中華民國第 58 屆中小學科學展覽會

作品說明書

高級中等學校組　電腦與資訊學科

(鄉土)教材獎

第三名

052504

以幾何圖形發想來建構覆蓋方格數之演算法

—探究台灣 AED 設置數量與分佈

學校名稱：新北市立北大高級中學

作者：	指導老師：
高二　胡佩欣	黃冠閔
高二　吳宣宏	沈昌懋
高二　許子潔	

關鍵詞：方格子、演算法、地理資訊系統

摘要

AED(Automated External Defibrillator)，中文稱為「自動體外心臟電擊去顫器」，是一台能夠自動偵測傷病患心律脈搏、並施以電擊使心臟恢復正常運作的儀器，因此傷病患的存活往往是和時間和死神的賽跑，如果將 AED 設置於人潮眾多的公共場所，供民眾搶救時使用，可降低該類傷病患到院前死亡率。

本研究是以政府資料開放平臺網提供的 AED 位置資訊，將資料導入地理資訊系統(QGIS)，呈現實際分佈狀況之網格化的結果，因應台灣行政區域的不同形狀，試圖探究數學上所學的幾何圖形，利用數學原理來設計覆蓋方格總數的演算法，藉此找出覆蓋的範圍內，是否需要再增設 AED 裝置，提供一既有效率且平均的急救體系。

壹、 研究動機

健護課提到 AED 裝置在我們生活當中扮演了相當重要的醫療角色，若能夠在發生猝死時取得 AED，我們便可以在實施 CPR 的時候同時進行 AED 急救，大大縮減了過去等待醫護人員到達才施予電擊除顫的時間差，而越早進行電擊、則病人搶救成功的機率也越大。

根據政府資料開放平臺網提供的 AED 位置資訊顯示，其實台灣的每一個行政區域，AED 裝置的分佈並不平均，將這些資料匯入地理資訊系統(QGIS)，AED 位置會呈現在網格圖上，則我們觀察到分佈位置的網格圖可以依據台灣行政區域的形狀(如:三角形)，利用幾何圖形來覆蓋它並計算在幾何圖形內最多覆蓋的方格子數，就可以計算出無 AED 的分佈之方格子數，這勾起了我們的興趣，想進一步研究數學的幾何圖形在平面座標上的關係，最後利用電腦程式語言 C 設計覆蓋方格子總數的演算法。

因此我們試圖跨領域，結合資訊的「基礎程式設計」、「演算法概論」、數學的「平面上的幾何圖形」、地理的「地理資訊系統(QGIS)」，期待透過撰寫程式來改善 AED 裝置不均的分佈，藉以達到降低該類傷病患到院前死亡率之目標。

貳、 研究目的

透過數學理論與電腦演算法的結合，製作出可以利用三頂點座標計算方格數量的演算法來解決問題，其中我們利用 QGIS 系統將取得的資料製作成網格圖，之後將影像資料放入演算法轉換成數據，再利用積分影像原理跟統計學的變異數來把它進行統計，最後嘗試找出三角形內部方格總數的數學通式與如何找出最佳覆蓋方格子之三角形三頂點的方法，將研究的理論程式化，藉此統計 AED 分佈的狀況，來改善 AED 裝置不均的分佈之情況。

參、 研究設備和器材

一、程式語言 C 與編譯程式語言系統：

C 語言是一種被廣泛使用電腦程式語言，是產業界最流行的程式開發工具，從硬體、軟體的設計到韌體、系統的開發；Microsoft Visual C++ 2010 Express 是一套由微軟公司開發的免費整合開環境，它具有 VB、VC++、VC#編譯器，我們作為這次研究撰寫程式的工具。

二、數學軟體 GeoGebra

GeoGebra 是一套動態數學教育軟體，它不但包含動態幾何尺規作圖、幾何轉換等功能，還加入函數繪圖、代數運算和基本微積分功能。我們用來繪製二維平面中三頂點連的三角形。

