臺灣二００六年國際科學展覽會

科　　別：電腦科學

作 品 名 稱：數位公車

得 獎 獎 項：佳作

學校 / 作者　：新竹市立光華國民中學　江盛浩

作者簡介：

1981 年生；就讀新竹市光華國中三年級數理資優班。小四開始學習程式設計，初始以 Microsoft Visual Basic 6.0 設計遊戲程式，之後經常利用程式設計驗証數學，對數學的學習幫助頗大。國一接觸 ROBOLAB(2.5.2 版)軟體，以原有的程式設計爲基礎，很快的能自如撰寫操控該軟體，因而延伸出以數位公車來探討改善未來交通的想法。

數位公車

目次

附件一 ：

【數位公車】結構說明（電腦樂高機器人控制系統 ROBOLAB)

附件一 ：

【數位公車】順路乘載程式碼（Microsoft Visual Basic 6.0)

英文摘要(Abstract)

Digital Buses

This study investigates the possibility that “Digital Buses” would actually be used in real life. In addition to the basic mathematics knowledge that I have acquired over the years, I have used “Microsoft Visual Basic” and “LEGO ROBOLAB SOFTWARE” to implement the idea as a program. It simulates the way in which a digital bus travels in a city with a rectangular grid. Various plans are tested to find the best paths for providing the most efficient, convenient and speedy transportation.

This study has not only shown that “Digital Buses” are sure to be used in a modern city when wireless communication networks has developed to a certain point, but also supplied a framework for future researchers who may wish to study the optimal way in which more than one digital buses could efficiently run in cities according to their population distributions and road arguments, in order to overcome the traffic problems from the current bus systems.

中文摘要

數位公車

本研究探討數位公車在人類未來生活中實際運行的可行性。本人以所學的數學知識進行公車路徑規劃，並透過「樂高機器人控制系統」以及 Microsoft Visual Basic 軟體程式之撰寫，在棋盤式城市區域中模擬公車行駛情境，靈活搭配各種方案找出最佳路徑，達到便利快捷的高運輸效能。

本研究顯示當現代化都市無線通訊網路發達到一定的程度時，數位公車的發展是可以預期的。本研究之結果可提供後續研究者繼續探討多台數位公車在實際都市中依人口分佈、道路狀況來規劃最佳行駛路徑，以解決現今機械公車無法克服之繁雜交通困境。

研究報告

壹、 研究動機：

小時候看「漢聲小百科」，提到未來人類的交通工具會是由電腦控制的機動公車。當開始接觸到直角平面座標、三角形的邊角關係、及商高定理等數學知識後，決定融合數學與電腦程式設計，來探討『數位公車』在現實生活行駛的可行性。

貳、 研究目的

一、 如何使電腦操控的『數位公車』找出最佳路線。

(一). 僅一人欲搭乘而公車上尚無乘客

1. 「兩點之間直線最近」公車取直線前進，轉彎點儘量最少
2. 公車如何判斷最佳載客路線

(二). 當公車接到一位以上的乘客要搭車的訊息

1. 以距離公車最近的站牌爲第一優先
2. 以先要求上車的乘客爲第一優先

二、 當二位以上的乘客先後分別在不同的站牌上搭乘時，數位公車如何找出最佳路線？

參、 理論探討

『數位公車』的行駛必須包含許多先進的數位配套措施，牽涉甚廣無法由個人一一驗証，所以僅針對『數位公車』行駛的路線、運輸效率以及公車和乘客間的互動深入探討。

首先假設『數位公車』的駕駛情境：

『數位公車』是行駛在一個路線呈棋盤狀；格局方正且無線通訊網路極爲發達的現代化都市。

設：『數位公車』一部

A～H 共八個固定數位站牌

A 在(0,0)，B 在(1,5)

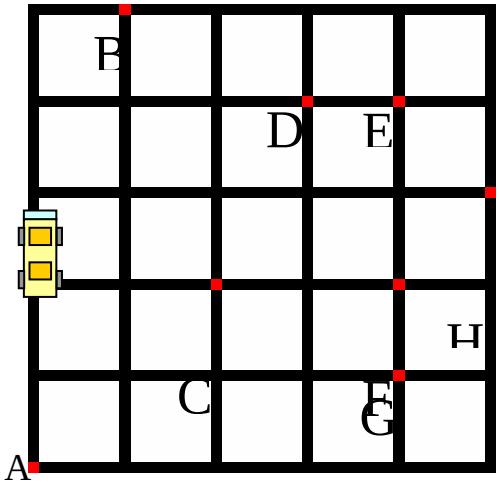
C 在(2,2)，D 在(3,4)

E 在(4,4)，F 在(4,2)

G 在(4,1)，H 在(5,3)

X 在(x,y)

設；兩個交叉點之間的距離爲 1 單位：



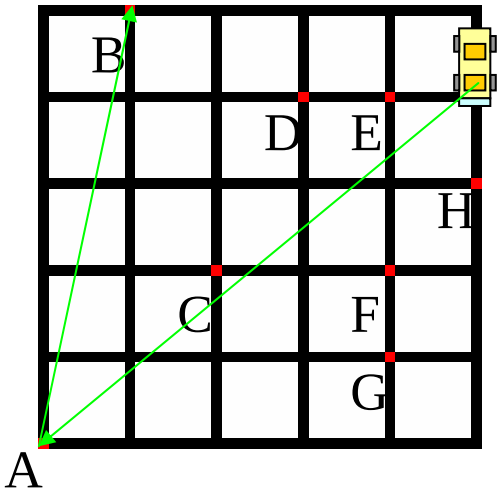
(圖一)

一、 如何使電腦操控的『數位公車』找出最佳路線。

(一). 僅一人欲搭乘而公車上尚無乘客

當乘客要由 A 到 B；而 X 點上的公車尚無乘客。公車行進路線；

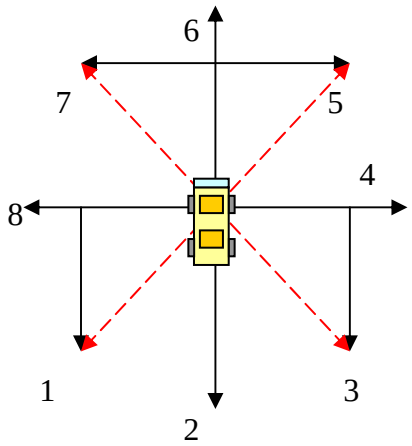
X ----→ A ----→ B。



（圖二）

1. **「兩點之間直線最近」公車取直線前進，轉彎點儘量最少**

最重要的物理原則：「兩點之間直線最近」，公車從 X 點到 A 應儘量取直線行進，讓轉彎點越少越省時間；運輸效率越高。以下以車頭爲準的八個方向探討



（圖三）

1：往左後方行駛，需二度左轉

2：往後方行駛，需原地迴轉再前進

3：往右後方行駛，需二度右轉

4：往右方行駛，只需右轉

5：要往右前方行駛，需前進再右轉

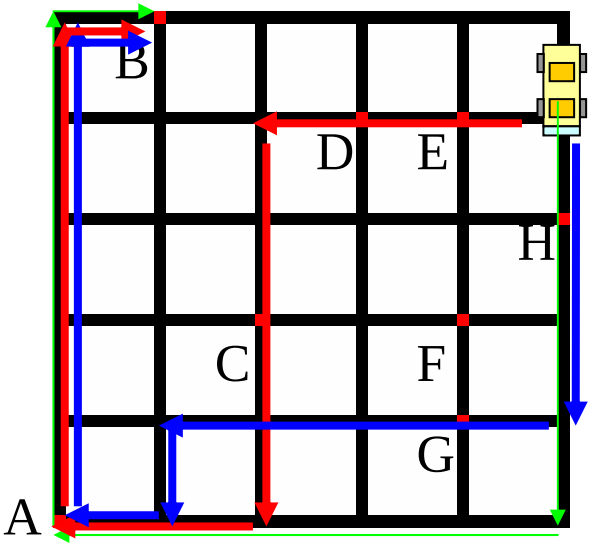
6：要往前方行駛，只需前進

7：要往左前方行駛，需前進再左轉

8：要往左方行駛，只需左轉

2. **公車如何判斷最佳載客路線**

當乘客要從 A 前往 B，最佳路線是； X ----→ A ----→ B 的走法：



（圖四）

自 X 點出發綠線轉了三個彎和行駛 15 單位，紅線或藍線同樣行駛 15 單位，卻分別轉了四個和五個彎。距離相等但行駛時間較綠線長，因此應以綠線爲最佳路線。

(二). 當公車接到一位以上的乘客要搭車的訊息

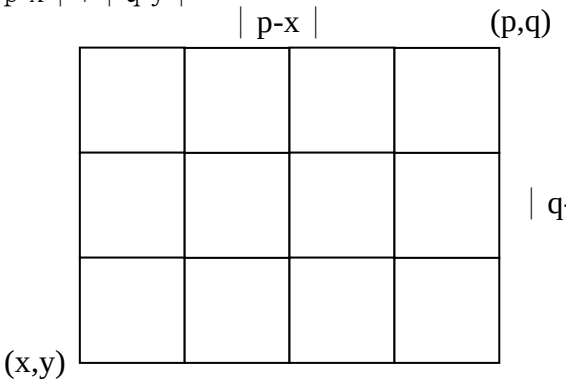
超過一個以上的乘客從不同的地方上下車時。乘載方式可從距離公車較近的乘客，和第一個要求上車的乘客爲先兩方面來考慮

1. 以距離公車最近的站牌爲第一優先

設 X=(x,y)　P=(p,q)

從 X 點開到 P 點，則 X 點到 P 點的距離是：

｜p-x｜+｜q-y｜



（圖五）

(1).各站牌與公車距離不等時：

A 在(0,0)，B 在(1,5) ，C 在(2,2)，D 在(3,4)

E 在(4,4)，F 在(4,2) ，G 在(4,1)，H 在(5,3)

X 在(2,3)

設 A、C、D、G、H 站牌都有乘客要上車，而公車在 X 點。

則 A 和 X 的距離 ｜0-2｜+｜0-3｜=5

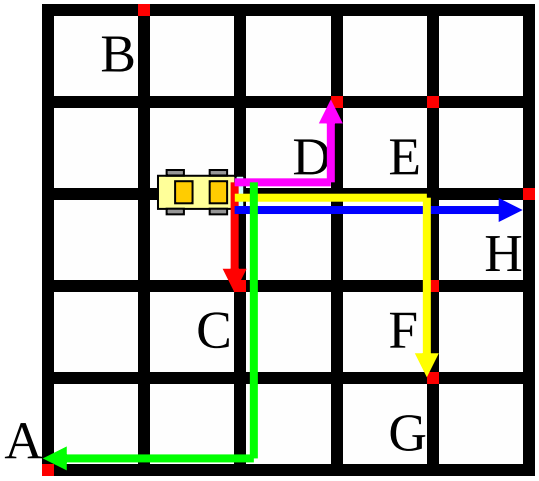
則 C 和 X 的距離 ｜2-2｜+｜2-3｜=1

則 D 和 X 的距離 ｜3-2｜+｜4-3｜=2

則 G 和 X 的距離 ｜4-2｜+｜1-3｜=4

則 H 和 X 的距離 ｜5-2｜+｜3-3｜=3

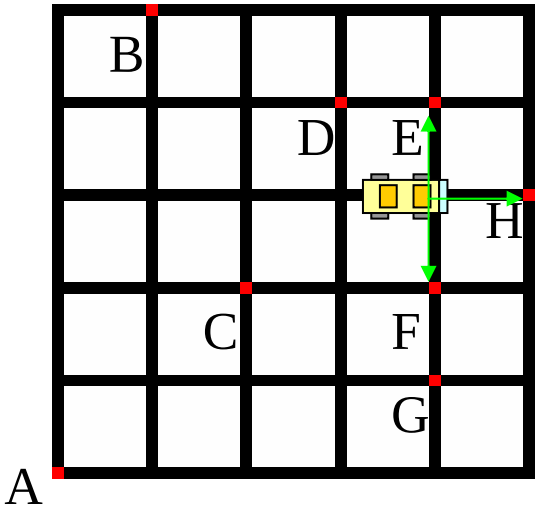
可知 C 距離最近，公車會依程式指示先右轉到 C 站牌載客。到達 C 點後則程式必須重新計算與剩下來的乘客間之最短距離。



