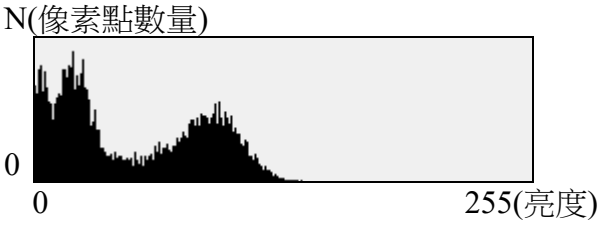
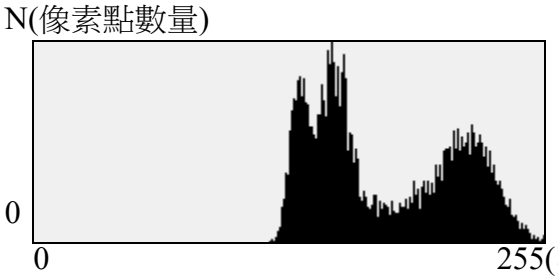
第 3 章　目標二值化與輪廓化

3-1 目標切割是目的，二值化與輪廓化是手段

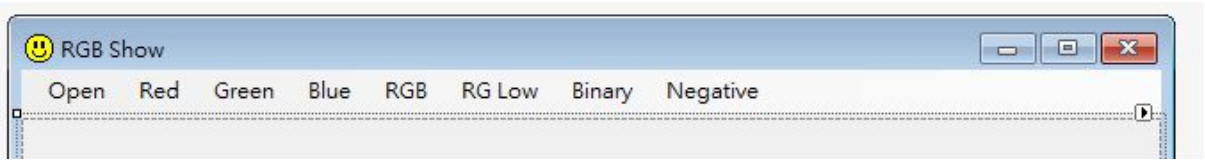
前一章已經介紹了將全彩影像簡化成灰階，到做出最簡易的二值化圖形。我們的目標是將車牌字元切割出來成為獨立目標做後續的辨識，但因為我們一開始不知道車牌在哪裡？所以必須針對全圖做二值化的處理，事實上就是在灰階圖上看起來「像個字」的較「深色」區塊，為何稱之為「區塊」？也就是指它的周邊圍繞著較「淺色」的區域。

但所謂的深色與淺色是相對的觀念，下面兩個車牌影像一亮一暗，用一般人的視力看都算清晰，灰階分布直方圖如右。如果你只以簡單的灰階 128 亮度門檻做二值化，兩者都是無法看到字元的，一個會變全白，一個全黑。這表示要找到所有可能的目標，我們的二值化門檻必須「因地制宜」，在全圖的不同區域要動態調整，這就是本章將介紹的第一個主題。





請先建立一個程式專案，匯入前一章建立的 FastPixel 模組，建立主功能表的下列幾個按鍵：Open, Gray, Ave, Binary 與 Outline。其中 Open 功能與前一章相同，用於開啟待處理的影像，Gray 的內容與前一章的 Green 按鍵相同，就是以綠光為基礎的灰階影像，程式碼可以直接複製過來。



3-2 建立區塊亮度平均值作為二值化門檻

這裡的概念是先將影像分成 40X40 大小的方塊，計算每個方塊內的平均亮度，在此方塊區域內，灰階亮度大於或等於平均值的將被視為白色背景，小於平均值的視為黑色目標，也就是以區塊的平均亮度作為二值化處理的門檻值。首先我們用 Ave 按鍵功能看看這個區塊平均值計算的結果：

```
'平均亮度方塊圖
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
Private Sub AveToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles AveToolStripMenuItem.Click
    Dim kx As Integer = nx \ Gdim, ky As Integer = ny \ Gdim
    ReDim Th(kx, ky)
    '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
            Th(x, y) += Gv(i, j)
        Next
    Next
    '建立亮度塊狀圖
    Dim A(nx - 1, ny - 1) As Byte
    For i As Integer = 0 To kx - 1
        For j As Integer = 0 To ky - 1
            Th(i, j) /= Gdim * Gdim '區塊亮度平均值計算
            For ii As Integer = 0 To Gdim - 1
                For jj As Integer = 0 To Gdim - 1
```

```
                        A(i * Gdim + ii, j * Gdim + jj) = Th(i, j)
                    Next
                Next
            Next
        Next
        PictureBox1.Image = GrayImg(A) '建立灰階圖
    End Sub
```

