# 第 7 章 字元影像目標的正規化


## 7-1 為何需要正規化？

本書介紹的車牌辨識流程，與大多數傳統車牌辨識的主要差異是：我們的處理單位是個別的**字元**目標，而非**整張車牌**，這其實比較接近標準的 OCR(光學影像辨識)作法，好處之一是：經由字元組的排列我們很容易掌握並修正車牌的傾斜角度。截至目前為止，多數市售的車牌辨識傾斜 **10** 度以上就無法辨識了！因此如何辨識較傾斜車牌？就很值得研究了！本章將以一個傾斜約 **33** 度的車牌作為示範，來展現這種辨識能力。

本章重點是盡量用可視化的過程，讓大家看到一個原始的字元目標，如何經過幾何運算變成與字模相同大小影像的過程！車牌字元原本當然是一樣大的，但是在立體空間攝影時，就會因為距離遠近與拍攝角度不同而產生變形，所以必須有此正規化的程序。具體來說，就是將字元目標先轉正，再縮放到標準字模的大小，以供比對符合度。

首先請複製第五章的程式專案至此修改，公用之全域變數須宣告如下：

```
Dim B(,) As Byte '灰階陣列
Dim Z(,) As Byte '全圖二值化陣列
Dim Q(,) As Byte '輪廓線陣列
Dim Gdim As Integer = 40 '計算區域亮度區塊的寬與高
Dim Th(,) As Integer '每一區塊的平均亮度，二值化門檻值
Dim C As ArrayList '目標物件集合
Dim Tsel As TgInfo, Tbin(,) As Byte '選擇處理之目標與其二值化陣列
```

```
Dim Pw As Integer = 25, Ph As Integer = 50 '標準字模影像之寬與高
Dim mw As Integer, mh As Integer '標準字元目標寬高
Dim MC() As Array '正規化完成後的字元二值化陣列
Dim Inc As Double '車牌傾斜角度(>0 為順時針傾斜)
```

表單之主功能表應該包括：Open, Align, Rotate, Normalize, Correct All 與 Add Dash 等幾個按鍵。Open 按鍵功能與之前一樣不再覆述，Align 則是整合第五章之前的處理步驟，程式碼如下：

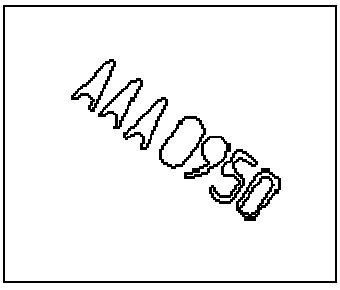
```
'找車牌字元目標群組
Private Sub AlignToolStripMenuItem_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles AlignToolStripMenuItem.Click
    Z = DoBinary(B) '二值化
    Q = Outline(Z) '建立輪廓點陣列
    C = getTargets(Q) '建立目標物件集合
    C = AlignTgs(C) '找到最多七個的字元目標
    Dim n As Integer = C.Count
    '末字中心點與首字中心點的偏移量，斜率計算參數
    Dim dx As Integer = C(n - 1).cx - C(0).cx
    Dim dy As Integer = C(n - 1).cy - C(0).cy
    Inc = Math.Atan2(dy, dx) '字元排列傾角
    '繪製字元輪廓
    ReDim Tbin(nx - 1, ny - 1)
    For k As Integer = 0 To C.Count - 1
        Dim T As TgInfo = C(k)
        For m As Integer = 0 To T.P.Count - 1
            Dim pt As Point = T.P(m)
            For i As Integer = 0 To T.P.Count - 1
                Dim p As Point = T.P(i)
                Tbin(p.X, p.Y) = 1 '字元輪廓點
            Next
        Next
    Next
    PictureBox1.Image = BWImg(Tbin)
End Sub
```

其中 AlignTgs 副程式需要加入左右排序的功能，當我們選擇群組時，是依據目標的對比度挑選的，但挑選出可能是車牌的字元後，那些目標應該以空間上從左到右的順序排列，才能依序組織出合理的車牌，程式碼如下：

```
'找車牌字元目標群組
Private Function AlignTgs(ByVal C As ArrayList) As ArrayList
    Dim R As New ArrayList, pmx As Integer = 0 '最佳目標組合與最佳度比度
    For i As Integer = 0 To C.Count - 1
        Dim T As TgInfo = C(i) '核心目標
        Dim D As New ArrayList, Dm As Integer = 0 '此輪搜尋的目標集合
```

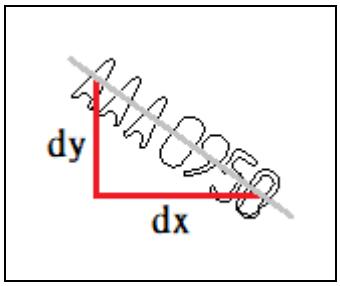
```
            D.Add(T) : Dm = T.pm '加入搜尋起點目標
            Dim x1 As Integer = T.cx - T.height * 2.5, x2 As Integer = T.cx + T.height * 2.5 '搜尋 X 範圍
            Dim y1 As Integer = T.cy - T.height * 1.5, y2 As Integer = T.cy + T.height * 1.5 '搜尋 Y 範圍
            For j As Integer = 0 To C.Count - 1
                If i = j Then Continue For '與起點重複略過
                Dim G As TgInfo = C(j)
                If G.cx < x1 Then Continue For
                If G.cx > x2 Then Continue For
                If G.cy < y1 Then Continue For
                If G.cy > y2 Then Continue For
                If G.width > T.height Then Continue For '目標寬度太大略過
                If G.height > T.height * 1.5 Then Continue For '目標高度太大略過
                D.Add(G) : Dm += G.pm '合格目標加入集合
                If D.Count >= 7 Then Exit For '目標蒐集個數已滿跳離迴圈
            Next
            If Dm > pmx Then '對比度高於之前的目標集合
                pmx = Dm : R = D
            End If
        Next
        '目標群位置左右排序
        If R.Count > 1 Then
            Dim n As Integer = R.Count
            For i As Integer = 0 To n - 2
                For j As Integer = i + 1 To n - 1
                    Dim Ti As TgInfo = R(i), Tj As TgInfo = R(j)
                    If Ti.cx > Tj.cx Then
                        R(i) = Tj : R(j) = Ti
                    End If
                Next
            Next
        End If
        Return R
    End Function
```

