行政院國家科學委員會補助專題研究計畫成果報告

開發一個高品質的 IEEE 802.11p 及 IEEE 1609 車輛無線網路模擬系統

計畫類別：☑個別型計畫　□整合型計畫
計畫編號：NSC　97-2221-E-009-065-MY3
執行期間： 2008　年 8 月 1　日至 2011　年 7 月 31　日

執行機構及系所：國立交通大學資訊工程系

計畫主持人：王協源 教授
計畫參與人員 ：

成果報告類型(依經費核定清單規定繳交)：□精簡報告　☑完整報告

本計畫除繳交成果報告外，另須繳交以下出國心得報告：
□赴國外出差或研習心得報告
□赴大陸地區出差或研習心得報告
☑出席國際學術會議心得報告
□國際合作研究計畫國外研究報告

處理方式：除列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年 ☑ 二年後可公開查詢

中　華　民　國　100　年　9　月 20　日

摘要

智慧型運輸系統(ITS)是近年一項備受矚目的重要議題，其藉由應用車輛無線網路，提供即時的資訊而達到增進運輸系統的安全，效率與舒適性。然而舊有的 IEEE 802.11 系列標準已不適用於車輛無線網路此一新興的網路型態，因此 IEEE 訂定了 802.11p/1609 標準，使其能符合智慧型運輸系統的相關應用。由於智慧型運輸系統與車用無線網路在商業發展及學術研究上的巨大潛力，有越來越多團隊投入此一研究領域。但目前市場上並沒有任何產品可供實驗進行研究，因此開發一個能模擬此類網路的軟體網路模擬器是非常重要的。

本計畫第一年度已完整實作 IEEE 802.11a/p 於 NCTUns 上，第二年度則成功實作 IEEE 1609 於 NCTUns 上，並與第一年的結果整合為一套完整的車用無線網路模擬系統，在第三年度我們成功開發了道路車流的模擬，並與前兩年所開發的網路模擬結合，使得 NCTUns 模擬器兼具網路與車流路網模擬的功能，之後我們在這個平台上陸續做了智慧型紅綠燈控制以及縮短緊急救護車輛到達事故現場時間的研究，都有很不錯的研究成果，藉由這些研究的產出，我們證實了 NCTUns 網路與車流模擬器已經足夠進行 ITS 的相關研究。

1. 前言

智慧型運輸系統(Intelligent Transport Systems, ITS)是近年一項備受矚目的重要議題,其藉由應用車輛無線網路,提供即時的資訊而達到增進運輸系統的安全,效率與舒適性。

在第一年以及第二年的計畫中，我們已經將 IEEE 802.11p 以及 1609 的規格開發且驗證完畢，但是在 ITS 的研究中，只有網路封包的模擬是不夠的，我們必須要有一套符合真實世界的道路網路以及車輛運動行為的車流模擬功能,才能支持 ITS 的研究，因此在我們第三年的計畫中，我們會完成道路網路的開發。

2. 計畫介紹

在第三年的計畫中，我們預計要完成的項目條列如下：

(1)道路網路的模擬
(2)符合真實世界車流運動行為的車輛行為模擬

第三年我們預計要完成真實到路網路的模擬，以方便開發和研究車輛模擬的相關應用與行為。我們會在 NCTUns 網路模擬器上，利用 GUI 來產生道路、紅綠燈、汽車以及 Roadside Unit(RSU)…等元件，使用者可依自己的需求拉出想要研究的路網系統。有了道路網路，我們接下來會開發車子的行進模型，讓車子的運動行為符合真實世界的車流模型，之後再搭配前兩年所整合開發的 802.11p 和 1609 網路模擬系統，來模擬車子在移動時的封包傳遞和訊息交換…等網路行為。之後我們會進行大規模的模擬，例如在複雜的路網上，放至上千輛的車，觀看跑大型模擬所耗費的系統資源，來做持續的效能改善，以期達到最佳的模擬結果。

參與的工作人員將藉由不斷的研究有關車輛行為的資料，瞭解車輛間行為的相互影響，並開發出符合真實世界車流運動的車輛行為模擬。車輛行為的模擬將會依據道路狀況的不同而改變，紅綠燈就是最好的例子，而車輛也會因收到其他車輛的資訊而在行為上做適當的判斷。當車輛到達十字路口時,將會把自身相關的訊息傳到 Roadside Access Points，並經由 Roadside Access Points 經由有線網路將訊息傳到行控中心,讓行控中心可依照十字路口的車流量而做出恰當的判斷，例如改變紅綠燈號誌變換的時間，來紓解塞車的情況。

最終我們將在此 NCTUns 模擬器上做一系列智慧型運輸系統的相關研究，例如:車輛藉由網路模擬系統來達到車輛與車輛的訊息交換，並研究車輛對收到的訊息如何反應才可將車禍發生率降到最低、行控中心如何依據目前的車流量來達

成車流的舒緩。這一系列的相關研究將對智慧型運輸系統這領域上有巨大的貢獻。

3. 道路網路設計實作與成果

在我們開發的道路網路中，我們新增了三個新的元件，如圖 1 所示，從左而右分別是道路、十字路口、擴充道路。

圖 1. 道路元件

我們用這三個元件建構了一個路網範例，如圖 2 所示。

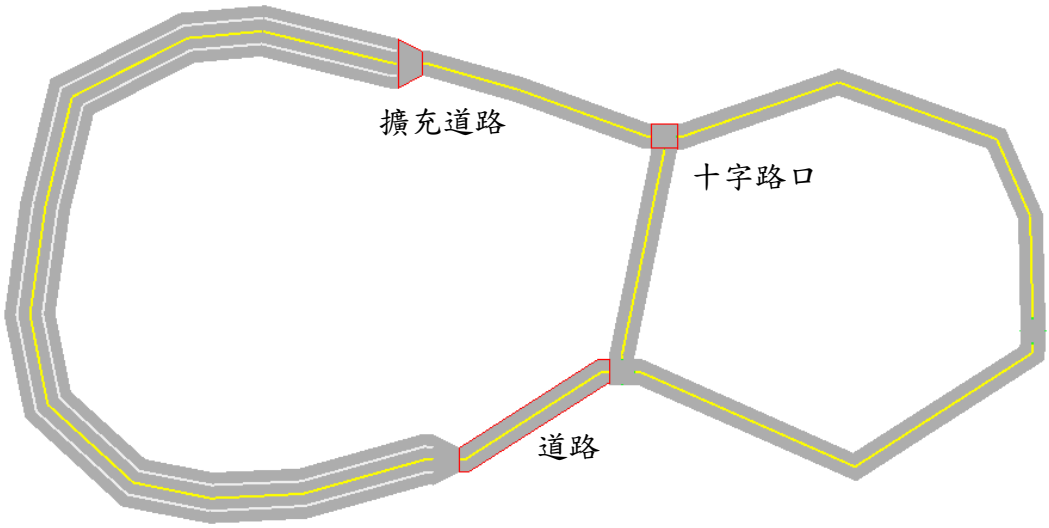


圖 2. 道路建構範例

等道路路網建構完成後，在模擬開始前，我們會先將這個路網以特定的資料結構儲存成.sim 檔，之後模擬引擎再從.sim 檔中把檔案讀取出來進行模擬。

接下來我們介紹一下路網.sim 檔的資料結構，在 NCTUns 的模擬引擎中路網的結構與 GUI 中的分類不同，在模擬引擎中的一條道路是由許多塊區塊所組成的，我們將路網分類的資料整理成表 1。

表 1. 路網組成元件

路網組成元件	說明
Road Block	路網組成的最小元件，由四個點座標形成的一個四方形。
Lane Block	許多的 Road Block 會形成一條 Lane，Lane 裡面會

	記錄它是由哪些 Road Block 所組成。
Edge Block	由許多的 Lane Block 所組成，例如：一條雙線道是由兩條不同方向的 Lane Block 所組成。

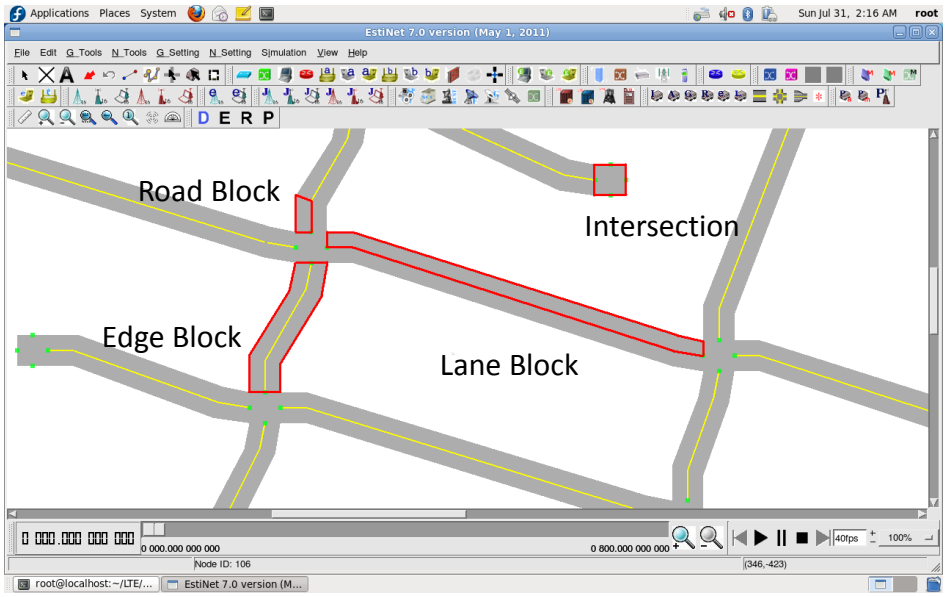


圖 3. 路網架構說明

如圖 3 所示，Road Block 是路網最小塊的區塊，許多的 Road Block 會組成一條 Lane Block，而數條 Lane Block 就會組成 Edge Block。而每個十字路口我們稱為 Intersection。接著我們繼續介紹各個階層的資料結構。

