中華民國 第 49 屆中小學科學展覽會

作品說明書

高職組　電子、電機及資訊科

最佳創意獎

091013

互動式魔術方塊教學軟體

學校名稱：國立東勢高級工業職業學校

作者：	指導老師：
職二 林旻翰	徐毅
職二 巫柏松	
職二 高銓尉	
職二 張貿帆	

關鍵詞：VB、DirectX、魔術方塊

摘要

當我們在學魔術方塊時，我們買了書，還到網路搜尋一些相關的教學資料，我們發現都只是圖形、公式和方法的解說，這些內容既多且雜，大家的疑問：「我手上的魔術方塊現在要套用哪個公式呢？」，非常不容易學會，因此，我們想做一個『互動式魔術方塊教學軟體』。

我們使用 Visual Basic 2008 Expressed + DirectX 9.0，魔術方塊的 3D 畫面使用 Direct3D，音效使用 DirectSound，此軟體的兩個創新想法爲：

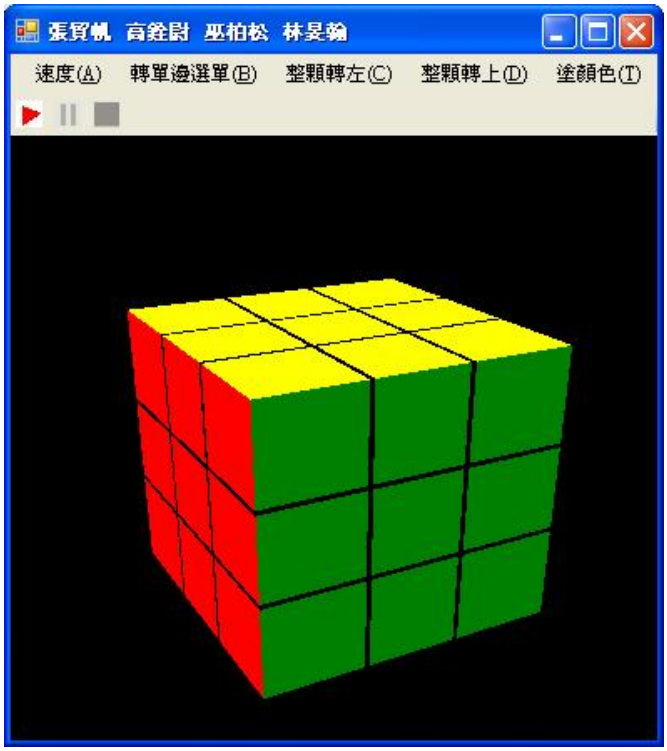
（1）增加互動性：使用者可設定電腦上的魔術方塊與自己手上的一模一樣，執行程式後你只要跟著電腦轉和聽語音解說，就可學會。

（2）圖形與公式的簡化：

圖形：網路上有約 27 種，我們只有用『1 種』簡單又好記的方法。

公式：現在的做法有 10 幾種，我們只用『5 種』方便又好用的做法。

下圖爲軟體的操作畫面，從使用者的意見反應得知，這是第一次有這種教學軟體，此種互動式實況教學也真的有助學習。



壹、研究動機

前一陣子校園裡流行魔術方塊，不管走到哪都看得到人手一顆魔術方塊，當初剛開始接觸魔術方塊的時候，覺得它很酷，明明只是用 6 種顏色和 26 顆小正方體組合起來的東西，卻有千變萬化種變化，使得許多人爲了解開它而苦惱。因此我們就上網找尋魔術方塊教學網站，希望可以一夕之間就可以把它解開，但是我發現網路上的確有許多的資料讓我參考與學習，奇怪的是，也不知道從何下手。

我們又再買了一本書，發現跟網路上的教學都差不多，還是一直背很多公式和記一些步驟，無法實際看到方塊的狀況，所以，它們都沒有辦法將教學方塊設定成跟自己手中方塊顏色一模一樣，也就不容易知道現在轉到的位置該如何解決，因此，網路上的教學只能解出固定的狀況，如果沒有其他人當場教導，真的很難學會。

於是，我們想做一個更簡單更有順序且可以互動的『**魔術方塊教學軟體**』。

貳、研究目的

一、解決現有學習者對網路或書籍中無互動學習的問題。

二、把多樣化的公式和圖形重新整理，只用更少、簡單、好記的方法呈現。

參、研究設備及材料

一、研究設備：

電腦、麥克風、喇叭。

二、研究材料：

VB 2008 Expressed version

Microsoft DirectX 9.0 SDK

錄音軟體。

肆、研究過程或方法

此次研究分成四部份，第一部份爲製作出魔術方塊基本圖型，第二部份爲轉動魔術方塊、塗顏色與速度，第三部份分爲製作魔術方塊第一層又一面解法、第二層解法、第三層又一面解法，第四部份爲製作跑馬燈與語音功能。