三、數學運算軟體 Mathematica

Mathematica 是一套整合數字以及符號運算的數學工具軟體，廣泛使用不同的領上，本研究藉由 Mathematica 高速運算特色，來驗證我們的數學通式。

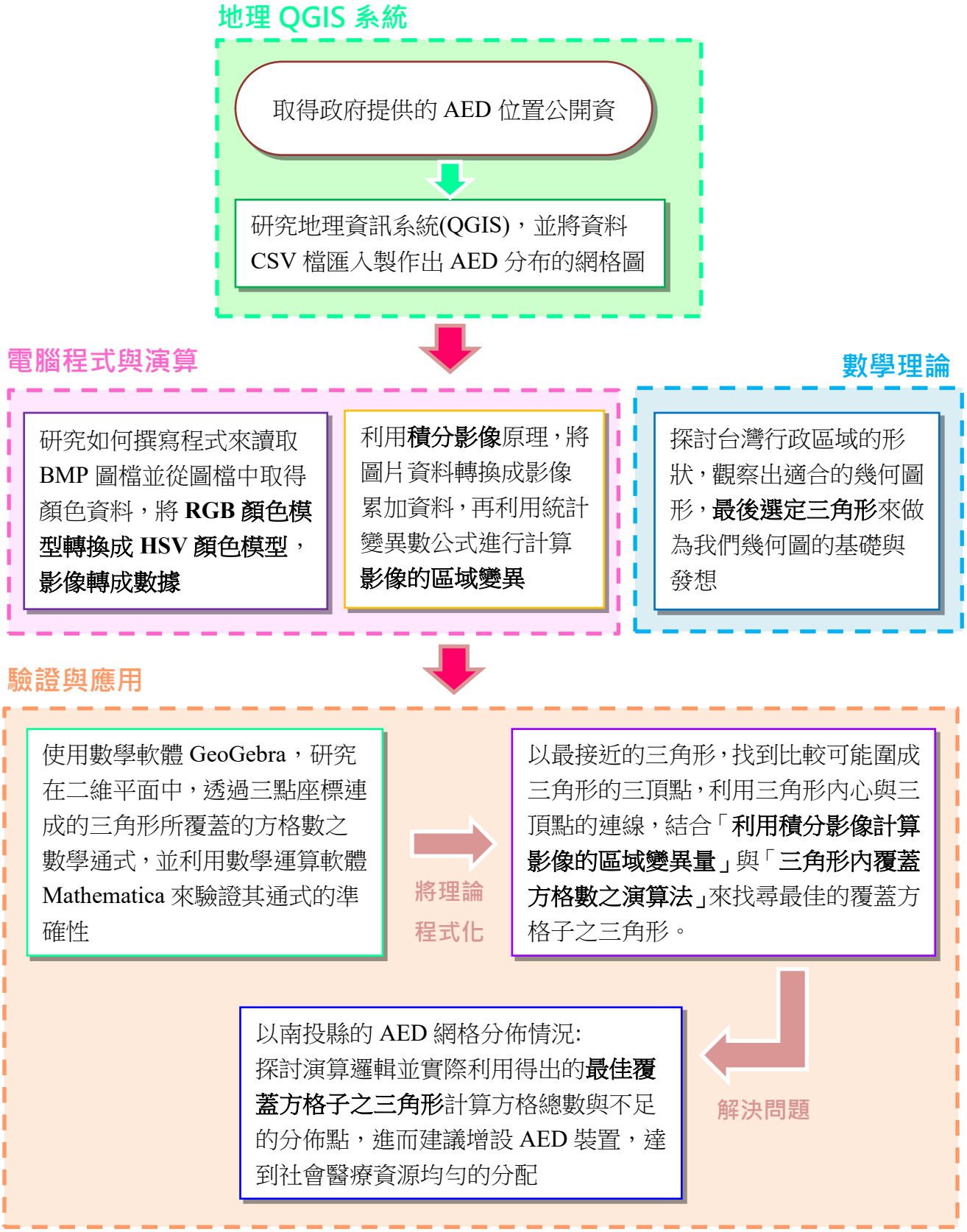
四、地理資訊系統(QGIS)：

原稱 Quantum GIS，是一個開放且免費的地理資訊系統，處理許多的地圖文件，包括地形圖、分布圖與多重疊圖等，我們利用它來處理 AED 公開資料。

五、筆電與桌上型電腦

肆、　研究過程和方法

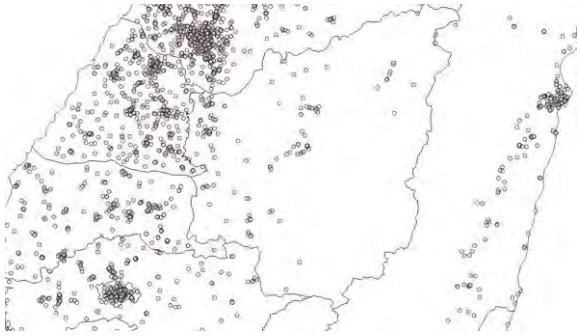
一、研究流程



【圖一：研究的流程示意圖】

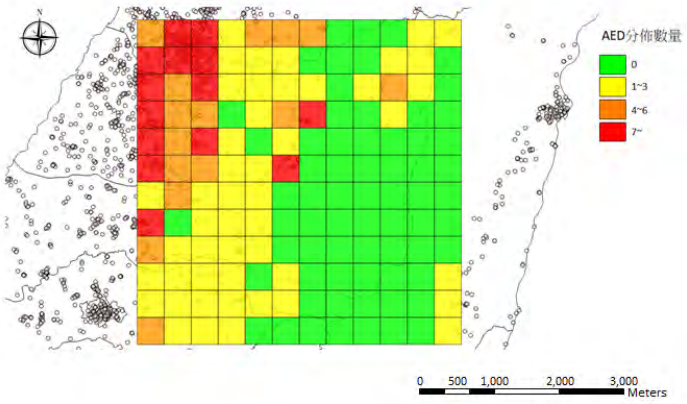
二、研究如何將 CSV 檔匯入地理資訊系統(QGIS)並繪製分佈網格圖

首先我們先取得政府資料開放平臺網提供的 AED 位置公開資料，資料檔為 CSV 檔，將所有的 AED 位置與台灣的行政區圖進行疊圖，可以清楚觀察到從北至南的行政區域中，AED 裝置的分佈是不均勻地，甚至在台灣偏中間的位置，如:南投山區，幾乎是沒有的，因此我們必須重新規劃 AED 裝置的分佈，以確保民眾在需要時，可以即時享有醫療的資源。我們選擇先以台灣中部地區「南投縣」作為基礎，開始進行製做 AED 裝置的分佈點狀圖(圖二）。



【圖二：台灣 AED 裝置的分佈之示意圖】

接著再利用 QGIS 系統內的網格系統進行網格化（圖三），將一個方格子覆蓋 AED 分佈點之密集程度作為分類的依據，轉化成顏色，以顏色區分每個地方不同的密集程度依序為「紅＞橙＞黃＞綠」，以方便我們更快的辨識 AED 裝置的位置，這可以讓整個區域概況更加容易辨識。