表 2. Road Block 的資料結構

```
RoadBlock
double x[4];
double y[4];
double a,b,c;
double direction;
int RID;
double width;
```

表 2 是 Road Block 的資料結構，x[4]與 y[4]分別代表了這個 Road Block 的四個角落的點座標，a,b,c 會構成一條線性方程式 ax+by+c 來表示這條 Road Block 的方向，direction 會以 0~360 來表示這個 Road Block 的方向，RID 是這個 Road Block 的唯一 ID，width 代表著這個 Road Block 的寬度。

表 3. Road Lane 的資料結構

RoadLane
vector<RoadBlock> RoadBlockID;

表 3 是 Lane 的資料結構，RoadBlockID 是一個 vector 串起構成這條 Lane 的所有 RoadBlock。

表 4. Road Edge 的資料結構

RoadEdge
int EID;
int NumOfLane;
int NumOfNode;
int *NID;
int L_Number;
int R_Number;
double length;
RoadLane *lane;

表 4 是 Edge 的資料結構，EID 代表的是這條 Edge 的唯一 ID，NumOfLane 是這指這條 Edge 由幾條 Lane 組成，NumOfNode 是指這條 Edge 與幾個十字路口連接，*NID 指向一個 Intersection 陣列，陣列大小與 NumOfNode 相同，L_Number 指的是左線道的數量，R_Number 指的是右線道的數量，length 是指這條 Edge 的長度，*lane 指向一個 RoadLane 陣列，陣列大小與 NumOfLane 相同。

表 5. Intersection 的資料結構

Intersection
int Nid;
int NumOfEdges;
int *EID;
double *direction;

表 5 是 Intersection 的資料結構，Nid 代表的是這個 Intersection 的唯一 ID，NumOfEdges 指的是這個 Intersection 連接的 Edge 數，*EID 指向一個 RoadEdge 陣列，陣列大小與 NumOfEdges 相同，*direction 與*EID 相同，指的是每條 Edge 的方向。

在模擬開始的時候，模擬引擎會讀取.sim 檔的資料存成以上的這些資料結構，由每個 Road Block 為最小單位組合成 Lane Block 再組成 Edge Block，然後各個 Edge Block 與 Intersection 再彼此連接而組成路網，之後每台車子只要知道自己的座標，就可以算出這台車子坐落在哪一條 Edge 的哪個 Lane 的哪一個 Road Block 上，達到車子與路網的互動。

在道路網路的設計方面，我們為了研究的真實性，所以我們讓模擬器能夠支援 shape file 格式的道路地圖，因此我們可以直接讀取真實世界的地圖檔進入模擬器中當作模擬的路網來使用，如圖 4 就是台北市文山區的真實路網。

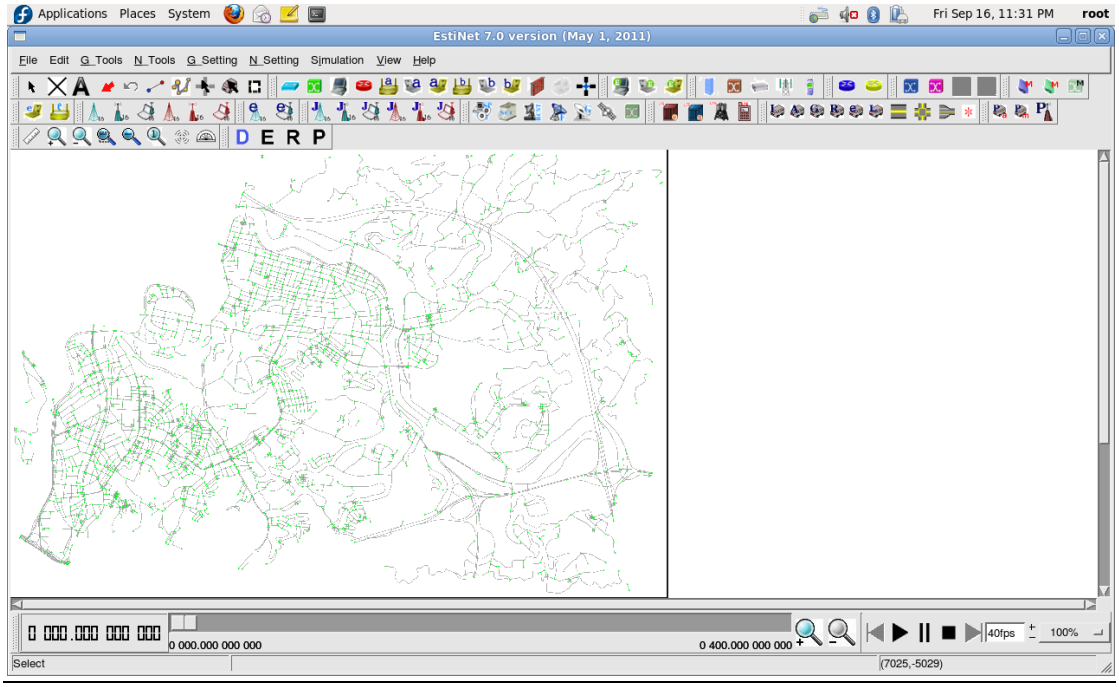


圖 4. 台北市文山區的路網

4. 真實世界車流運動行為設計實作與成果

在車流的開發上，我們新增了兩種不同種類的車子，如圖 5 所示，由左而右分別是，Agent Mode 的車子以及 Module Mode 的車子。

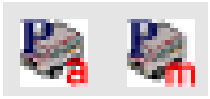


圖 5. 車輛元件

這兩種不同種類的車子是為了不同的用途而設計的，Agent Mode 的車子在模擬進行的時候，模擬引擎會 fork 出新的程序去控制這輛車子的行為，而我們可以在這類型的車輛上運行應用程式，來達到車輛間的互動或訊息的傳遞。而 Module Mode 的車子是由模擬引擎統一控管的車子，其目的是以較少的資源運作大量的車子來擔任背景車流，這類型的車輛可以進行數千台的車輛行為模擬。

以上兩種車輛除了控制角色不同之外，其餘的行為是大同小異。我們可以為車輛配置車子的性能，車輛的性能有最大車速、最大加速度以及最大減速度，而這些性能會寫在 Car Profile 裡面，然後我們可以隨機或指定分配每台車輛所擁有的 Car Profile，其設定如圖 6 所示。

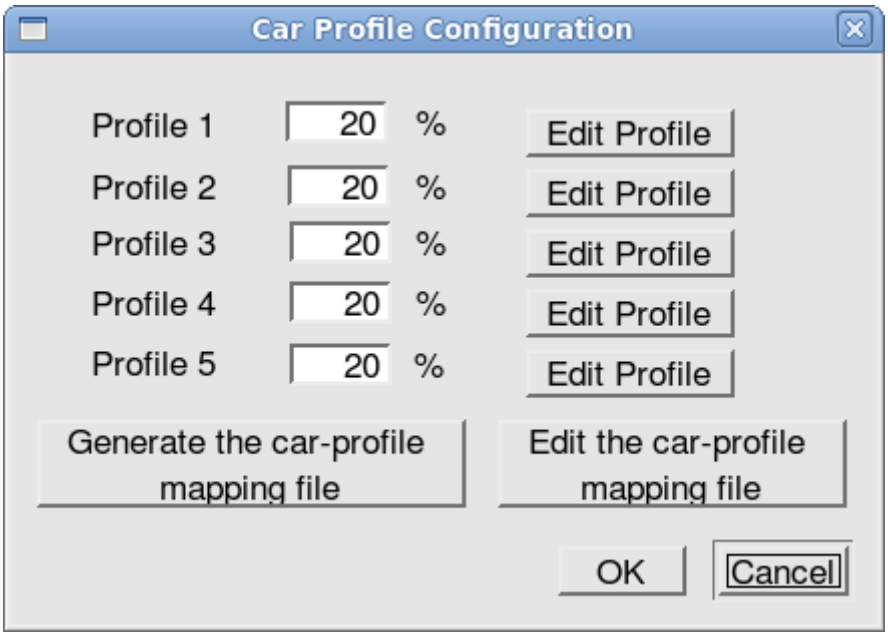


圖 6. Car Profile 的設定

設定完車輛的性能後，車輛就會依據這些性能模擬車輛的運動行為，在前方沒車的時候會依照最大加速度來加速、前方有車輛或遇到紅燈的時候，會減速，

藉由這些動作來模擬車輛的運動行為。

在車輛的運動行為中，還有很重要的一點是移動路徑的選擇，當車輛行駛到十字路口時，該往哪個方向轉向，哪個行為能符合真實車輛移動的情形。在我們的模擬器中，我們開發了一個 Landmark-based 的車輛行進模型，以下會詳細的說明這個車輛行進模型。