圖 4-1 是製作程序流程圖：

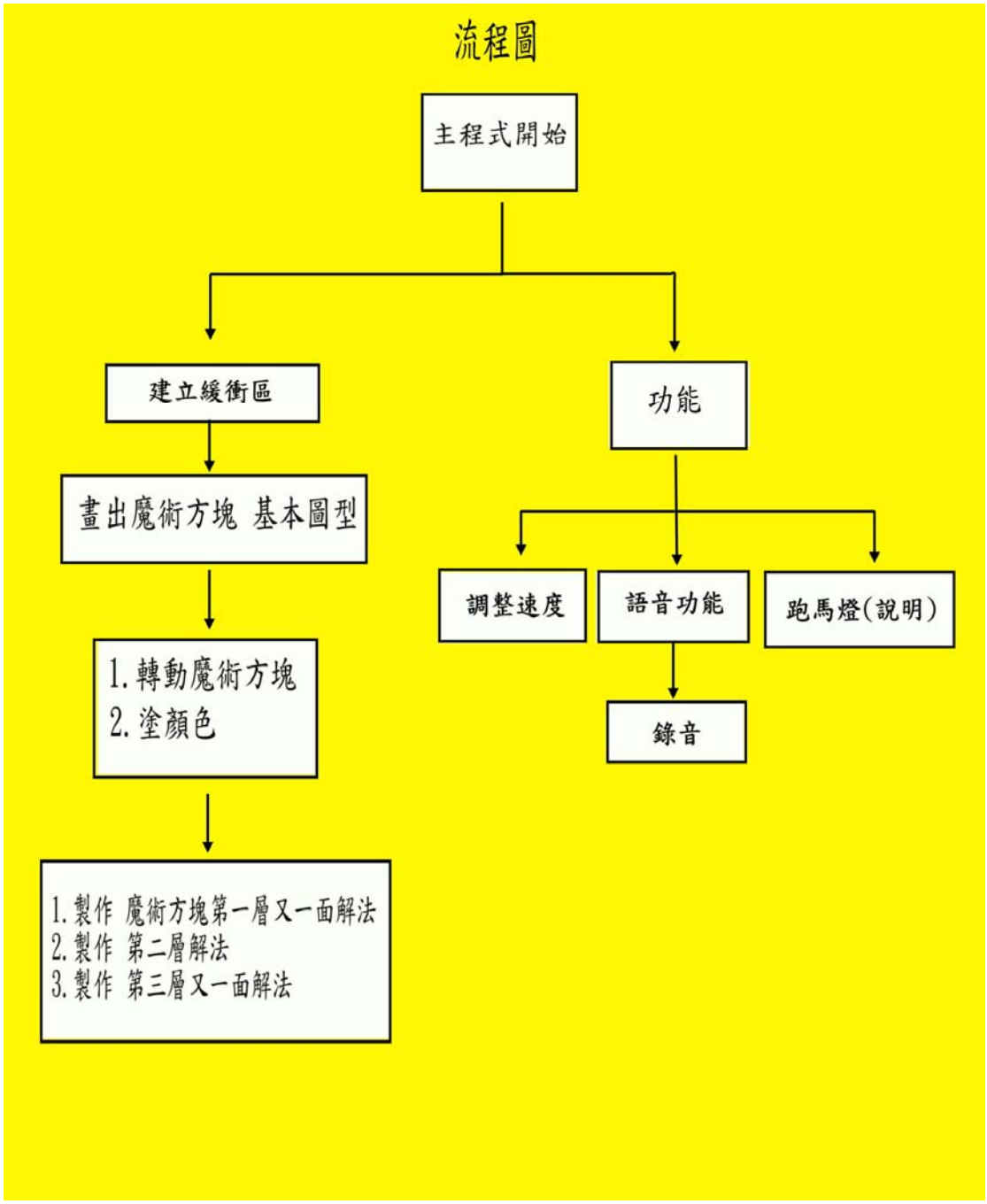


圖 4-1

一、3D 基本概念：

圖 4-2 爲 3D 畫面的基本架構，此傳輸管道爲將模型頂點轉換成畫面頂點，進入傳輸管道的是模型空間頂點，從傳輸管道末端出來的，則是畫面頂點，此理念是將 3D 空間用 2D 空間呈現（投影）。

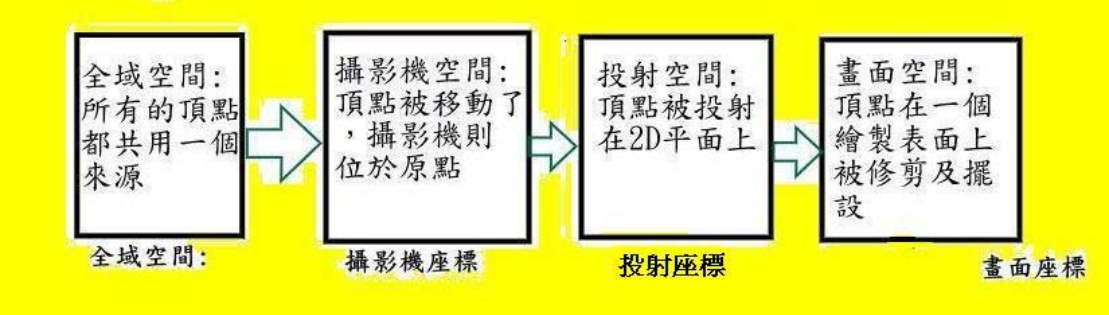


圖 4-2

1、什麼是模型

簡單來說模型是頂點(vertices)的集合，並包含每一頂點的一些附加屬性（該頂點些對應的材質座標、該頂點點的向量等等）。指定一個模型頂點相對於該模型的本地原點的X、Y、Z元件；換句話說，頂點係位於本地貨模型空間。模型可以依據使用它們的方法包含附加資料，不過那是基本概念。

(1) 頂點構成三角形的三種方法

爲了正確的繪製模型，Direct3D 必須解釋給它的頂點，而且必須將這些頂點形成三角形。它可以使用三種方式來完成：三角形清單(triangle lists)、三角形帶(triangle strips)及三角形扇(triangle fans)。Direct3D 將這些稱之爲原體類型(primitive types)。

(1-1)三角形清單

當給 Direxat3D 三角形清單時，必須給它每個三角形的頂點，如圖 4-3。使用三角形清單，會得到許多重複的頂點。使用越多的頂點，就會佔用越多的記憶體。

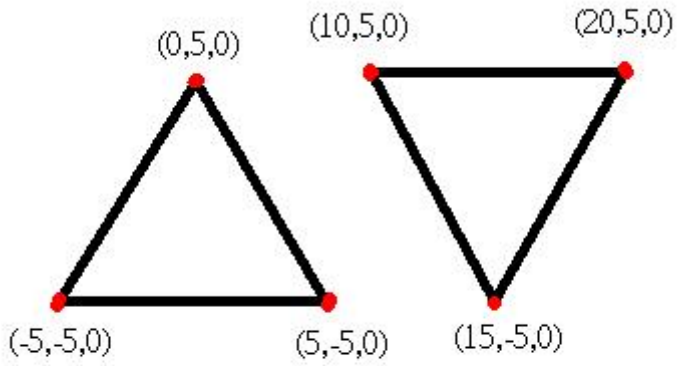


圖 4-3

(1-2)三角形帶

使用第一個三角形的最後兩個頂點及一個新頂點來構成下一個三角

形，基本上每個三角形都與它前一個三角形共用兩個頂點，如圖 4-4。

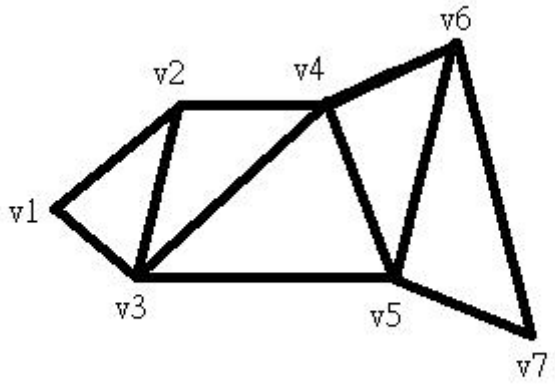


圖 4-4

(1-3)三角形扇

在這個模式裡，給 Direct3D 該扇形基礎點的頂點，之後再給它所有扇面上的頂點，它會瞭解每個三角形的結束頂點是另一個三角形的開始頂點，如圖 4-5。

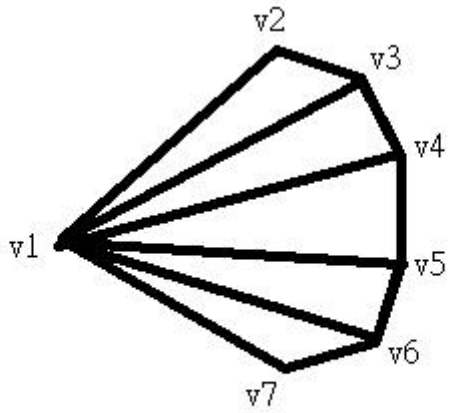


圖 4-5

2、本地座標：模型頂點的參考座標，通常是每個物體的中心點爲本地原點。

3、全域空間：３Ｄ世界裡全部物體的單一原點座標，本地座標經過此轉換就成爲全域座標。

4、攝影機空間：全域座標代表每個畫面的觀看者位置在全域座標的原點朝+Z 軸方向，但是，事實上你可以採取任何觀看位置與角度來產生畫面，因此，攝影機空間的區塊功能就是把整個全域空間予以轉換，讓場景觀看者的位子（攝影機的位子）移至全域原點。