（圖六）

(2). 兩個以上站牌與公車距離相等時

當 X 點和兩個以上站牌距離相等，就必須依車頭的方向決定優先順序。轉彎的時間也應計算在內。如圖七：E、F、H 三位乘客與公車的距離相等，因 H 處在正前方可直線前進，爲第一優先乘載。至 H 點後與 E 和 F 的距離仍然相等，則以先得到乘坐訊息者爲優先。



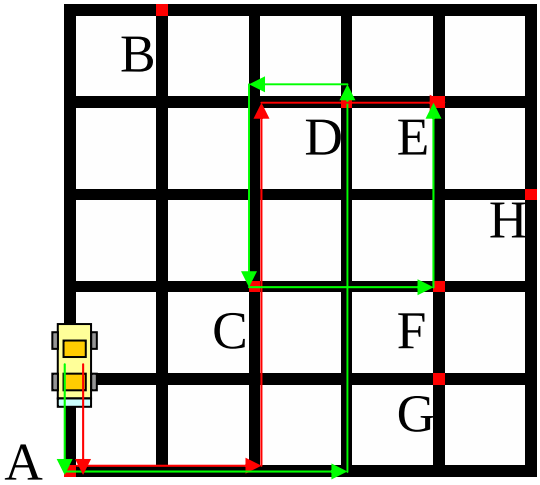
（圖七）

近距離乘載的優點是省時省力。缺點則是；設許多 D、E、H 點的乘客要到 F、G 點參加集會，因以距離最近的站牌爲第一優先，公車勢必在這附近來回移動。而此時在 A、B、C 的乘客在短時間內等不到車，因此該方式較適合在公車量多空車率高的地區。

2. **以先要求上車的乘客爲第一優先。**

該乘載的優點是按照乘坐訊息先後載客，每一個人都可以搭上車。缺點則是；例：

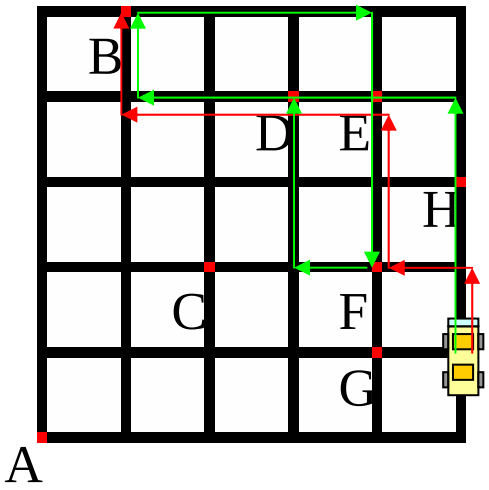
(1) 第一位乘客從 A 到 D，第二位乘客從 C 到 E。公車可以從 A 到 C 到 D 再到 E。行駛路線共 15 單位和 6 個彎（綠線），順路乘載卻只走 9 單位和 3 個彎（紅線）。



（圖八）

	行徑路線	X	→	A	→	D	→	C	→	E	合計
→	行徑距離	1		7		3		4			15
	轉彎次數	0		2		2		2			6
	行徑路線	X	→	A	→	C	→	D	→	E	合計
→	行徑距離	1		4		3		1			9
	轉彎次數	0		2		1		0			3

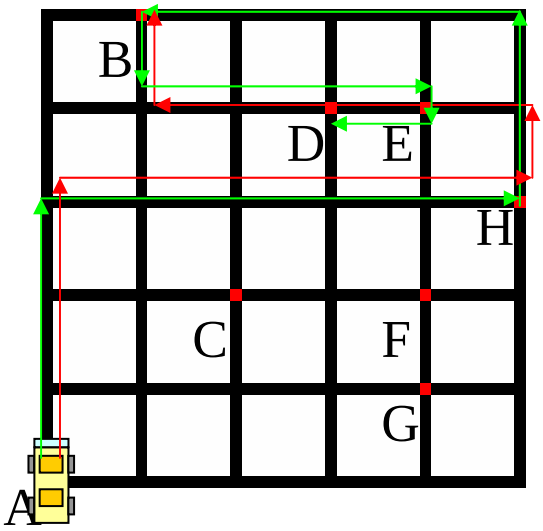
(2).第一位乘客從 E 到 B，第二位乘客從 F 到 D。公車從 F 到 E 到 D 再到 B。行駛路線共 17 單位和 6 個彎（綠線），順路乘載卻只走 8 單位和 4 個彎（紅線）。



（圖九）

	行徑路線	X	→	E	→	B	→	F	→	D	合計
→	行徑距離	4		4		6		3			17
	轉彎次數	1		1		2		2			6
	行徑路線	X	→	F	→	E	→	D	→	B	合計
→	行徑距離	2		2		1		3			8
	轉彎次數	1		1		1		1			4

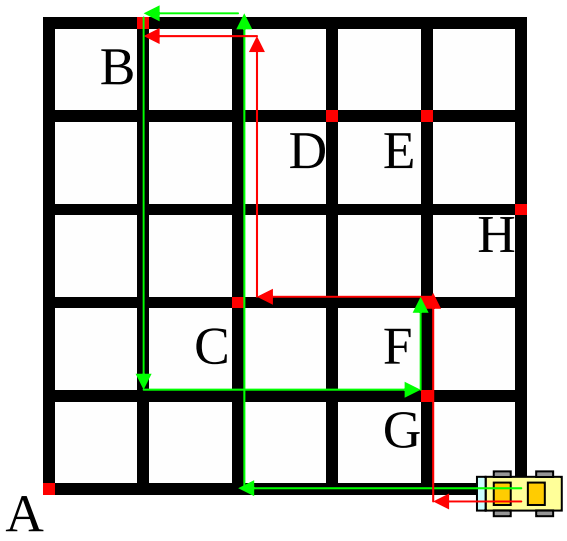
(3). 第一位乘客從 H 到 B，第二位乘客從 E 到 D。公車從 H 到 E 到 D 再到 B。行駛路線共 19 單位和 5 個彎、1 個迴轉（綠線）。順路乘載的走法卻只走 14 單位和 4 個彎（紅線）。



（圖十）

→	行徑路線	X	→	H	→	B	→	E	→	D	合計
	行徑距離	8		6		4		2			20
	轉彎次數	1		1		2		2			6
→	行徑路線	X	→	H	→	E	→	D	→	B	合計
	行徑距離	8		2		1		3			14
	轉彎次數	1		2		0		1			4

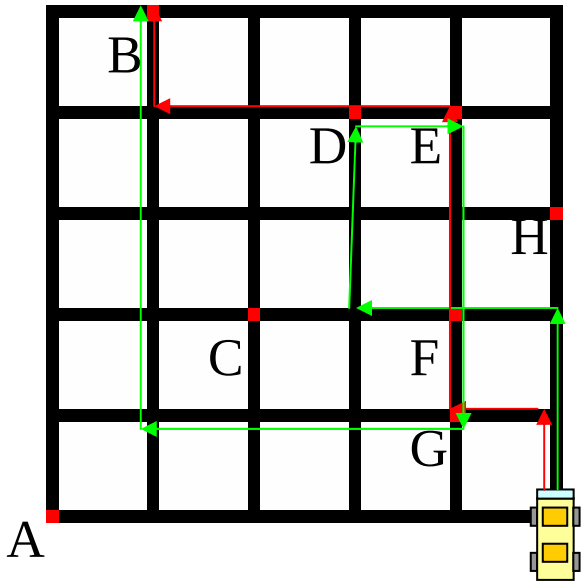
(4). 第一位乘客從 C 到 B，第二位乘客從 G 到 F。公車從 G 到 F 到 C 再到 B。行駛路線共 17 單位和 5 個彎（綠線），順路乘載卻只走 9 單位和 4 個彎（紅線）。



（圖十一）

→	行徑路線	X	→	C	→	B	→	G	→	F	合計
	行徑距離	5		4		7		1			17
	轉彎次數	1		1		2		1			5
→	行徑路線	X	→	G	→	F	→	C	→	B	合計
	行徑距離	2		1		2		4			9
	轉彎次數	1		0		1		2			4

(5). 第一位乘客從 F 到 D，第二位乘客從 G 到 B。公車從 G 到 F 到 D 再到 B。行駛路線共 17 單位和 6 個彎（綠線），而順路乘載的走法只走 9 單位和 4 個彎（紅線）。



（圖十二）

→	行徑路線	X	→	F	→	D	→	G	→	B	合計
	行徑距離	3		3		4		7			17
	轉彎次數	1		1		2		2			6
→	行徑路線	X	→	G	→	F	→	D	→	B	合計
	行徑距離	2		1		3		3			9
	轉彎次數	1		1		1		1			4

因此可以發現以「先要求上車的乘客爲第一優先」的乘載方式費時耗力，爲力求改善並提高運輸效率，則需另外考慮「順路的乘載」的配合方案。

二、 當二位以上的乘客先後分別在不同的點上搭乘時，數位公車如何找出最佳路線？

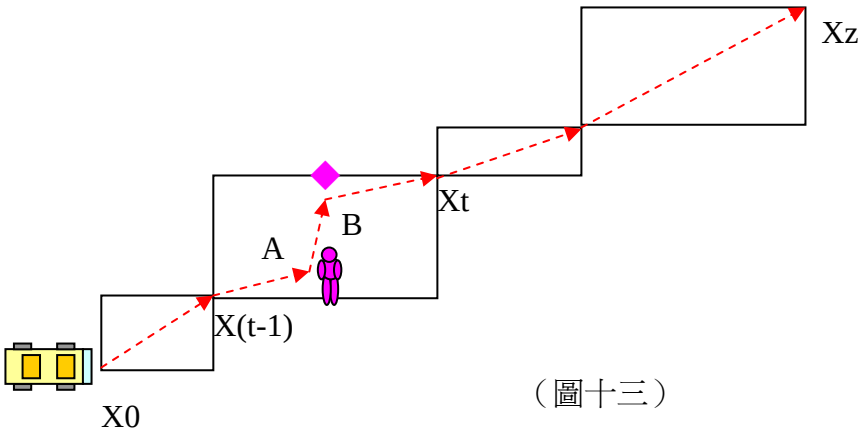
(一). 順路乘載儘量以不增加原本路線長度爲考量

設車子欲從 X0 出發至 X1、X2、…、Xz。途中接到另有他人要搭車的訊息時，順路乘載的最佳路線應以不增加 X1→X2→…→Xz 的路線長度爲最佳，路線越短運輸效率愈高。