執行此功能結果如下：




這段程式比第 5 幾章的 Align 多搭載了一個**計算車牌傾斜度 inc** 的功能，作法是找到找到最左與最右目標的中心點，算出它們的垂直變化量 **dy**，與水平變化量 **dx**，再以反三角函數計算，稍後我們要將字元轉正時，就是依據這個參數了！因為需要重複使

用，所以 inc 被定義為全域變數。示意圖如下：

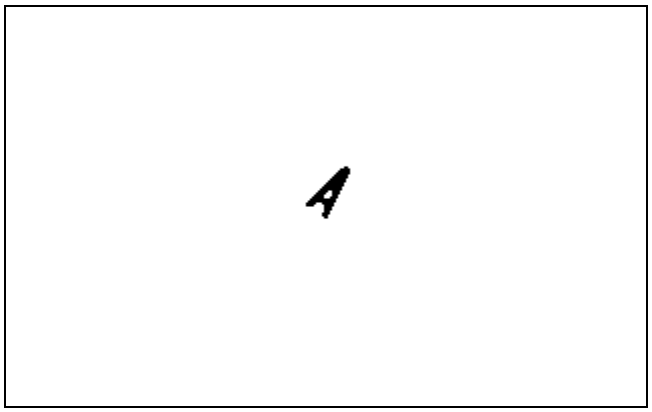



## 7-2 建立單一目標之二值化圖

因為我們是以單一字元目標為處理標的，所以先在 PicturBox1_MouseDown 事件中寫程式做選取動作，也順勢產生一個**只有該字元**的二值化矩陣，作為後續旋轉目標的基礎，程式碼與點選 A 字元後的結果如下：

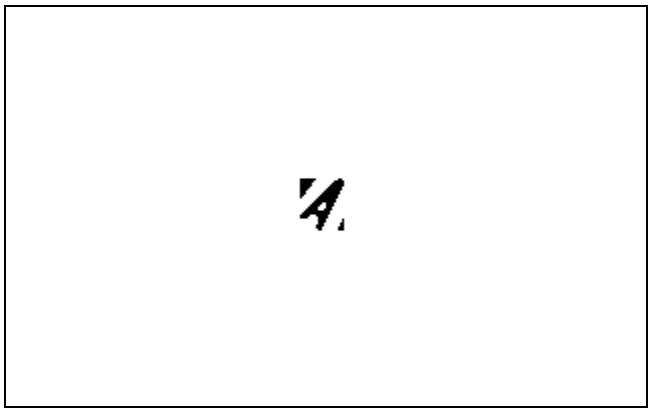
```
'點選目標
Private Sub PictureBox1_MouseDown(ByVal sender As Object, ByVal e As MouseEventArgs) _
    Handles PictureBox1.MouseDown
    If e.Button = MouseButtons.Left Then
        Dim m As Integer = -1
        For k As Integer = 0 To C.Count - 1
            Dim T As TgInfo = C(k)
            If e.X < T.xmn Then Continue For
            If e.X > T.xmx Then Continue For
            If e.Y < T.ymn Then Continue For
            If e.Y > T.ymx Then Continue For
            m = k '被點選目標
            Exit For
        Next
        If m >= 0 Then '有選取目標時
            Tsel = C(m) '點選之目標
            ReDim Tbin(nx - 1, ny - 1) '選取目標的二值化陣列
            For k As Integer = 0 To Tsel.P.Count - 1
                Dim p As Point = Tsel.P(k)
                Tbin(p.X, p.Y) = 1 '起點
                '向右連通成實心影像
                Dim i As Integer = p.X + 1
                Do While Z(i, p.Y) = 1
                    Tbin(i, p.Y) = 1
                    i += 1
                Loop
                '向左連通成實心影像
                i = p.X - 1
                Do While Z(i, p.Y) = 1
                    Tbin(i, p.Y) = 1
```

```
                    i -= 1
                Loop
            Next
            PictureBox1.Image = BWImg(Tbin) '繪製二值化圖
        End If
    End If
End Sub
```



在此我們以輪廓點為基礎，再次將目標填滿為實心圖案，這是必須的！因為比對字模時還是必須以實心目標為基礎。那為何不用簡單一點的邏輯？就是**目標物件範圍內的所有黑點**呢？我們可以試試看改用以下程式碼建立二值化圖形：

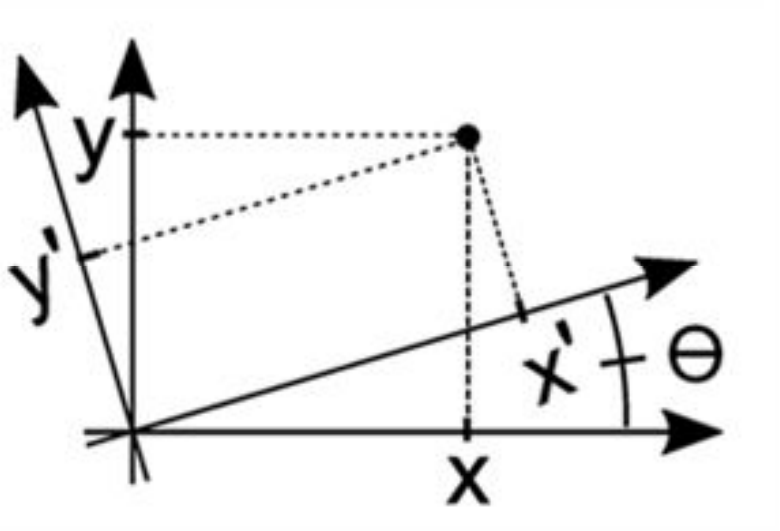
```
For i As Integer = Tsel.xmn To Tsel.xmx
    For j As Integer = Tsel.ymn To Tsel.ymx
        Tbin(i, j) = Z(i, j)
    Next
Next
```



結果會變成這樣，就是並非屬於此目標的黑色區域也會出現，因為目標的矩形範圍內不保證都是此目標的資料，尤其是在實際目標傾斜時鄰近目標交疊的狀況會更加嚴重，這會讓我們後續的處理程序變得很困難。所以以本目標的輪廓點為起點，左右延伸到連續黑點的盡頭，就不會跨過目標實際的邊界，找到其他目標的黑點了！