5、投射空間：把３Ｄ座標投射到一個２Ｄ平面（螢幕）上後之座標。

6、畫面空間：所有投射座標不一定會同時都可看到，畫面空間系統會檢視每個頂點，去掉無法見到的頂點後形成所謂的畫面座標。

二、製作魔術方塊基本圖型

1、魔方基本架構

運用三個點組成一個小三角形，再用兩個小三角形組成一個小正方形，再用九個小正方形組成一個面，魔術方塊有六個面，所以總共運用了３２４個點、１０８個小三角形、５４個小正方形組成了魔術方塊基本圖形。圖 4-6 是畫一個三角形所運用到的點。

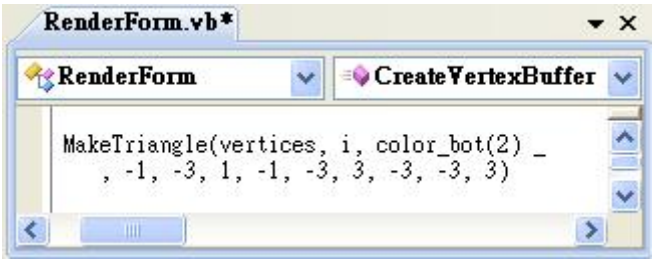


圖 4-6

2、基本上色

運用陣列以及內建的顏色，把每一個小正方形取一個名稱，設定顏色。圖 4-7 是每個小方形初始化的顏色。

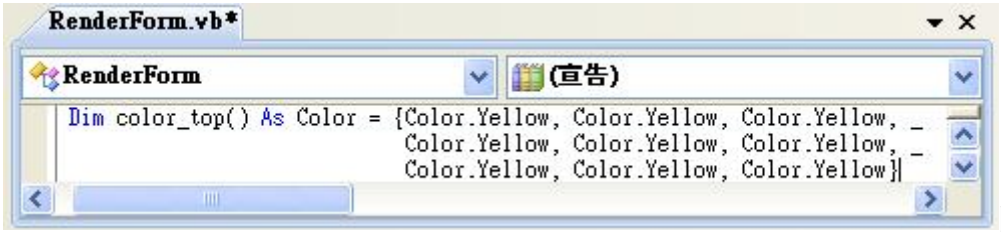


圖 4-7

三、轉動魔術方塊：

１、**3D 畫面製作七大步驟**：圖 4-8 爲實際程式碼，搭配來說明。

Step１：呼叫 m_Device.Clear 方法來清除後緩衝區（存放畫面頂點）。

Step２：呼叫 m_Device.BeginScene 指示 Direct3D 開始指揮畫製場景。

呼叫 SetupMatrices，將資料流來源設置到繪製的頂點緩衝區的位址，此時將會做全域座標、攝影機座標、投影座標的轉換。

Step 3： 呼叫 m_Device.SetStreamSource 將要繪製的頂點緩衝區位址設成資料流的來源。

Step 4： 呼叫 m_Device.VertexFormat，設定頂點著色引擎，包含頂點所有資料，例如：顏色。

Step 5： 呼叫 m_Device.DrawPrimitives，此時會依據上面步驟設定好的頂點與其資料來繪製原始畫面。

Step 6： 呼叫 m_Device.EndScene 代表你已經發佈完繪製場景所需指令。

Step７： 呼叫 m_Device.Present 來呈現後緩衝區，也就是顯示實際畫面。

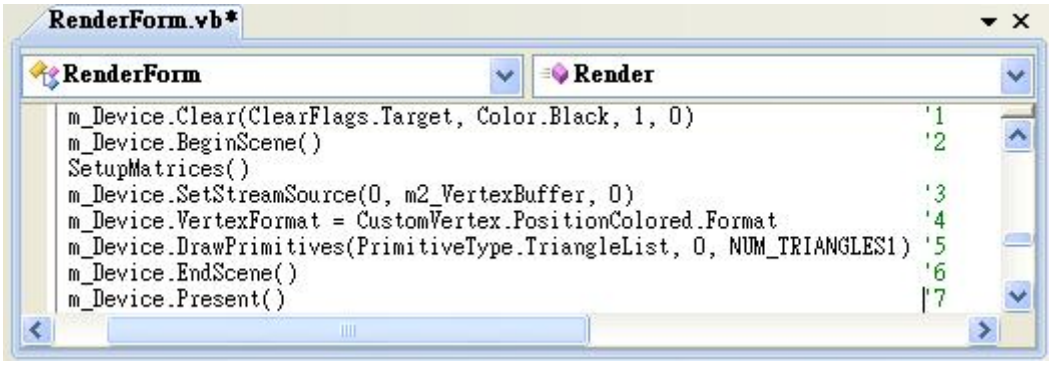


圖 4-8

２、點的存放位置

圖 4-9 是不轉動的點存放的緩衝區位置，圖 4-10 是轉動的點存放的緩衝區位置。

圖 4-9

圖 4-10

３、換顏色以及轉ｘ、ｙ、ｚ軸

轉動魔術方塊是以換顏色的方式來達成，程式碼如圖 4-11。

編號１：轉動前的一邊顏色存到設定的變數中。

編號２～５：把其他顏色覆蓋到未轉動前的顏色，再把存起來得一邊顏色覆蓋到上一個顏色。

編號６：用 DirectX 內建轉動方法依Ｘ、Ｙ或Ｚ軸轉動。

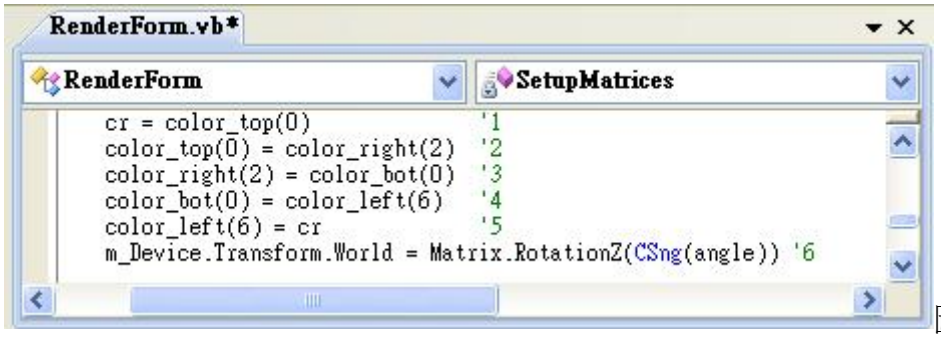


圖 4-11

四、塗顏色功能：

這是很重要的一個步驟，就是能做成互動感受的一個先備基礎。

１、顏色鈕：圖 4-12 爲實際程式碼。

編號１：爲顏色鈕。

編號２：在顏色按鈕的程式碼裡設一變數（ｌｏｖｅ）再把要變成的顏色存到變數裡。

圖 4-12

２、位置鈕：圖 4-13 爲實際程式碼。

編號１：爲位置鈕。

編號２：在位置按鈕的程式碼裡儲存要變色的位置。

圖 4-13

伍、速度：此概念是依據使用者操作速度不同而產生。

1、速度的變換：圖 4-14 爲實際程式碼。

編號１：速度從中變快的情形。

編號２：設定換顏色的時間點。

編號３：速度累加的值。

要變快就把數值改成 ”２０”換顏色時間點改成 ”９８０”，要變中就

把累加數值改成 ”１０”換顏色時間點改成 ”９９０”，要變慢就把累加數值改成 ”５”換顏色時間點改成 ”９９５”。

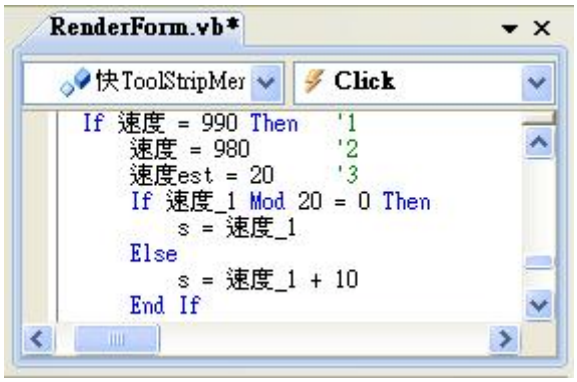


圖 4-14

六、製作魔術方塊：

１、一面又第一層的解法

(1) 第一面的四邊，轉到與它對面的四邊， 如圖 4-15-1。

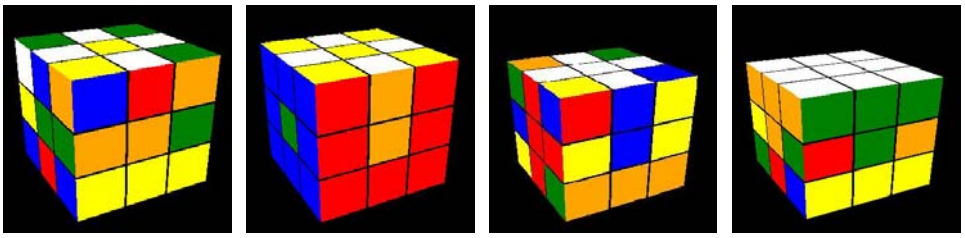
(2) 接下來把剛轉上去的邊對齊與中心的顏色，如圖 4-15-2。

(3) 在轉 180 度下去，其他三邊也以同樣的方式做，轉完後再把整個方塊轉上來，如圖 4-15-3。

(4) 把顏色對齊完後，接下來就把角塞進去並且對齊旁邊的顏色，如圖 4-15-4。

(5) 角也塞完後，一面一層就已做完了，接下來就要做第二層了。

圖 4-15-1 圖 4-15-2 圖 4-15-3 圖 4-15-4



２、第二層的解法

首先把整個方塊轉下，再來要對齊以下橘色及藍色的方塊，如圖 4-16-1。

或是是對齊以下綠色及紅色的方塊，如圖 4-17-1。

（1）假如出現的是圖 4-16-1，首先把橘色及藍色的方塊轉道橘色面的對面，如圖 4-16-2。

接下來轉以下公式：左逆時鐘 X 90 → 上順時鐘 Y 90 → 左順時鐘 X 90 → 上順時鐘 Y 90 → 前順時鐘 Z 90 →上逆時鐘 Y 90 →前逆時鐘 Z 90。

程式碼如圖 4-18，就能把第二層做好，如圖 4-16-2。

（2）假如出現的是圖 4-17-1，首先把綠色及紅色的方塊轉到綠色面的對面，如圖 4-17-2。

接下來轉以下公式：前順時鐘 Z 90 → 上逆時鐘 Y 90 → 前逆時鐘 Z 90 → 上逆時鐘 Y 90 → 左逆時鐘 X 90 →上順時鐘 Y 90 →左順時鐘 X 90。程式碼如圖 4-19，就能把第二層做好，如圖 4-17-3。