例如：車子原本載客的路線是 X0→X1→X2→…→Xz，若另有一個人要從 A→B，產生的路線有以下這幾種：

1、乘客上下車地點介於 X(t-1)和 Xt 之間。

順路乘載路線爲：X0→…→X(t-1)→A→B→Xt→…→Xz



（圖十三）

順路乘載具備的條件

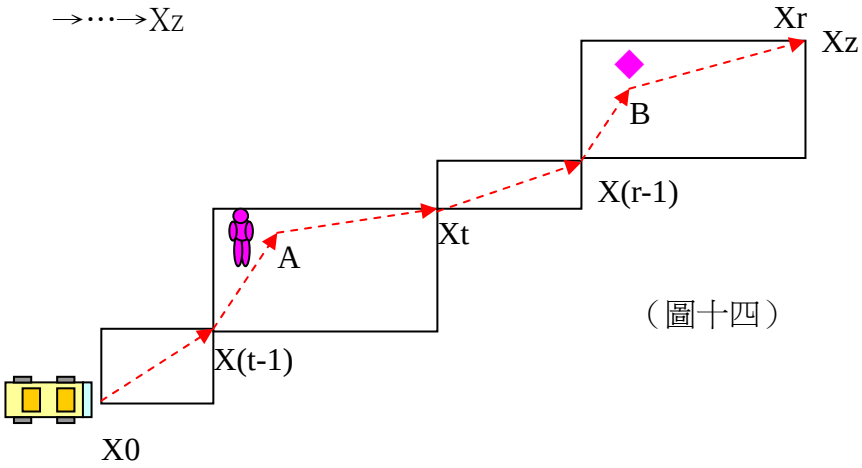
1.A 介於 X(t-1)和 Xt 之間。

2.A→B 的方向和 A→Xt 的方向爲同一方向。

3.B 介於 A 和 Xt 之間。

2、乘客上車地點介於 X(t-1)和 Xt 之間，下車地點介於 X(r-1)和 Xr 之間。

順路乘載路線爲：X0→…→X(t-1)→A→Xt→…→X(r-1)→B→Xr→…→Xz



（圖十四）

順路乘載具備的條件

1.A 介於 X(t-1)和 Xt 之間。

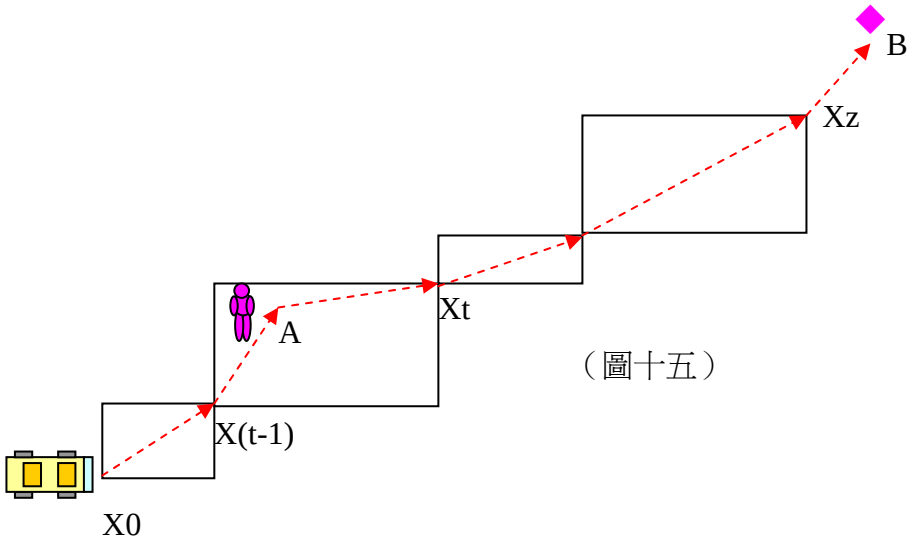
2.A→B 的方向和 A→Xt 的方向爲同一方向。

3.X(r-1)→Xr 的方向和 A→B 的方向爲同一方向。

4.B 介於 X(r-1) 和 Xr 之間。

3、乘客上車地點介於 X(t-1)和 Xt 之間，下車地點在 Xz 後。

順路乘載路線爲：X0→…→X(t-1)→A→Xt→…→Xz→B



（圖十五）

順路乘載具備的條件

1.A 介於 X(t-1)和 Xt 之間。

2.A→B 和 A→Xt 爲同一方向。

3.X(r-1)→Xr 和 A→B 爲同一方向。

4. X(r-1)→Xr 和 Xr→B 爲同一方向。

5. Xr→X(r+1)和 X(r+1)→B 爲同一方向。

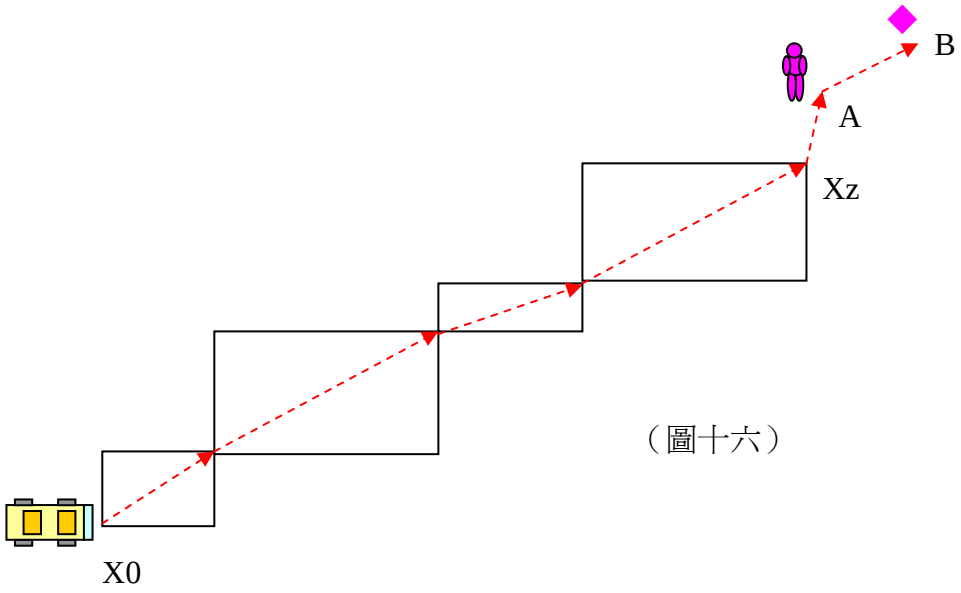
：

：

n.X(z-1)→Xz 和 Xz→B 爲同一方向。

4、乘客上下車地點在 Xz 之後

順路乘載路線爲：X0→…→Xz→A→B



（圖十六）

乘客上下車地點在 Xz 之後；則不具備順路乘載的條件，因爲上下車地點不在 X0 和 Xz 之間，該乘客乘車條件與第一位上車的乘客相同，所以不做順路乘載的考慮。

肆、 研究器材及裝置

一、Microsoft Visual Basic 6.0 程式軟體

二、樂高機器人控制系統 ROBOLAB(2.5.2 版)軟體

三、可程式控制的馬達、光源感應器。

四、各式樂高積木一大箱

五、製作路線圖用黑色電膠布一捲

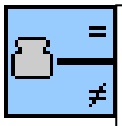
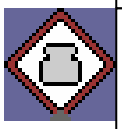
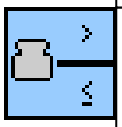
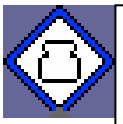
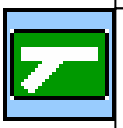
六、135cm × 135cm 路線圖四張

伍、 實作過程：

上述論述以樂高機器人控制系統 ROBOLAB 實際操作

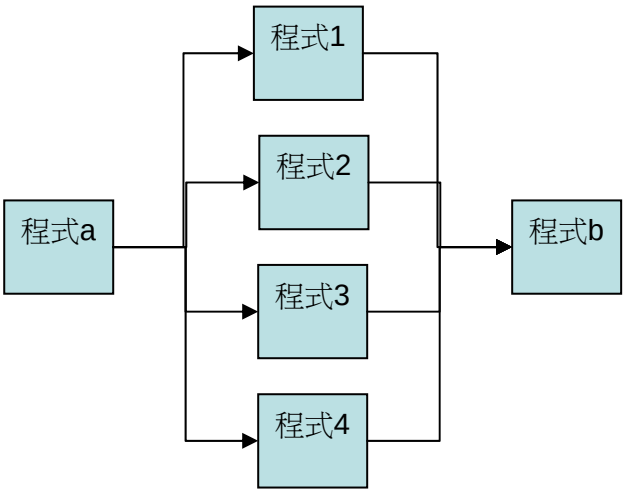
一、 公車應取直線前進，讓轉彎點儘量最少

(一)、圖例說明：

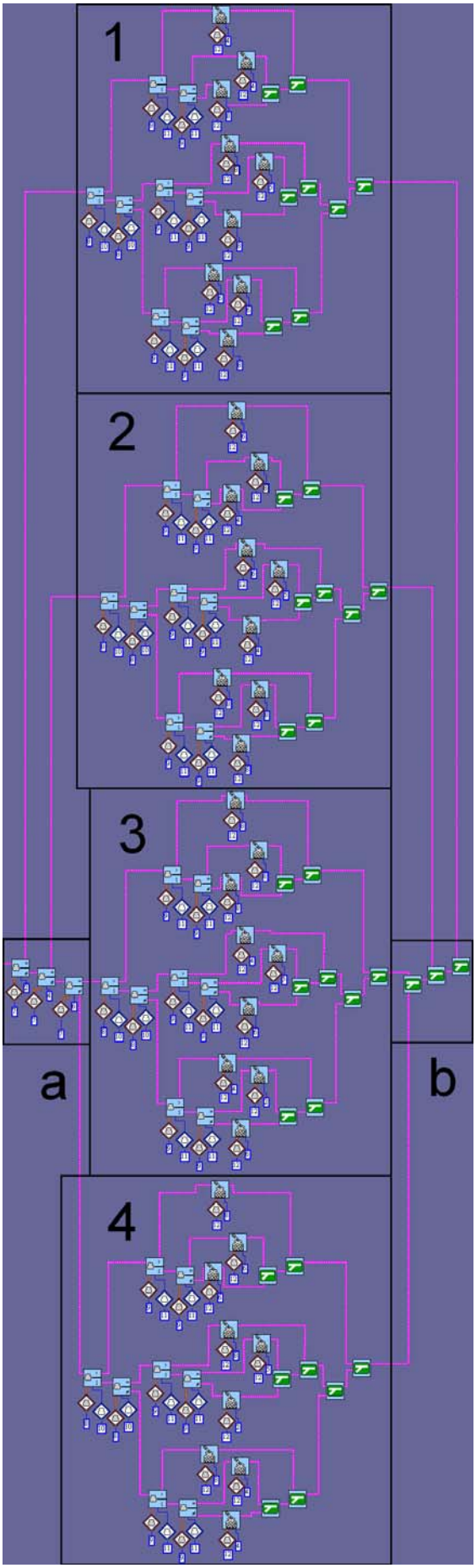
	計數器分叉 A：計數器等於指定數值時，程式走上方路線；不等於指定數值時，程式走下方路線。
	x 號計數器：代表這是 x 號的計數器。
	計數器分叉 B：計數器大於指定數值時，程式走上方路線；小於等於指定數值時，程式走下方路線。
	x 號計數器數值：指 x 號計數器內的數值。
	填滿計數器：設定計數器爲某一指定數值。
	分叉整合：整合分叉指令。所有分叉指令，均需配合一個分叉整合。
	數值常數：設定等待時間、或是與感應器有關的數值。在此爲計數器編號、或一數值。

(二)、程式路線說明：

以〔樂高機器人控制系統〕設計〔數位公車〕程式控制行駛各方向全圖。爲便於詳細說明，把整張劃分成〔a 圖〕、〔b 圖〕、〔1 圖〕、〔2 圖〕、〔3 圖〕、〔4 圖〕共六張圖。

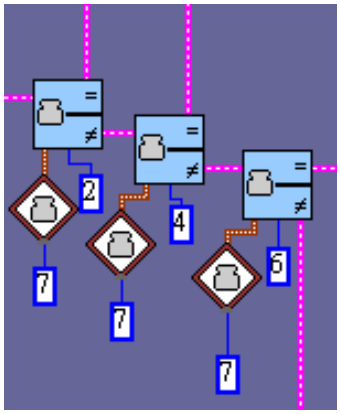


程式控制行駛各方向樹狀圖



1、【a 圖】:

設定〔7 號計數器〕內的值代表公車的方向是朝↓、←、→、↑行駛。(數值 2 代表方向是朝↓、數值 4 代表方向是朝←、數值 6 代表方向是朝→、數值 8 代表方向是朝↑)。



(1)、如果〔7 號計數器〕內的值爲 2，則代表公車方向朝↓方向行駛（程式控制公車往 1 號圖）。

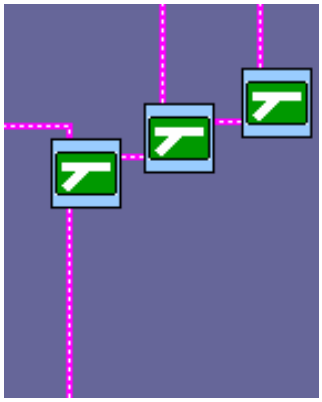
(2)、如果〔7 號計數器〕內的值不爲 2、而爲 4，則代表公車方向朝←方向行駛（程式控制公車往 2 號圖）。

(3)、如果〔7 號計數器〕內的值不爲 2、不爲 4、而爲 6，則代表公車方向朝→方向行駛（程式控制公車往 3 號圖）。

(4)、如果〔7 號計數器〕內的值不爲 2、不爲 4、不爲 6，則 7 號計數器內的值一定是 8，代表公車方向朝↑方向行駛（程式控制公車往 4 號圖）。

2、【b 圖】:

設定該圖三個分叉整合指令，所有分叉指令均需配合一個〔分叉整合〕。因此【b 圖】是整合由【a 圖】分叉出來的【1 圖】【2 圖】【3 圖】【4 圖】。



3、【1 圖】

(1)、〔8 號計數器〕、〔9 號計數器〕內的值分別代表目的地的 x 軸和 y 軸

(2)、〔10 號計數器〕、〔11 號計數器〕內的值分別代表公車位置的 x 軸和 y 軸

(3)、〔12 號計數器〕內的值代表公車應該迴轉、左轉、右轉、前進(數值 2 代表迴轉、數值 4 代表左轉、數值 6 代表右轉、數值 8 代表前進)

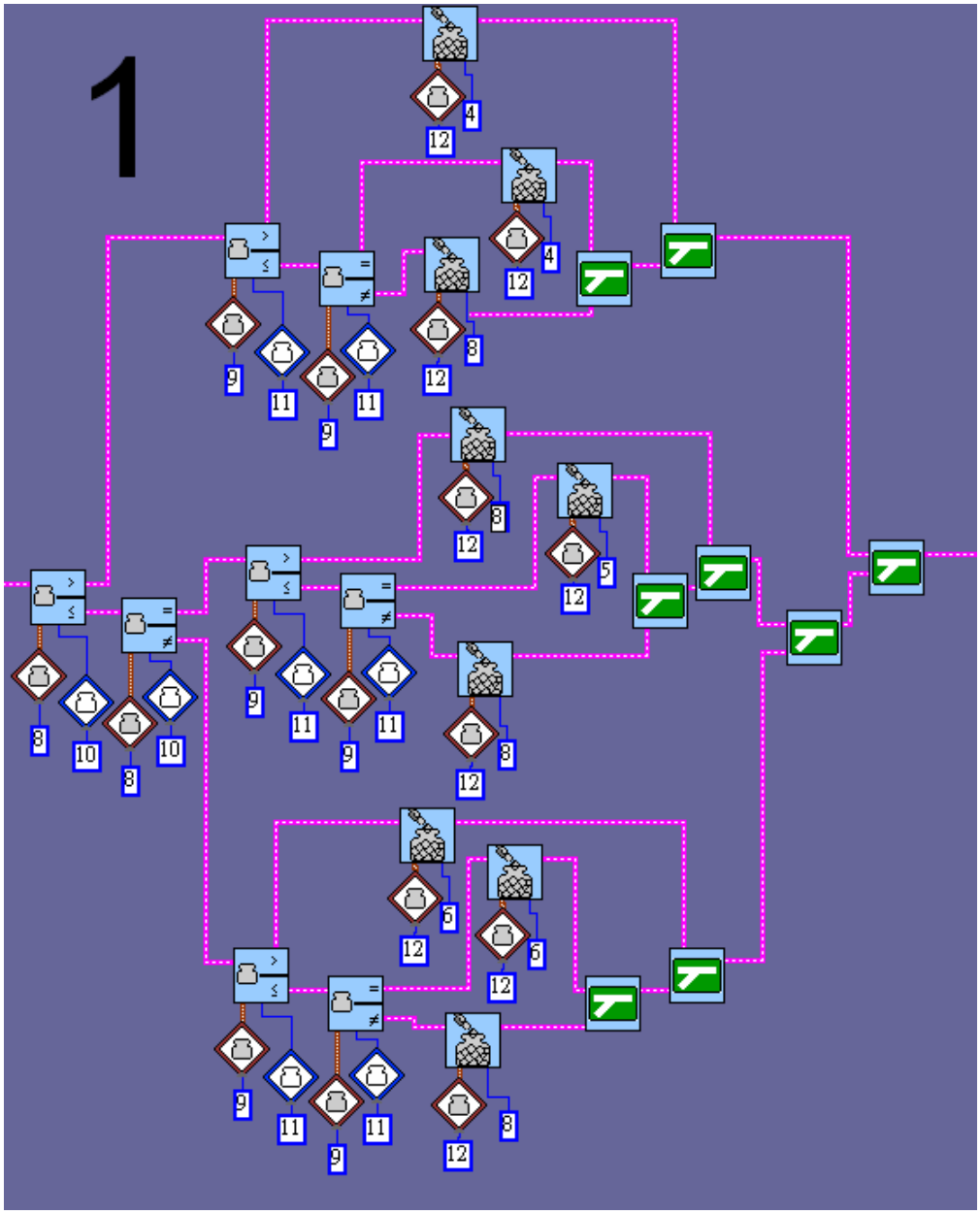
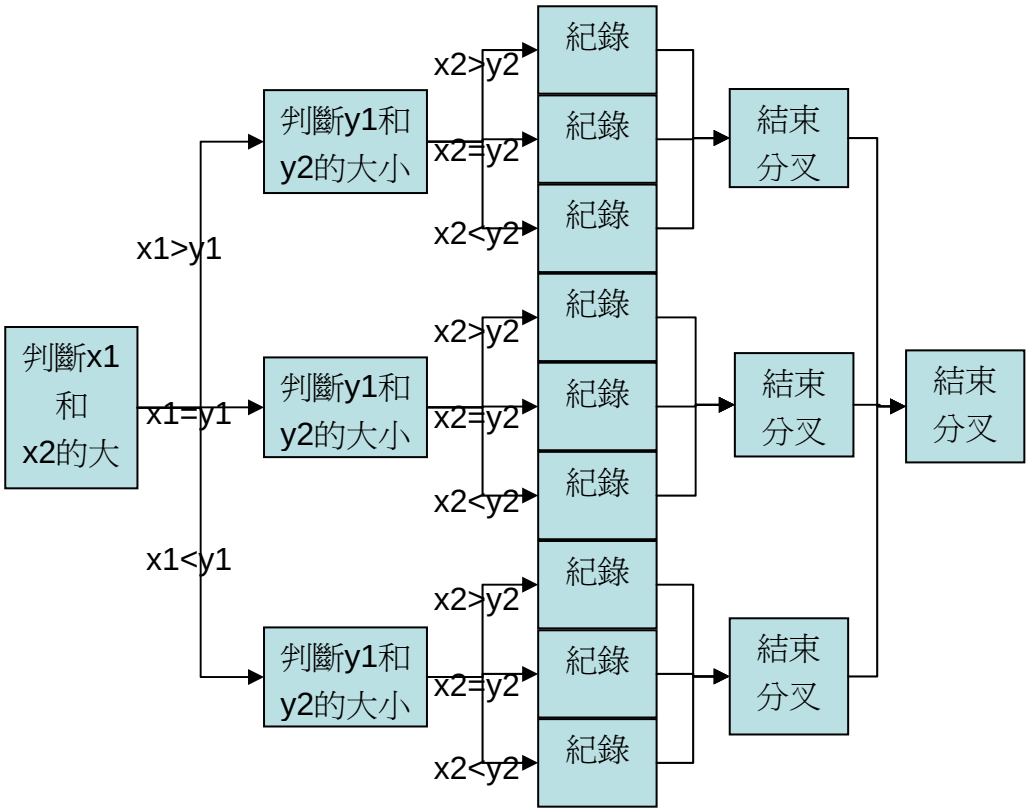


圖1、2、3、4樹狀圖



(4)、〔計數器分叉 B〕將 x 軸程式路線分成兩個部分，上方路線爲(〔8 號計數器〕內的值大於〔10 號計數器〕內的值)；下方路線爲〔8 號計數器〕內的值小於等於〔10 號計數器〕內的值。〔計數器分叉 A〕又將下方路線分成兩個部分。分別是；

