第 8 章　車牌的字元辨識

8-1 匯入字模資料與建立專案

不論我們將字元目標經過影像處理計數調整到多麼端正，它們依舊還是「**影像**」資料！要讓車牌辨識產生可以使用的資料，就必須經過字元比對，讓影像資料變成文字模式的資料，這也是所有 OCR 必經的過程，這一步沒有完成的話就只是影像「**處理**」，不能稱為「**辨識**」了！

我們在第 6 章已經做好比對字模的準備工作，在此就是先複製第 7 章的專案做修改，首先是將第 6 章製作的二進位字模檔案匯入為資源檔。在表單的主功能表建立 Open, Align, Correct All 與 Recognize All 幾個按鍵，必須使用的全域變數宣告如下：

```
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 Pw As Integer = 25, Ph As Integer = 50 '字模的寬與高
Dim P(1, 35, Pw - 1, Ph - 1) As Byte '六七碼車牌所有英數字二值化陣列
Dim P69(1, Pw - 1, Ph - 1) As Byte '變形的 6 與 9
Dim MC() As Array '正規化完成後的字元二值化陣列
Dim mw As Integer, mh As Integer '標準字元目標寬高
```

```
Dim LP As String '車牌號碼
Dim Inc As Double '車牌傾斜角度(>0 為順時針傾斜)
'字元對照表
'0-9→0-9
'10→A，11→B，12→C，13→D，14→E，15→F，16→G，17→H，18→I，19→J
'20→K，21→L，22→M，23→N，24→O，25→P，26→Q，27→R，28→S，29→T
'30→U，31→V，32→W，33→X，34→Y，35→Z
Dim Ch() As Char = {"0"c, "1"c, "2"c, "3"c, "4"c, "5"c, "6"c, "7"c, "8"c, "9"c,
                    "A"c, "B"c, "C"c, "D"c, "E"c, "F"c, "G"c, "H"c, "I"c, "J"c,
                    "K"c, "L"c, "M"c, "N"c, "O"c, "P"c, "Q"c, "R"c, "S"c, "T"c,
                    "U"c, "V"c, "W"c, "X"c, "Y"c, "Z"c}
```

新增的部分主要是 0-9 與 A-Z 的字元陣列，為了方便查詢，我們也用註解方式做了一個查詢表，可以很快知道譬如 Ch(27)=R 或 Ch(17)=H 等等。另外當我們比對字元時，除了回傳最佳字元，通常也需要知道符合度，最符合的字元屬於六或七碼字型也有參考價值，所以建立了一個資料結構，每次比對完一個目標就可以把這些資訊一起封裝回傳供主程式使用。

```
'最佳字元結構
Public Structure ChInfo
    Dim Ch As Char '最符合字元
    Dim ft As Integer '符合度評分
    Dim kind As Integer '六或七碼字元，0→六碼，1→七碼
End Structure
```

主功能表中前三個功能完全與前一章的內容相同，無須修改，必須先做的是在 Form_Load 事件時匯入字模資料，程式碼如下：

```
'啟動程式載入字模
Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load
    FontLoad()
End Sub
Private Sub FontLoad()
    Dim q() As Byte = My.Resources.font '使用字模資源檔
    ReDim P(1, 35, Pw - 1, Ph - 1)
    Dim n As Integer = 0
    '匯入六與七碼字模
    For m As Integer = 0 To 1
        For k As Integer = 0 To 35
            For j As Integer = 0 To Ph - 1
                For i As Integer = 0 To Pw - 1
                    P(m, k, i, j) = q(n)
                    n += 1
                Next
            Next
```

```
            Next
        Next
        '匯入六碼變形 69 字模
        For k As Integer = 0 To 1
            For j As Integer = 0 To Ph - 1
                For i As Integer = 0 To Pw - 1
                    P69(k, i, j) = q(n)
                    n += 1
                Next
            Next
        Next
    End Sub
```

8-2 比對字元目標

比對字元的基本概念就是將一個待檢測的目標陣列，逐一與所有字模陣列比對，看哪一個字元的符合度最高？同時也用 Chinfo 資料結構記錄符合度與字型種類(六或七碼)，我們先建立一個副程式如下：