我們認為車輛在道路上行駛，都會朝向一個目的地前進，所以為了讓車輛的運動行為符合現實，我們在模擬器中，新增了一個新的元件 Landmark，如圖 7 所示。

圖 7. Landmark 元件

‘

Landmark 是一個城市熱點，車輛可以選擇朝向某一個 Landmark 前進，而行進路線會選擇最短路徑，當有很多車輛選擇同一個 Landmark 為目的地前進的時候，在這個 Landmark 附近會有大量車流湧入而造成塞車。為了讓 Landmark 的能力更多元，我們設計了三種不同熱門程度的 Landmark，高熱門程度的 Landmark 就好比 101 大樓、著名風景區，會吸引大量車潮，而中熱門程度的 Landmark 就好比電影院、大賣場，會吸引不少車輛，而低熱門程度的 Landmark 就像是一間餐廳、會吸引少數車輛，在地圖上巧妙的配置這些不同熱門程度的 Landmark，可以讓車流行進的路線貼近現實狀況。

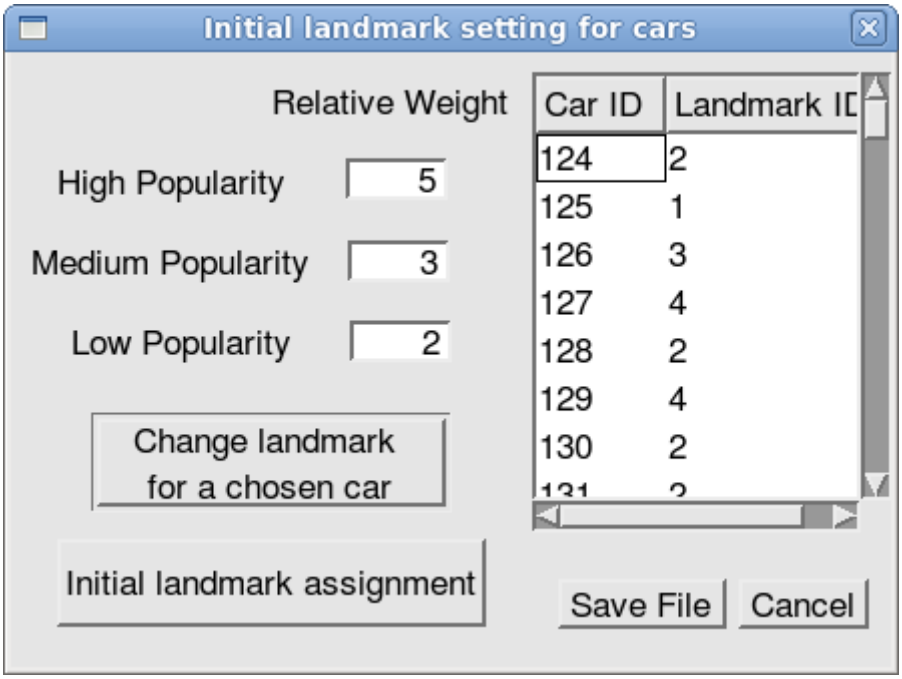


圖 8. Landmark 與車輛的配對

在我們模擬器的設定中，可以為三種不同熱門程度的 Landmark 設定不同的權重，之後隨機分配車輛與 Landmark 的對應時，就會依照權重隨機分配一定比例的車輛到所屬的 Landmark，之後這些車就會朝向所指定的 Landmark 前進，當然也可以為某幾台車輛指定配置其所屬的 Landmark，如圖 8 所示。

接下來我們來看佈署 Landmark 對車流的行進路線能有什麼樣的影響，首先我們先測試不使用 Landmark，然後讓車輛到達路口就隨機轉彎，模擬 400 秒後看車輛的分佈情形，以及我們另外佈置 Landmark 的兩個案例，同樣模擬 400 秒後，看車輛的分佈情形有哪些差異。

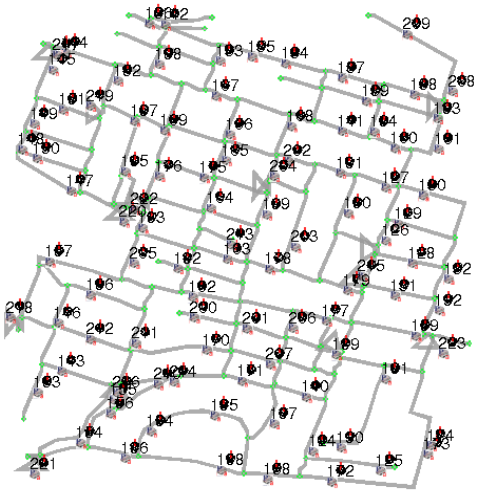


圖 9.100 台車輛初始佈署情形

看圖 9，這個地圖是使用台北市文山區路網的其中一個小片段，地圖大小是 700 公尺乘 700 公尺，我們在上面隨機分佈了 100 輛車，目前我們還沒有放置 Landmark，所以車輛到達路口是隨機轉彎，圖 10 是模擬 400 秒後的結果。

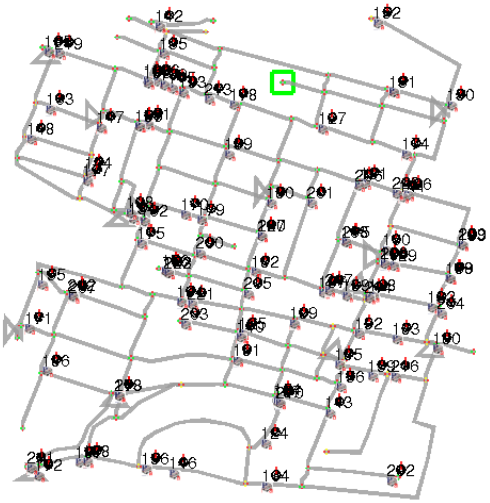


圖 10. 使用隨機轉彎模擬 400 秒後的結果

由圖 10 的結果我們可以看出車輛還是蠻平均分佈在道路上，沒有特別壅塞而造成塞車的道路，與現實世界的車輛分佈情形差距甚遠，接下來我們要看有佈置 Landmark 之後的情形。

圖 11 與圖 11 分別是佈置了 4 個 Landmark 以及 10 個 Landmark 的地圖，我們來看模擬 400 秒後車輛的分佈情形。

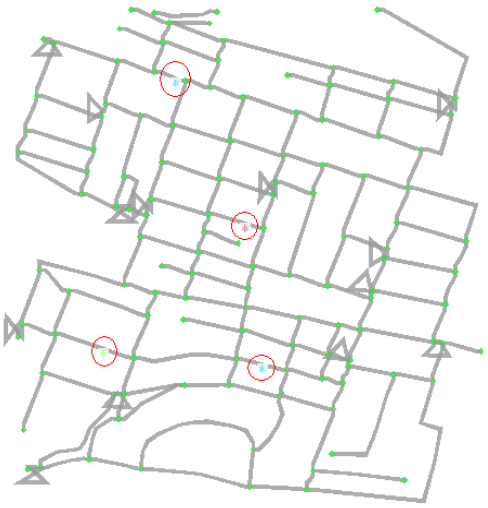


圖 11. 佈署 4 個 Landmark

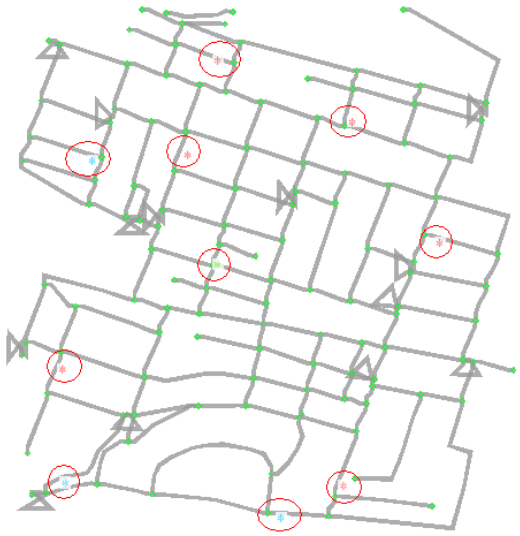
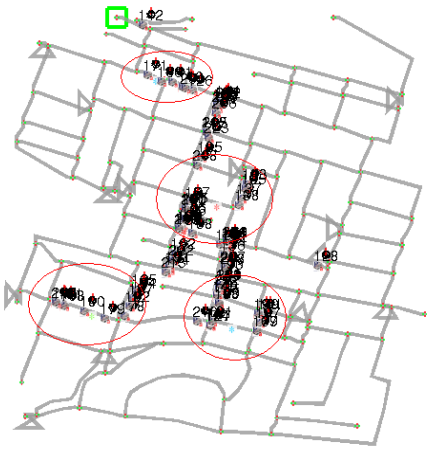


圖 12. 佈署 10 個 Landmark

圖 13 與圖 13 是模擬 400 秒後的結果，我們可以看到有 Landmark 的附近會有車流匯聚的情形，而在各個 Landmark 之間的道路也會有大量車輛行走而形成主要幹道，左邊的圖因為 Landmark 分佈個數少而且集中，所以車輛的分佈較為緊密，而右邊的圖因為 Landmark 分佈的個數較多而且分散，所以車流也會比較分散，所以不同的 Landmark 擺設的位置不同，就可以巧妙的控制車輛的行走方式，因此可以依照使用者的需求佈置 Landmark 讓車流的分佈情形符合使用者的需求。