(5)、〔計數器分叉 B〕分出上、下方路線後的結果：

A、〔8 號計數器〕內的值大於〔10 號計數器〕內的值。

B、〔8 號計數器〕內的值等於、小於〔10 號計數器〕內的值。

(6)〔計數器分叉 A〕再將第 (2) 項分出等於、小於路線後的結果：

A、〔8 號計數器〕內的值大於〔10 號計數器〕內的值。

B、〔8 號計數器〕內的值等於〔10 號計數器〕內的值。

C、〔8 號計數器〕內的值小於〔10 號計數器〕內的值。

(7)、即：假設公車的座標爲(x1,y1)，目的地的座標爲(x2,y2)，則：

A、爲 x2>x1

B、爲 x2=x1

C、爲 x2<x1

(8)、此三個部分叉各有兩個計數器分叉用前次方法將 y 軸程式路線再分成三個部分，分別是：

A、〔計數器分叉 B〕分出上、下方路線後的結果：

(A)、〔9 號計數器〕內的值大於〔11 號計數器〕內的值。

(B)、〔9 號計數器〕內的值等於、小於〔11 號計數器〕內的值。

B、〔計數器分叉 A〕再將第 (2) 項分出等於、小於路線後的結果：

(A)、〔9 號計數器〕內的值大於〔11 號計數器〕內的值。

(B)、〔9 號計數器〕內的值等於〔11 號計數器〕內的值。

(C)、〔9 號計數器〕內的值小於〔11 號計數器〕內的值。

(9)、即：假設公車的座標爲(x1,y1)，目的地的座標爲(x2,y2)，則：

A、爲 y2>y1

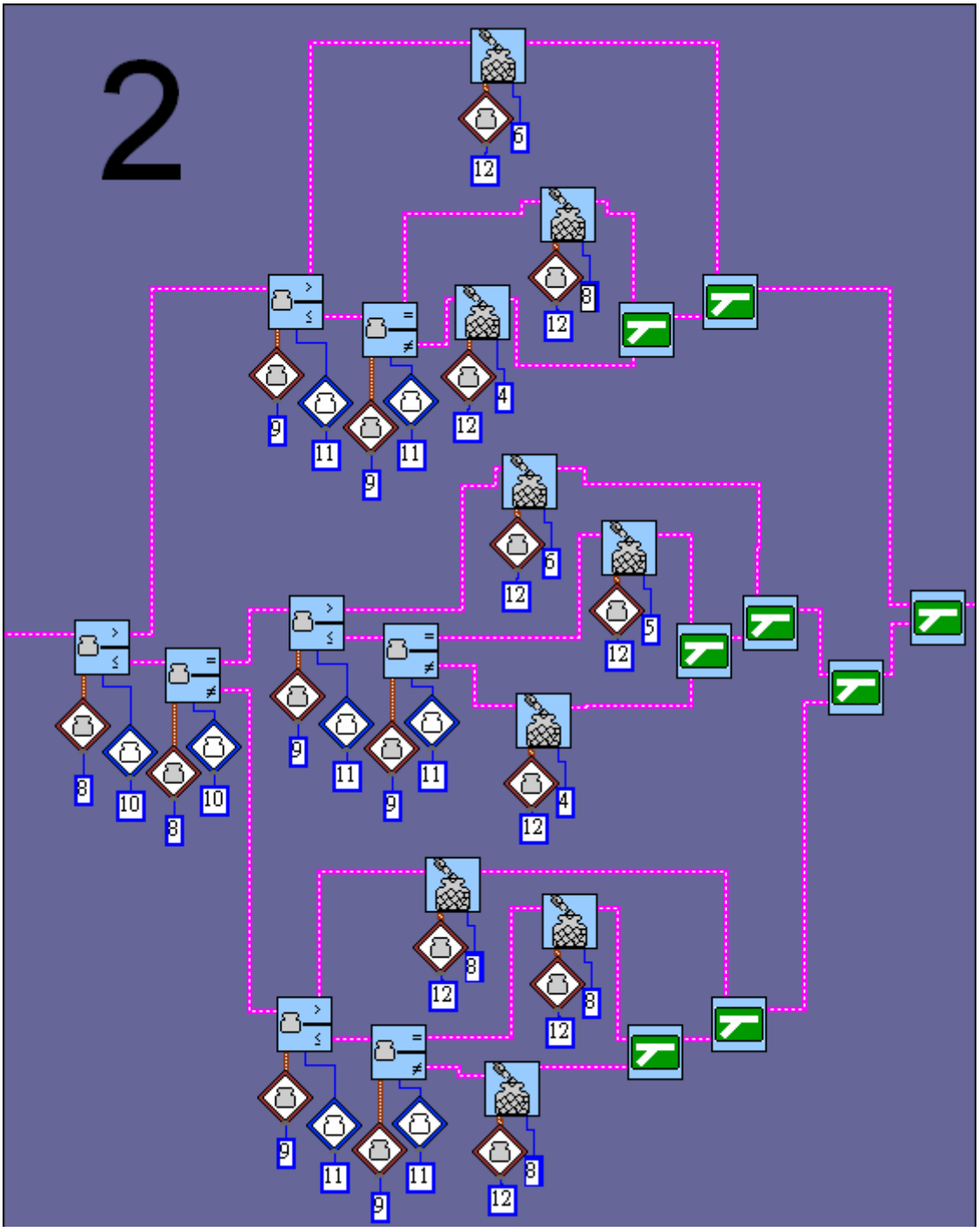
B、爲 y2=y1

C、爲 y2<y1

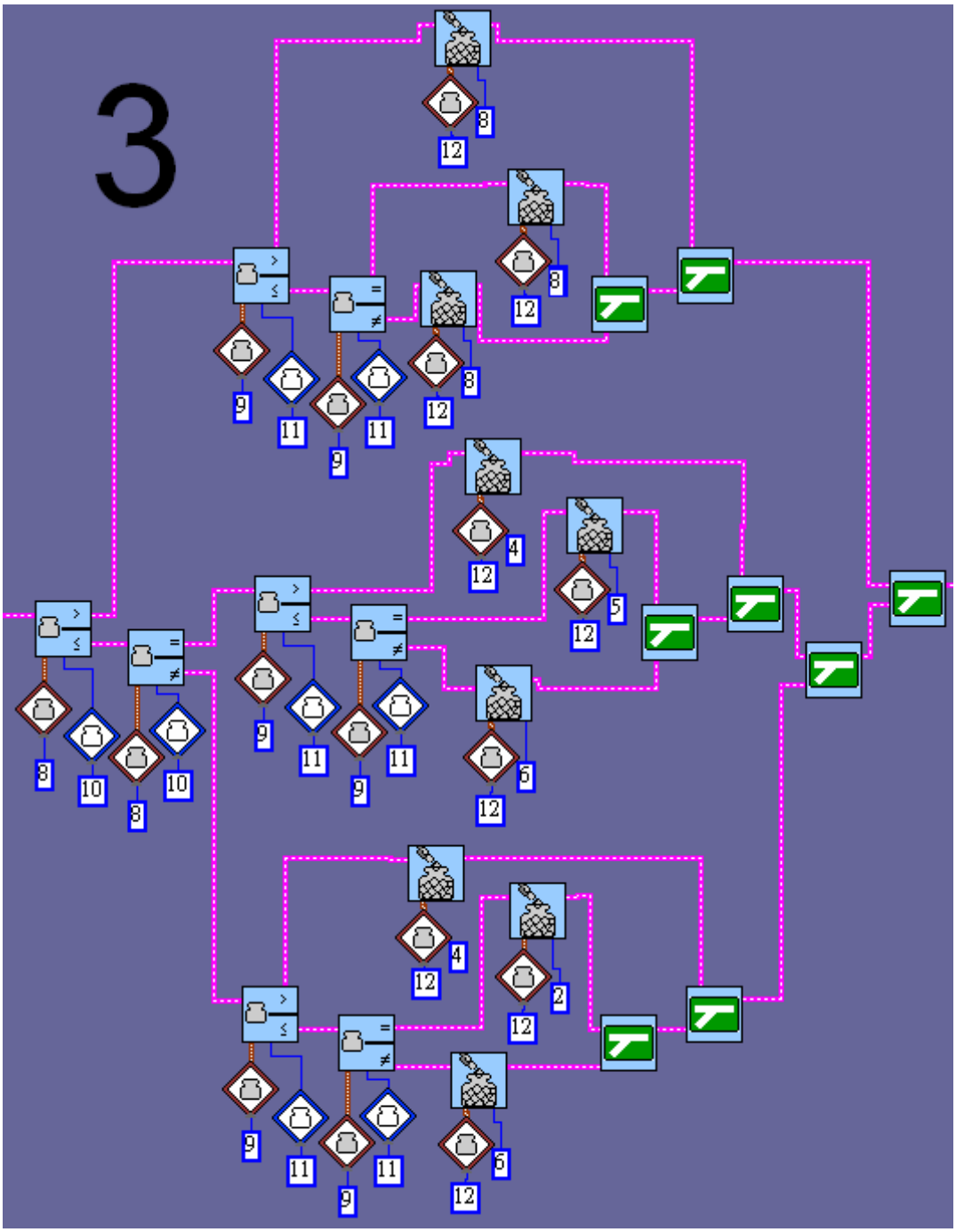
(10)、經過〔計數器分叉 B〕、〔計數器分叉 A〕兩段細分後，程式路線共分成九個部分，分別是：目的地的位置在公車的↑方、↓方、←方、→方、↖方、↗方、↙方、↘方和公車已經處在目的地(正下方)，再依照前述方法將迴轉、左轉、右轉、前進紀錄到〔12 號計數器〕內。

4、以下所列【2 圖】、【3 圖】、【4 圖】之程式紀錄說明與【1 圖】所述相同，在此只呈現程式碼作圖，就不再重複敘述。

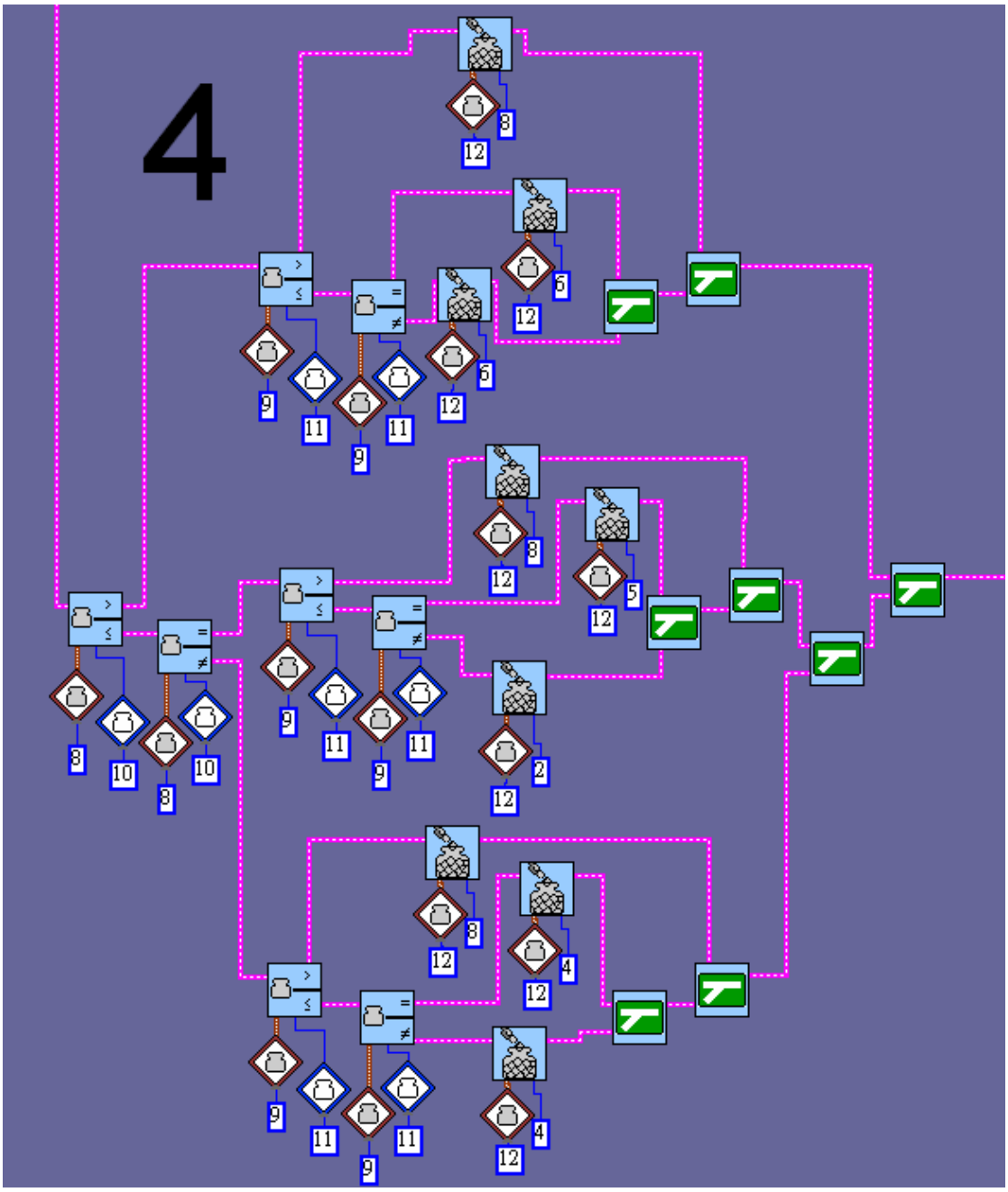
【2 圖】



【3 圖】



【4 圖】



二、 順路乘載的最佳路線

〔數位公車〕判斷順路乘載的最佳路線，經由樂高機器人控制系統 ROBOLAB 實際操作，發現一個 RCX 只有三個輸入端無法控制大量的按鍵。當三個輸入端都接上，設定按下後爲 1；不按爲 0，則三個觸控感應器最多只能控制 8-1 共 7 個車站。

爲了解決這個問題我設計了一個 16 個角度的感應器，用途分別負責：公車要由 A 地出發、公車要到 B 地和確定等。先將兩個輸入端接上角度感應器，讓每一個角度都代表一個車站。其中一個角度感應器代表某一個人要從某站出發，另一個代表從某站下車，剩下的輸入端加上觸控感應器代表確定。

但是實驗的結果發現這個角度感應器的功能還是太少，所以我放棄這一部份，另外設法融入 Microsoft Visual Basic 6.0 程式軟體續寫。

以 Microsoft Visual Basic 程式軟體模擬〔數位公車〕判斷順路乘載的可行性。

(一)、公車位置在 X0，要前往的車站依序是 X1、X2、X3、……、Xz。**順路乘載的程式書寫首先必須將 X0 到 Xz 中連續且重複出現的車站數去掉不計。即以總數 z 減掉連續且重複的車站數，以去除指令重複讓公車無法判斷而於原地動作耗時耗力後才會繼續進行下一個動作的困境。**