因為區塊的大小 Gdim 與門檻值陣列 Th 會被後續程式繼續使用，所以需要提升為全域變數。寫好程式後開啟前面的影像檔，Gray 與 Ave 顯示的影像應該如下：

3-3 二值化處理

接下來是依據這個門檻陣列(Th)產生全圖的二值化影像，就是 Binary 按鍵的功能，程式碼如下：

```
'二值化
Dim Z(,) As Byte '全圖二值化陣列
Private Sub BinaryToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles BinaryToolStripMenuItem.Click
    ReDim Z(nx - 1, ny - 1)
    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 Gv(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
    PictureBox1.Image = BWImg(Z) '建立二值化圖
End Sub
```

二值化結果如下圖：

3-4 輪廓化處理

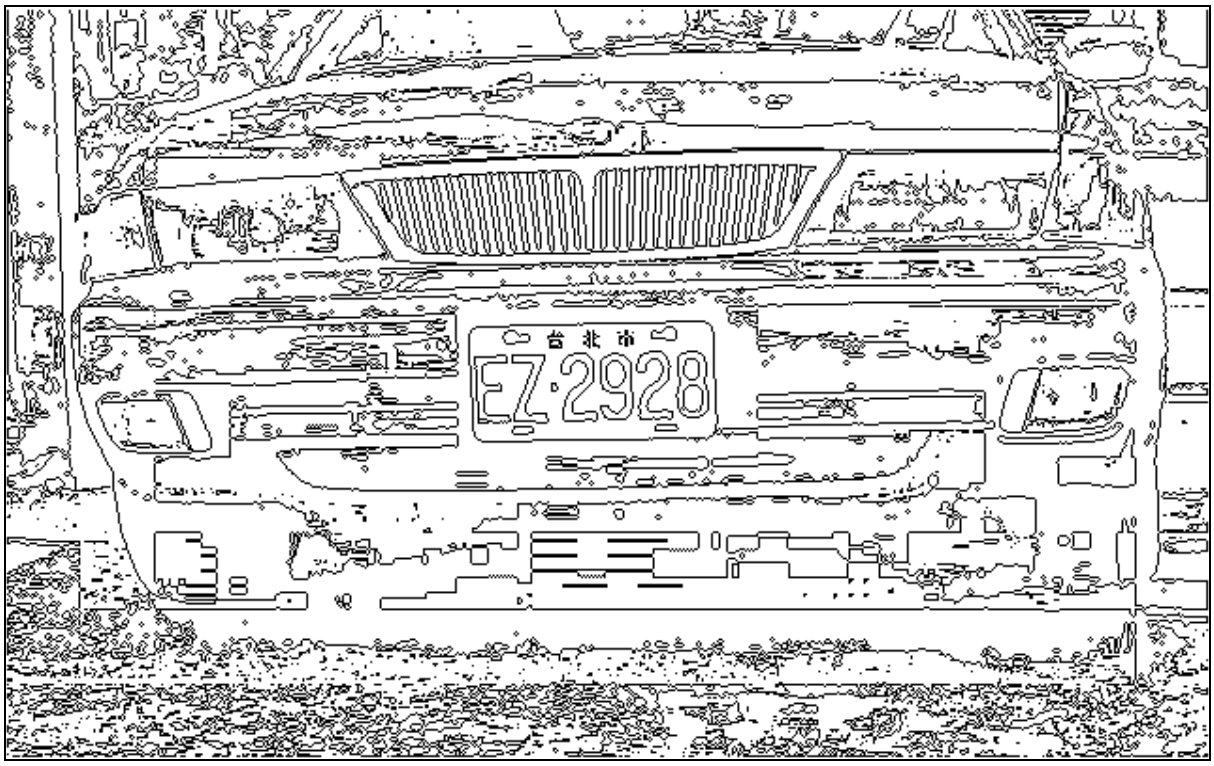
接下來要如何將這些大大小小的黑色塊狀目標，變成後續的辨識程式演算法可以方便操作的「**物件**」呢？這應該就是業餘與專業技術的分水嶺了！我們必須可以直接用程式指定上圖中那些 E 或 Z 等「**目標**」該去做甚麼處理？就是要將它們一一從全圖中擷取出來，建立成一個個明確的資料結構。在數學上就是將相連的黑點視為一個獨立目標，此時最適用的演算法就是所謂的 **Flood fill Algorithm**，中文稱為**氾濫式演算法**，大家最熟悉的應用就是小畫家裡面的**油漆桶**填色功能了！