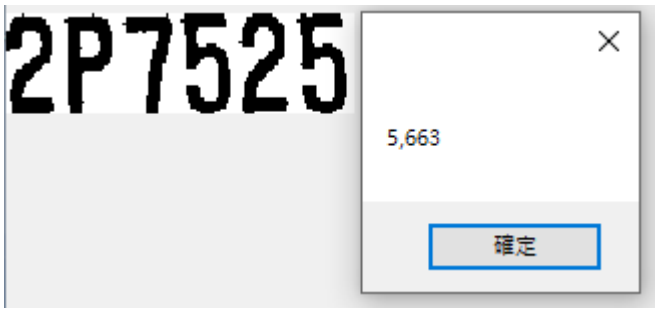
```
    '最佳字元
    Private Function BestC(ByVal A(,) As Byte) As ChInfo
        Dim C As New ChInfo
        For m As Integer = 0 To 1
            For k As Integer = 0 To 35
                Dim n0 As Integer = 0 '字模黑點數
                Dim nf As Integer = 0 '符合的黑點數
                For i As Integer = 0 To Pw - 1
                    For j As Integer = 0 To Ph - 1
                        If P(m, k, i, j) = 0 Then
                            If A(i, j) = 1 Then nf -= 1 '目標與字模符合點數
                        Else
                            n0 += 1 '字模黑點數累計
                            If A(i, j) = 1 Then nf += 1 '目標與字模符合點數
                        End If
                    Next
                Next
                Dim v As Integer = nf * 1000 / n0 '符合點數百分比
                If v > C.ft Then
                    C.ft = v
                    C.Ch = Ch(k)
                    C.kind = m
                End If
            Next
        Next
        For k As Integer = 0 To 1
            Dim n0 As Integer = 0, nf As Integer = 0
```

```
            For i As Integer = 0 To Pw - 1
                For j As Integer = 0 To Ph - 1
                    If P69(k, i, j) = 0 Then
                        If A(i, j) = 1 Then nf -= 1 '目標與字模符合點數
                    Else
                        n0 += 1 '字模黑點數累計
                        If A(i, j) = 1 Then nf += 1 '目標與字模符合點數
                    End If
                Next
            Next
            Dim v As Integer = nf * 1000 / n0 '符合點數百分比
            If v > C.ft Then
                C.ft = v
                If k = 0 Then
                    C.Ch = "6"c
                Else
                    C.Ch = "9"c
                End If
            End If
        Next
        Return C
    End Function
```

其功能就是饋入待辨識的已正規化目標陣列(MC(m))，看看是甚麼字元？符合度如何等等。我們先用本章開頭的影像為例，執行 Align 與 Correct All 之後會變成如下的影像：

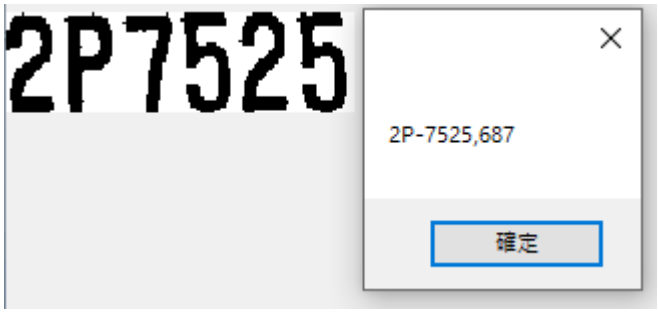
接著建立 PictureBox1_MouseDown 的事件副程式，就是點選某字之後做字元辨識，以訊息框顯示結果：

```
'點選字元
Private Sub PictureBox1_MouseDown(ByVal sender As Object, ByVal e As MouseEventArgs) _
    Handles PictureBox1.MouseDown
    If e.Button = Windows.Forms.MouseButtons.Left Then
        Dim m As Integer = e.X \ (Pw + 4)
        If m < 0 Or m > MC.Count - 1 Then Exit Sub
        Dim C As ChInfo = BestC(MC(m))
        MsgBox(C.Ch + "," + C.ft.ToString)
    End If
End Sub
```