**7-3 旋轉目標**

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$




接下來我們要將傾倒的 A 字元目標扶正，基本上我們是用如上的旋轉座標公式計算的，但是因為影像座標的 Y 軸是以下為正，與一般直角座標以上為正相反，所以實際的程式碼會做一些調整，旋轉原點變成目標的左下角而非左上角等等，程式碼如下：

```
'旋轉目標
Private Sub RotataToolStripMenuItem_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles RotateToolStripMenuItem.Click
    Tbin = RotateTg(Tbin, Tsel, Inc) '旋轉目標二值化影像
    PictureBox1.Image = BWImg(Tbin) '繪製轉正後之影像
End Sub
'將單一目標轉正
Private Function RotateTg(ByVal b(,) As Byte, ByRef T As TgInfo, ByVal A As Double) As Byte(,)
    If A = 0 Then Return b '無傾斜不須旋轉
    If A > 0 Then A = -A '順或逆時針傾斜時需要旋轉方向相反，經過推導 A 應該永遠為負值
    Dim R(1, 1) As Double '旋轉矩陣
    R(0, 0) = Math.Cos(A) : R(0, 1) = Math.Sin(A)
    R(1, 0) = -R(0, 1) : R(1, 1) = R(0, 0)
    Dim x0 As Integer = T.xmn, y0 As Integer = T.ymx '左下角座標
    '旋轉後之目標範圍
    Dim xmn As Integer = nx, xmx As Integer = 0, ymn As Integer = ny, ymx As Integer = 0
    For i As Integer = T.xmn To T.xmx
        For j As Integer = T.ymn To T.ymx
            If b(i, j) = 0 Then Continue For '空點無須旋轉
            Dim x As Integer = i - x0, y As Integer = y0 - j '轉換螢幕座標為直角座標
            Dim xx As Integer = x * R(0, 0) + y * R(0, 1) + x0 '旋轉後 X 座標
            If xx < 1 Or xx > nx - 2 Then Continue For '邊界淨空
```

```
                Dim yy As Integer = y0 - (x * R(1, 0) + y * R(1, 1)) '旋轉後 Y 座標
                If yy < 1 Or yy > ny - 2 Then Continue For '邊界淨空
                b(i, j) = 0 '清除舊點
                b(xx, yy) = 1 '繪製新點
                '旋轉後目標的範圍偵測
                If xx < xmn Then xmn = xx
                If xx > xmx Then xmx = xx
                If yy < ymn Then ymn = yy
                If yy > ymx Then ymx = yy
            Next
        Next
        '重設目標屬性
        T.xmn = xmn : T.xmx = xmx : T.ymn = ymn : T.ymx = ymx
        T.width = T.xmx - T.xmn + 1 : T.height = T.ymx - T.ymn + 1
        T.cx = (T.xmx + T.xmn) / 2 : T.cy = (T.ymx + T.ymn) / 2
        Return b
    End Function
```

在此要注意的是旋轉後的點有可能超出原圖的範圍，所以必須做檢查，如果超出原圖就直接忽略即可。還有必須重新設定旋轉之後的目標(TgInfo)邊界，因為轉正之後這些目標的寬高都變了，下一步驟的正規化時又還要使用這些新資訊，不修正是不行的！可惜這樣做之後的結果還是怪怪的？角度是轉正了，但不是我們預期的實心狀態，字元內部好多洞，放大看是這個樣子：

這是因為做旋轉運算時浮點數被**強制取整數**產生的數位化誤差所致，還好這些缺口一定都是很獨孤立零散的狀況，請在上述副程式的最後加上一段補洞程式即可：

```
        '補足因為旋轉運算實產生的數位化誤差造成的資料空點
        For i As Integer = T.xmn To T.xmx
            For j As Integer = T.ymn To T.ymx
                If b(i, j) = 1 Then Continue For
                If b(i + 1, j) + b(i - 1, j) + b(i, j + 1) + b(i, j - 1) >= 3 Then
                    b(i, j) = 1
                End If
            Next
```

```
    Next
```

補洞之後 A 就變成這樣了：

## 7-4 寬高正規化

現在我們有了一個大致上水平的字元目標了，只要縮放到寬高與字模寬高一致就好了！此時的縮放方式寬與高的縮放程度不必一樣，如果車牌是側視拍照的話，多半會比標準的寬高比 1:2 要窄一點，強制寬高等比例縮放時，字元就會偏瘦了！Normalize 按鍵的程式碼，與執行結果如下：

```
'字元目標正規化到字模寬高
Private Sub NormalizeToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles NormalizeToolStripMenuItem.Click
    Dim fx As Double = Tsel.width / Pw, fy As Double = Tsel.height / Ph
    Dim V(Pw - 1, Ph - 1) As Byte
    For i As Integer = 0 To Pw - 1
        Dim x As Integer = Tsel.xmn + i * fx
        For j As Integer = 0 To Ph - 1
            Dim y As Integer = Tsel.ymn + j * fy
            V(i, j) = Tbin(x, y)
        Next
    Next
    PictureBox1.Image = BWImg(V)
End Sub
```

現在得到的二值化陣列 V 就是可以正確與字模比對的矩陣了，我們在下一章會繼續介紹如何比對的過程，以及一些必須處理的問題。在此或許已經有人想到，如果是字元”1”或”I”會不會有問題？不限寬高比的縮放不就會變成一個實心方塊了嗎？是的！


如上的影像中車牌的”1”字元會變成這樣：

要解決這個問題就必須對**已經找到的車牌字元作一個資料統計**，理論上同一車牌上的車牌字元都應該一樣高，寬度除了”1”與”I”之外也都應該一樣寬，那合理的寬與高應該是多少？這就不是只看單一字元可以確認的資訊了！下一節就來介紹一個完整呈現整個車牌正規化結果的程序 Correct All，基本上就是引進「**同一車牌**」的字元大小應該有一致性的合理假設！

**7-5 完整車牌正規化的呈現**

最終的 Correct All 按鍵功能，事實上是將 Align 之後的旋轉與縮放程序乃至參數，統一用在每一個目標上，再將所有結果整合呈現為一個完整的車牌。首先是一一旋轉各個目標，我們將之前建立單一目標二值化圖的程式寫成 Tg2Bin 副程式，並將旋轉目標的程式寫成 RotateTg 副程式。