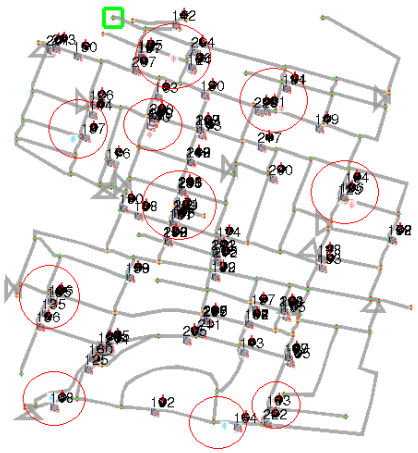


圖 13.4 Landmark 模擬 400 秒後的結果 圖 14.10 Landmark 模擬 400 秒後的結果

5.後續研究 – 智慧型交通號誌燈控制

在開發出路網以及真實的車流模型後,我們就藉由這平台陸續做了一些研究,其中一篇是智慧型交通號誌燈控制的研究。

5.1 摘要

在現今各大城市中,壅塞的交通一直是亟欲解決的問題,而隨著資訊產業的快速發展,智慧型交通運輸的研究漸趨熱門,其中交通號誌燈的控制是智慧型交通運輸中重要的一環。交通號誌燈控制的研究分為兩個種類,一種是固定時間式的交通號誌燈控制,也就是事先幫每個交通號誌燈安排好紅綠燈的循環周期,藉此達到疏通交通阻塞的目的。而另一種是變動時間式的交通號誌燈控制,藉由動態的評估車況來配給紅綠燈的時間,以應付隨時變化的交通狀況。

在本論文的研究中,我們選擇研究變動時間式的交通號誌控制,我們會設計不同的車流拓樸 (traffic pattern)、不同的時間間隔、以及不同的車流資訊蒐集程度,來深入的研究變動時間式的交通號誌控制演算法的運作情形,並觀察演算法在不同情境下的運作情形,然後提出改善的建議,透過模擬案例實驗證實,我們的改良確實有效。

5.2 智慧型控制演算法改良

我們將智慧型交通號誌燈的演算法實做到 NCTUns 模擬器,並且研究各種拓樸的模擬數據後,我們發現原有的智慧型交通號誌控制演算法有幾個部分還可以繼續改良。分別說明如下:

5.2.1 固定時間的探查車況改成動態時間

在原本的演算法中會每隔一個固定的時間間隔去探查現在道路的車況如何,然後視車況配置紅燈與綠燈,因為是固定的時間間隔,所以每次配置完紅綠燈後,必須等待下一次的車況探查才有可能改變紅綠燈,但車流的狀況並不對等,並且差異很大,如果某條路上只有少數一兩台車要通過,卻配置給他 20~30 秒的綠燈時間,此時如果另外一條道路上有車輛到來,會增加額外的等候時間。

所以在我們改良的演算法中,我們並不把探查車況的時間固定住,我們會先評估道路上是否有衝突的車流在競爭紅綠燈,如果只有單向的車流,那我們可以給予一段比較長的綠燈時間,然後等候下一次探查車況的時間,如

果有不同方向的車流要競爭紅綠燈，那我們會觀察車流的數量是否差異很大，如果一邊只有少數一兩台車，那我們會給予他短時間的綠燈，然後馬上把綠燈還給較多車流的道路。

固定時間探查車況還有另外一個缺點，探查時間如果過長，會導致上述我們說的問題，配置大量時間給少數車輛，造成其他路的車輛不必要的等候時間。但如果探查時間過短，會有另外一個問題是，當兩條道路的車流差不多壅塞，頻繁的探查車況有可能會導致頻繁的紅綠燈變換，然而每次紅綠燈的轉換，都有會有 5 秒的黃燈時間，這個時間雙向的車都無法通行，所以頻繁的變換紅綠燈，對雙向的車輛而言都是額外的負擔。

因此在我們改良的演算法中，遇到雙向車流都壅擠的時候，我們會配置一個較長的綠燈時間，之後再換另一條道路有較長的綠燈時間，這樣可以避免因為頻繁變換紅綠燈所帶來的負面影響。

5.2.2 使用老化技術取代最長紅燈時間

在未改良的智慧型交通號誌控制演算法中，我們會評估車流狀況，給予較多車流的道路綠燈以舒緩交通，但如果兩條車流的流量差異過大，例如主要幹道與鄉間小路的交接，有可能造成演算法每次都給予主要幹道綠燈，而導致鄉間小路上的車流無限期的等待，因此在舊有的演算法中會使用最長的紅燈時間，以避免少數車輛被無限期的紅燈困住，而我們實作的演算法中將最長紅燈時間制訂為 90 秒。

而在我們改良的演算法中，我們要引進老化技術來取代最長紅燈時間，這邊可以分兩個部分來說，第一是確保車子不會被無限期的紅燈困住，第二是盡快恢復主要幹道的通暢。

為了達到第一點的目標，確保車子不會被無限期的紅燈困住，我們將使用老化技術，也就是說，當演算法第一次將綠燈給予主要幹道的車流時，遇到紅燈的車流會增加自身的權重，以便在下一次演算法的判斷中，比較容易搶得綠燈，但如果還是被判定為紅燈，則再次加重權重，最多連拿兩次紅燈，第三次就一定會搶到綠燈，在這個機制下，車子不會被無限期的紅燈困住，也不用等到 90 秒這個最長的紅燈時間，而且主要幹道的車在沒有車流競爭的道路上，也不用因為到了 90 秒最長的紅燈時間就被迫將綠燈換給沒有車流的道路，因而增加車子的延遲。

而為了達到第二點的目標，當我們因為老化技術而要把綠燈讓給車流量較小的道路時，我們會觀察此時道路上的車輛數，然後針對車輛數給予相對

應的綠燈時間，讓這一批車通過十字路口後，馬上把綠燈還給主要幹道的大批車流，如此一來可以在顧及公平性的同時又不至於拖慢多數車輛的行車時間。

5.3 模擬設計與結果分析

由於篇幅的關係，我們這邊會提到論文中的兩個模擬案例，分別是模擬案例 C 以及模擬案例 D，以下將詳細介紹。

5.3.1 模擬設計 C

模擬設計 C 是為了比較我們改良過後的演算法與原設計演算法在效能上的差異，關於模擬設計 C 的設計目的與說明請見表 6 與表 7：

表 6. 模擬設計 C 的設計目的與說明

模擬設計編號	C
設計目的	為了比較我們改良過後的演算法與原設計演算法在效能上的差異。
模擬設計說明	我們改良的演算法是使用動態時間間隔，與原本設計的固定時間間隔的演算法有設計上的差異，我們想知道這兩種演算法對於行車延遲的改善有多少的效果。

表 7. 模擬設計 C 的模擬參數

組別	**演算法**	**時間間隔 (最短綠燈時間)**	**車流資訊蒐集**	**車流拓樸**
0	不使用智慧型控制演算法	每 40 秒變換一次燈號	100 %	拓樸 A,B,C
1	原智慧型控制演算法	每 15 秒評估一次車況	100 %	拓樸 A,B,C
2	改良後智慧型控制演算法	動態時間評估車況	100 %	拓樸 A,B,C

5.3.2 模擬設計 D

模擬設計 D 是想研究，在車流資訊蒐集不完整的情況下，依照車況來變動交通號誌燈的演算法是否能正常運作，以及運作的效能如何，關於模擬設計 D 的設計說明與目的請見表 8 與表 9：

表 8. 模擬設計 D 的設計目的與說明

模擬設計編號	D
設計目的	研究在車流資訊蒐集不完整的情況下，依照車況來變

	動交通號誌燈的演算法是否能正常運作，以及運作的效能如何。
模擬設計說明	假設我們擁有所有車輛的位置與行車速度，我們可以很適當的配置交通號誌燈的燈號，但現實狀況不見得所有的車流資訊我們都能夠掌握，所以我們想知道在車流資訊掌握不充足的情況下，演算法的運作是否正常，我們也想知道當我們掌握車流資訊到哪個程度就能夠準確的使用智慧型交通號誌。

表 9. 模擬設計 D 的模擬參數

組別	演算法	時間間隔 (最短綠燈時間)	車流資訊 蒐集	車流拓樸
0	不使用智慧型控制演算法	每 40 秒變換一次燈號	0%~100%	拓樸 A,B,C
1	原智慧型控制演算法	每 15 秒評估一次車況	0%~100%	拓樸 A,B,C
2	改良後智慧型控制演算法	動態時間評估車況	0%~100%	拓樸 A,B,C

5.3.3 模擬設計 C 之結果分析

在模擬設計 C 中我們希望比較原始演算法以及改良過的演算法的效能，在圖 15 與圖 15 中可以看到，在車流密度不高的環境下，我們改良過的演算法能比原始演算法減少更多的行車延遲時間，因此效能表現上比起原始演算法更好，而且對於被延遲的車輛來說，能減少他們的延遲時間，增加演算法的公平性，所以在公平性的表現上，我們改良過的演算法也比原始演算法更好。