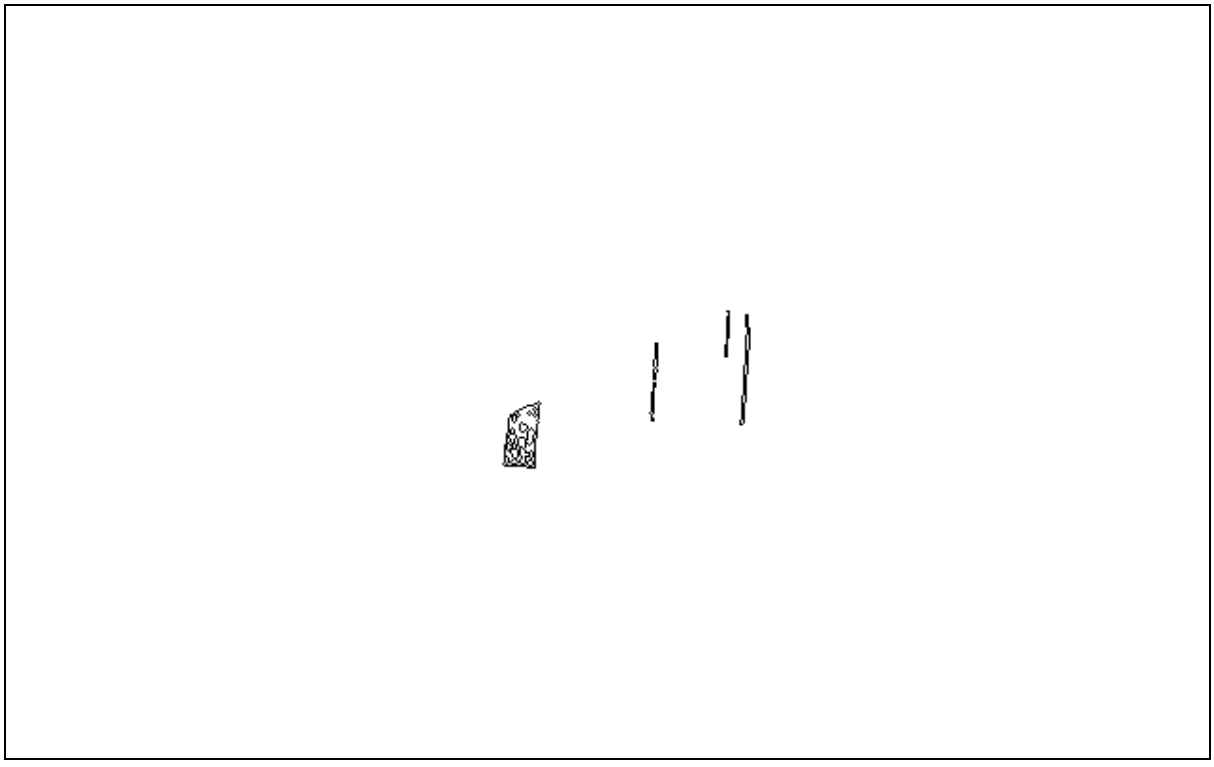
接下來是「Recognize All」按鍵，也就是辨識完整車牌的程式，此時就必須先計算車牌短隔線的位置了！如上一章的介紹，完整程式碼與執行結果如下：

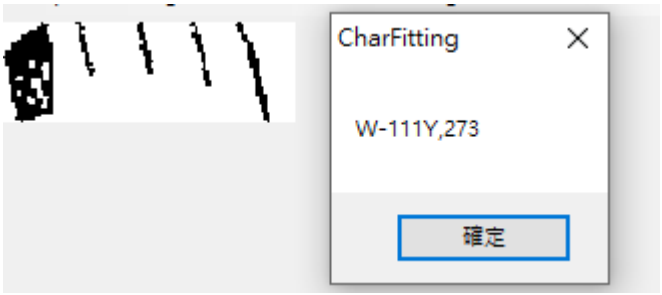
```
'辨識整個車牌
Private Sub RecognizeAllToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles RecognizeAllToolStripMenuItem.Click
    Dim n As Integer = C.Count
    '計算最大字元間距與位置
    Dim dmx As Integer = 0, mi As Integer = 0
    For i As Integer = 0 To n - 2
        Dim d As 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
    LP = ""
    Dim sc As Integer = 0 '車牌字串與符合度
    For i As Integer = 0 To MC.Count - 1
        Dim C As ChInfo = BestC(MC(i))
        LP += C.Ch
        sc += C.ft
        If i = mi Then LP += "-"c
    Next
    sc /= MC.Count
    MsgBox(LP + "," + sc.ToString)
End Sub
```



8-3 你找到的答案是對的嗎？

截至目前為止，我們都是以容易辨識的車牌影像做示範，但是實際情況不會這麼理想，我們常會誤認到不是車牌的目標，譬如告示牌上的文字，甚至背景的光影雜訊等等。如果我們可以用某些條件判斷這不是車牌，就應該直接攔截這些明顯錯誤的答案，回應主程式說「**沒有車牌**」！譬如下面這張影像以目前程式辨識結果如下：





W-111Y 是強制辨識的車牌答案，273 是符合度的分數。我們目前的符合度計算方式滿分是 1000 分，273 當然是極低的值，所以僅用此值就可以確定這個答案中的目標根本不是文字了！而且按照台灣車牌的格式規則，不可能有 W-111Y 這種車牌。第一、五碼的車牌只會有 2-2 或 3-2 的兩種字數區段格式，1-4 格式是沒有的！第二、台灣車牌中字數較長的區段一定全部都是數字，所以 111Y 是不可能的！