如前節所述，為了避免，因為字寬不同造成的誤差，我們將旋轉後的目標寬度與高度排序，取**第二大**的字寬與字高作為原始字元目標的標準大小 mw 與 mh。為何是第二大？原因是想避開可能的兩字沾連意外，然後取較大的寬高作基準，可以盡量涵蓋每個目標的實際大小。接著就是使用統一的寬高 mw 與 mh 去作個別字元的正規化縮放，並將每一個字以 4 畫素的間隔分開，並排呈現為一個類似車牌的樣貌！完整程式碼如下：

```
'完整處理
Private Sub CorrectAllToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles CorrectAllToolStripMenuItem.Click
    Dim n As Integer = C.Count '目標總數
    '旋轉所有目標
    Dim T(n - 1) As TgInfo, M(n - 1) As Array, w(n - 1) As Integer, h(n - 1) As Integer
    For k As Integer = 0 To n - 1
        Dim G As TgInfo = C(k)
        M(k) = Tg2Bin(G) '建立單一目標的二值化矩陣
        M(k) = RotateTg(M(k), G, Inc) '旋轉目標
        T(k) = G '儲存旋轉後的目標物件
        w(k) = G.width '寬度陣列
        h(k) = G.height '高度陣列
    Next
    Array.Sort(w) '寬度排序小到大
    Array.Sort(h) '高度排序小到大
    mw = w(n - 2) '取第二寬的目標為標準，避開意外沾連的極端目標
    mh = h(n - 2) '取第二高的目標為標準，避開意外沾連的極端目標
    '車牌全圖矩陣，字元間隔 4 畫素
    Dim R((Pw + 4) * n, Ph - 1) As Byte
    ReDim MC(n - 1)
    For k As Integer = 0 To n - 1
        MC(k) = NmBin(T(k), M(k), mw, mh) '個別字元正規化矩陣
        Dim xs As Integer = (Pw + 4) * k 'X 偏移量
        For i As Integer = 0 To Pw - 1
            For j As Integer = 0 To Ph - 1
                R(xs + i, j) = MC(k)(i, j)
            Next
        Next
    Next
    PictureBox1.Image = BWImg(R) '顯示正規化之後的車牌
End Sub
```

Tg2Bin 副程式的程式碼如下：

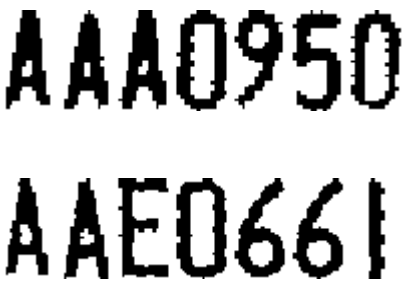
```
'建立單一目標的二值化矩陣
Private Function Tg2Bin(ByVal T As TgInfo) As Byte(,)
    Dim b(nx - 1, ny - 1) As Byte '二值化陣列
    For k As Integer = 0 To T.P.Count - 1
        Dim p As Point = T.P(k)
        b(p.X, p.Y) = 1 '起點
        '向右連通成實心影像
        Dim i As Integer = p.X + 1
        Do While Z(i, p.Y) = 1
            b(i, p.Y) = 1
```

```
            i += 1
        Loop
        '向左連通成實心影像
        i = p.X - 1
        Do While Z(i, p.Y) = 1
            b(i, p.Y) = 1
            i -= 1
        Loop
    Next
    Return b
End Function
```

NmBin 副程式的程式碼如下，比較特殊的是如果碰到特別窄的 1 或 I 時，目標必須平移置中，因為我們設計字模時，1 或 I 的筆劃都是在中間的！平移量就是標準寬度 mw 減去窄目標寬度 T.width 的一半！

```
'建立正規化目標二值化陣列
Private Function NmBin(ByVal T As TgInfo, ByVal M(,) As Byte,
                       ByVal mw As Integer, ByVal mh As Integer) As Byte(,)
    Dim fx As Double = mw / Pw, fy As Double = mh / Ph
    Dim V(Pw - 1, Ph - 1) As Byte
    For i As Integer = 0 To Pw - 1
        Dim sx As Integer = 0 '過窄字元的平移量，預設不平移
        If T.width / mw < 0.75 Then '過窄字元，可能為 1 或 I
            sx = (mw - T.width) / 2 '平移寬度差之一半
        End If
        Dim x As Integer = T.xmn + i * fx - sx
        If x < 0 Or x > nx - 1 Then Continue For
        For j As Integer = 0 To Ph - 1
            Dim y As Integer = T.ymn + j * fy
            V(i, j) = M(x, y)
        Next
    Next
    Return V
End Function
```

執行時請注意，讀入圖檔後還是要先執行 Align 的功能，建立車牌原始目標群組，但接下來不必選特定單一字元目標了！直接按 Correct All 按鍵即刻看到如下結果：



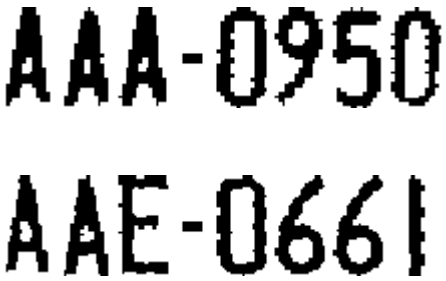
## 7-6 如何辨識車牌的分隔短線？

台灣車牌都有一個分隔短線，譬如「ABC-1234」或「AB-5678」等等，很多車牌辨識系統會直接忽略這個短線，只辨識出字元，這會變成車牌辨識系統的一個缺點！因為它們在閱讀上是個輔助，也是一個**驗證車牌格式是否正確的重要參考資訊**！但是因為那個短線目標相對於字元實在太小，我們通常在目標大小篩選時就會將它排除了。但我們不必真的辨識到它的存在，只要精確計算相鄰的車牌字元間的間距，相距明顯較遠的兩字之間，就可以判斷有格線了！