該部份在這份程式書寫中極爲重要，它要不斷的被考慮到，所以在下面的圖表敘述中我均以【※】符號代替。

1、順路乘載路程式碼見【附件二】，以下依 Microsoft Visual Basic 程式碼書寫方式依序說明：

(1)、設此時有一位乘客要從 a 站到 b 站，公車行進路線爲 X0、X1、X2、X3、……、Xz。我以 X0 代表公車的所在位置；X1、X2、X3……Xz，是代表公車原先行駛的站牌順序，即公車在接到搭乘訊息決定移動的位置。它視情況判斷則每次所走的路線都不盡相同。

(2)、設 z 爲公車要移動的站牌總數；t 爲判斷 a 的所在位置；r 爲判斷 b 的所在位置；s 是判斷 b 是否處在 Xz 的後面。

例如：z=0 爲目前無人搭車；公車並不需移動。

z=1 則公車要移動一個站牌；z=2 則公車要移動二個站牌；z=2 則……。

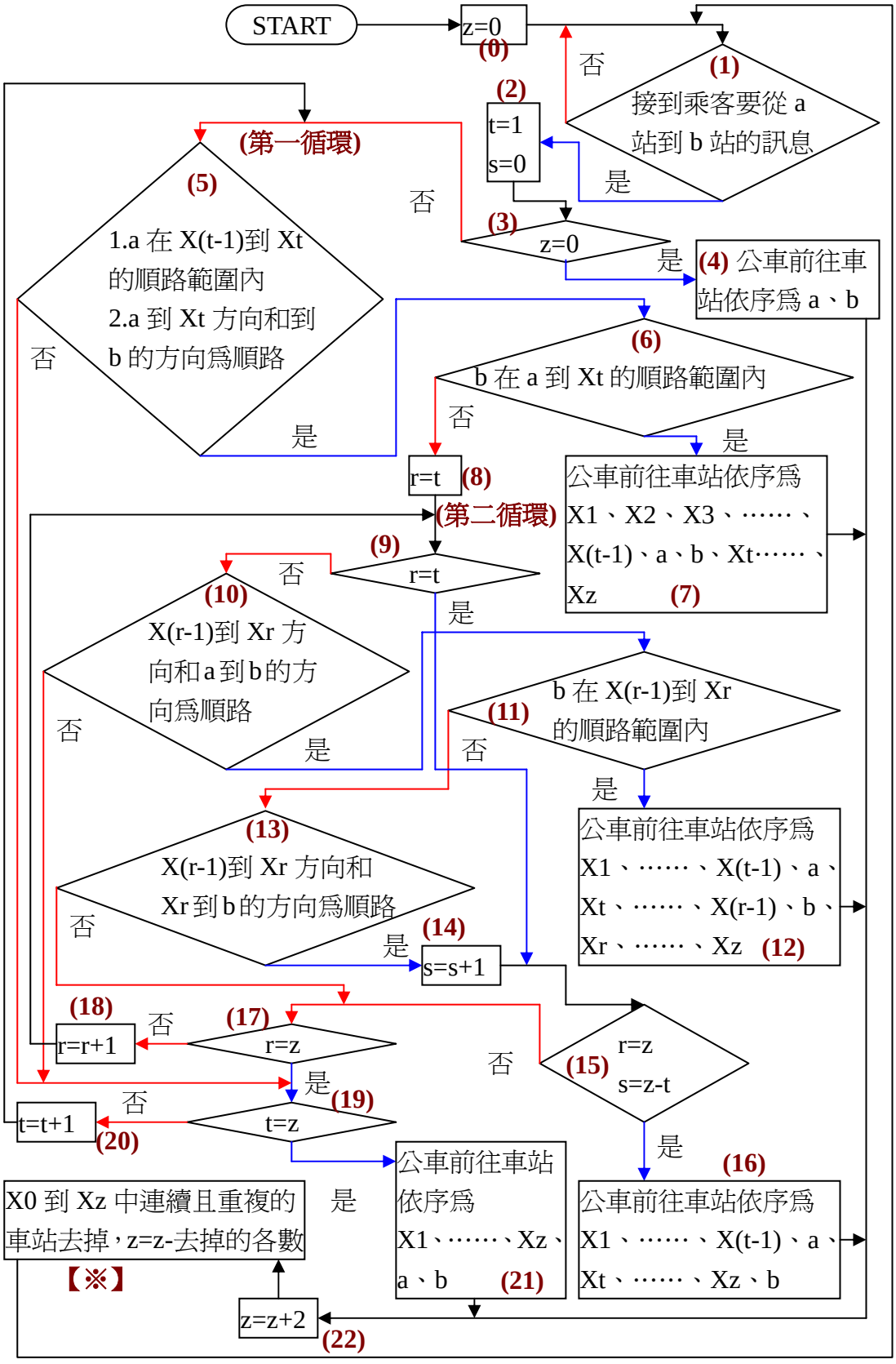
t =1 時；表示公車將判斷 a 是否在 X0 和 X1 之間，當 t =2 時是判斷 a 是否在 X1 和 X2 之間；t =3 ……

r =1 時；表示公車將判斷 b 是否在 X0 和 X1 之間，當 r =2 時是判斷 b 是否在 X1 和 X2 之間；t =3 ……

s =0 時 a 到 Xz 中的每一小段，沒有和 a 到 b 的方向是順路

s =1 時 a 到 Xz 中的每一小段，有一段和 a 到 b 的方向是順路

2、爲了詳細說明我根據程式碼整理出行駛動線圖：



(1) 判斷是否接到乘客要從 a 站到 b 站的訊息，是則進到(2)，否則回到(1)繼續等待。

(2) t=1 是因爲**第一循環** t 要從 1 到 z，所以 t 一開始是 1；s=0 是因爲(13)要判斷 a 到 Xz 中的每一小段都和 a 到 b 的方向是順路，因此 s 一開始是 0。

(3) 當 z=0；公車接到乘客要從 a 站到 b 站的訊息時前進到(4)，否則到(5) 。

(4) 公車路線是從 a 站至 b 站。此時公車已經找到所要進行的路線，所以不用繼續判斷，但必須考慮【※】這種情況以去除重複站牌的部分。
進入**第一循環**； t 從 1 到 z，爲了判斷 a 是否在 X(t-1)和 Xt 之間。

(5) 判斷 a 在 X(t-1)到 Xt 的順路範圍內；a 到 Xt 方向和到 b 的方向這兩個方向是否爲順路乘載。如果有一項是錯誤的，代表 a 不在 X(t-1)到 Xt 的順路範圍內或是 a 到 Xt 方向和到 b 的方向爲不順路，所以重新進入**第一循環**，但此時 t 必須加 1，因爲所要判斷位置已移動至下一個，所以再進入第一循環則 t=t+1。

(6) 判斷 b 是否在 a 到 Xt 的順路範圍內，因(5) 的判斷是正確的才能繼續判斷。如果接下來的判斷也是正確的，很明顯的可以看出 a 和 b 都順路，所以前進到(7)，否則到(8)。

(7) 公車前往車站依序爲 X1、X2、X3、……、X(t-1)、a、b、Xt……、Xz。此時公車已經找到所要進行的路線，所以不用繼續做判斷，但仍須考慮【※】的情況以去除重複站牌的部分。

(8) r=t 是因爲 b 在 a 的後面，而 t 是從 1 到 z，所以 r 是從 t 到 z。
進入**第二循環**；r 從 t 到 z，爲了判斷 b 是否在 X(r-1)和 Xr 之間。

(9) 因爲 r=t 時，是在判斷 b 是否在 X(t-1)和 Xt 之間，之前就已經判斷過了，所以跳到(15)直接判斷 X(r-1)到 Xr 方向和 Xr 到 b 的方向是否爲順路。

(10) 判斷 X(r-1)到 Xr 方向和 a 到 b 的方向是否爲順路，如果不順路，則結束**第二循環**。

(11) 判斷 b 是否在 X(r-1)到 Xr 的順路範圍內，如果是，則前進到(12)，否則到(13)。

(12) 公車前往車站依序爲 X1、……、X(t-1)、a、Xt、……、X(r-1)、b、Xr、……、Xz 此時公車已經找到所要進行的路線，所以不用繼續做判斷，但仍須考慮【※】的情況以去除重複站牌的部分。

(13) 除了前幾種順路的可能，另有一種；就是 a 在 X0 到 Xz 之間，b 在 Xz 後面，但是必須從 a 到 Xz 中的每一小段都和 a 到 b 的方向是順路。所以判斷 X(r-1)到 Xr 方向和 Xr 到 b 的方向是否順路，如果是則前進到

(14)，否則到(17)。

(14)s=s+1 是爲了要判斷(13)，只要有一個方向順路，s 就會加 1，再判斷(15)。

(15)當 r=z 時，s 又等於 z-t，代表 a 到 Xz 中的每一小段都和 a 到 b 的方向是順路，所以前進到(16)，否則到(17)。

(16)公車前往車站依序爲 X1、……、X(t-1)、a、Xt、……、Xz、b。此時公車已經找到所要進行的路線，所以不用再繼續做判斷，但此時仍必須考慮【※】去除重複站牌的部分。

(17)如果 r≠z，則重新進入**第二循環**，但重新進入循環時前進到(18)，否則到(19)。

(18)因爲所要判斷位置已移動至下一個，所以 r=r+1。

(19)如果 t≠z，則重新進入**第一循環**，但重新進入循環時前進到(20)，否則到(21)。

(20)因爲所要判斷位置已移動至下一個，所以 t=t+1。

(21)如果到最後還沒把 a、b 安排進去，代表 a、b 不順路，所以公車前往車站依序爲 X1、……、Xz、a、b 此時公車也已經找到所要進行的路線，所以不用繼續做判斷，但仍必須考慮【※】去除重複站牌的部分。