要實作這些檢驗機制的程式技術是很簡單的，譬如 If 分數<及格分數 Then Return False，用 IsNumeric("X")就知道 X 是不是數字，看"-"的位置就可以知道字數的區段是?-?。所以你一定可以檢驗七碼車牌的千位是不是都是英文字？後四碼是不是都是數字等等。

但是甚麼是合理的「及格分數」？這跟你設計的計分方式與影像品質都有關係，不經過大量資料的分析測試就無法得到最接近合理的數值。另一方面，車牌格式的驗證條件可達數十個之多！有些還與字元或車牌背景的顏色有關係，所以我們無法在此做出並展示完整的程式，只能點到為止！檢驗車牌是否正確的副程式可以寫成這個形式：

```
'檢驗車牌是否正確的副程式
Private Function ChkLP(ByVal S As String, ByVal V As Integer) As String
    If V < 500 Then Return "" '符合度低於及格分數
    If S.Length < 5 Then Return "" '包含分隔線在內字數小於 5
    Dim m As Integer = S.IndexOf("-") '格線位置
    If m = 1 Then Return "" '沒有 1-x 的字數區段格式
    Return S '合格車牌
End Function
```

檢驗通過就回傳原來的車牌答案，不通過的就直接變成空字串，程式碼只需插入一行：LP=ChkLP(LP，Sc)即可。這些條件要寫得完整又不互相衝突，做出相當合理的判斷，到達可以商業運轉的程度，至少會擴充到數百行之多！就留待大家各顯神通了！

前面說過台灣車牌中的格線是有實質意義的！譬如七碼車牌格線前三碼一定是英文字，而且確定不會有 I 與 O，而且所有格式中較長的區段一定全部都是數字。所以我們可以檢查答案是否合乎這些規則？不合的就是錯誤的答案，我們可以嘗試把它們改成近似的英或數字，譬如 B8 容易誤認，BC-100B 很可能是 BC-1008，此時竄改答案就很合理了！

下面是一個程式範例，可以部份實現這種因應英數字格式的強制修改：

```
'嘗試依據英數字規範修改車牌答案
Private Function ChkED(ByVal S As String) As String
    Dim C() As Char = S.ToCharArray '字串轉成字元陣列
    Dim n1 As Integer = S.IndexOf("-") '第一區段長度
    Dim n2 As Integer = C.Length - n1 - 1 '第二區段長度
    Dim d1 As Integer = 0, d2 As Integer = 0 '數字區的起終點
    If n1 > n2 Then '第一區段較長
        d1 = 0 : d2 = n1 - 1
    End If
    If n2 > n1 Then '第一區段較長
        d1 = n1 + 1 : d2 = C.Length - 1
    End If
    If d2 = 0 Then Return S '無法判定純數字區段(2-2 或 3-3)
    '嘗試將純數字區段的英文字改成數字
    For i As Integer = d1 To d2
        C(i) = E2D(C(i))
    Next
    '如果是七碼車牌，強制將前三碼中的數字改成英文
    If n1 = 3 And n2 = 4 Then
        For i As Integer = 0 To 2
            C(i) = D2E(C(i))
        Next
    End If
    S = "" '重組字串
    For i As Integer = 0 To C.Length - 1
        S += C(i)
    Next
    Return S '回傳字串
End Function
'嘗試將英文字母變成相似的數字
Private Function E2D(ByVal C As Char) As Char
    If C = "B"c Then C = "8"c
    If C = "D"c Then C = "0"c
    If C = "O"c Then C = "0"c
    Return C
End Function
'嘗試將數字變成相似的英文字母
Private Function D2E(ByVal C As Char) As Char
    If C = "8"c Then C = "B"c
    If C = "0"c Then C = "D"c
    Return C
End Function
```

請在 Form_Load 事件加入程式：MsgBox(ChkED("A8C-01D6"))，執行後即可看到：

8-4 可以補字嗎？

上面這張影像使用目前的程式結果會變成左邊少一個字，F-1965 當然不合乎台灣車牌的格式！原因應該是最左邊的 5 字有些反光之故，但少字的原因不重要，即使不反光也可能因為六碼車牌邊緣的白色淨空區極窄，讓左或右邊的字元與背景沾連切不成獨立目標。我們想問的是有可能用強制「**補字**」的方式補救嗎？

在此，我們從格式知道 F 的左邊應該還有一個字，但正常的二值化與目標切割過程無法鎖定這個字，我們要示範如何使用已有的其他字元資訊「**外插**」這個 5 字出來！如