這個功能我們用一個 Add Dash 的功能展示，首先是計算字元間的**中心點**間距找到最大間距的位置，記得必須使用已經左右排序好的目標，而且要算各字元中心點的**平面**距離，如果只用 X 或 Y 軸距離計算誤差出錯的機率較大，尤其是碰到傾斜的車牌時。接著就是根據上節作好的各字元正規化陣列重繪車牌，並在**空格最大處**用程式畫出短線即可！程式碼與執行結果如下：

```
'加隔線
Private Sub AddDashToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles AddDashToolStripMenuItem.Click
    Dim n As Integer = C.Count
    '計算最大字元間距
    Dim dmx As Integer = 0, mi As Integer = 0
    For i As Integer = 0 To n - 2
        Dim d As Integer = (C(i + 1).cx - C(i).cx) ^ 2 + (C(i + 1).cy - C(i).cy) ^ 2
        If d > dmx Then
            dmx = d
            mi = i
        End If
    Next
    '繪製含隔線車牌
    Dim R((Pw + 4) * n + 20, Ph) As Byte
    For k As Integer = 0 To n - 1
        Dim xs As Integer = (Pw + 4) * k
        If k > mi Then xs += 20
        For i As Integer = 0 To Pw - 1
            For j As Integer = 0 To Ph - 1
                R(xs + i, j) = MC(k)(i, j)
            Next
        Next
        If k = mi Then '隔線
            xs += Pw + 2
            For i As Integer = 5 To 14
                For j As Integer = 23 To 27
                    R(xs + i, j) = 1
                Next
```

```
                Next
            End If
        Next
        PictureBox1.Image = BWImg(R)
    End Sub
```



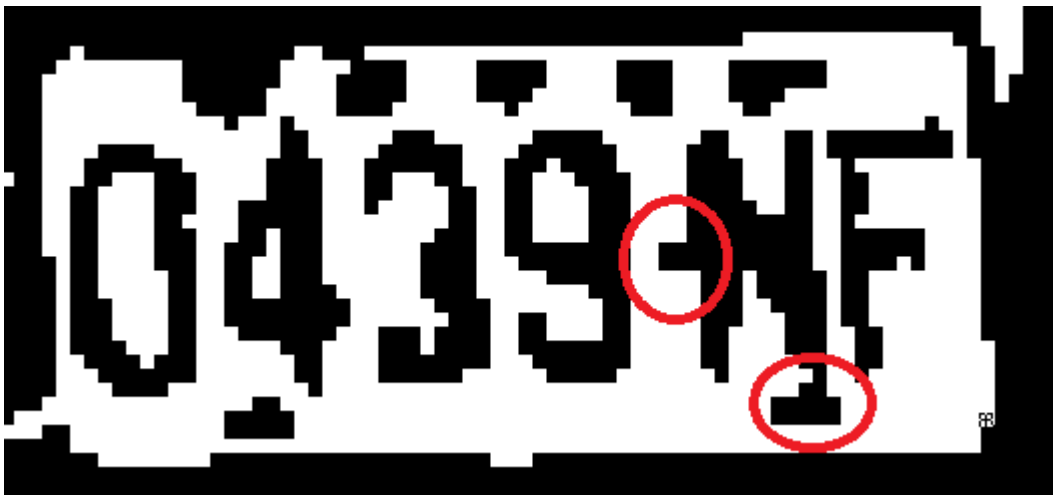




## 7-7 革命尚未成功，同志仍須努力

做到此處應該已經讓你信心大增了吧？但是要做出一個夠聰明的車牌辨識軟體，聰明到可以真正變成商業等級的產品，距離還是相當遙遠的！如果是在一般的停車場使用，各位讀者確實可以在詳讀本書後就做出很接近商業等級的辨識軟體，但如果希望在較為模糊困難略有雜訊(車牌汙損失焦之類)的狀態下，還是可以「猜對」，需要處理的例外狀況實在太多了！

最常見的是六碼字元的間距很小，影像畫素略低或微微失焦時，字元沾連在一起，兩字(甚至三字)變成一個目標，或車牌邊緣的字元與背景沾連，或者短隔線與相鄰字元沾連，字元長瘤突出一塊就讓字模比對失敗等等，這些常見錯誤都必須要有針對性的例外處理能力，否則你的產品就只是廉價品了！給大家幾張範例體會一下，怎麼處理就留給大家思考解題了！