（圖三）中顯示，紅色的方格表示 AED 裝置分佈非常密集(數量:7)，橘色的方格表示次多(數量:4~6)，黃色表示少許(數量:1~3)，綠色方格表示無 AED 裝置的位置分佈之地(數量:0)，也就是一般的土地，我們依據四種顏色的不同，呈現在二維平面上的方格子分佈，便能夠更好判斷一個區塊內的 AED 裝置分佈情形，並將網格化的圖存成 BMP 格式檔。



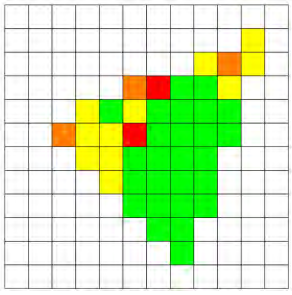
【圖三：台灣 AED 裝置之網格化示意圖】

三、撰寫 C 語言來讀取南投縣 AED 位置的分佈之點陣圖 BMP 與顏色的分類研究

點陣圖 BMP 取自點陣圖 BitMap 的縮寫，檔案的預設副檔名是 .BMP 或 .bmp，此檔案格式通常包含四個資訊：檔案標頭、檔案資訊標頭、調色板與點陣圖資料，其中這四個資訊所佔的檔案大小分別為: 14bytes、40bytes、1024bytes 與依據圖片的像素大小。這裡我們著重在第四個資訊「點陣圖資料」，因為一張點陣圖實際的像素資料會存放在 Raw 檔裡，其中像素存放的元素為三原色，即為紅(R)、綠(G)、藍(B)三種原色，這三種色皆可由 16 進位表示，如:#ff0000 表示紅色、#ff0000 表示綠色、#ff0000 表示藍色等依此類推，值得注意的是點陣圖中像素的存放順序分別為藍(B)、綠(G)、紅(R)。

在此我們嘗試利用 C 語言來撰寫一個程式讀取一張南投縣 AED 位置的分佈之點陣圖 BMP 檔，並作方格子的顏色分類研究，我們分以下步驟來介紹:

(一)利用 QGIS 系統繪製出的網格化後的南投縣 AED 位置分佈圖:



我們利用 QGIS 系統繪製出的網格化後的南投縣 AED 位置分佈圖並存成尺寸為 320×320 的點陣圖，命名為「nantou_aed_distribution.bmp」(如右圖) 來進行測試，其中紅色的方格表示 AED 裝置分佈非常密集，橘色的方格表示次多，黃色表示少許，綠色方格表示無 AED 裝置之地。