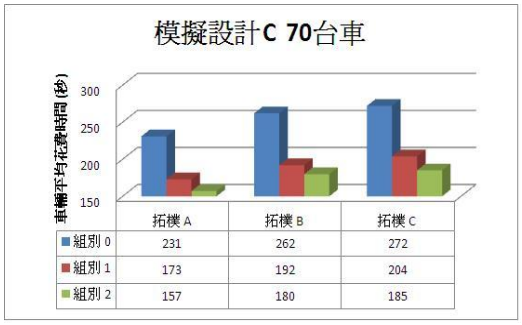


圖 15. 70 台車的車輛平均花費時間

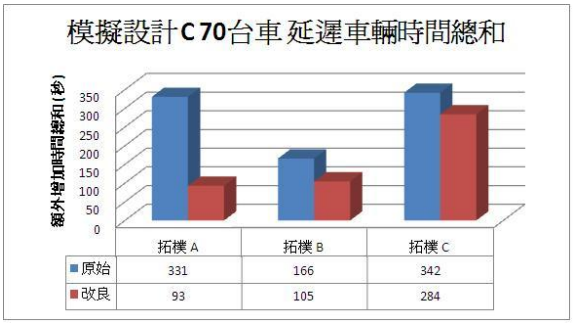


圖 16. 70 台車的車輛延遲時間總和

而在圖 17 中可以看到，在車流密度高的環境下，我們改良過的演算法在車輛平均行車延遲上的效能表現跟原始演算法差不多，這是因為車流密度高的環境下，兩邊的車流都呈現飽和，因此不管給予哪邊綠燈都會有另一邊的車流在等待，

所以整體的效能表現上很難提升，不過我們可以提升演算法的公平性，在圖 17 中我們可以看到，改良過的演算法能有效的減少被犧牲車輛的延遲時間，因而提升演算法的公平性，所以在公平性的表現上，改良過的演算法的確比較好。

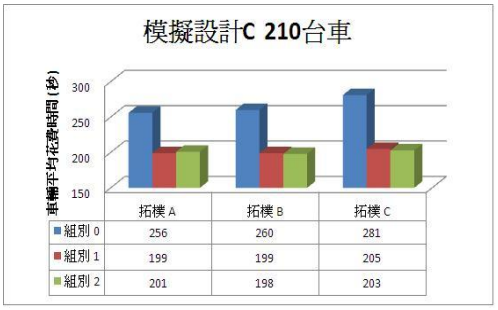


圖 17.210 台車的車輛平均花費時間

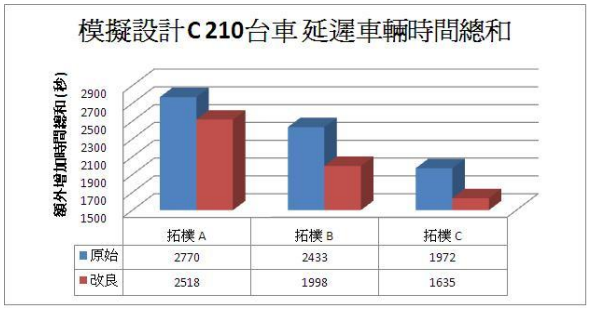


圖 18.210 台車的車輛延遲時間總和

5.3.4 模擬設計 D 之結果分析

在模擬設計 D 中，我們希望比較原始演算法以及改良過的演算法在車流資訊蒐集不齊全的情況下的效能表現。由圖 19 圖 19 中可知，原始演算法在車輛資訊低於 40%的環境下，效能表現會急速下降，到了只剩 20%的環境時，甚至比不使用演算法的交通號誌控制還差，但我們改良過的演算法，卻能夠在車輛資訊只掌握 20%的環境下，還保有一定程度的運作效能，這是因為我們改良過的演算法是根據車輛的數量配給秒數，所以當車輛資訊掌握不足的時候，我們會低估現有的車輛數而給予較少的綠燈時間，但也因此可以快速的進入下一次的判定，所以如果判斷錯誤，就能很快的在下一次的判定中修正回來。

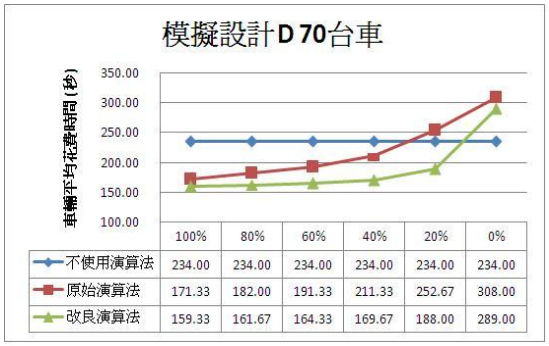


圖 19.70 台車的車輛平均花費時間

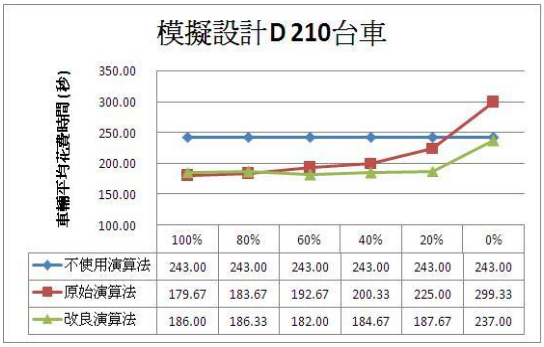


圖 20.210 台車的車輛平均花費時間

5.4 結論

在這個資訊產業迅速發展的時代，智慧型交通運輸系統也迅速發展，智慧型的交通號誌燈控制也是相當重要的一環，交通號誌控制又分為固定時間式的交通號誌控制與動態時間式的交通號誌控制。

在本論文中，我們選擇動態時間式的交通號誌控制進行深入的研究與分

析，透過不同的車流拓樸、時間間隔以及車流資訊蒐集完整程度，對動態時間式的交通號誌控制演算法進行觀察與分析，藉此了解演算法的運作效能，並觀察演算法在不同環境下的模擬結果，提出改良的建議，然後透過模擬案例證實，我們的改良確實有效。

我們將本論文中，經由觀察、模擬以及改良原本演算法後所得到的結果整理成以下的結論：

1. 在原本的演算法運作下，比起沒有使用演算法的交通號誌燈，確實能有效的減少車輛到達目的地的平均花費時間，而我們改良過後的演算法，比起原始的演算法，在車流密度低的時候，能更有效的減少車輛到達目的地的平均花費時間，而在車流密度高的時候，也能維持跟原本演算法差不多的效能。

2. 在原本的演算法運作下，會有少數車輛增加行車延遲時間，而在我們改良後的演算法中，能確實減少這些被犧牲車輛的延遲時間，提升整體的公平性。

3. 在我們使用不同時間間隔的分析下，發現原本動態時間評估車況的演算法，評估的時間越短越好，但是太短暫又會有頻繁變換交通號誌的問題，所以以十秒左右為評估車況的時間間隔，能帶來比較大的利益。

4. 在車流資訊蒐集不足的環境中，原本演算法的延遲時間會隨著資訊蒐集程度的減少而快速增加，而我們改良過後的演算法在車流資訊蒐集減少到 20%以前都還能維持不錯的效能表現，所以我們改良過的演算法比起原本的演算法更能承受資訊的遺失。

6. 後續研究 - 利用IEEE802.11p/1609技術來縮短緊急車輛到達事故地點的時間

在開發出路網以及真實的車流模型後，我們就藉由這平台做的另外一項研究是利用 IEEE802.11p/1609 技術來縮短緊急車輛到達事故地點的時間，以下將詳細介紹。

6.1 摘要

事故救援對拯救生命和緊急救護是一個重要的論題。事故救援中的一項議題就是如何縮短緊急車輛（例如救護車、消防車、警車）到達事故現場的時間使得救護作業可以順利的進行。舉例來說，救護車應盡快地到達車禍現場，如此一來傷患才能盡早地被照顧到。然而，在大都市的區域裡，塞車可能使得緊急車輛難以快速地到達事故現場。這是因為在道路上的車輛有時會難以讓出道路讓緊急車輛經過。在這個情況下，由於無法預期的塞車，拯救傷患或消滅火災的寶貴時間將會被浪費掉而造成無法挽回的悲劇。

為了解決上述的問題，在此篇論文中，我們提出一種基於IEEE 802.11p/1609無線技術的交通控管機制來控制車流狀況和交通號誌來縮短緊急車輛到達事故現場的時間。在所提出的機制裡，行控中心用來監控道路上的車流情況並提供交通資訊給緊急車輛。行控中心會動態地改變道路上的交通號誌來幫助緊急車輛能快速地到達事故現場。

我們的模擬結果顯示，在我們所研究的案例中，我們所提出的機制能夠將緊急車輛到達事故現場的時間大幅地縮短 30%至 50%。

6.2 機制的設計與實作

我們提出的車流控制機制，利用了三種功能，(1)移動路徑安排(2)疏散車輛的訊息傳遞(3)動態交通號誌燈的控制，來達到讓緊急救護車輛快速且順暢的到達事故現場。

(1)移動路徑安排

交通控制中心會安排緊急救護車輛到達事故現場的移動路徑，這種移動路徑必須是最短時間的路徑，以避免緊急救護車輛花費較多時間才到達事故現場，最短時間路徑是使用 A*演算法找出來的。A*演算法是最好的搜尋演算法，該演算法可以找到從源頭到目的地的最小成本路徑。產生最短時間路徑的程序如下所述。