0439NF

## 完整專案

```
Public Class Form1
    Dim B(,) As Byte '灰階陣列
    Dim Z(,) As Byte '全圖二值化陣列
    Dim Q(,) As Byte '輪廓線陣列
    Dim Gdim As Integer = 40 '計算區域亮度區塊的寬與高
    Dim Th(,) As Integer '每一區塊的平均亮度，二值化門檻值
    Dim C As ArrayList '目標物件集合
    Dim Tsel As TgInfo, Tbin(,) As Byte '選擇處理之目標與其二值化陣列
    Dim Pw As Integer = 25, Ph As Integer = 50 '標準字模影像之寬與高
    Dim mw As Integer, mh As Integer '標準字元目標寬高
    Dim MC() As Array '正規化完成後的字元二值化陣列
    Dim Inc As Double '車牌傾斜角度(>0 為順時針傾斜)
    '目標物件結構
    Public Structure TgInfo
        Dim np As Integer '目標點數
        Dim P As ArrayList '目標點的集合
        Dim xmn As Short, xmx As Short, ymn As Short, ymx As Short '四面座標極值
        Dim cx As Integer, cy As Integer '目標中心點座標
        Dim width As Integer, height As Integer '寬與高
        Dim pm As Integer '目標與背景的對比強度
        Dim ID As Integer '目標依據對比度的排序
    End Structure
    '開啟檔案
    Private Sub OpenToolStripMenuItem_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
        Handles OpenToolStripMenuItem.Click
        If OpenFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            Dim bmp As New Bitmap(OpenFileDialog1.FileName)
            Bmp2RGB(bmp) '擷取影像資訊
            B = Gv '以綠光為灰階
            PictureBox1.Image = bmp '顯示
        End If
    End Sub
    '找車牌字元目標群組
    Private Sub AlignToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles AlignToolStripMenuItem.Click
        Z = DoBinary(B) '二值化
        Q = Outline(Z) '建立輪廓點陣列
        C = getTargets(Q) '建立目標物件集合
        C = AlignTgs(C) '找到最多七個的字元目標
        Dim n As Integer = C.Count
        '末字中心點與首字中心點的偏移量，斜率計算參數
        Dim dx As Integer = C(n - 1).cx - C(0).cx
        Dim dy As Integer = C(n - 1).cy - C(0).cy
        Inc = Math.Atan2(dy, dx) '字元排列傾角
        '繪製字元輪廓
        ReDim Tbin(nx - 1, ny - 1)
        For k As Integer = 0 To C.Count - 1
            Dim T As TgInfo = C(k)
            For m As Integer = 0 To T.P.Count - 1
                Dim pt As Point = T.P(m)
                For i As Integer = 0 To T.P.Count - 1
                    Dim p As Point = T.P(i)
                    Tbin(p.X, p.Y) = 1 '字元輪廓點
                Next
            Next
```

```
    Next
    PictureBox1.Image = BWImg(Tbin)
End Sub
'點選目標
Private Sub PictureBox1_MouseDown(ByVal sender As Object, ByVal e As MouseEventArgs) _
    Handles PictureBox1.MouseDown
    If e.Button = MouseButtons.Left Then
        Dim m As Integer = -1
        For k As Integer = 0 To C.Count - 1
            Dim T As TgInfo = C(k)
            If e.X < T.xmn Then Continue For
            If e.X > T.xmx Then Continue For
            If e.Y < T.ymn Then Continue For
            If e.Y > T.ymx Then Continue For
            m = k '被點選目標
            Exit For
        Next
        If m >= 0 Then '有選取目標時
            Tsel = C(m) '點選之目標
            ReDim Tbin(nx - 1, ny - 1) '選取目標的二值化陣列
            For k As Integer = 0 To Tsel.P.Count - 1
                Dim p As Point = Tsel.P(k)
                Tbin(p.X, p.Y) = 1 '起點
                '向右連通成實心影像
                Dim i As Integer = p.X + 1
                Do While Z(i, p.Y) = 1
                    Tbin(i, p.Y) = 1
                    i += 1
                Loop
                '向左連通成實心影像
                i = p.X - 1
                Do While Z(i, p.Y) = 1
                    Tbin(i, p.Y) = 1
                    i -= 1
                Loop
            Next
            PictureBox1.Image = BWImg(Tbin) '繪製二值化圖
        End If
    End If
End Sub
'找車牌字元目標群組
Private Function AlignTgs(ByVal C As ArrayList) As ArrayList
    Dim R As New ArrayList, pmx As Integer = 0 '最佳目標組合與最佳度比度
    For i As Integer = 0 To C.Count - 1
        Dim T As TgInfo = C(i) '核心目標
        Dim D As New ArrayList, Dm As Integer = 0 '此輪搜尋的目標集合
        D.Add(T) : Dm = T.pm '加入搜尋起點目標
        Dim x1 As Integer = T.cx - T.height * 2.5, x2 As Integer = T.cx + T.height * 2.5 '搜尋 X 範圍
        Dim y1 As Integer = T.cy - T.height * 1.5, y2 As Integer = T.cy + T.height * 1.5 '搜尋 Y 範圍
        For j As Integer = 0 To C.Count - 1
            If i = j Then Continue For '與起點重複略過
            Dim G As TgInfo = C(j)
            If G.cx < x1 Then Continue For
            If G.cx > x2 Then Continue For
            If G.cy < y1 Then Continue For
            If G.cy > y2 Then Continue For
            If G.width > T.height Then Continue For '目標寬度太大略過
            If G.height > T.height * 1.5 Then Continue For '目標高度太大略過
```

```
                D.Add(G) : Dm += G.pm '合格目標加入集合
                If D.Count >= 7 Then Exit For '目標蒐集個數已滿跳離迴圈
            Next
            If Dm > pmx Then '對比度高於之前的目標集合
                pmx = Dm : R = D
            End If
        Next
        '目標群位置左右排序
        If R.Count > 1 Then
            Dim n As Integer = R.Count
            For i As Integer = 0 To n - 2
                For j As Integer = i + 1 To n - 1
                    Dim Ti As TgInfo = R(i), Tj As TgInfo = R(j)
                    If Ti.cx > Tj.cx Then
                        R(i) = Tj : R(j) = Ti
                    End If
                Next
            Next
        End If
        Return R
    End Function
    '旋轉目標
    Private Sub RotataToolStripMenuItem_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
        Handles RotateToolStripMenuItem.Click
        Tbin = RotateTg(Tbin, Tsel, Inc) '旋轉目標二值化影像
        PictureBox1.Image = BWImg(Tbin) '繪製轉正後之影像
    End Sub
    '將單一目標轉正
    Private Function RotateTg(ByVal b(,) As Byte, ByRef T As TgInfo, ByVal A As Double) As Byte(,)
        If A = 0 Then Return b '無傾斜不須旋轉
        If A > 0 Then A = -A '順或逆時針傾斜時需要旋轉方向相反，經過推導 A 應該永遠為負值
        Dim R(1, 1) As Double '旋轉矩陣
        R(0, 0) = Math.Cos(A) : R(0, 1) = Math.Sin(A)
        R(1, 0) = -R(0, 1) : R(1, 1) = R(0, 0)