(22)因爲把 a、b 都安排進去了，所以 z=z+2。

陸、 結論

一、 爲達到省時省力的乘載方式，公車應取直線前進，讓轉彎點儘量最少。

二、 一位以上的乘客要搭車時的兩種判斷方法，一種是「以距離公車最近的站牌爲第一優先」；一種是「以先要求上車的乘客爲第一優先」。

三、 「距離公車最近的站牌爲第一優先」這種方法適合用在公車量多空車率高的地區。

四、 「以先要求上車的乘客爲第一優先」必須同時配合「順路乘載」方案，才能有效的達到便利快捷的高運輸效能。

五、 計算公車行進距離時，公車要從 X 到 P，X=(x,y)、P=(p,q)，X 點到 P 點的距離是｜p-x｜+｜q-y｜。

六、 順路乘載應以不增加原本路線長度爲考量，因此必需在限定的條件下方可實施。

七、 順路乘載的靈活運用，更能增進〔數位公車〕與乘客間的良好互動。

八、 僅以一部公車實驗探討如何找出便捷路徑，當實務操作時可考慮由多部公車行駛分別判斷、運用以上各方案，以達到高運輸的最佳效能。

九、 本報告僅針對〔數位公車〕在順境中行駛而加以研究。

十、 逆境中行駛；諸如停電、電腦操控疏失、通訊網路的承載不足…等導致〔數位公車〕拋錨以及車禍等突發狀況的應變部份，留待日後另案探討。

柒、 參考資料

（一）、漢聲小百科十二月份第 28 頁～去看未來的生活

（二）、北一女動力機械社～速食餐點系統

（三）、電腦樂高機器人控制系統 ROBOLAB 軟體使用手冊

附件一

【數位公車】結構說明

由〔電腦樂高機器人控制系統〕所製作的〔數位公車〕各部份構造詳見以下圖片：

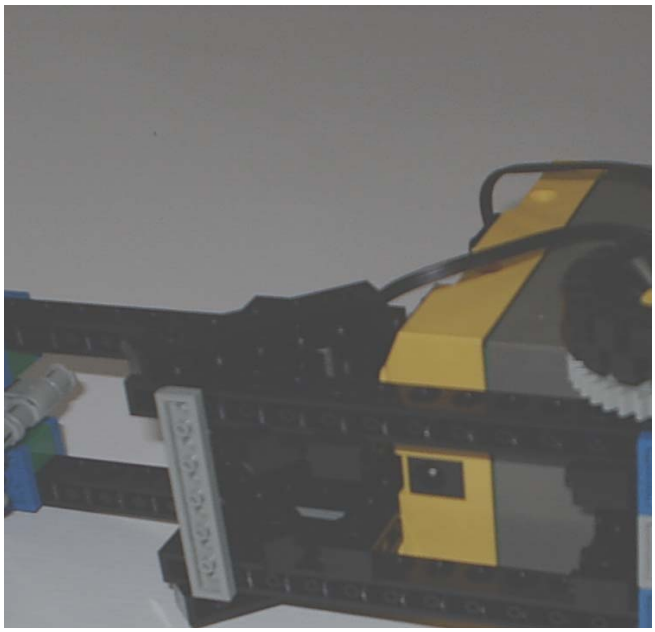
實驗用〔數位公車〕外貌

前輪裝置用小齒輪連接大齒輪，以減緩馬達的速度，可控制車子的速度。

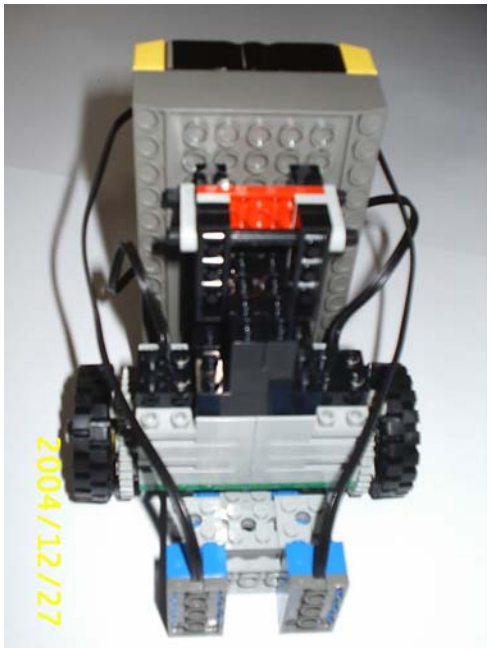
後輪可作 360 度旋轉，車子行進中轉彎時，不因角度改變所產生的摩擦力影響行進速度。

公車在運行中方向會改變，所以將 RCX 的紅外線接收端朝上，較容易接收到訊息。

該部份結構體的下方裝置的是馬達。當馬達的動力讓輪子轉動，而輪子本身和地面的摩擦力卻也很容易讓馬達彈起造成空轉，所以用此結構固定。

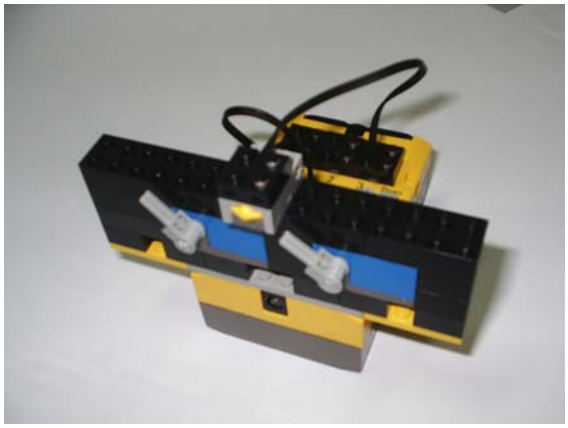


爲了克服車子在行進轉彎時，會和地面產生摩擦力，容易和車子的後半部分離，所以以這部份結構體加強固定。

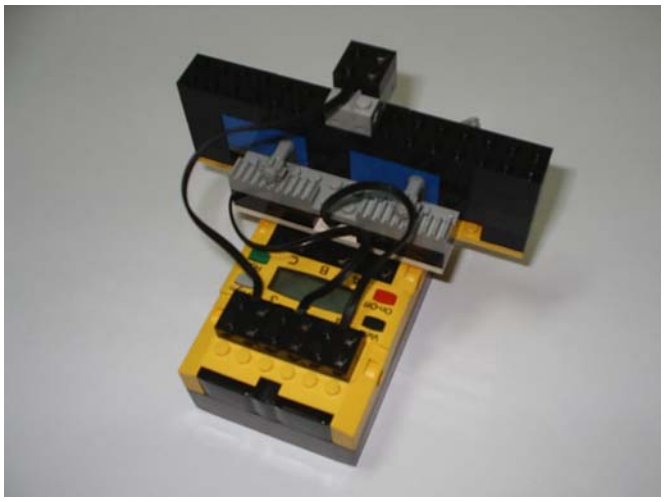


車子前面裝置兩個光源感應器，是爲了用來尋找地圖上的黑線（車子行走的路線），以便在接收車上駕設的微電腦指令後，能準確的判斷所在位置。

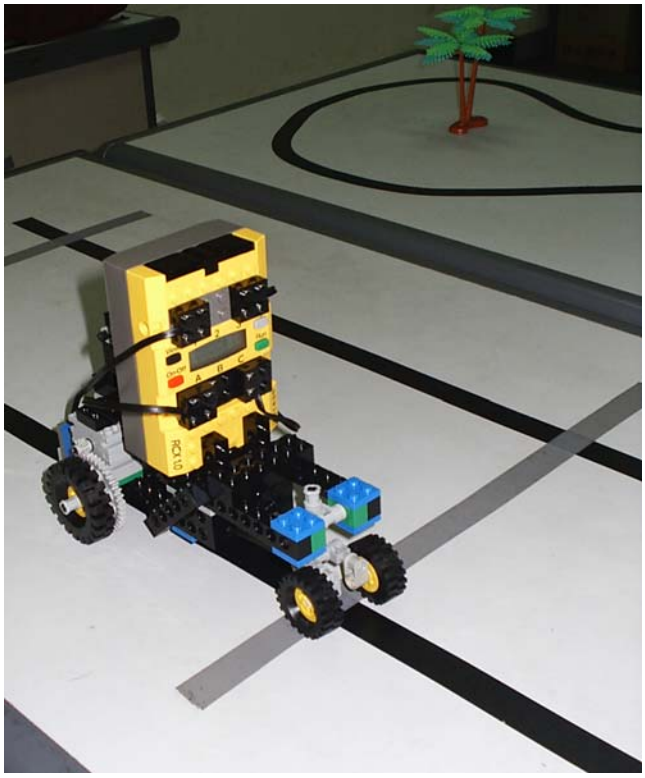
RCX 是整部車子的心臟，它接受由個人電腦所撰寫的程式，並透過各種輸入、輸出設備，將撰寫的程式執行出來。整部車子的運作就靠它來傳授指令。



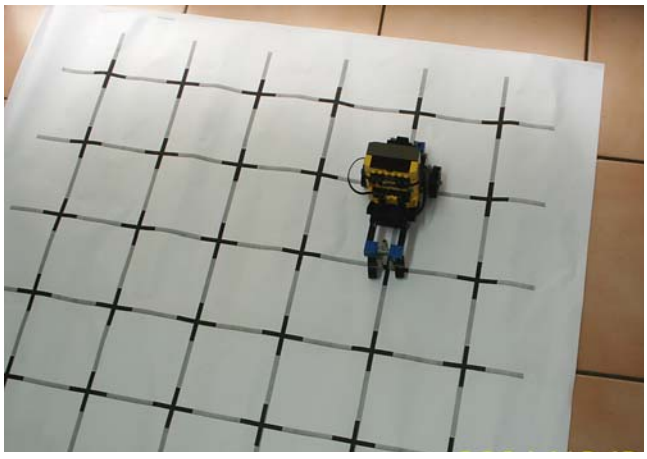
這是我原先構想下的產物：兩個角度感應器和一個按鈕，感應器共可感應十六個角度；用途分別負責：公車要由 A 地出發、公車要到 B 地和確定等。但是實驗的結果發現這樣的功能還是太少，所以我放棄這一部份，另外設法融入 Microsoft Visual Basic 6.0 程式軟體續寫。



這個地方是用來固定前面的轉盤的，這樣才不會再按確定時滑動造成錯誤。

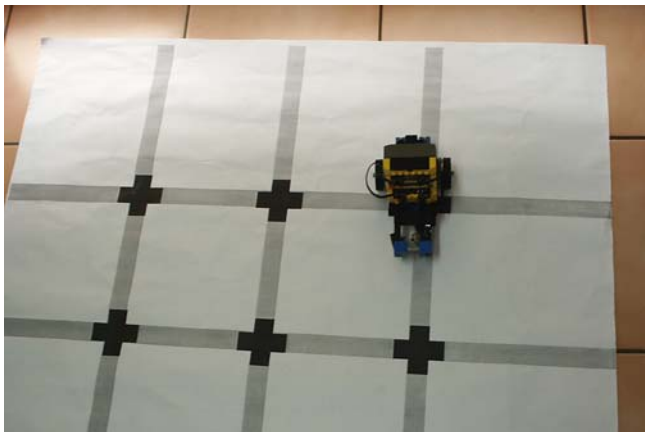


在實驗過程中我爲了操作方便，直接在工作桌上粘貼黑色電膠布做爲路線圖。



第一張路線圖：
程式指令完成後，將電腦繪製的路線圖列出來讓數位公車實際走一趟，發現：

1.車子經過路口要轉彎的時候，會先衝過去再轉彎。
2.轉彎轉過去後，兩個光源感應器有時候不會在線上。
3.兩個路口的間距太小，車頭已經到路口，車尾還沒離開前一個路口。