但是有一個問題是：氾濫式演算法要填充那麼多凌亂的**實心**黑色區域當然是很耗時的，想想一個字元區塊內的黑點可能就上千個！但是目標內部的實心黑點，對於我們後續辨識字元的寬高、大小、形狀甚至字形的特徵都沒有幫助，所以我們最好先將二值化圖簡化為輪廓圖，就是以輪廓圖進行擷取目標的計算，這樣就可以節省很多的處理時間。這就是 Outline 按鍵的功能了，程式碼及執行結果如下：

```
'輪廓線
Dim Q(,) As Byte '輪廓線陣列
Private Sub OutlineToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
    Handles OutlineToolStripMenuItem.Click
    If IsNothing(Z) Then Exit Sub '無二值化圖忽略
    Q = Outline(Z) '建立輪廓點陣列
    PictureBox1.Image = BWImg(Q) '建立輪廓圖
End Sub
'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 Sub
```

所謂輪廓點的定義就是某個黑點(目標點)的上下左右，至少有一點是白點(背景點)，這樣它就應該是目標區塊的邊界點了！程式中我們是以 Z 陣列代表二值化的資料，Q 則代表輪廓點的陣列，這是我們自行建立的習慣用法，本書後續都會盡量承襲這些慣例。

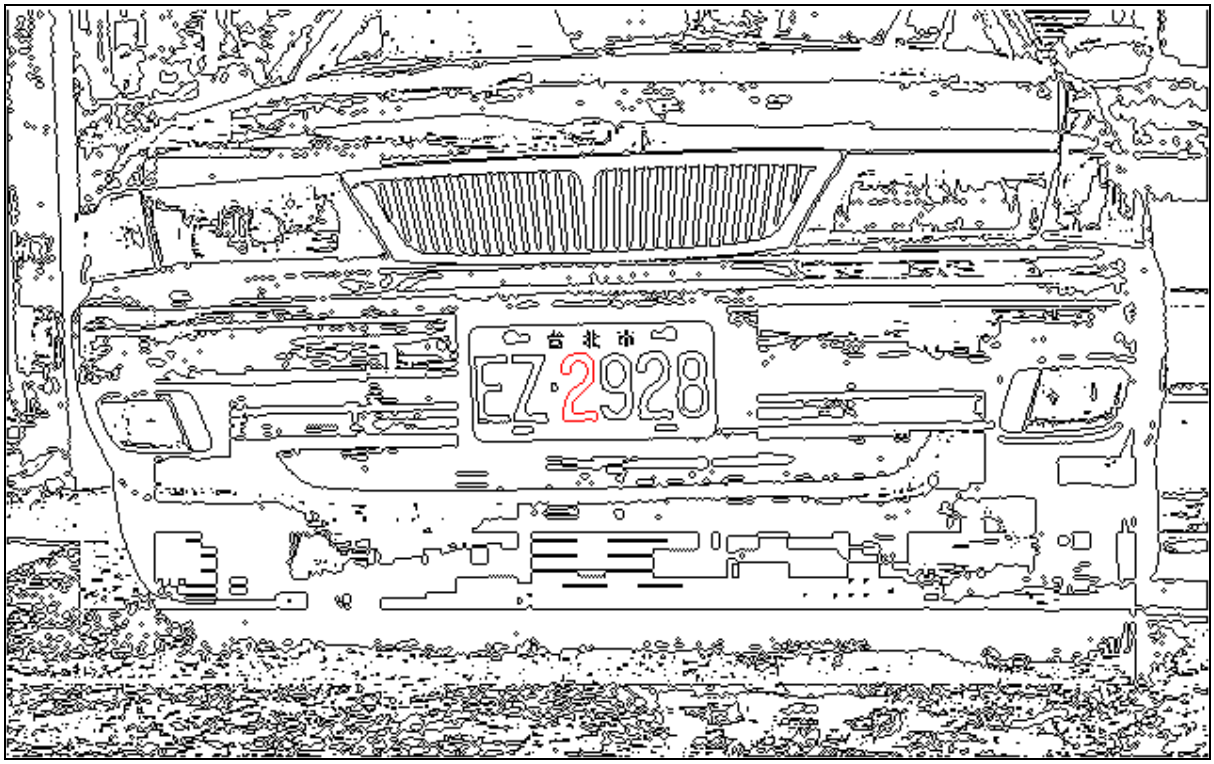


3-5 用氾濫式演算法檢視封閉曲線

在輪廓線圖上要找出獨立目標，其實就是使用氾濫式演算法(flood fill algorithm)找出封閉的曲線，所有可以相連互通的點集合就是同一目標了！如何建立有完整屬性(如寬、高、位置與點數等)的獨立目標物件？是下一章的議題，在此我們先用一個氾濫式演算法讓大家看到哪些輪廓是相連的？也就是可辨識目標的原型！請在 PictureBox1 的 MouseDown 事件中寫程式如下：

```
'選擇顯示某目標之輪廓線
Private Sub PictureBox1_MouseDown(ByVal sender As Object, ByVal e As MouseEventArgs) _
    Handles PictureBox1.MouseDown
    If e.Button = MouseButtons.Left Then
        If IsNothing(Q) Then Exit Sub
        Dim x As Integer = e.X, y As Integer = e.Y
        '尋找左方最近之輪廓點
        Do While Q(x, y) = 0 And x > 0
            x -= 1
        Loop
        Dim A As ArrayList = getGrp(Q, x, y) '搜尋此目標所有輪廓點
        Dim bmp As Bitmap = BWImg(Q) '建立輪廓圖