果要做出涵蓋所有狀況，判斷何時該補字？該補哪一邊的字？等等的完整判斷程式是個很複雜的工作，在此我們只介紹如何向左方補一個字的動作，請將主功能表加一個按鍵「Left Ext.」只針對這個案例補字吧！程式碼如下：

```
'左方外插一字
Private Sub LeftExtToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles LeftExtToolStripMenuItem.Click
    '字元間的最小間距
    Dim xd As Integer = nx, yd As Integer = ny
    For i As Integer = 0 To C.Count - 2
        Dim dx As Integer = C(i + 1).cx - C(i).cx
        If dx < xd Then xd = dx
        Dim dy As Integer = C(i + 1).cy - C(i).cy
        If dy < yd Then yd = dy
    Next
    Dim G As TgInfo = C(0) '複製一個外插的目標
    '目標位置移動
    G.cx -= xd : G.cy -= yd
    G.xmn = G.cx - mw / 2 : G.xmx = G.cx + mw / 2
    G.ymn = G.cy - mh / 2 : G.ymx = G.cy + mh / 2
    '外插目標二值化陣列
    Dim A(nx - 1, ny - 1) As Byte
    For i As Integer = G.xmn To G.xmx
        For j As Integer = G.ymn To G.ymx
            A(i, j) = Z(i, j)
        Next
    Next
    A = RotateTg(A, G, Inc) '旋轉目標
    Dim D(,) As Byte = NmBin(G, A, mw, mh) '外插字元正規化
    Dim k As ChInfo = BestC(D) '辨識字元
    LP = k.Ch + LP '外插字元加入車牌
    '繪圖顯示外插目標框線
    Dim bmp As Bitmap = New Bitmap(OpenFileDialog1.FileName)
    Dim Rec As New Rectangle(New Point(G.xmn, G.ymn), New Size(mw, mh))
    Dim Gr As Graphics = Graphics.FromImage(bmp)
    Gr.DrawRectangle(Pens.Red, Rec)
    PictureBox1.Image = bmp
    MsgBox(LP) '顯示新車牌答案
End Sub
```

這段程式首先要找到最小的字元間距，其實主要是避開有隔線的較大字元間距，在此例終究是要避免以 F 和 1 字元的間距做外插的標準。之後就是在最左邊的 F 左方製造一個新的虛擬目標，位置是往 F 字的左方平移一個字的距離。將此目標按照其他字元的處理程序：旋轉→正規化→比對字元，就會得到 5 這個字了！執行程式的結果如下圖：

由上可知，在我們建立好字元目標辨識的標準程序之後，可以很容易的在我們預期有字元的位置新增一個虛擬的目標，迅速補足可能遺漏的字元！程式操作並不困難，比較需要時間實驗調整的是決策的過程！這一部份就留待讀者自己去慢慢補足了！

8-5 影像辨識不只是辨識「影像」

從本章內容讀者應該可以體會，「聰明」的車牌辨識不會只是死板的根據「影像」資料作處理。既然知道要辨識的是車牌，當然可以盡量參考車牌應有的格式與特性，來輔助我們判斷辨識是否正確？甚至協助我們做一些「**破壞**」性的處理！就是修改直接來自影像的辨識結果，以符合車牌格式。

事實上多數特定目的的辨識都會有類似的「影像外」的資訊可以使用！當影像本身比較複雜或模糊時，這些先備的條件知識其實是比影像更為真實可靠的資訊，譬如指紋只會有三叉點，如果你發現你辨識出來的校條中有兩線交叉成十字路口的狀況，那一定是錯的！可能是雜訊干擾，或根本就不是指紋影像！最好就是忽略或直接用程式改掉！

我們做車牌辨識時通常不會只有一張車牌占據整個影像畫面，背景雜訊甚麼東西都有可能出現，為甚麼好的車牌辨識系統不會常常將其他文字目標當作車牌報告出來？原因就是我們會參考很多外部資訊，即使它們也是清楚的文字，如果顏色或格式不符合已知的車牌就會被否定忽略。

我們使用這種 OCR 的程序辨識車牌時，最大的危機是車牌字元的沾連，尤其是字元與背景連在一起時，就鐵定不會變成我們可以處理的目標，所以強行切割補字的功能是必要的程序！很像我們生病就必須進行特殊的醫療一樣！沒有醫院與醫生當然人類

的死亡率就會高出很多了！如果是相連的兩個字元沾連呢？就是直接從中切開為兩個目標了！總之，因為影像總有模糊的時候，某些依據外部資訊發現錯誤而進行的強制補救演算法是必須的！否則我們就會失去很多邊緣狀況的辨識成功機會。

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
```