（3）按照以上的兩種公式，就能完成了第二層。

圖 4-16-1

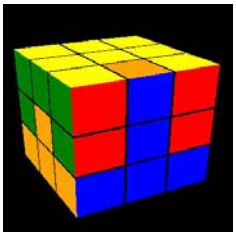


圖 4-16-2

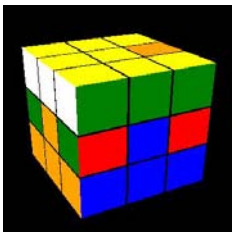


圖 4-16-3

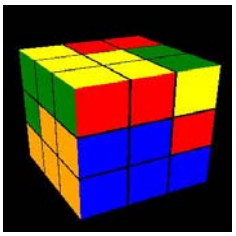


圖 4-17-1

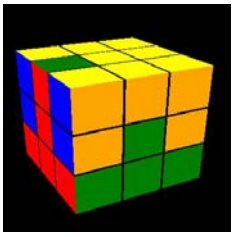


圖 4-17-2

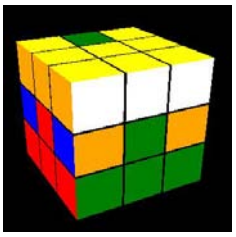
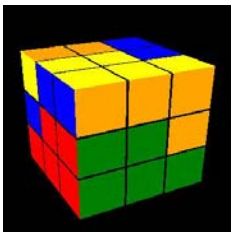


圖 4-17-3



SubMain.vb | RenderForm.vb

RenderForm | 第二層

```
3593                If color_left(1) = color_back(4) And _
3594                   color_top(3) = color_right(4) Then
3595                    m = 5 '左逆時鐘X 90
3596                    x = 5
3597                    If color_back(0) = color_bot(4) Then
3598                        m = 1 '上順時鐘Y 90
3599                        x = 1
3600                    End If
3601                End If
3602                If color_right(0) = color_bot(4) And _
3603                   color_top(8) = color_back(4) And _
3604                   color_front(2) = color_right(4) Then
3605                    m = 2 '左順時鐘X 90
3606                    x = 2
3607                End If
3608                If color_right(2) = color_bot(4) And _
3609                   color_top(1) = color_top(2) And _
3610                   color_back(0) = color_back(4) Then
3611                    m = 1 '上順時鐘Y 90
3612                    x = 1
3613                End If
3614                If color_front(2) = color_bot(4) And _
3615                   color_right(5) <> color_bot(4) And _
3616                   color_top(8) = color_right(4) And _
3617                   color_right(0) = color_back(4) Then
3618                    m = 3 '前順時鐘Z 90
3619                    x = 3
3620                End If
3621                If color_front(2) = color_bot(4) And _
3622                   color_right(5) = color_bot(4) Then
3623                    m = 4 '上逆時鐘Y 90
3624                    x = 4
3625                End If
3626                If color_top(1) = color_top(2) And _
3627                   color_back(0) = color_back(4) And _
3628                   color_right(2) = color_bot(4) And _
3629                   color_right(2) = color_right(5) And _
3630                   color_back(1) = color_back(4) Then
3631                    m = 6 '前逆時鐘Z 90
3632                    x = 6
3633                End If
```

