artif. Intell.*, Edmonton, AB, Canada, Jul. 2002, pp. 476 – 483.

[17] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum-cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100 – 107, Jul. 1968.

[18] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 343 – 352, Jun. 2005.

[19] S. Choi and W. Yu, "Any-angle path planning on non-uniform costmaps," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Shanghai, China, May 2011, pp. 5615 – 5621.

[20] B. Angulo, A. I. Panov, and K. Yakovlev, "Policy optimization to learn adaptive motion primitives in path planning with dynamic obstacles," *arXiv*, arXiv:2212.14307, Dec. 2022.

[21] C. Zhou, B. Huang, and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing*, vol. 33, pp. 387 – 424, 2022, doi: 10.1007/s10845-021-01867-z.

[22] A. Jones, M. Schwager, and C. Belta, "A receding horizon algorithm for informative path planning with temporal logic constraints," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Karlsruhe, Germany, May 2013, pp. 5019 – 5024.

[23] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269 – 271, 1959.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.

[25] A. Buluç, S. Beamer, K. Madduri, K. Asanović, and D. Patterson, "Distributed-memory breadth-first search on massive graphs," in *Parallel Graph Algorithms*, D. Bader, Ed. Boca Raton, FL, USA: CRC Press, 2015, ch. 6, pp. 87 – 106.

[26] R. E. Tarjan, "Depth-first search and linear graph algorithms," *SIAM J. Comput.*, vol. 1, no. 2, pp. 146 – 160, 1972.

[27] D. Ferguson, M. Likhachev, and A. Stentz, "A guide to heuristic-based path planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Edmonton, AB, Canada, Aug. 2005, pp. 212–217.

[28] S. Koenig, M. Likhachev, and D. Furcy, "Lifelong planning A*," *Artificial Intelligence*, vol. 155, no. 1–2, pp. 93–146, 2004.

[29] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[30] M. Likhachev, D. Ferguson, G. J. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic A*: An anytime, replanning algorithm," in *Proc. AAAI Conf. Artif. Intell.*, Pittsburgh, PA, USA, Jul. 2005, pp. 262–267.

[31] L. Guo, S. Huang, J. Zhuang, and A. W. Sadek, "Modeling parking behavior under uncertainty: A static game theoretic versus a sequential neo-additive capacity modeling approach," *Networks and Spatial Economics*, vol. 13, no. 3, pp. 327–350, 2013.

[32] M. Chen and T. Chang, "A parking guidance and information system based on wireless sensor network," in *Proc. IEEE Int. Conf. Inf. Autom.*, Shenzhen, China, Jun. 2011, pp. 601–605.

[33] J. Yang, J. Portilla, and T. Riesgo, "Smart parking service based on wireless sensor networks," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Cape Town, South Africa, Feb. 2013, pp. 6029–6034.

[34] M. Quiñones, V. González, L. Quiñones, C. Valdivieso, and W. Yaguana, "Design of a smart parking system using wireless sensor network," in *Proc. IEEE Int. Conf. Inf. Autom.*, Loja, Ecuador, Aug. 2015.

[35] X. Chi, Z. Liu, J. Huang, F. Hong, and H. Su, "Optimization-based motion planning for autonomous parking considering dynamic obstacle: A hierarchical framework," *arXiv*, arXiv:2210.13112, Oct. 2023.

[36] D. Ferguson and A. Stentz, "Field D*: An interpolation-based path planner and replanner," in *Robotics Research: The 11th Int. Symp.*, Berlin, Germany: Springer, 2007, pp. 239–253.

[37] J. W. Tukey, *Exploratory Data Analysis*. Reading, MA, USA: Addison-Wesley, 1977.

[38] J. Portley, "5 ways smart parking systems improve urban transportation," *KnowHow*, May 22 2024. [Online]. Available: https://knowhow.distrelec.com/transportation/5-ways-smart-parking-systems-improve-urban-transportation/. [Accessed: May 2 2025].

[39] "Smart parking system: A solution for narrow, high-traffic lanes," LED Traffic Pro, 2025. [Online]. Available: https://www.ledtrafficpro.com/blogs/knowledge-share/smart-parking-system. [Accessed: May 2 2025].

[40] R. T. Dunphy, "SFpark: Solving parking and traffic problems with pricing," *Urban Land Magazine*, Sep. 1 2015. [Online]. Available: https://urbanland.uli.org/infrastructure-transit/sfpark-solving-parking-traffic-problems-pricing. [Accessed: May 2 2025].

[41] K. Barry, "City parking smartens up with Streetline," *WIRED*, Nov. 29 2010.[Online].Available: https://www.wired.com/2010/11/city-parking-smartens-up-with-streetline/. [Accessed: May 2 2025].

[42] H. Choset *et al.*, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA, USA: MIT Press, 2005.