1. 當萬芳醫院收到一個事故地點的緊急意外救護，他會跟交通控制中心報告

事故地點的位置。

2. 交通控制中心收到萬芳醫院的報告時，它會產生一個從院方醫院到事故地點的最短時間路徑，然後將此路徑傳回醫院，讓緊急救護車輛依照此路徑行駛。

(2)疏散車輛的訊息傳遞

疏散車輛的訊息式封裝在 WSM (WAVE Short Message)中，是用來通知鄰近車輛，緊急救護車輛已經靠近，讓鄰近車輛讓路給緊急救護車輛通行，其訊息傳遞的程序如下。

1. 緊急救護車輛會定期的廣播疏散車輛的訊息給路旁的 RSU，這訊息裡面夾帶著緊急救護車輛的 ID 以及位置並封裝在 WSM 中。
2. 當 RSU 收到周圍的緊急救護車輛廣播的疏散車輛訊息，它會將此訊息透過有線網路傳輸給交通控制中心。
3. 當交通控制中心收到疏散車輛的訊息，因為它知道緊急救護車輛的移動路徑，所以他會送出新的疏散車輛訊息給緊急救護車輛前面的十字路口上的 RSU，新的疏散車輛訊息中包含緊急救護車輛現在以及未來會經過的道路 ID。
4. 當十字路口上的 RSU 收到新的疏散車輛訊息，它會使用 WSM 將訊息廣播給周圍的車輛。
5. 當車輛收到 WSM 的訊息，因為裡面包含緊急救護車輛會經過的道路 ID，所以在這到路上的車輛如果會阻擋到緊急救護車輛的通行，則他們會讓路給緊急救護車輛通行，因此緊急救護車輛可以無障礙的快速通行。

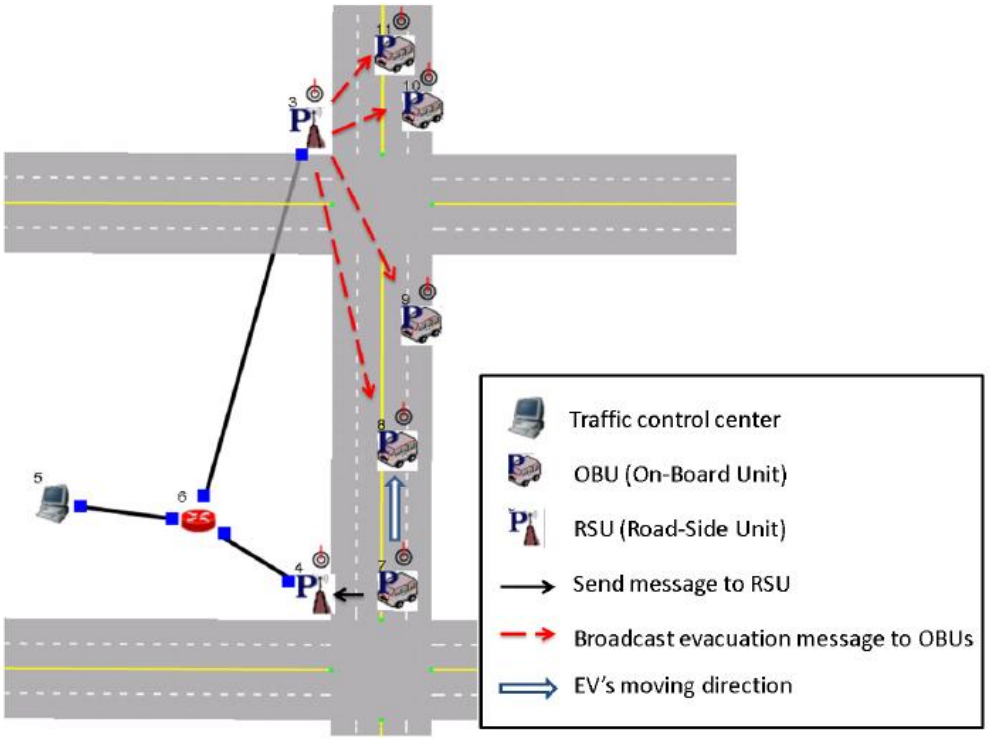


圖 21.疏散車輛訊息傳遞的範例

如圖 21 所示，緊急救護車輛(編號 7)會透過 WSM 定期廣播疏散車輛的訊息，訊息裡面包括緊急救護車輛的 ID 跟位置。十字路口的 RSU(編號 4)收到 WSM 的訊息後會把訊息轉送到交通控制中心(編號 5)，當交通控制中心收到訊息後，它會找出緊急救護車輛現在以及之後會經過的道路編號，然後將這些訊息傳送到十字路口的 RSU(編號 3)，之後 RSU 會將疏散車輛的訊息廣播給周圍的車輛(編號 8,9,10,11)，當車輛(編號 8~11)收到廣播的 WSM 訊息後他們會保留道路給緊急救護車輛，讓緊急救護車輛可以快速通過。

(3)動態交通號誌燈的控制

交通號誌燈由交通號誌控制中心控制，緊急救護車輛會定期廣播紅綠燈燈號請求、車輛 ID 以及所在位置的 WSM 訊息，當 RSU 收到訊息後會將此訊息傳送到交通控制中心，它會將此訊息轉送到交通號誌控制中心，當交通號誌控制中心收到紅綠燈燈號的請求，它會將緊急救護車輛前方需要疏散的車輛所在的十字路口的燈號轉成綠燈，讓需要疏散的車輛能順利通行，以避免緊急救護車輛被阻塞在車陣後。

6.3 機制的設計

我們設計的交通控制機制是為了要減少緊急救護車輛到達事故現場的時間，如之前所描述的，交通控制機制有三種主要的功能(1)移動路徑的安排(2)疏散車輛的訊息傳遞(3)動態交通號誌燈的控制。我們設計了數種不同的機制來進行評估，以確認每種機制對於緊急救護車輛到達事故現場的時間有什麼樣的影響，如下表所示。

	機制 A	機制 B	機制 C	機制 D
移動路徑的安排	∨	∨	∨	∨
疏散車輛的訊息傳遞		∨		∨
動態交通號誌燈的控制			∨	∨

機制 A：這個交通控制的機制只有提供移動路徑的規劃，緊急救護車輛會走最短時間的路徑去事故現場。這個機制的效能是用來衡量其他機制效能的基礎指標。

機制 B：這個交通控制機制提供了移動路徑規畫與疏散車輛訊息傳遞，透過機制 B 與機制 A 的比較，能幫我們分析疏散車輛訊息傳遞所帶來的影響。

機制 C：這個交通控制機制提供了移動路徑規劃與動態交通號誌燈控制，透過機制 C 與機制 A 的比較，能幫我們分析動態交通號誌燈控制所帶來的影響。

機制 D：這個交通控制機制提供了移動路徑規畫、疏散車輛訊息傳遞與動態交通號誌燈控制這三種主要功能，透過機制 D 與機制 A 的比較，能幫我們分析最佳的機制所能帶來的影響。

6.4 模擬情境

我們模擬的地圖是位於台北市大安區的萬芳醫院附近，如圖 22 所示，其中 S 是我們的源頭萬芳醫院，有兩個事故地點 A 跟 B，因此在我們的模擬中會有兩條路徑 S → A 以及 S → B。

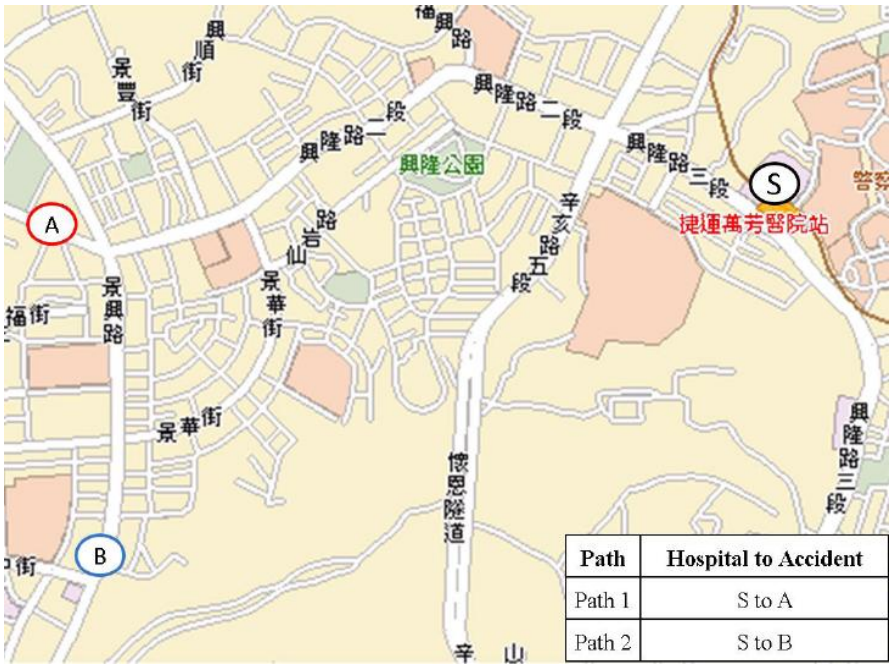


圖 22. UrMap 上萬芳醫院附近的道路地圖

圖 22 是電子地圖，但是在模擬器中我們必須建構真實的路網地圖，此時我們選用讀取真實世界的路網 shapefile 格式，讓模擬的情境更貼近於現實。如圖 23 所示。