圖 4-18

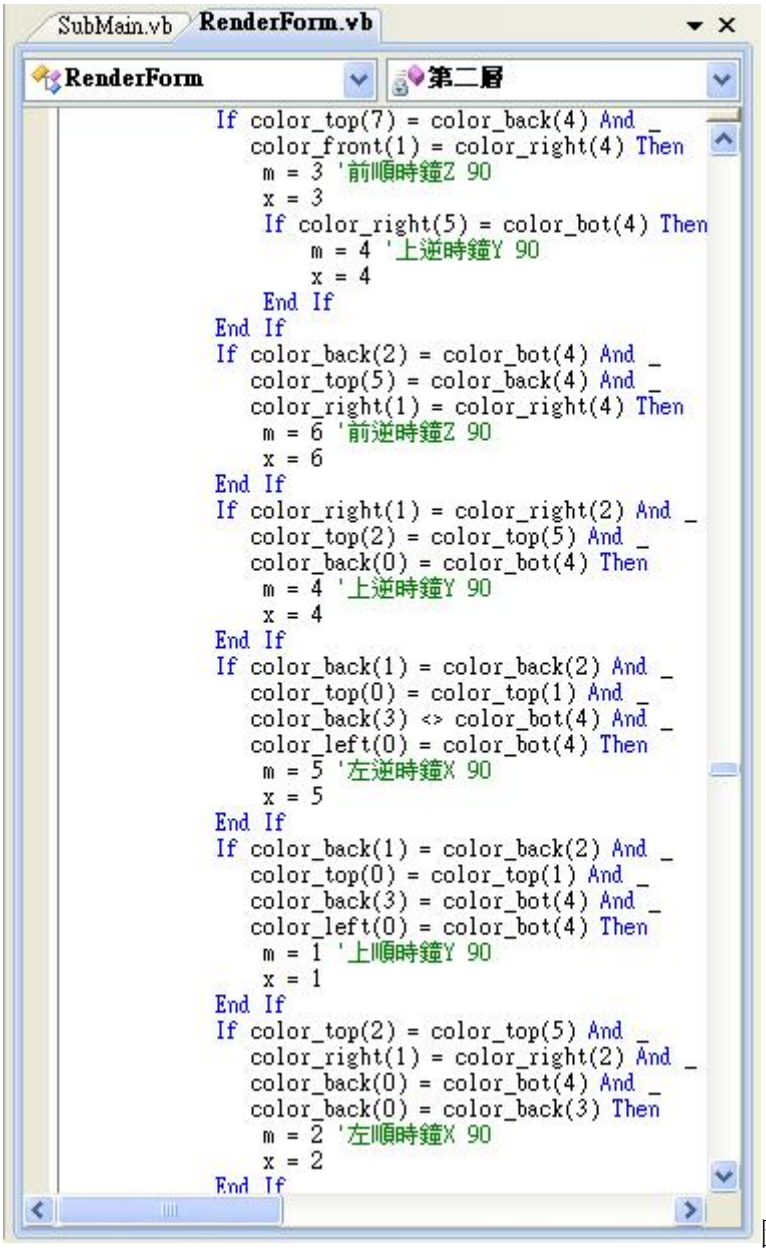


圖 4-19

3、第二面又第三層的解法

我們以四個簡單的公式來完成最後一面一層。第一個公式是用來整理第 2 面，以方便找，做第二個公式時所需要的圖形，如圖 4-20-1，完成第二個公式以後，會判斷是該執行第三個或第四個公式（來完成第三層）。完成上面的步驟後，我們又輸入了幾個判斷式來判斷從哪邊開始執行整理的公式會比較容易找到圖形，如圖 4-20-1。因爲我們是以一些沒有玩過或者是轉不出六面的學習者爲重心。所以我們只用了 4 個簡單的公式加上幾個圖形的位置來判斷並且完成第三面與第三層。

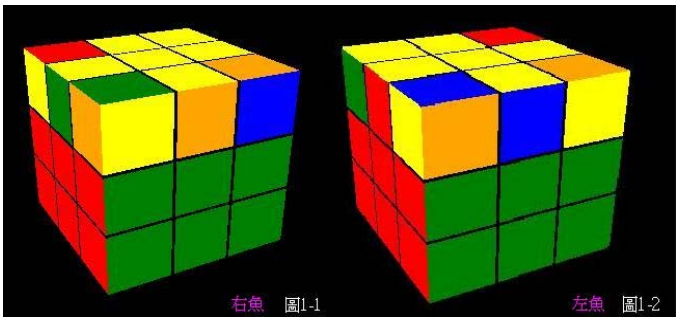


圖 4-20-1　　圖 4-20-2

圖 4-21 中的變數(trans)代表執行中的步驟。
共 6 步：『右逆、上逆、前逆、上順、前順、右順』
（逆＝逆時鐘，順＝順時鐘）
圖 4-22 中 s = 1000 代表方塊已經轉動 90 度。
假如 Bot38 = true(程式=啓動)每轉動方塊一個地方，變數值(trans)就會+1

Form1.vb*

(Form1 事件) | Load

```
If find_picture = True Then
    Select Case trans
        Case Is = 1
            m = 14
            x = 14
        Case Is = 2
            x = 4
            m = 4
        Case Is = 3
            x = 6
            m = 6
        Case Is = 4
            m = 1
            x = 1
        Case Is = 5
            m = 3
            x = 3
        Case Is = 6
            m = 17
            x = 17
        Case Is = 8
            trans = 1
    End Select
End If
```

圖 4-21

Form1.vb*

(Form1 | Load

```
If s = 1000 Then
    x = 0
    s = 0
    If bot38 = True Then
        trans += 1
    End If
End If
```

圖 4-22

圖 4-23 中的程式碼爲判斷該執行 圖 4-20-1 或 圖 4-20-2 的 執行步驟，圖 4-24 的程式碼在圖形同圖形 4-20-1 時執行，圖 4-25 的程式碼在圖形同圖形 4-20-2 時執行。

RenderForm.vb*

RenderForm | fish

```
If photo_fish = True Then
    If color_top(5) = color_top(4) And color_top(8) = color_top(4) And _
    color_back(0) <> color_top(4) Then
        fish_ture = 1
        trans = 1
        photo_fish = False
    ElseIf color_top(3) = color_top(4) And color_top(6) = color_top(4) And _
    color_back(2) <> color_top(4) Then
        fish_ture = 2
        trans = 1
        photo_fish = False
    Else
        x = 1
        m = 1
    End If
End If
```

圖 4-23

RenderForm.vb*

RenderFor | fish

```
If fish_ture = 2 Then '右魚
    Select Case trans
        Case Is = 1
            x = 17
            m = 17
        Case Is = 2
            x = 4
            m = 4
        Case Is = 3
            x = 4
            m = 4
        Case Is = 4
            x = 14
            m = 14
        Case Is = 5
            x = 4
            m = 4
        Case Is = 6
            x = 17
            m = 17
        Case Is = 7
            x = 4
            m = 4
        Case Is = 8
            x = 14
            m = 14
        Case Is = 9
            fish_ture = 0
    End Select
End If
```

圖 4-24

RenderForm.vb

RenderFor | fish

```
If fish_ture = 1 Then '左魚
    Select Case trans
        Case Is = 1
            x = 5
            m = 5
        Case Is = 2
            x = 1
            m = 1
        Case Is = 3
            x = 1
            m = 1
        Case Is = 4
            x = 2
            m = 2
        Case Is = 5
            x = 1
            m = 1
        Case Is = 6
            x = 5
            m = 5
        Case Is = 7
            x = 1
            m = 1
        Case Is = 8
            x = 2
            m = 2
        Case Is = 9
            fish_ture = 0
    End Select
End If
```