        Dim x0 As Integer = T.xmn, y0 As Integer = T.ymx '左下角座標
        '旋轉後之目標範圍
        Dim xmn As Integer = nx, xmx As Integer = 0, ymn As Integer = ny, ymx As Integer = 0
        For i As Integer = T.xmn To T.xmx
            For j As Integer = T.ymn To T.ymx
                If b(i, j) = 0 Then Continue For '空點無須旋轉
                Dim x As Integer = i - x0, y As Integer = y0 - j '轉換螢幕座標為直角座標
                Dim xx As Integer = x * R(0, 0) + y * R(0, 1) + x0 '旋轉後 X 座標
                If xx < 1 Or xx > nx - 2 Then Continue For '邊界淨空
                Dim yy As Integer = y0 - (x * R(1, 0) + y * R(1, 1)) '旋轉後 Y 座標
                If yy < 1 Or yy > ny - 2 Then Continue For '邊界淨空
                b(i, j) = 0 '清除舊點
                b(xx, yy) = 1 '繪製新點
                '旋轉後目標的範圍偵測
                If xx < xmn Then xmn = xx
                If xx > xmx Then xmx = xx
                If yy < ymn Then ymn = yy
                If yy > ymx Then ymx = yy
            Next
        Next
        '重設目標屬性
        T.xmn = xmn : T.xmx = xmx : T.ymn = ymn : T.ymx = ymx
        T.width = T.xmx - T.xmn + 1 : T.height = T.ymx - T.ymn + 1
        T.cx = (T.xmx + T.xmn) / 2 : T.cy = (T.ymx + T.ymn) / 2
```

```
        '補足因為旋轉運算實產生的數位化誤差造成的資料空點
        For i As Integer = T.xmn To T.xmx
            For j As Integer = T.ymn To T.ymx
                If b(i, j) = 1 Then Continue For
                If b(i + 1, j) + b(i - 1, j) + b(i, j + 1) + b(i, j - 1) >= 3 Then
                    b(i, j) = 1
                End If
            Next
        Next
        Return b
    End Function
    '字元目標正規化到字模寬高
    Private Sub NormalizeToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles NormalizeToolStripMenuItem.Click
        Dim fx As Double = Tsel.width / Pw, fy As Double = Tsel.height / Ph
        Dim V(Pw - 1, Ph - 1) As Byte
        For i As Integer = 0 To Pw - 1
            Dim x As Integer = Tsel.xmn + i * fx
            For j As Integer = 0 To Ph - 1
                Dim y As Integer = Tsel.ymn + j * fy
                V(i, j) = Tbin(x, y)
            Next
        Next
        PictureBox1.Image = BWImg(V)
    End Sub
    '二值化
    Private Function DoBinary(ByVal b(,) As Byte) As Byte(,)
        Th = ThresholdBuild(b) '建立二值化使用之門檻值陣列
        Dim Z(nx - 1, ny - 1) As Byte '建立二值化陣列
        For i As Integer = 1 To nx - 2
            Dim x As Integer = i \ Gdim 'x 座標換算
            For j As Integer = 1 To ny - 2
                Dim y As Integer = j \ Gdim 'y 座標換算
                If b(i, j) < Th(x, y) Then
                    Z(i, j) = 1 '低於亮度門檻設為目標點
                End If
            Next
        Next
        Return Z
    End Function
    '門檻值陣列建立
    Private Function ThresholdBuild(ByVal b(,) As Byte) As Integer(,)
        Dim kx As Integer = nx \ Gdim, ky As Integer = ny \ Gdim
        Dim T(kx, ky) As Integer
        '累計各區塊亮度值總和
        For i As Integer = 0 To nx - 1
            Dim x As Integer = i \ Gdim
            For j As Integer = 0 To ny - 1
                Dim y As Integer = j \ Gdim
                T(x, y) += b(i, j) '亮度值累加
            Next
        Next
        '區塊亮度平均值計算
        For i As Integer = 0 To kx - 1
            For j As Integer = 0 To ky - 1
                T(i, j) /= Gdim * Gdim
            Next
        Next
```

```
    Return T
End Function
'建立輪廓點陣列
Private Function Outline(ByVal b(,) As Byte) As Byte(,)
    Dim Q(nx - 1, ny - 1) As Byte '輪廓點陣列
    For i As Integer = 1 To nx - 2
        For j As Integer = 1 + 1 To ny - 2
            If b(i, j) = 0 Then Continue For '非輪廓點忽略
            If b(i, j - 1) = 0 Then Q(i, j) = 1 : Continue For '確認為輪廓點
            If b(i - 1, j) = 0 Then Q(i, j) = 1 : Continue For '確認為輪廓點
            If b(i + 1, j) = 0 Then Q(i, j) = 1 : Continue For '確認為輪廓點
            If b(i, j + 1) = 0 Then Q(i, j) = 1 '確認為輪廓點
        Next
    Next
    Return Q
End Function
'以輪廓點建立目標陣列，排除負目標
Dim minHeight As Integer = 20, maxHeight As Integer = 80 '有效目標高度範圍
Dim minwidth As Integer = 2, maxWidth As Integer = 80 '有效目標寬度範圍
Dim Tgmax As Integer = 20 '進入決選範圍的最明顯目標上限
Function getTargets(ByVal q(,) As Byte) As ArrayList
    Dim A As New ArrayList
    Dim b(,) As Byte = q.Clone '建立輪廓點陣列副本
    For i As Integer = 1 To nx - 2
        For j As Integer = 1 To ny - 2
            If b(i, j) = 0 Then Continue For
            Dim G As New TgInfo
            G.xmn = i : G.xmx = i : G.ymn = j : G.ymx = j : G.P = New ArrayList
            Dim nc As New ArrayList '每一輪搜尋的起點集合
            nc.Add(New Point(i, j)) '輸入之搜尋起點
            G.P.Add(New Point(i, j))
            b(i, j) = 0 '清除此起點之輪廓點標記
            Do
                Dim nb As ArrayList = nc.Clone '複製此輪之搜尋起點集合
                nc = New ArrayList '清除準備蒐集下一輪搜尋起點之集合
                For m As Integer = 0 To nb.Count - 1
                    Dim p As Point = nb(m) '搜尋起點
                    '在此點周邊 3X3 區域內找輪廓點
                    For ii As Integer = p.X - 1 To p.X + 1
                        For jj As Integer = p.Y - 1 To p.Y + 1
                            If b(ii, jj) = 0 Then Continue For '非輪廓點忽略
                            Dim k As New Point(ii, jj) '建立點物件
                            nc.Add(k) '本輪搜尋新增的輪廓點
                            G.P.Add(k) '點集合
                            If ii < G.xmn Then G.xmn = ii
                            If ii > G.xmx Then G.xmx = ii
                            If jj < G.ymn Then G.ymn = jj
                            If jj > G.ymx Then G.ymx = jj
                            b(ii, jj) = 0 '清除輪廓點點標記
                        Next
                    Next
                Next
            Loop While nc.Count > 0 '此輪搜尋有新發現輪廓點時繼續搜尋
            If Z(i - 1, j) = 1 Then Continue For '排除白色區塊的負目標，起點左邊是黑點
            G.width = G.xmx - G.xmn + 1 '寬度計算
            G.height = G.ymx - G.ymn + 1 '高度計算