        For k As Integer = 0 To A.Count - 1
            Dim p As Point = A(k)
            bmp.SetPixel(p.X, p.Y, Color.Red)
        Next
        PictureBox1.Image = bmp
    End If
End Sub
'氾濫式演算法取得某目標之輪廓點
Function getGrp(ByVal q(,) As Byte, ByVal i As Integer, ByVal j As Integer) As ArrayList
    If q(i, j) = 0 Then Return New ArrayList
    Dim b(,) As Byte = q.Clone '建立輪廓點陣列副本
    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
    b(i, j) = 0 '清除此起點之輪廓點標記
    Dim A As ArrayList = nc '此目標中所有目標點的集合
    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
                    A.Add(k) '加入所有已蒐集到的目標點集合
                    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
    Return A '回傳所有目標點的集合
End Function
```

寫好之後用滑鼠點選輪廓圖的畫面，應該可以看到最接近該點的封閉曲線，大概如下圖：



3-6 本節相關議題討論

本書的目的是想清楚介紹傳統的影像辨識概念與實作方法，但是距離要做出真正商業化的軟體還是有很多細節差距。譬如使用區塊亮度平均作二值化門檻時，大家可能已經發現在區塊邊緣的門檻值會有跳躍的現象，如果要辨識的字元目標剛好跨越兩個亮度差異較大的區塊，就有可能產生邊界效應，字元左右使用差異大的門檻做二值化就會有些異常了！

我們實作商業軟體時一定會使用線性內插等方式，讓門檻值不會隨區塊跨界而跳躍，內外插的方式也有很多種，但是為免陷入太多枝節，模糊了主要辨識流程的介紹，就留待讀者自己去補足了！那些不是太困難的數學，但需要時間寫程式與充分實驗。事實上，以區塊的亮度平均值作為門檻也不是唯一或最好的方式！我們想表達的重點是：**每一點的門檻值必須參考該點周遭的環境因素來決定**。按照這個原則，讀者也可以設計自己的門檻值演算法，只要結果好就是好演算法。

此外，所謂氾濫式演算法是以某目標物中的一個點為起始，反覆嘗試擴張版圖到周圍相連的目標點，直到每一點的周圍都是已搜尋確認過的目標點，或背景點為止。在動態門檻的方式下做出的二值圖，即使簡化成輪廓，還是相當複雜的！大量使用氾濫式演算法多少會影響辨識速度，所以我們在商業軟體中會有很多技巧去簡化輪廓，或提升氾濫式演算的速度，整體來說就是所謂的「**優化**」效能了！