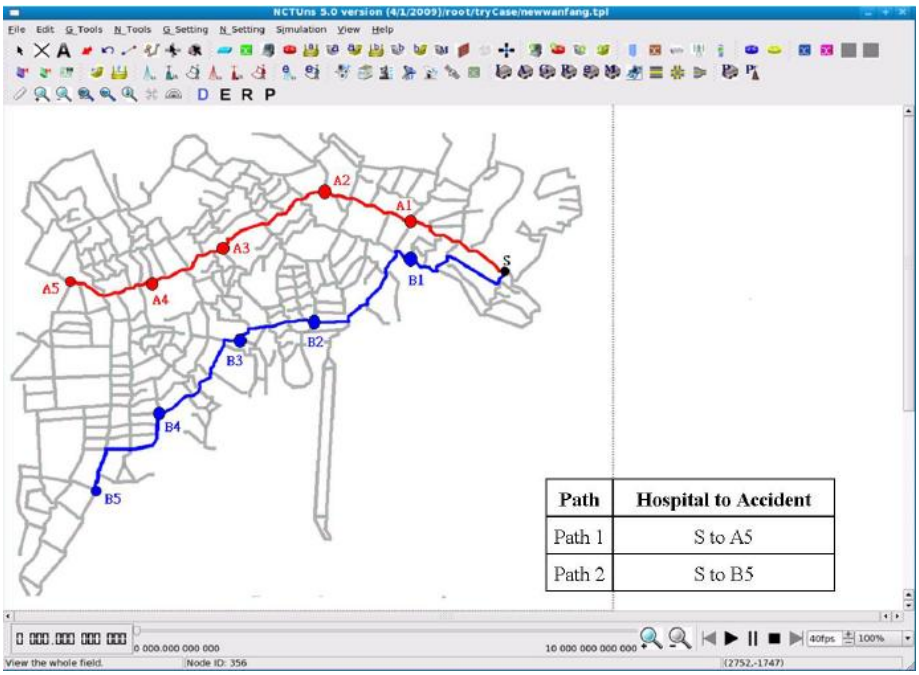


圖 23. NCTUns 上萬芳醫院附近的街道圖

在圖 23 中，S 代表是緊急救護車輛的起始位置萬芳醫院，而 A5 與 B5 是距離源頭 S 有 25 個十字路口遠的事故地點，模擬的參數如表 10 所示，對每一條路徑，我們會每隔 5、10、15、20 個十字路口會設定一個終點 A1 到 A5 以及 B1 到 B5。我們也會設定不同的紅綠燈周期，有 30 秒、60 秒以及 90 秒，以分析不同的紅綠燈周期對於各種交通控制機制的影響，在表 11 中我們會評估兩條不同的路徑，路徑 1 以及路徑 2，去觀察不同的路徑對於各個機制效能的影響，在表 12 中，我們展示了不同的車輛間距 10、20、30(公尺/車輛/lane)。

表 10. 模擬情境參數設定

參數	設定
訊息長度(bytes)	< 1400
網路種類	IEEE 802.11p/1609 WSMP
源頭	萬芳醫院
模擬時間	源頭到目的地
網路與路網模擬器	NCTUns
地形尺寸	2000*1500
車輛數	1500~4000
路寬(公尺)	3
車子佔的道路表面積(公尺^2/車輛)	30, 60, 90
終點 (從源頭開始算經過幾個十字路口)	5, 10, 15, 20, 25
紅綠燈周期(秒)	30, 60, 90

表 11.從萬芳醫院到事故現場的路徑

路徑	醫院到事故現場
路徑 1	S → A
路徑 2	S → B

表 12.平均車輛間距

交通條件	平均車輛間距 （公尺/車輛/lane）
A	10
B	20
C	30

6.5 模擬結果

我們使用 ATRP (arrival time reduction percentage)來當作衡量的指標，以機制 A 當成基準來衡量其他機制在不同的交通環境、路徑、交通號誌燈的周期下效能的表現如何。ATRP 的值越大代表這個機制的效能越好，關於 ATRP 的定義如下。

假設在機制 A 底下，緊急救護車輛到達事故現場所花費的時間為 X 秒，在機制 B、C、D 底下需要花費 Y 秒，則 ATRP 的值為：

$$\text{ATRP}(\%) = \frac{X-Y}{X} * 100\%$$

為了方便，表 13 為這篇論文中所使用的縮詞：

表 13.原句與縮詞的對照

原句	縮詞
Arrival Time Reduction Percentage	ATRP
Traffic Lights Period	TLP
Average Vehicle Separation Distance	AVSD

如下圖所示，ATRP 的結果顯示，在 TLP 是 30 秒的環境下，效能表現是機制 A < 機制 B < 機制 C < 機制 D，原因如下。

一般來說，機制 B、C、D 的效能比機制 A 好，這是因為疏散車輛的訊息傳遞以及動態交通號誌燈的控制這兩種方法都能有效的縮減緊急救護車輛到達事故現場的時間。而結果顯示機制 C 的效能表現比機制 B 要來的好，則代表動態交通號誌燈控制比疏散車輛的訊息傳遞更能疏散緊急救護車輛前方壅塞的車潮，因此效能表現比較優異，而機制 D 的效能最佳是因為，動態交通號誌燈的控制以及疏散車輛的訊息傳遞同時使用時，能有效的清除緊急救護車輛前方的壅塞。

如圖 24 圖 24 圖 26 中所示，當 AVSD 的值增加時，各個機制的效能卻降低了，這是因為當 AVSD 的值比較大時，代表緊急救護車輛前方的車輛數比較少，因此也比較少有車會阻擋緊急救護車輛的通行，因此在機制 A 底下，緊急救護車輛到達事故現場的時間就會比較少。可以比較圖 26 圖 28 圖 28，就能清楚看見當 AVSD 減少時，緊急救護車輛在機制 A 底下，到達事故現場的時間確實會增加，而機制 A 運作下的時間減少會導致機制 B、C、D 的 ATRP 值的下降。

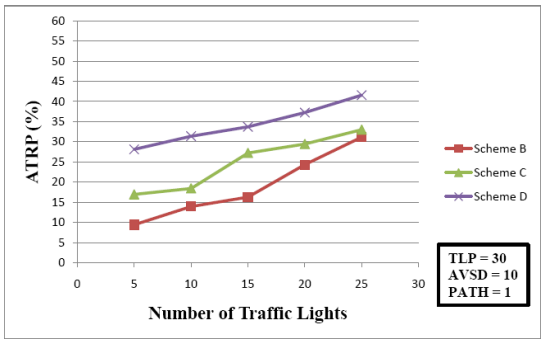


圖 24. AVSD=10 各個機制下的 ATRP

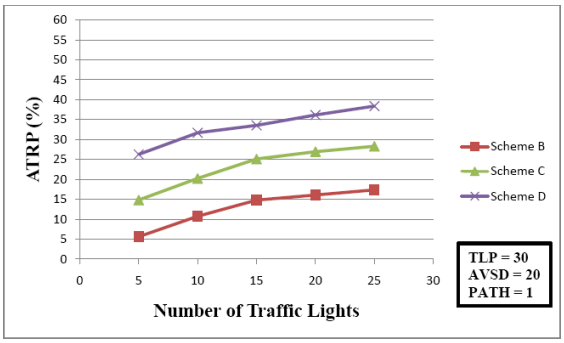


圖 25. ASVD=20 各個機制下的 ATRP

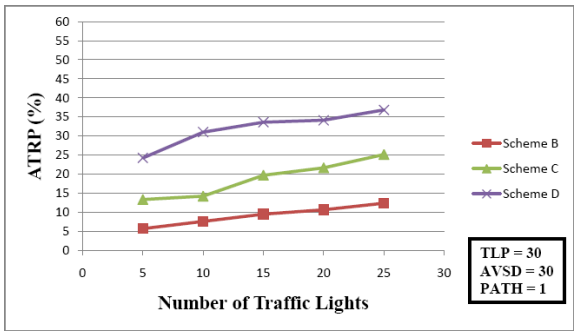


圖 26. AVSD=30 各個機制下的 ATRP

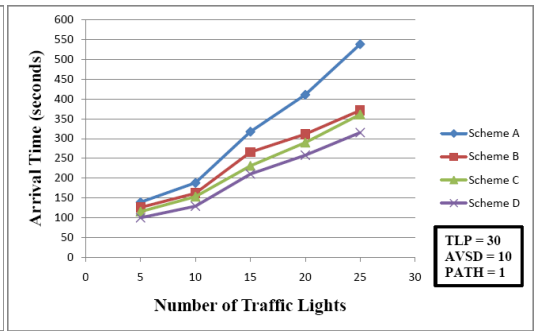


圖 27. AVSD=10 各個機制下的到達時間

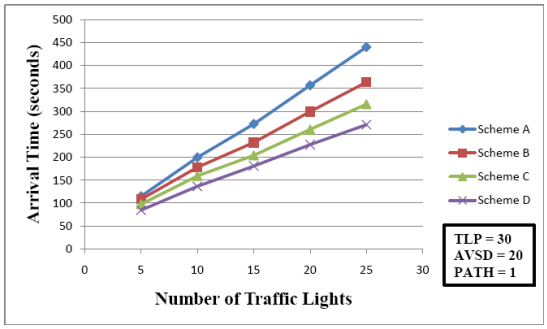


圖 28. AVSD=20 各個機制下的到達時間

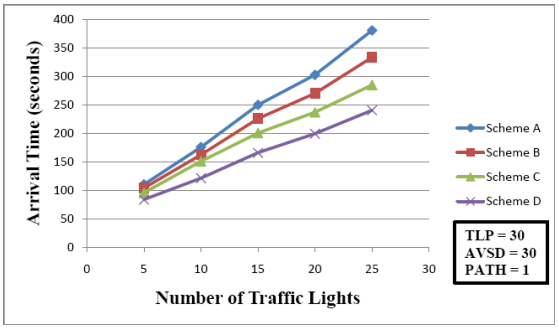


圖 29. AVSD=30 各個機制下的到達時間

比較圖 24 和圖 30，會發現機制 B、C、D 在路徑 2 的效能比在路徑 1 的效能好，這是因為路徑 2 的總長度比路徑 1 的總長度還要長，所以緊急救護車輛在機制 A 的運作下，在路徑 2 上會遭遇更多的車輛阻擋，因此會花費比較多的時間到達事故現場。由圖 26 和圖 30 中就可以觀察到上述說的情形，在機制 A 的運作下，在圖 30 中緊急救護車輛到達事故現場的時間比圖 267 中多，但是在機制 B、C、D 的運作下圖 30 中只比圖 24 中多了一點點的時間，所以在機制 B、C、D 運作下在路徑 2 的效能會比路徑 1 的效能好。