圖 4-25

七、跑馬燈及語音功能

1.跑馬燈做法：程式碼如圖 4-26。

（1）首先把要說明的文字用陣列變數來宣告成字串。

（2）在開啓 Timer1 這個物件用時間來讓它跑動。

（3）第 5 行程式碼，是把第一個字先移出去到 temp 暫存。

（4）第 6~8 行程式碼，是用 For 迴圈，首先，i=1，LCD(i - 1) = LCD(i)即爲 LCD(0) = LCD(1)，也就是把第二個字移到第一個字原本的位置，LCD(1)也就可以更改。

（5）第 9 行程式碼就是把 temp 暫存的字存放到最後一個位置。

（6）第 4 步驟可以把它顯示出來，每次讀取顯示的字前要先清掉前一個內容，才不會把舊的資料也顯示出來，清空之後就要把新的所有字依序讀出並顯示，就是程式碼 6~8 行做的事，i=0 就先把第一個字讀出，與空字串連起來後送到標籤上顯示，i =1 就把第二個字讀出並且將剛剛的第一個字做連結後送到標籤上顯示，如此做直到全部字都串起來爲止。

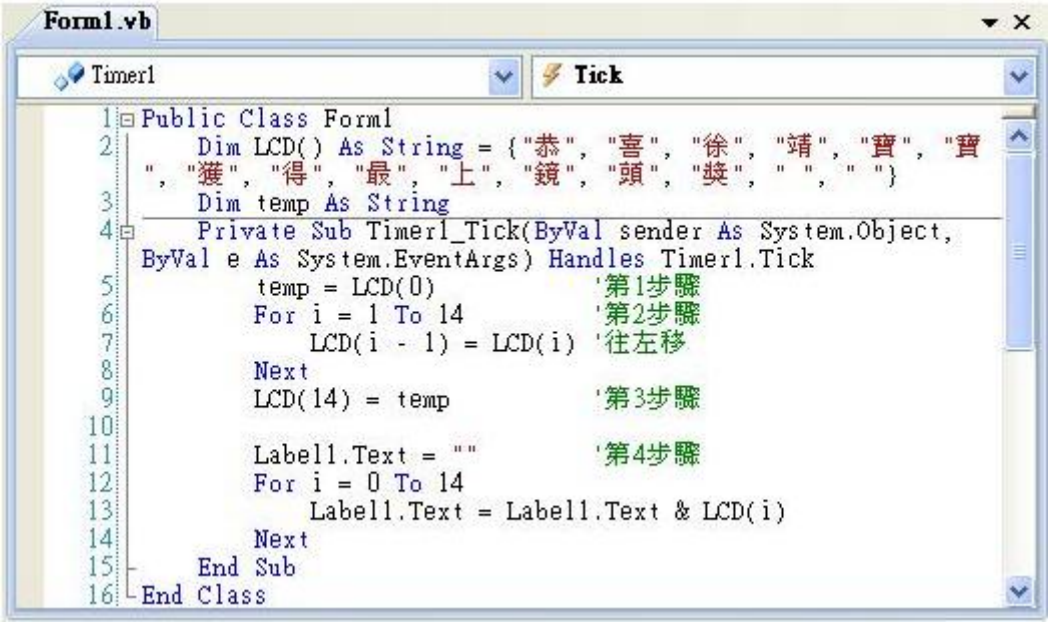


圖 4-26

2.音效播放器：使用 DirectSound，程式碼如圖 4-27。

（1）首先要把 DirectSound 開出來，在 vb2008 裡面，從專案→加入參考→點選 Microsoft.DirectX.DirectSound 把它加入到 VB2008 裡面，這是要播放音樂的必需程序。

（2）先宣告 2 個名稱定義爲 Microsoft.DirectX.DirectSound 和一個 Buffer。

（3）第９～１１行是播放音樂的程序。

（4）第１３～１５行是載入音樂的路徑。

（5）第１７行是讓音樂重複播放。

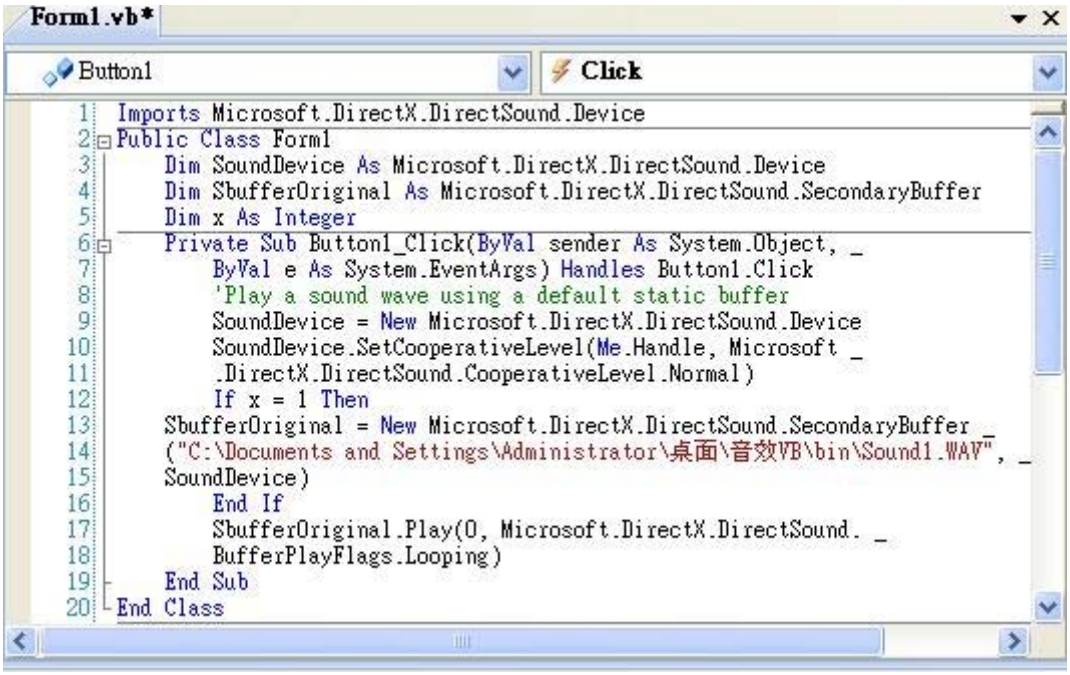

Form1.vb*
Button1
Click
Imports Microsoft.DirectX.DirectSound.Device
Public Class Form1
Dim SoundDevice As Microsoft.DirectX.DirectSound.Device
Dim SbufferOriginal As Microsoft.DirectX.DirectSound.SecondaryBuffer
Dim x As Integer
Private Sub Button1_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles Button1.Click
'Play a sound wave using a default static buffer
SoundDevice = New Microsoft.DirectX.DirectSound.Device
SoundDevice.SetCooperativeLevel(Me.Handle, Microsoft _
.DirectX.DirectSound.CooperativeLevel.Normal)
If x = 1 Then
SbufferOriginal = New Microsoft.DirectX.DirectSound.SecondaryBuffer _
("C:\Documents and Settings\Administrator\桌面\音效VB\bin\Sound1.WAV", _
SoundDevice)
End If
SbufferOriginal.Play(0, Microsoft.DirectX.DirectSound. _
BufferPlayFlags.Looping)
End Sub
End Class