            '以寬高大小篩選目標
            If G.height < minHeight Then Continue For
```

```
                If G.height > maxHeight Then Continue For
                If G.width < minwidth Then Continue For
                If G.width > maxWidth Then Continue For
                G.cx = (G.xmn + G.xmx) / 2 : G.cy = (G.ymn + G.ymx) / 2 '中心點
                G.np = G.P.Count
                '計算目標的對比度
                For m As Integer = 0 To G.P.Count - 1
                    Dim pm As Integer = PointPm(G.P(m))
                    If pm > G.pm Then G.pm = pm '最高對比度的輪廓點
                Next
                A.Add(G) '加入有效目標集合
            Next
        Next
        '以對比度排序
        For i As Integer = 0 To A.Count - 2
            For j As Integer = i + 1 To A.Count - 1
                Dim T As TgInfo = A(i), G As TgInfo = A(j)
                If T.pm < G.pm Then A(i) = G : A(j) = T '互換位置，高對比目標在前
            Next
        Next
        '取得 Tgmax 個最明顯的目標輸出
        Dim C As New ArrayList
        For i As Integer = 0 To Tgmax - 1
            If i > A.Count - 1 Then Exit For '超過總目標數
            Dim T As TgInfo = A(i) : T.ID = i '建立以對比度排序的序號
            C.Add(T)
        Next
        Return C '回傳目標物件集合
    End Function
    '輪廓點與背景的對比度
    Private Function PointPm(ByVal p As Point) As Integer
        Dim x As Integer = p.X, y As Integer = p.Y
        Dim mx As Integer = 0 '周邊最亮點，依據灰階陣列 B
        If mx < B(x - 1, y) Then mx = B(x - 1, y)
        If mx < B(x + 1, y) Then mx = B(x + 1, y)
        If mx < B(x, y + 1) Then mx = B(x, y + 1)
        If mx < B(x, y - 1) Then mx = B(x, y - 1)
        Return mx - B(x, y) '最亮點與輪廓點的差值
    End Function
    '儲存目前影像
    Private Sub SaveImageToolStripMenuItem_Click(ByVal sender As Object, ByVal e As System.EventArgs)
_
        Handles SaveImageToolStripMenuItem.Click
        If SaveFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            PictureBox1.Image.Save(SaveFileDialog1.FileName)
        End If
    End Sub
    '完整處理
    Private Sub CorrectAllToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles CorrectAllToolStripMenuItem.Click
        Dim n As Integer = C.Count '目標總數
        '旋轉所有目標
        Dim T(n - 1) As TgInfo, M(n - 1) As Array, w(n - 1) As Integer, h(n - 1) As Integer
        For k As Integer = 0 To n - 1
            Dim G As TgInfo = C(k)
            M(k) = Tg2Bin(G) '建立單一目標的二值化矩陣
            M(k) = RotateTg(M(k), G, Inc) '旋轉目標
            T(k) = G '儲存旋轉後的目標物件
```

```
            w(k) = G.width '寬度陣列
            h(k) = G.height '高度陣列
        Next
        Array.Sort(w) '寬度排序小到大
        Array.Sort(h) '高度排序小到大
        mw = w(n - 2) '取第二寬的目標為標準，避開意外沾連的極端目標
        mh = h(n - 2) '取第二高的目標為標準，避開意外沾連的極端目標
        '車牌全圖矩陣，字元間隔 4 畫素
        Dim R((Pw + 4) * n, Ph - 1) As Byte
        ReDim MC(n - 1)
        For k As Integer = 0 To n - 1
            MC(k) = NmBin(T(k), M(k), mw, mh) '個別字元正規化矩陣
            Dim xs As Integer = (Pw + 4) * k 'X 偏移量
            For i As Integer = 0 To Pw - 1
                For j As Integer = 0 To Ph - 1
                    R(xs + i, j) = MC(k)(i, j)
                Next
            Next
        Next
        PictureBox1.Image = BWImg(R) '顯示正規化之後的車牌
    End Sub
    '建立單一目標的二值化矩陣
    Private Function Tg2Bin(ByVal T As TgInfo) As Byte(,)
        Dim b(nx - 1, ny - 1) As Byte '二值化陣列
        For k As Integer = 0 To T.P.Count - 1
            Dim p As Point = T.P(k)
            b(p.X, p.Y) = 1 '起點
            '向右連通成實心影像
            Dim i As Integer = p.X + 1
            Do While Z(i, p.Y) = 1
                b(i, p.Y) = 1
                i += 1
            Loop
            '向左連通成實心影像
            i = p.X - 1
            Do While Z(i, p.Y) = 1
                b(i, p.Y) = 1
                i -= 1
            Loop
        Next
        Return b
    End Function
    '建立正規化目標二值化陣列
    Private Function NmBin(ByVal T As TgInfo, ByVal M(,) As Byte,
                           ByVal mw As Integer, ByVal mh As Integer) As Byte(,)
        Dim fx As Double = mw / Pw, fy As Double = mh / Ph
        Dim V(Pw - 1, Ph - 1) As Byte
        For i As Integer = 0 To Pw - 1
            Dim sx As Integer = 0 '過窄字元的平移量，預設不平移
            If T.width / mw < 0.75 Then '過窄字元，可能為 1 或 I
                sx = (mw - T.width) / 2 '平移寬度差之一半
            End If
            Dim x As Integer = T.xmn + i * fx - sx
            If x < 0 Or x > nx - 1 Then Continue For
            For j As Integer = 0 To Ph - 1
                Dim y As Integer = T.ymn + j * fy
                V(i, j) = M(x, y)
            Next
```

```
        Next
        Return V
    End Function
    '加隔線
    Private Sub AddDashToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles AddDashToolStripMenuItem.Click
        Dim n As Integer = C.Count
        '計算最大字元間距
        Dim dmx As Integer = 0, mi As Integer = 0
        For i As Integer = 0 To n - 2
            Dim d As Integer = (C(i + 1).cx - C(i).cx) ^ 2 + (C(i + 1).cy - C(i).cy) ^ 2
            If d > dmx Then
                dmx = d
                mi = i
            End If
        Next
        '繪製含隔線車牌
        Dim R((Pw + 4) * n + 20, Ph) As Byte
        For k As Integer = 0 To n - 1
            Dim xs As Integer = (Pw + 4) * k
            If k > mi Then xs += 20
            For i As Integer = 0 To Pw - 1
                For j As Integer = 0 To Ph - 1
                    R(xs + i, j) = MC(k)(i, j)
                Next
            Next
            If k = mi Then '隔線
                xs += Pw + 2
                For i As Integer = 5 To 14
                    For j As Integer = 23 To 27
                        R(xs + i, j) = 1
                    Next
                Next
            End If
        Next
        PictureBox1.Image = BWImg(R)
    End Sub
End Class
```