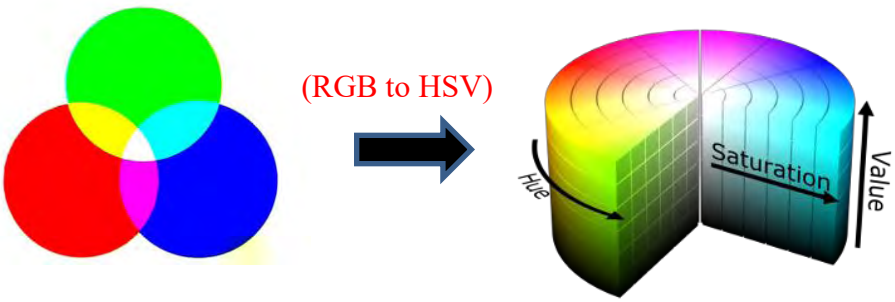
(二)撰寫 C 語言來讀取南投縣 AED 位置的分佈之點陣圖 BMP:

首先我們先宣告陣列 header[54]、陣列 OutputImage[320*320*3]與陣列 block[12][12][3]，前兩個陣列分別用來存取點陣圖「nantou_aed_distribution.bmp」之檔案標頭與資訊標頭的大小(共 54bytes)與點陣圖資料(共 320×320×3 bytes)，利用函數 fopen( )來開啟點陣圖檔，再使用函數 fgetc( )分別讀取前 54 bybes 與 320×320×3 bytes 的資料，最後利用陣列 block[12][12][3]，配合三層巢狀迴圈來存取點陣圖的像素資料(如下圖)，共存取 144 個位置中 RGB 的數值，其中 block[*i*][*j*][0]、block[*i*][*j*][1]、block[*i*][*j*][2]分別存取藍(B)、綠(G)、紅(R)之值。

```
for(i=0; i<12; i++)
{
    for(j=0; j<12; j++)
    {
        for(k=0; k<3; k++)
        {
            block[i][j][k]= OutputImage[(320*13*(25-2*(i+1)) +13*(2*(j+1)-1))*3+k];
        }
    }
}
```

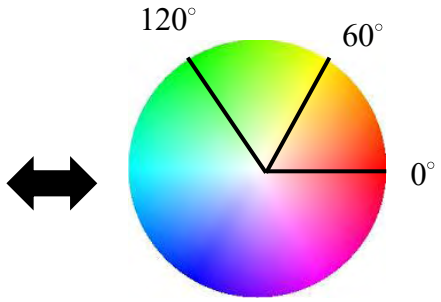
(三)方格子的顏色分類之研究:RGB 顏色模型轉成 HSV 顏色模型

為了將讀取出來的每一個方格中的 RGB 值作顏色標準的分類，我們利用到轉換 HSV 顏色模型的概念，HSV 模型定義了另一種顏色空間，它更接近人類視覺感知顏色屬性的方式，HSV 模型空間分別由有三個部分組成：色調(Hue)，飽和度(Saturatio)和值(Value)

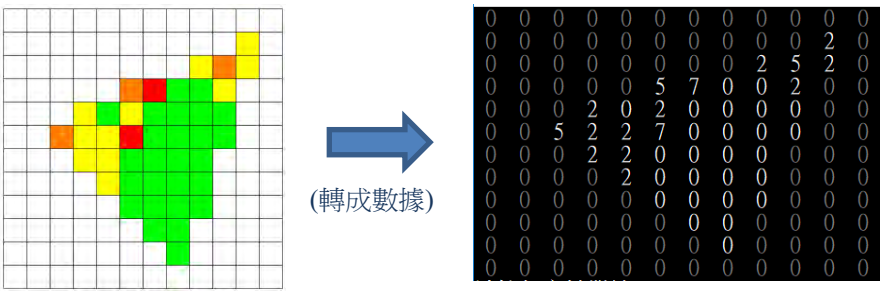


在 HSV 模型中，我們主要是利用「色調」的特性，因為它將顏色對應到角度，角度範圍為 0 度到 360 度，如右表與圖:

角度	顏色
0	紅色
0~60	橘色
60	黃色
120	綠色



因此我們開始撰寫程式，宣告兩個資料結構 RGB 與 HSV 並對應的變數 data 與 value，來存取每個方格的 RGB 資料與 HSV 資料，再利用設計好的函數 *RGBToHSV*(data)，進行兩個顏色模型的轉換，接著判別函數 *RGBToHSV*(data)回傳的值 value，其中轉換後的 value.H 範圍為 0 至 120 度，可以得到四種顏色:紅色、橘色、黃色與綠色;再來我們宣告陣列 output[12][12] 並根據南投縣 AED 分佈數量的標準，判別為紅色與綠色時，output[i][j]設定分別為 7 與 0，其他兩種顏色的值都是範圍分別為 4~6 與 1~3，因此皆取組中點，橘色設定為 5，黃色設定為 2，白色也設定為 0，利用函數 print( )將 output[12][12]呈現在電腦螢幕上，轉化的過程如下圖:



我們成功地讀取影像資料並將之轉化成一組數據後，接下來開始對這組數據做分析與研究。

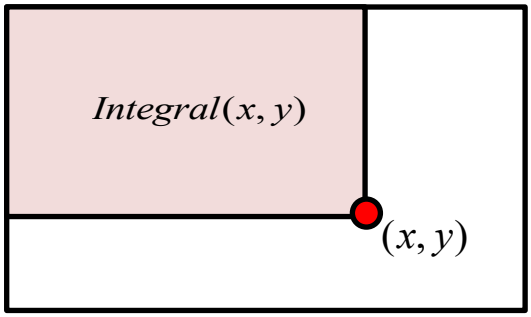
四、研究如何應用「積分影像」(Integral image)計算「影像的區域變異量」

積分影像 (Integral image) 是一個能快速且有效率地計算網格或矩形區域內影像資料累加的方式，可將原始影像轉換成積分影像，藉由透過積分影像的數值或特徵，不需探究每個像素點的資料，進而可分析影像的特色或取得有用的資料。我們以簡單的步驟來介紹積分影像的計算原理與計算區域變異量如下:

(一)首先考慮原始灰階影像的每個像素點值 $image(i,j)$ ，其中 i 與 j 值範圍由影像大小決定，積分影像 $Integral(x,y)$ 定義數學式子表示如下:

$$Integral(x,y)=\sum_{i=0}^{x}\sum_{j=0}^{y}image(i,j)$$

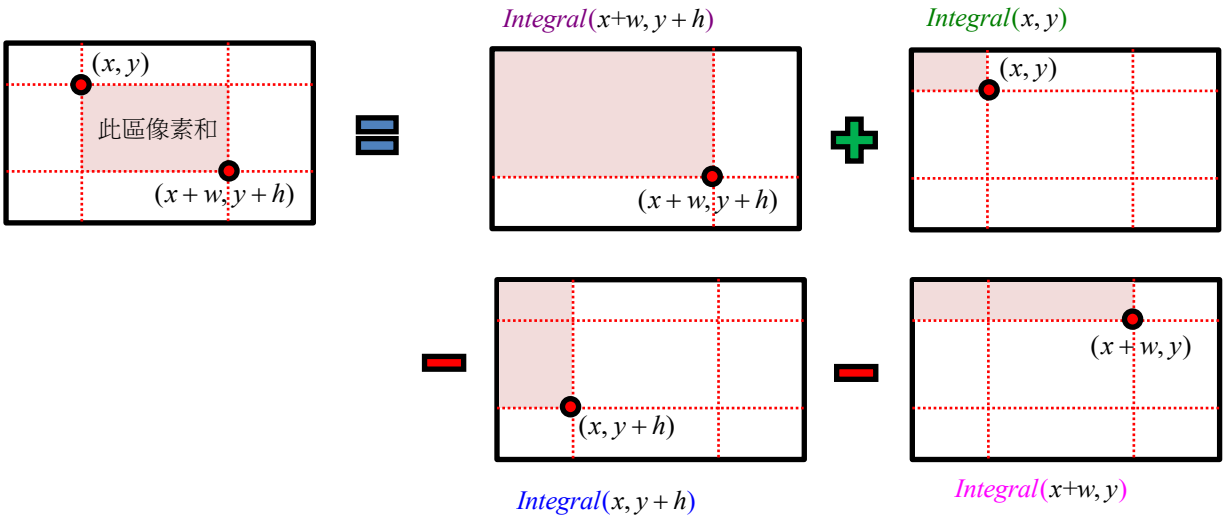
以粉紅色區域表示從影像左上角進行像素值累加至位置 (x,y) 之總和，以圖示意如下:



(二)取得影像中 x 座標為 x 至 $x+w$ 與 y 座標為 y 至 $y+h$ 中所有的像素值的總和，利用數學式子表示如下:

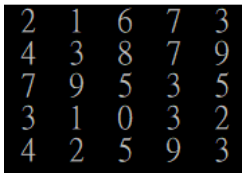
所求像素總和=$Integral(x+w,y+h)+Integral(x,y)-Integral(x+w,y)-Integral(x,y+h)$

我們利用下面圖來解釋上式的運算過程如下:



(三)應用「積分影像」的運算原理，學習撰寫程式將原影像資料快速轉化成積分影像資料:

在學習積分影像的數學式子之後，接著我們以積分影像運算原理為演算的核心，開始撰寫 C 語言程式，藉此模擬原始資料轉化成累加資料，其中原始資料設計為5×5整數陣列，並利用迴圈與函數 printf( )呈現於電腦螢幕上，如下圖四:

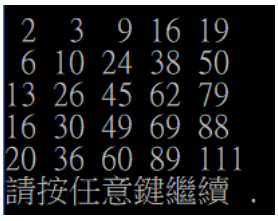


【圖四:設計原始資料設計為5×5整數陣列之示意圖】

接著將積分影像的運算原理寫成演算法，程式碼設計如圖五，計算出累加資料，並利用迴圈與函數 printf( )呈現於電腦螢幕上，如圖六如下:

```
for(i=0;i<5;i++)
{
    for (j=0;j<5;j++)
    {
        if(i==0 && j>0){
            integral[0][j] = integral[0][j-1] + image[0][j];
        }else if(j==0 && i>0){
            integral[i][0] = integral[i-1][0] + image[i][0];
        }else if(i>0 && j>0){
            integral[i][j] = integral[i-1][j] + integral[i][j-1] - integral[i-1][j-1] +image[i][j];
        }
    }
}
```

【圖五：利用迴圈與積分影像的運算原理，計算求得累加資料】



【圖六：經過計算後的累加資料之示意圖】

(四)應用積分影像的原理計算影像的區域變異量」

這裡我們先介紹如何定義「影像的區域變異量」，利用統計學上常用於分析的兩個統計量「平均數 μ 」與「變異數 σ^2 」，若給定一組 n 筆數據資料 x_i ，則變異數為計算所有資料到平均數的距離平方之平均，可以做為一個用來量測資料分散程度之指標值，此指標值越小表示

資料越集中，越大則情況反之，兩個統計量以數學式子分別定義如下:

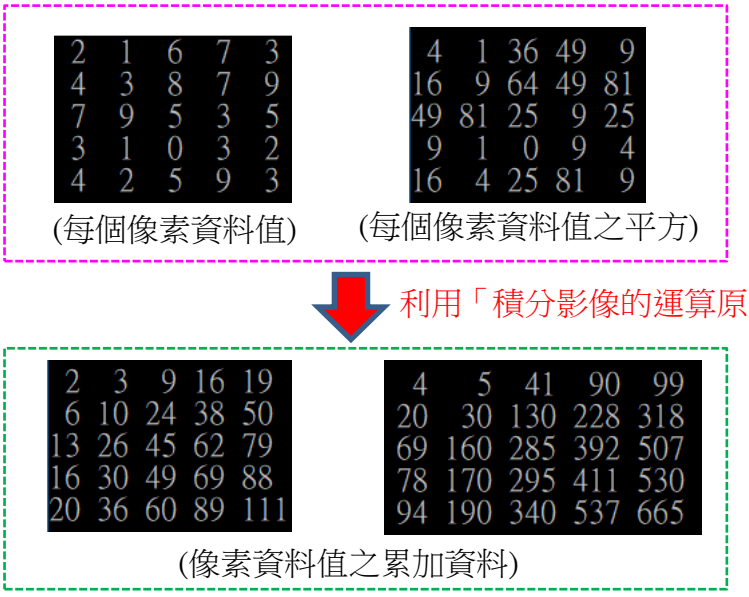
$$\mu = \frac{\sum_{i=1}^{n} x_i}{n} \quad , \quad \sigma^2 = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n}$$

將以上的一組 n 筆數據資料 x_i 想像為影像的每個像素點值，因此 μ 值為所有像素點值和之平均值，變異數 σ^2 可以解釋為所有像素點值與平均值的平方之平均，其中 σ^2 值可以化簡如下:

$$\sigma^2 = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n} = \frac{\sum_{i=1}^{n} \left(x_i^2 - 2\mu x_i + \mu^2 \right)}{n} = \frac{\sum_{i=1}^{n} x_i^2}{n} - 2\mu \frac{\sum_{i=1}^{n} x_i}{n} + \frac{\sum_{i=1}^{n} \mu^2}{n} = \frac{\sum_{i=1}^{n} x_i^2}{n} - 2\mu^2 + \mu^2$$

$$= \frac{\sum_{i=1}^{n} x_i^2}{n} - \mu^2 = \text{像素值平方和的平均} - \text{像素值平均的平方}$$