圖 4-27

伍、研究結果

一、軟體流程架構圖：

魔術方塊是 3D 畫面，所以有一個建立 3D 畫面的七大步驟（橘色流程圖），顯示的畫面都由此決定。成功轉出六面的功能流程分三大部分（黃色流程圖），互動過程中還有很多單一功能（藍色流程圖）。

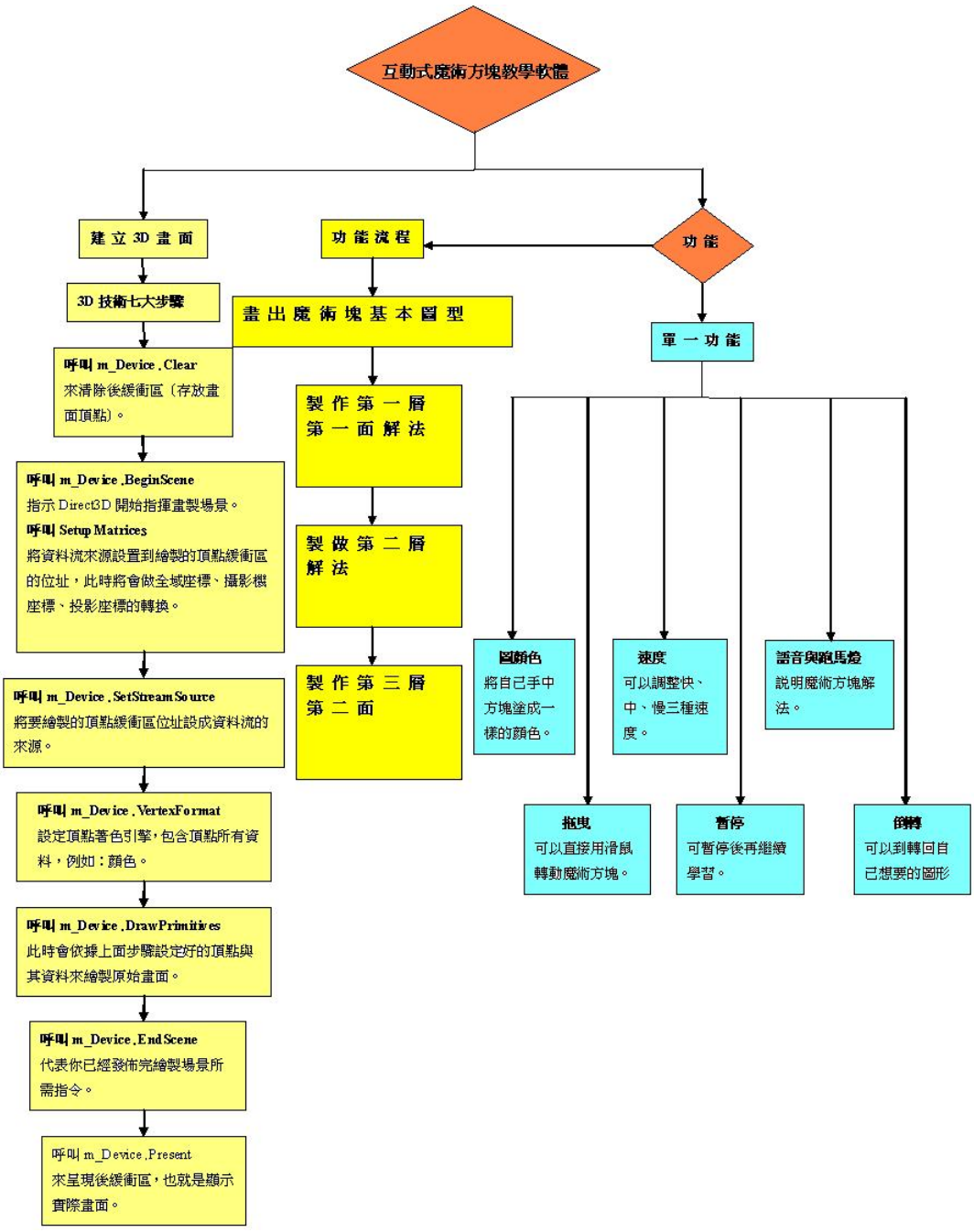


圖 5-0

（一）、塗顏色功能：使學者方塊與軟體上的一樣，畫面如圖 5-1，電腦端塗好顏色的方塊如圖 5-2，使用者方塊如圖 5-3。

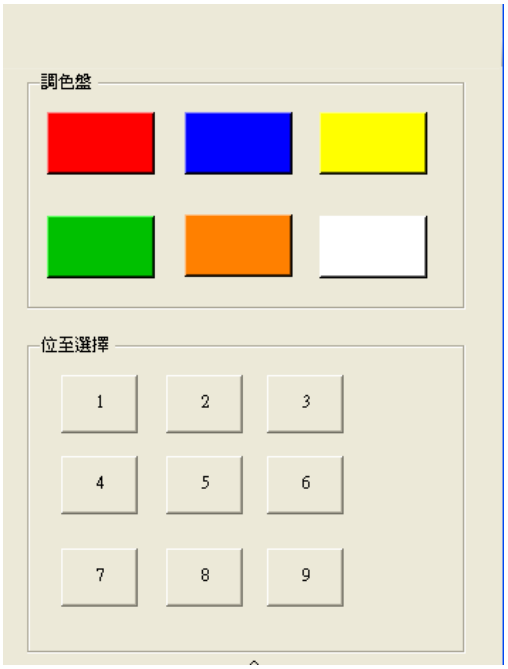


圖 5-1

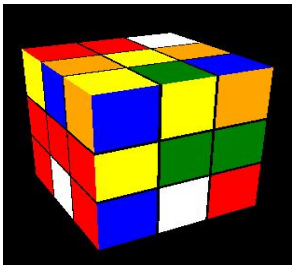


圖 5-2

圖 5-3

（二）、轉出六面功能說明：

1、學習者手上的方塊顏色設定完後，接下來轉一面一層，圖 5-4 爲電腦端的方塊，圖 5-5 爲使用者的方塊。

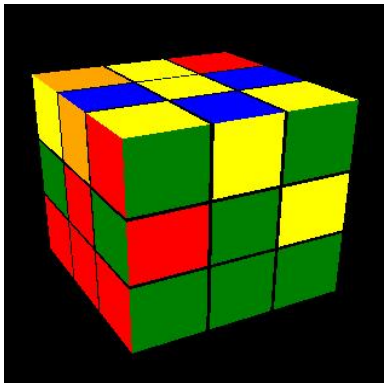


圖 5-4

圖 5-5

2、做完一面一層，接著轉第二層，圖 5-6 爲電腦端的方塊，圖 5-7 爲使用者的方塊。

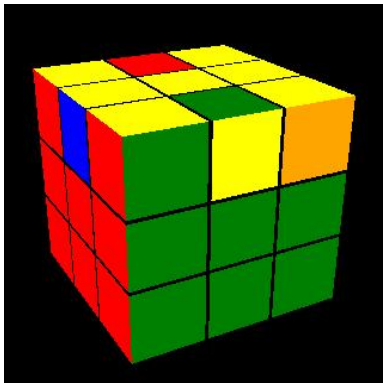


圖 5-6

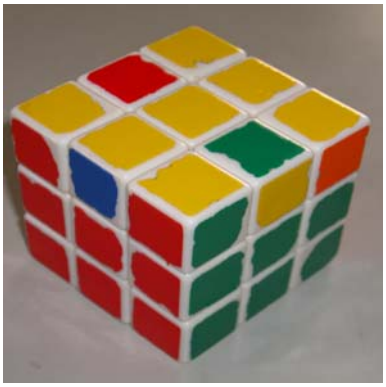


圖 5-7

3、再來轉第二面，圖 5-8 爲電腦端的方塊，圖 5-9 爲使用者的方塊。

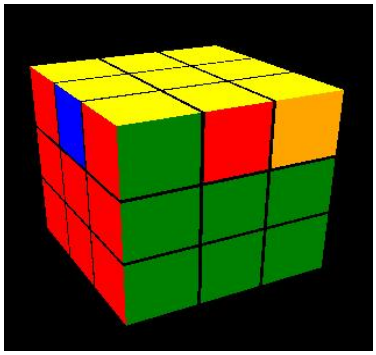


圖 5-8

圖 5-9

4、轉完第二面，再來轉第三層，轉完後方塊就六面完成了，圖 5-10 爲電腦端的方塊，圖 5-11 爲使用者的方塊。

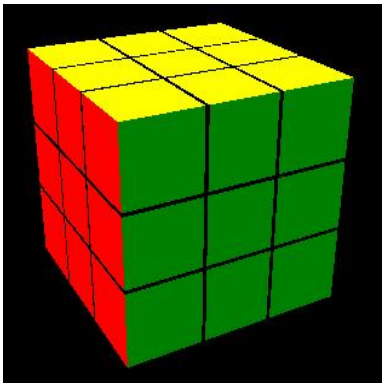


圖 5-10

圖 5-11

（三）、速度功能說明：

電腦上方塊的轉動速度分快、中、慢三種速度，學習者可依照自己的能力作最適當的調整。

（四）、語音與跑馬燈功能說明：

使用者可自己決定是否要聽語音教學說明，文字說明會一直顯示，兩者都隨著目前方塊的狀況而做最適當的說明。

（五）、拖曳功能說明：

如果使用者只是要用電腦玩魔術方塊，一樣可以利用拖曳功能來達到目的。

（六）、暫停功能說明：

如果玩到一半要暫停或是太快跟不上或是不小心轉錯了，沒關係，都可先用暫停功能停止方塊的轉動。

（七）、回覆功能說明：

如果不小心跟著轉轉錯了或是電腦太快跟不上，甚至要看看前幾個步驟到底是使用哪些圖形或公式，都可用回覆功能讓方塊，每按一下就會倒退到前一個狀態，我們的可是可以倒退好幾千步喔。

（八）、3D 功能說明：

1. 第一步驟：清除後緩衝區，這是呈現畫面前放置頂點座標的記憶體。
2. 第二步驟：開始繪製場景，把所有頂點資料流作座標的轉換。
3. 第三步驟：決定要繪製的頂點是哪些並設成來源資料流。
4. 第四步驟：來源資料流的頂點資訊加入。
5. 第五步驟：依據上面兩個步驟繪製畫面。
6. 第六步驟：後緩衝區的畫面已經會置完畢。
7. 第七步驟：被畫面呈現到螢幕上去。

陸、討論

一、讓魔術方塊整顆轉動，沒辦法只轉動單邊的問題？

經過看書來學習如何製作緩衝區，依緩衝區七大步驟我們新建立出兩個緩衝區，把要轉動的點與不轉動的點分開來放，之後再寫兩個副程式（SetupMatrices1）讓轉動的點隨著某一個軸轉動（圍繞），讓不轉動的點固定在軸的同一個地方。