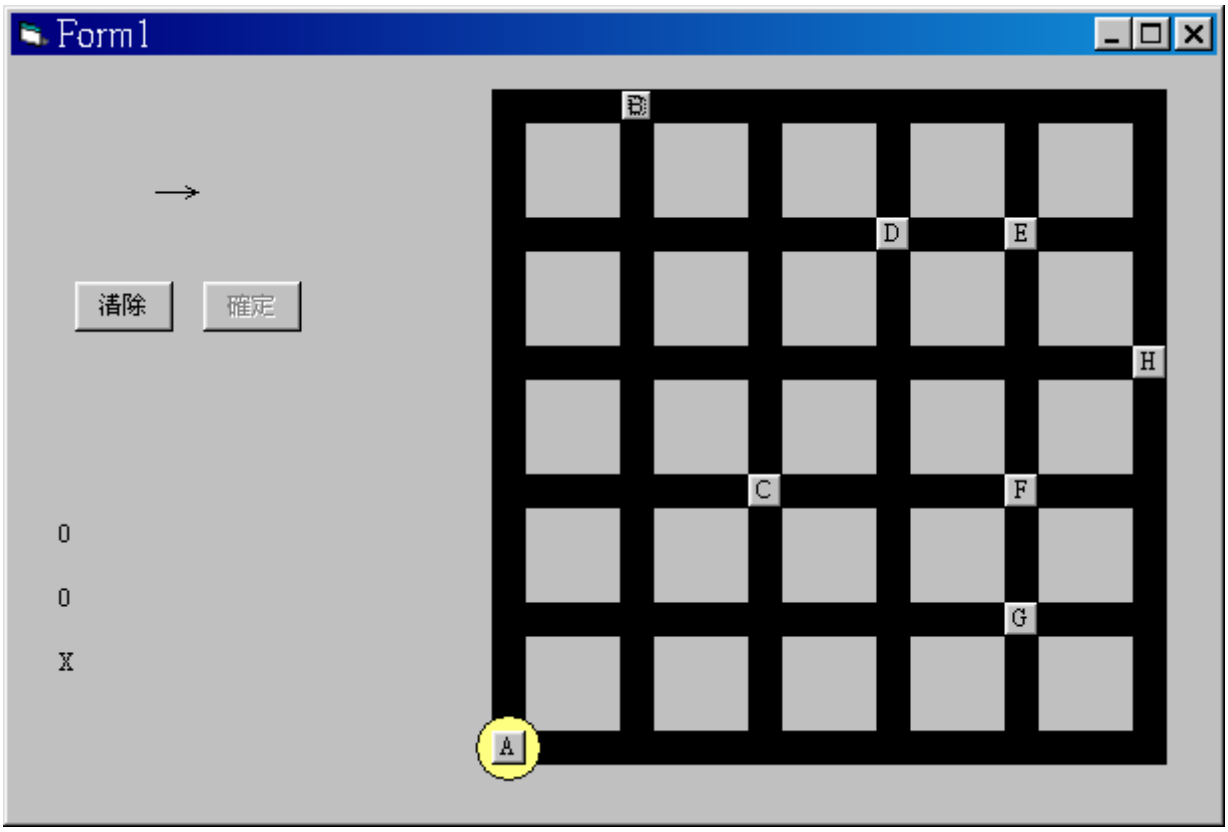


第二張路線圖：
這一張路線圖改善了前一張的缺點。我把路的寬度和兩個路口的間距加一倍，但是路口的黑線只把寬度加一倍，長度則維持原樣，再印製第二張路線圖。

第三張路線圖：
爲了操作的完整性，將第二張路線圖列印四大張銜接，供數位公車行駛。

附件二

由於受限於〔電腦樂高機器人控制系統 ROBOLAB〕所能寫入的程式無法全盤達到我要呈現的想法，所以我再加入 Microsoft Visual Basic 6.0 程式軟體續寫〔**順路乘載**〕的這一部份，並設法融合這兩支程式軟體，將 Microsoft Visual Basic 所寫成的程式碼輸入原來由〔電腦樂高機器人控制系統〕程式所控制的〔數位公車〕內，讓它們共同完成我交付未來世界的數位公車的任務。



【數位公車】順路乘載程式碼

```
Dim x, y, a, b, c, d, w, z, h5
Private Sub Command1_Click(Index As Integer)

If Label2.Tag = 0 Then
    Label1.Caption = Command1(Index).Caption
    Label2.Tag = Label2.Tag + 1
    a = Int(Index / 10)
```

```
        b = Index Mod 10
        Command1(Index).Enabled = False
Else
        Label3.Caption = Command1(Index).Caption
        Label2.Tag = Label2.Tag + 1
        c = Int(Index / 10)
        d = Index Mod 10
        Command1(10 * a + b).Enabled = True
End If

If Label2.Tag = 2 Then
        Command3.Enabled = True
End If

End Sub

Private Sub Command2_Click()

Label1.Caption = ""
Label2.Tag = 0
Label3.Caption = ""
Command3.Enabled = False
Command1(10 * a + b).Enabled = True

End Sub

Private Sub Command3_Click()

Timer1.Interval = 0

p = 0
q = 0
s = 0

If z = 0 Then
```

```
    Label4.Caption = Label4.Caption + Trim(Str(a)) + Trim(Str(c))
    Label5.Caption = Label5.Caption + Trim(Str(b)) + Trim(Str(d))
    Label6.Caption = Label6.Caption + Command1(10 * a + b).Caption + Command1(10 * c +
d).Caption
    z = z + 2
Else
    For t = 1 To z
        i = Mid(Label4.Caption, t, 1)
        j = Mid(Label4.Caption, t + 1, 1)
        k = Mid(Label5.Caption, t, 1)
        l = Mid(Label5.Caption, t + 1, 1)
        If Abs(i - a) + Abs(a - j) = Abs(j - i) And Abs(k - b) + Abs(b - l) = Abs(l - k) Then

            h1 = j
            h2 = a
            h3 = l
            h4 = b
            Call aaa(h1, h2, h3, h4 * 1)
            m = h5

            h1 = c
            h2 = a
            h3 = d
            h4 = b
            Call aaa(h1, h2, h3, h4 * 1)
            n = h5
            If Abs(m - n) <= 1 Or Abs(m - n) = 7 Or m = 100 Or n = 100 Then
                s = s + 1
                If Abs(a - c) + Abs(c - j) = Abs(j - a) And Abs(b - d) + Abs(d - l) = Abs(l - b)
Then
                    p = t
                    q = t + 1
                    GoTo yyy:
                Else
                    For r = t To z
```

```
If r = t Then
    GoTo xxx:
End If

i = Mid(Label4.Caption, r, 1)
j = Mid(Label4.Caption, r + 1, 1)
k = Mid(Label5.Caption, r, 1)
l = Mid(Label5.Caption, r + 1, 1)

h1 = j
h2 = i
h3 = l
h4 = k
Call aaa(h1, h2, h3, h4 * 1)
m = h5

h1 = c
h2 = a
h3 = d
h4 = b
Call aaa(h1, h2, h3, h4 * 1)
n = h5

If Abs(m - n) <= 1 Or Abs(m - n) = 7 Or m = 100 Or n = 100 Then
    If Abs(i - c) + Abs(c - j) = Abs(j - i) And Abs(k - d) + Abs(d - l) =
Abs(l - k) Then

        p = t
        q = r + 1
        GoTo yyy:
    End If

    h1 = c
    h2 = j
    h3 = d
```

```
                    h4 = l
                    Call aaa(h1, h2, h3, h4 * 1)
                    m = h5

                    h1 = j
                    h2 = i
                    h3 = l
                    h4 = k
                    Call aaa(h1, h2, h3, h4 * 1)
                    n = h5

                    If Abs(m - n) <= 1 Or Abs(m - n) = 7 Then
                        s = s + 1
                    Else
                        Exit For
                    End If
                Else
                    Exit For
                End If
xxx:
                If r = z And s = z - t + 1 And s <> 0 Then
                    p = t
                    q = z + 2
                End If

            Next r


                End If
            End If
        End If
    Next t
    If p = 0 And q = 0 Then
        p = z + 1
        q = z + 2
```

```
    End If
yyy:
    Label4.Caption = Left(Label4.Caption, p) + Trim(Str(a)) + Right(Label4.Caption, z - p + 1)
    Label5.Caption = Left(Label5.Caption, p) + Trim(Str(b)) + Right(Label5.Caption, z - p + 1)
    Label6.Caption = Left(Label6.Caption, p) + Command1(10 * a + b).Caption +
Right(Label6.Caption, z - p + 1)

    z = z + 1

    Label4.Caption = Left(Label4.Caption, q) + Trim(Str(c)) + Right(Label4.Caption, z - q + 1)
    Label5.Caption = Left(Label5.Caption, q) + Trim(Str(d)) + Right(Label5.Caption, z - q + 1)
    Label6.Caption = Left(Label6.Caption, q) + Command1(10 * c + d).Caption +
Right(Label6.Caption, z - q + 1)

    z = z + 1

End If
aaa:
For o = 1 To z
    If Mid(Label4.Caption, o + 1, 1) = Mid(Label4.Caption, o, 1) And Mid(Label5.Caption, o + 1,
1) = Mid(Label5.Caption, o, 1) Then
        Label4.Caption = Left(Label4.Caption, o) + Right(Label4.Caption, z - o)
        Label5.Caption = Left(Label5.Caption, o) + Right(Label5.Caption, z - o)
        Label6.Caption = Left(Label6.Caption, o) + Right(Label6.Caption, z - o)
        z = z - 1
        GoTo aaa:
    End If
Next o

Call Command2_Click

Spirit1.SetVar 8, 2, Val(Mid(Label4.Caption, 2, 1))
Spirit1.SetVar 9, 2, Val(Mid(Label5.Caption, 2, 1))

Timer1.Interval = 1
```

```
End Sub

Private Sub Form_Activate()

x = 0
y = 0
z = 0

End Sub

Private Sub aaa(h1, h2, h3, h4 As Integer)

If h1 - h2 > 0 Then
    If h3 - h4 > 0 Then
        h5 = 5
    End If
    If h3 - h4 = 0 Then
        h5 = 4
    End If
    If h3 - h4 < 0 Then
        h5 = 3
    End If
End If
If h1 - h2 = 0 Then
    If h3 - h4 > 0 Then
        h5 = 6
    End If
    If h3 - h4 = 0 Then
        h5 = 100
    End If
    If h3 - h4 < 0 Then
        h5 = 2
    End If
End If
```

```
If h1 - h2 < 0 Then
    If h3 - h4 > 0 Then
        h5 = 7
    End If
    If h3 - h4 = 0 Then
        h5 = 8
    End If
    If h3 - h4 < 0 Then
        h5 = 1
    End If
End If

End Sub
Private Sub Form_Load()

Spirit1.InitComm
w = 0

End Sub

Private Sub Form_Unload(Cancel As Integer)

Spirit1.CloseComm
End

End Sub

Private Sub Timer1_Timer()

g = Spirit1.Poll(0, 0)
u = Spirit1.Poll(0, 6)
v = Spirit1.Poll(0, 7)
x = Spirit1.Poll(0, 10)
y = Spirit1.Poll(0, 11)
```

```
If z <> 0 Then
    If w = 0 Then
        Spirit1.SetVar 8, 2, Val(Mid(Label4.Caption, 2, 1))
        Spirit1.SetVar 9, 2, Val(Mid(Label5.Caption, 2, 1))
        w = 1
    Else
        If u = 1 Then
            If z = 1 Then
                Label4.Caption = Trim(Str(x))
                Label5.Caption = Trim(Str(y))
                Label6.Caption = "X"
            Else
                Label4.Caption = Trim(Str(x)) + Right(Label4.Caption, z - 1)
                Label5.Caption = Trim(Str(y)) + Right(Label5.Caption, z - 1)
                Label6.Caption = "X" + Right(Label6.Caption, z - 1)
            End If
            z = z - 1
            u = 0
            Spirit1.SetVar 6, 2, 0
        End If

        Spirit1.SetVar 8, 2, Val(Mid(Label4.Caption, 2, 1))
        Spirit1.SetVar 9, 2, Val(Mid(Label5.Caption, 2, 1))
        Label4.Caption = Trim(Str(x)) + Right(Label4.Caption, z)
        Label5.Caption = Trim(Str(y)) + Right(Label5.Caption, z)
    End If
    Spirit1.SetVar 5, 2, 0
Else
    Spirit1.SetVar 5, 2, 1
    w = 0
End If

If Spirit1.Poll(0, 0) <> 0 Then
    Shape2.Left = 3480 + 960 * x
    Shape2.Top = 4920 - 960 * y
```

```
Else
    If v = 2 Then
        Shape2.Left = 3480 + 960 * x
        Shape2.Top = 4920 - 960 * y - 480
    End If
    If v = 4 Then
        Shape2.Left = 3480 + 960 * x + 480
        Shape2.Top = 4920 - 960 * y
    End If
    If v = 6 Then
        Shape2.Left = 3480 + 960 * x - 480
        Shape2.Top = 4920 - 960 * y
    End If
    If v = 8 Then
        Shape2.Left = 3480 + 960 * x
        Shape2.Top = 4920 - 960 * y + 480
    End If
End If

End Sub
```

評語

1. 公車的最佳行進路徑是現實生活中很有意思的問題。
2. 一個國三同學能以數學的觀點設計演算法來解決，是十分值得鼓勵的。
3. 實驗驗證方法做得十分精緻。
4. 改進的方向包括：(1)考慮更多的實際狀況，因爲目前的問題稍嫌簡化。

   (2)可以做演算法複雜度(complexity)分析。
5. 其實動能評估用電腦預測亦是可行的作法。