所以在此將變異數 σ^2 定義為一個「影像的區域變異量」，同時發現像素值平方和的平均也可以由「積分影像」的運算原理快速地求出，接下來我們繼續以上面設計好的 5×5 整數陣列資料為例，並以「積分影像的運算原理」為演算的核心，計算出此資料的區域變異量，數值轉化的過程呈現於電腦螢幕上如圖:



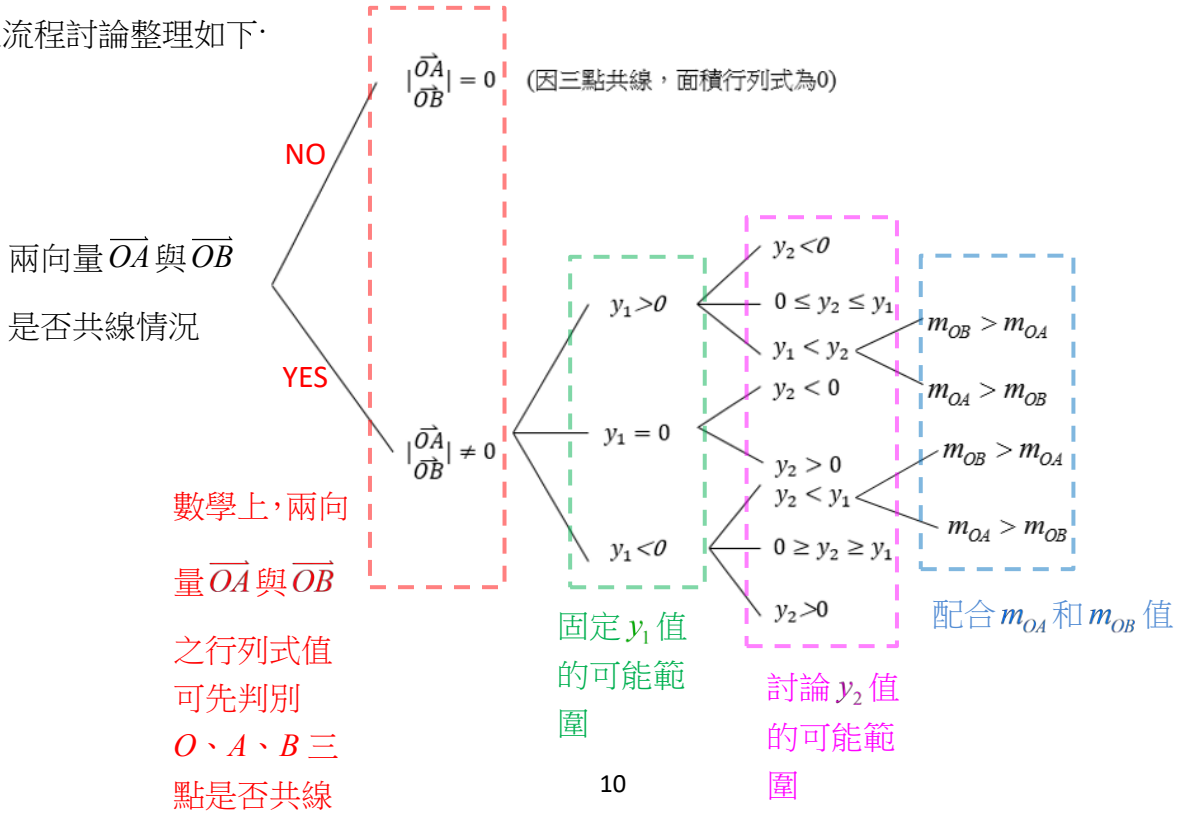
最後我們針對以上兩個累加資料，取最右下角的值分別為 $\sum_{i=1}^{25} x_i = 111$ 與 $\sum_{i=1}^{25} x_i^2 = 665$ ，即可快速計算出此資料的區域變異量 $\sigma^2 = \frac{665}{25} - (\frac{111}{25})^2 = 6.8864$ ；因此有了這些概念後，我們要使用資料越集中或分散的特性來應用於找尋最佳的幾何圖形來覆蓋 AED 的網格分佈圖。

