Computer Vision - Homework 2

開發環境

- OS: Windows 10 Pro
- Program Language: C# (with .Net Core 3.1)
- IDE: Visual Studio 2019
- Project: Console Application

程式說明

程式碼主要寫在 Program.cs，各題目程式皆已實作個別方法，由 Main entry 進行呼叫，答案結果儲存於 answers 資料夾。

各題目程式碼片段、參數及相關演算法說明如下：

(A). Binary Image (Threshold at 128): 藉由遞迴方式取得影像的所有 Pixel 值後進行二位元閥值(Threshold)轉換。

```
/// <summary>
/// 取得 Binary Image (Threshold at 128)
/// </summary>
/// <param name="srcImg">來源影像</param>
/// <returns>二位元影像</returns>
private static Bitmap GetBinaryBitmap(Bitmap srcImg)
{
    if (srcImg == null)
        throw new Exception("source image is null");

    var dstBmp = new Bitmap(srcImg.Width, srcImg.Height);

    // 兩層迴圈取得所有 pixel 值(x, y)
    for (var x = 0; x < srcImg.Width; x++)
    for (var y = 0; y < srcImg.Height; y++)
    {
        // 取得 Pixel RGB 值, 除以 3 後得到灰階值
        var color = srcImg.GetPixel(x, y);
        var gray = (color.R + color.G + color.B) / 3;

        // 值 >= 128 轉為白色(255, 255, 255), 否則轉為黑色(0, 0, 0)
        dstBmp.SetPixel(x, y, gray >= 128 ? WhiteColor : BlackColor);
    }

    return dstBmp;
}
```

(B). Histogram: 迴圈取得所有 Pixel 值後，根據灰階值的對應進行數量的加總後繪製直方圖，X 軸為 0 ~ 255 的灰階值，Y 軸為對應灰階值的數量加總。(本題加入 ScottPlot 的參考作為直方圖繪製的函式庫)

```
private static Plot GetHistogram(Bitmap srcImg)
{
    if (srcImg == null)
        throw new Exception("source image is null");
```

```
    // xs: x 軸: 0 ~ 255 的灰階值
    var xs = new double[256];
    for (var x = 0; x < 256; x++) xs[x] = x;

    // ys: y 軸: 影像內灰階值對應的數量加總
    var ys = new double[256];

    for (var x = 0; x < srcImg.Width; x++)
    for (var y = 0; y < srcImg.Height; y++)
    {
        var color = srcImg.GetPixel(x, y);
        var gray = (color.R + color.G + color.B) / 3;
        ys[gray] += 1;
    }

    // 藉由 ScottPlot 繪製直方圖
    var plt = new Plot();
    plt.Title("Histogram");
    plt.XLabel("Gray (0~255)");
    plt.YLabel("Count");
    plt.PlotBar(xs, ys, barWidth: 1.4D, outlineWidth: 0);

    return plt;
}
```

(C). Connection Components (regions with + at centroid, bounding box): 採用 Iterative Algorithm 取得四連通元件圖，並找出面積 >= 500 的 Connected Component, Retangle 及 Centrold。

- BoundingBox: 遞迴比對所有 Point，找出左上及右下的點。
- Centroid: 平均 Connected Component 內的 X 及 Y 值。

```
private static int[,] _labels;

/// <summary>
/// 取得連通元件圖 (四連通)
/// </summary>
/// <param name="srcImg">來源影像</param>
/// <returns>四連通圖(含 BoundingBox 及 Centroid Circle)</returns>
private static Mat GetConnectedComponents(Bitmap srcImg)
{
    // 初始化 int[,] 陣列，紀錄 (x, y) 的標籤值
    InitLabels(srcImg);

    var changed = true;

    // 判斷是否有值變更，若有則繼續遞迴更新標籤
    while (changed)
    {
        // UpDown: 遞迴比對所有標籤值，與左上標籤取最小值(非0)並更新至 Labels
        // BottomUp: 遞迴比對所有標籤值，與右下標籤比較取最小值(非0)並更新至 Labels
        changed = UpDown();
        if (!changed) break;
        changed = BottomUp();
    }

    return GetBoundingBoxImage(srcImg);
```

```
}

/// <summary>
/// 繪製 BoundingBox
/// </summary>
/// <param name="srcImg"></param>
private static Mat GetBoundingBoxImage(Bitmap srcImg)
{
    var dic = new Dictionary<int, ConnectedComponent>();

    // 遞迴取出所有 ConnectedComponent
    for (var x = 0; x < _labels.GetLength(0); x++)
    for (var y = 0; y < _labels.GetLength(1); y++)
    {
        var value = _labels[x, y];
        var cmp = dic.ContainsKey(value) ? dic[value] : new
ConnectedComponent();

        // 取得最左上值
        if (cmp.LeftTopPoint == null) cmp.LeftTopPoint = new Point(x, y);
        if (x < cmp.LeftTopPoint.X) cmp.LeftTopPoint.X = x;
        if (y < cmp.LeftTopPoint.Y) cmp.LeftTopPoint.Y = y;

        // 取得最右下值
        if (cmp.RightBottomPoint == null) cmp.RightBottomPoint = new Point(x,
y);
        if (x > cmp.RightBottomPoint.X) cmp.RightBottomPoint.X = x;
        if (y > cmp.RightBottomPoint.Y) cmp.RightBottomPoint.Y = y;

        cmp.Points.Add(new Point(x, y));
        dic[value] = cmp;
    }

    // 透過 Linq 篩選取得 >= 500 的 Connected Components
    var components = dic
        .Where(pair => pair.Key != 0 && pair.Value.Points.Count >= 500)
        .Select(pair => pair.Value);
    var mat = srcImg.ToMat();

    foreach (var cmp in components)
    {
        // Bounding Box
        Cv2.Rectangle(
            mat,
            new OpenCvSharp.Point(cmp.LeftTopPoint.X, cmp.LeftTopPoint.Y),
            new OpenCvSharp.Point(cmp.RightBottomPoint.X,
cmp.RightBottomPoint.Y),
            Scalar.Blue);

        // Centroid
        var x = 0;
        var y = 0;

        // 所有 x 及 y 值的加總後平均
        foreach (var point in cmp.Points)
        {
            x += point.X;
            y += point.Y;
```

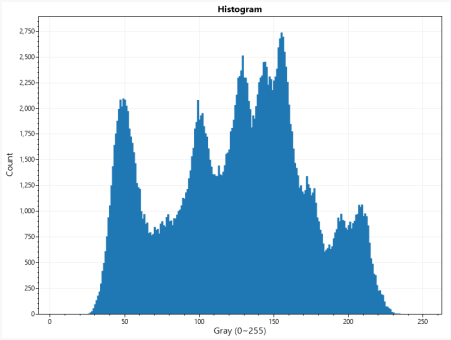
```
            }

            x /= cmp.Points.Count;
            y /= cmp.Points.Count;

            Cv2.Circle(mat, new OpenCvSharp.Point(x, y), 5, Scalar.Red, 2);
        }

        return mat;
    }
```

結果圖片

(A). 	(B). 	(C).