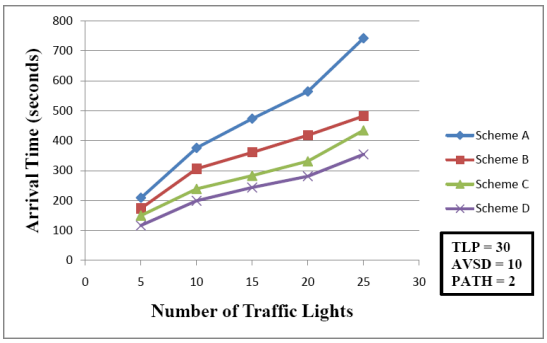


圖 30. 在路徑 2 各機制的到達時間

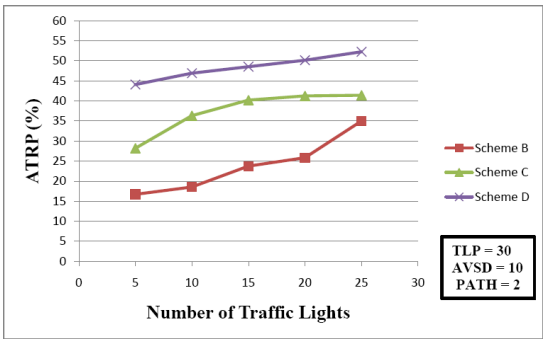


圖 31. 在路徑 2 各機制的 ATRP 值

當 TLP 增加時，因為綠燈時間變長，所以在機制 A 中緊急救護車輛可以一次通過更多的十字路口，因此可以比較快速的到達事故現場，而在機制 D 中，TLP 的增加幾乎不影響其效能，這是因為機制 D 能控制交通號誌燈而且能保留車道給緊急救護車輛。如圖所示，在機制 A 運作下緊急救護車輛到達事故現場的時間會隨著 TLP 增加而增加。同樣也可以看到在機制 D 的運作下，隨著 TLP 的增加緊急救護車輛到達事故現場的時間幾乎沒什麼變化。所以如圖所示，在機制 D 底下的 ATRP 會隨著 TLP 的增加而減少。

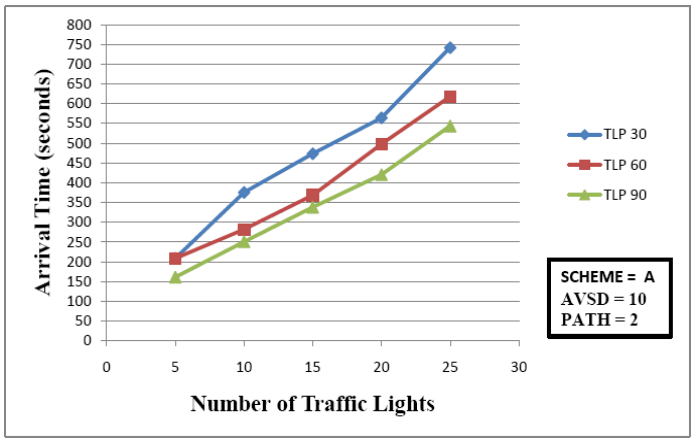


圖 32. 機制 A 在 TLP 變動的環境下的到達時間

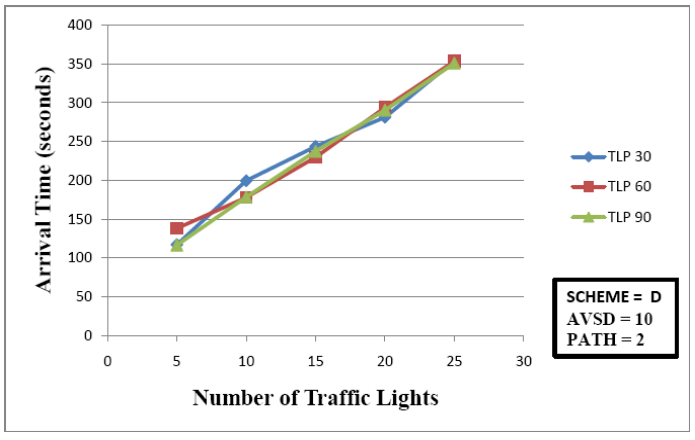


圖 33. 機制 D 在 TLP 變動的環境下的到達時間

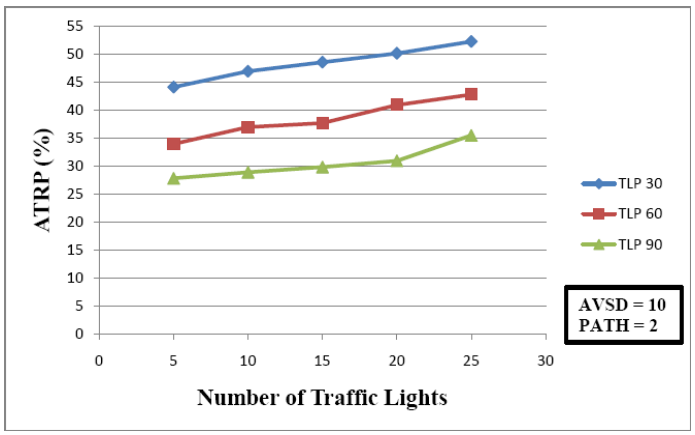


圖 34. 機制 D 在 TLP 變動的環境下的 ATRP 值

6.6 結論

在這篇論文中，我們提出的交通控制機制有三種主要的功能(1)移動路徑安排(2)疏散車輛的訊息傳遞(3)動態交通號誌燈的控制。

我們模擬的結果顯示，我們提出的交通控制機制能有效的減少緊急救護車輛到達事故現場的時間，而且動態交通號誌燈控制的效果比疏散車輛的訊息傳遞的效果還要好。另外在車輛密度較高的環境，我們提出的機制能更有效的減少緊急救護車輛到達事故現場的時間，這是因為，車輛密度高，則道路上會有更多不可預期的情形阻塞緊急救護車輛，而我們提出的機制能有效的排除這些問題讓緊急救護車輛更快速的到達事故現場。模擬的結果也顯示，當紅綠燈的周期變長時，我們提出的交通控制機制的效果會降低，這是因為綠燈的時間變長了，交通的狀況也會跟著改善，因此我們提出的交通控制機制的效果就沒這麼明顯了。

在這篇論文中顯示我們提出的交通控制機制能有效的減少 30%~50%的緊急救護車到達事故現場的時間，對於救護人命來說這是個非常有用且可行的機制。

7.研究成果自評

計畫第三年預期完成的工作項目條列如下：

(1)道路網路的模擬
(2)符合真實世界車流運動行為的車輛行為模擬

經過我們設計與實作，我們已經順利的把道路網路以及車流的運動行為順利開發完成，同時為了提升模擬的真實度，我們在道路網路方面增加了可以引進真實世界路網 shapefile 格式的功能，因此可以把現實世界中的地圖直接讀入模擬器中進行模擬。而在車流的運動行為，我們不只建構了車輛的前進與停止，更是實做了 Landmark-based 這個特別的機制，讓車流可以依照我們的需求運作，並且讓車流的行進更符合真實世界的運作。

在我們開發完預定的工作項目後，我們便在這個平台上著手進行研究，如同我們計畫當初所提到的，會進行紅綠燈的智慧型控制的研究，以及車輛間封包的傳遞以達到緊急救護功能的研究。

在紅綠燈的智慧型控制研究中，我們改良過的智慧型紅綠燈控制演算法的效能比起原本的智慧型控制演算法，在效能、公平性以及抗資訊流失的各方面都有更優良的表現。

而在緊急救護的議題中，我們提出的三個機制，經過交叉實驗比對後，我們發現綜合路徑規畫、疏散車輛訊息的傳遞以及動態交通號誌燈的控制，這三項功能的機制，能大幅縮短緊急救護車輛到達事故現場的時間，在分秒必爭的救護中，能大大提高救護性命的效率。

經過我們在這平台上的研究，已經證實這個平台的可用性，的確在道路網路以及封包網路的模擬中，都能提供研究者做各式各樣的研究。我們對於這計畫的研究成果完成度給予高度的評價。