二、顏色無法隨著方塊轉動而改變的問題？

參考資料，運用跑馬燈的原理，設一個變數把轉動前的一邊顏色存起來，在把其他顏色覆蓋到未轉動前的顏色，之後存起來的一邊顏色覆蓋到上一個顏色。

三、討論用什麼方法塗顏色？

我們選擇用按鈕的方式來呈現，因爲使用按鈕比較不會影響到我們其他的程式碼，所以感覺會比較好做。

四、緩衝區空間不夠的問題？

空間不足主要是因爲儲存點的空間不足夠，爲何會不足夠呢？因爲每轉動一次方塊，點的數目就會一直累加下去，並沒有清除掉，而且當初我們沒有把所使用的點及三角形的數目改成變數，所以會使程式不穩定，我們就先把使用的點及三角形數目改成變數，再把每邊轉動即不轉動使用到的點及三角形數分別存入變數中，在去上網找到(Dispose)是清除的語法，所以只要在緩衝區製作中再加上(Dispose)便會清除每轉動一次所累加的點。

柒、結論

當初做這個軟體的目的就是希望藉由軟體與使用者的互動與更少又簡單公式與做法，讓玩魔術方塊更容易、更簡單、更清楚。因此我們找了會玩但是解不出整顆六面，也找了從頭到尾都不會解但是很想要解開方塊的人，來使用我的方塊教學軟體來學習如何解開六面。

當他們使用軟體後，都覺得比網路上和書面上的教學內容更有實際互動，讓他們感受到有做中學的感覺，原因在於網路上和書面上都是很死板的，而我們的軟體可以自己設定顏色解決目前所遇到的問題和位置的解法，也還有實際在講解和語音的說明。

我也把他們的建議做了一些改善，例如：空間感不夠而讓學習者跟不上程式中的方塊，因此我們對於這個問題，我們做了一個暫停的按鈕，可以讓空間感不太夠的學習者，當他們跟不上程式裡面的方塊時，可以按下暫停鈕讓方塊停止，停下來看看說明和自己做到哪個步驟，了解以後再按下開始鈕繼續讓方塊轉動。但是某些學習者需要的介面，可能會對其他學習者造成困擾，例如：需要正反面的方塊以加以對應前後的位置，讓介面上可以看到兩顆方塊一個前面一個後面，但是這一個建議可能會對於其他學習者更混亂他的空間概念，因此像這一類的建議我們就沒有做實際的軟體更動。

所以，我們會加以討論使用者的建議是否加進去程式中，對於影響不太大的介面問題和個人感覺上的問題，我們都會以大多數人的需求下去做決定。

下表爲結論的統整表格。

	我的教學軟體	網路與書
公式	5	10 幾種
圖形	1	27
教學	互動式	單一式

捌、參考資料及其他

一、參考書籍

１、徐毅　編著

Visual Basic 程式設計（Easy try）

文魁資訊有限公司

2、Mason McCuskey 編著

DirectX 特效遊戲程式設計

博碩文化股份有限公司

二、參考網站

１、http://www.devx.com/dotnet/Article/36102/0/page/1

２、http://www.codeproject.com/KB/audio-video/Circular_Buffers.aspx

３、微軟 MSDN 線上論壇

４、微軟技術社群討論區

【評語】091013

本作品的主題具備學術性,並且為典型演算法之問題.。

本作品要解決的問題定義完整.。

本作品的 User Interface (人機界面) 開發完整.。

但若能精準的估算出在作品中所用演算法的複雜度，整個研究專題將更完整。