# 基於 Three.js 的 Web3D 智慧倉儲環境模擬系統

專題編號:114-2-CSIE-S021

執行期限:114年第1學期至114年第2學期

指導教授:林惠勇

專題參與人員:112590024 邱紹溏

112490004 邱昱綸

## 一、摘要

建構一套實時3D倉儲自動化模擬系統，整合Vue.js、FastAPI 與 Three.js，透過WebSocket實現前後端實時通訊，支援A*、BFS、動態避障及死鎖檢測四種路徑規劃演算法的並行比較。系統配置兩台自主車輛，具備動態優先級調整。實驗結果顯示各演算法在路徑效率、碰撞避免與死鎖處理上呈現明顯差異，並提供視覺化演算法選擇依據。

## 二、緣由與目的

現有模擬工具如Unreal Engine跨平台性不足，NVIDIA系統授權成本昂貴，且兩者皆難以與Python強化學習工具整合。選擇 Three.js 作為 3D 模擬核心，基於瀏覽器運行。

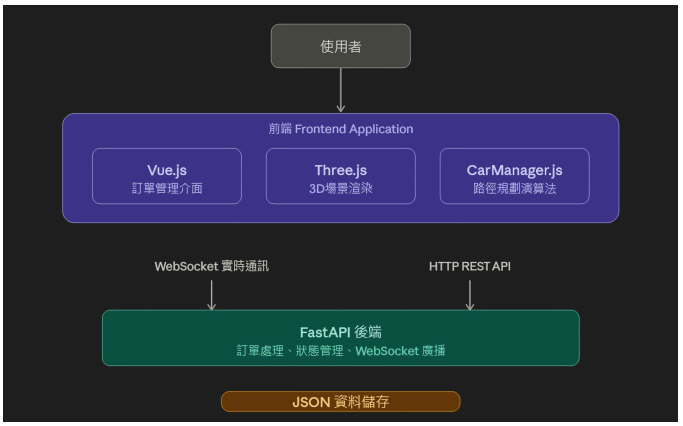
(一)建構虛擬倉儲環境:以Three.js建立含230個貨物箱的立體倉儲空間

(二)車輛協調:整合路徑規劃與碰撞避讓機制，實現兩台車輛的協同作業

(三)比較算法:並行比較A*、BFS、動態避障及死鎖檢測四種算法的差異

(四)開發完整管理系統:建立前端與後端，支援訂單處理與狀態追蹤。

## 三、架構流程




(圖1. 架構圖)

## 四、工具說明

前端採Vue.js建構訂單管理介面，搭配Three.js進行倉儲3D場景與車輛動畫的渲染，並以Vite作為開發建構工具。

後端以FastAPI處理訂單新增、查詢與狀態更新，透過WebSocket實現訂單狀態的廣播與前後端同步。

資料儲存採JSON檔案記錄訂單與貨物位置。演算法層面則整合A*路徑規劃、碰撞檢測與多車輛協作任務系統，共同實現自主車輛的智能協調。

## 五、實驗結果

以相同倉儲場景進行四種路徑規劃機制之比較，觀察指標包含:任務完成時間、路徑長度、碰撞次數、等待時間與死鎖發生率。測試情境涵蓋單筆訂單、連續多筆訂單及雙車同區域交會等高衝突狀況。

(一)**A***演算法：在大多數情境下能快速產生較短路徑，整體路徑效率最佳

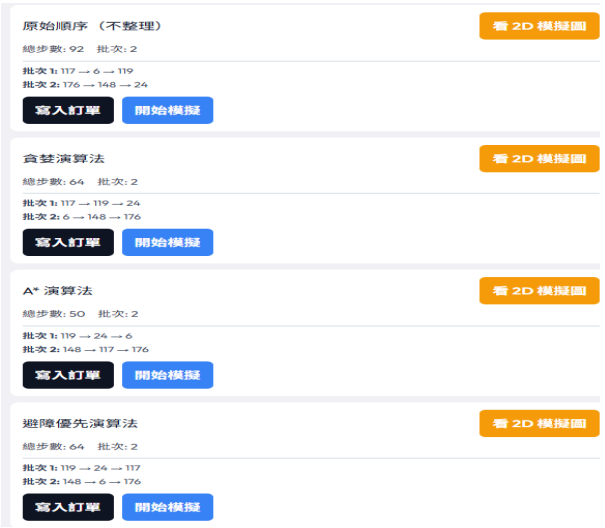
(二)**BFS**演算法：BFS可穩定找到可行路徑，實作簡單且結果一致性高。

(三)動態避障機制：導入基本等待與進階繞路策略後，車輛在交會路口的碰撞率明顯下降。基本等待可降低即時碰撞，但會增加任務延遲；進階繞路雖提高計算負擔，仍能有效縮短總等待時間並提升任務吞吐量。

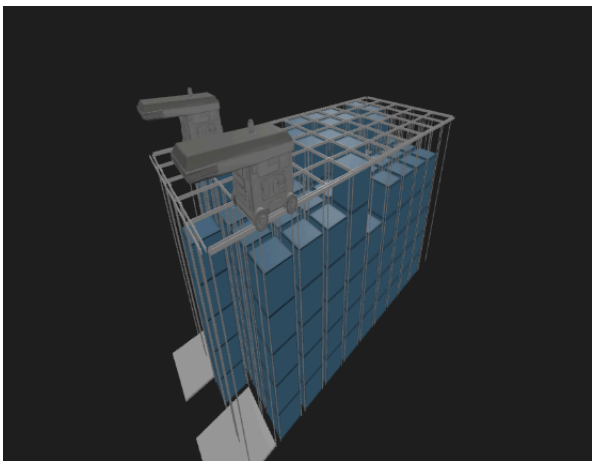
(四)整體比較結果

1. A*在路徑效率上表現最佳；
2. BFS具穩定性但效率較低；
3. 動態避障可顯著改善碰撞與等待問題；
4. 死鎖檢測機制可提升多車協作之可靠性。

因此，本系統已能有效呈現不同演算法(如圖2.)在智慧倉儲情境中的性能差異，提供後續演算法選型與調度策略優化之依據。




（圖2. 各個算法比較圖）



（圖3. 3D模擬圖）

## 六、結論

成功建構一套實時3D倉儲自動化模擬系統，實現兩台自主車輛的協同作業與四種路徑規劃演算法的視覺化比較。系統支援完整的訂單處理流程，並透過WebSocket實現前後端實時同步。

未來方向擴展：導入強化學習優化路徑規劃決策、支援更多車輛的協作調度，以及串接真實機器人系統，逐步從模擬平台發展為實際倉儲管理應用。

## 參考文獻

[1] E. You, "Vue.js Documentation," 2024. [Online]. Available: https://vuejs.org/
[2] mrdoob, "Three.js," GitHub, 2024. [Online]. Available: https://github.com/mrdoob/three.js
[3] S. Ramírez, "FastAPI Documentation," 2024. [Online]. Available: https://fastapi.tiangolo.com/
[4] E. You et al., "Vite Documentation," 2024. [Online]. Available: https://vitejs.dev/
[5] AutoStore, "AutoStore System," 2024. [Online]. Available: https://www.autostoresystem.com/