那些優化的細節技術都是基於實驗效果逐步研發加入的，也不是所有狀況都能通用，為免模糊焦點，本書也不會詳細介紹了！讀者可以自行研究，隨時增刪調整自己的優化程序。我們公司承接的各式影像辨識專案，也都是以本書介紹的主流程與概念為基礎，細節流程與方法，都是依據辨識目標特性而彈性調整的！我們的理念是：**沒有任何方法或程序是所有影像辨識目的都可以一體適用的**！能直接幫你準確擷取所有需要目標的影像辨識萬靈丹，應該是不存在的！**理解狀況**並**針對處理**才是製作出最佳化影像辨識軟體的王道！

完整專案：

```
Public Class Form1
    '開啟檔案
    Private Sub OpenToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles OpenToolStripMenuItem.Click
        If OpenFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            Dim bmp As New Bitmap(OpenFileDialog1.FileName)
            Bmp2RGB(bmp)
            PictureBox1.Image = bmp
        End If
    End Sub
    '灰階
    Private Sub GrayToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles GrayToolStripMenuItem.Click
        PictureBox1.Image = GrayImg(Gv)
    End Sub
    '平均亮度方塊圖
    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
    Private Sub AveToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
         Handles AveToolStripMenuItem.Click
        Dim kx As Integer = nx \ Gdim, ky As Integer = ny \ Gdim
        ReDim Th(kx, ky)
        '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
                Th(x, y) += Gv(i, j)
            Next
        Next
        '建立亮度塊狀圖
        Dim A(nx - 1, ny - 1) As Byte
        For i As Integer = 0 To kx - 1
            For j As Integer = 0 To ky - 1
                Th(i, j) /= Gdim * Gdim '區塊亮度平均值計算
                For ii As Integer = 0 To Gdim - 1
                    For jj As Integer = 0 To Gdim - 1
                        A(i * Gdim + ii, j * Gdim + jj) = Th(i, j)
                    Next
                Next
            Next
        Next
        PictureBox1.Image = GrayImg(A) '建立灰階圖
    End Sub
    '二值化
    Dim Z(,) As Byte '全圖二值化陣列
    Private Sub BinaryToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles BinaryToolStripMenuItem.Click
        ReDim Z(nx - 1, ny - 1)
        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 Gv(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
        PictureBox1.Image = BWImg(Z) '建立二值化圖
    End Sub
    '輪廓線
    Dim Q(,) As Byte '輪廓線陣列
    Private Sub OutlineToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles OutlineToolStripMenuItem.Click
        If IsNothing(Z) Then Exit Sub '無二值化圖忽略
        Q = Outline(Z) '建立輪廓點陣列
        PictureBox1.Image = BWImg(Q) '建立輪廓圖
    End Sub
    '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
    '選擇顯示某目標之輪廓線
    Private Sub PictureBox1_MouseDown(ByVal sender As Object, ByVal e As MouseEventArgs) _
        Handles PictureBox1.MouseDown
        If e.Button = MouseButtons.Left Then
            If IsNothing(Q) Then Exit Sub
            Dim x As Integer = e.X, y As Integer = e.Y
            '尋找左方最近之輪廓點
            Do While Q(x, y) = 0 And x > 0
                x -= 1
            Loop
            Dim A As ArrayList = getGrp(Q, x, y) '搜尋此目標所有輪廓點
            Dim bmp As Bitmap = BWImg(Q) '建立輪廓圖
            For k As Integer = 0 To A.Count - 1
                Dim p As Point = A(k)
                bmp.SetPixel(p.X, p.Y, Color.Red)
            Next
            PictureBox1.Image = bmp
        End If
    End Sub
    '氾濫式演算法取得某目標之輪廓點
    Function getGrp(ByVal q(,) As Byte, ByVal i As Integer, ByVal j As Integer) As ArrayList
        If q(i, j) = 0 Then Return New ArrayList
        Dim b(,) As Byte = q.Clone '建立輪廓點陣列副本
        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
        b(i, j) = 0 '清除此起點之輪廓點標記
        Dim A As ArrayList = nc '此目標中所有目標點的集合
        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
                        A.Add(k) '加入所有已蒐集到的目標點集合
                        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
        Return A '回傳所有目標點的集合
    End Function

    Private Sub SaveImageToolStripMenuItem_Click(ByVal sender As Object, ByVal e As EventArgs) _
        Handles SaveImageToolStripMenuItem.Click
        If SaveFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            PictureBox1.Image.Save(SaveFileDialog1.FileName)
        End If
    End Sub
End Classl
```