五、探討台灣行政區域的形狀，從幾何圖形「三角形」發想

為了找尋最佳的幾何圖形，在此我們先觀察台灣的每一個行政區域之形狀，從北至南，每個行政區都有 AED 的分佈點，而因為每個行政區的形狀必非是方正的，形狀皆不一致，所以就算我們有網格過後的 AED 位置分佈圖，也很難在單一的行政區裡計算具有 AED 裝置的方格子，因此我們研究最適合的幾何圖形來覆蓋每一個行政區，如右圖，最後選定三角形來做為我們幾何圖的基礎與發想。

六、研究在二維平面上，推導出透過三點座標連成的三角形內覆蓋的方格總數之數學通式

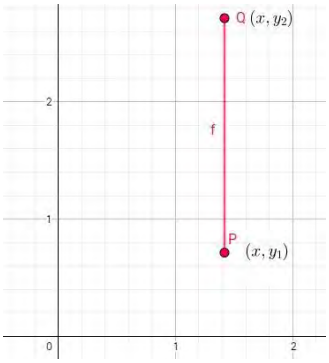
不失一般性，我們都由原點 O 出發，考慮 $A(x_1,y_1)$ 、 $B(x_2,y_2)$ ，其中 $0<x_1<x_2$ ，將這三點圍出的三角形區域定義為 ΔOAB 。為了要設計三角形內數方格子之演算法，我們必須要考慮平面上所有可能發生的三角形之三頂點情況，因此一開始先固定 A 點的 y 座標，討論分成三個區段: $y_1>0$、 $y_1=0$、 $y_1<0$ ，再討論 B 點的 y 座標可能情況，並配合 m_{OA} 和 m_{OB} 值，這樣就可以找出平面上所有可能的三角形之三頂點情況，總共會有十個數學通式，我將所有的結果之流程討論整理如下:



在研究上述的流程之前，首先要先了解一條鉛直線段上所通過的方格數量會有幾個，在此考慮鉛直線段的兩端點分別為 $P(x,y_1)$ 、 $Q(x,y_2)$ ，其中 $y_2 > y_1$ ，分兩種情況如下:

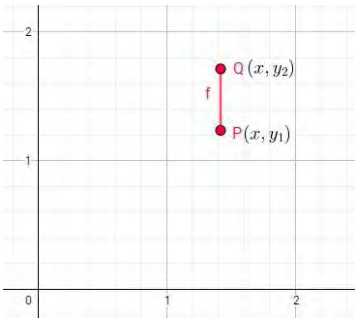
(一)兩端點不在同一塊方格裡面

將 y_2 值取向下取整數， y_1 值較小的向上取整數，分別用數學函數 $\lfloor y_2 \rfloor$ 與 $\lceil y_1 \rceil$ 表示，對應的程式函數即為 floor(y_2)與 ceil(y_1)，$\lfloor y_2 \rfloor - \lceil y_1 \rceil$ 之值就會是鉛直線段通過的方格數量。如右圖: $\lfloor y_2 \rfloor - \lceil y_1 \rceil = 2-1=1$ ，表示紅色鉛直線段 $\overline{PQ}$ 通過 1 個方格子。

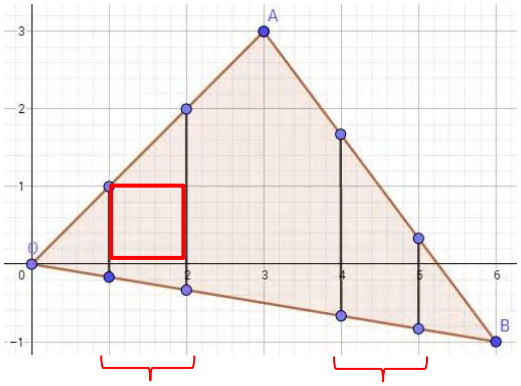


(二)兩端點在同一個方格內

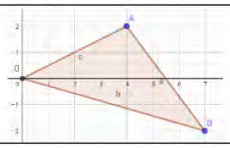
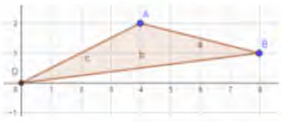
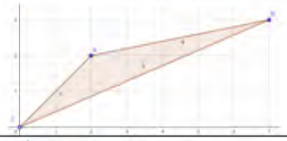
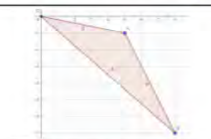
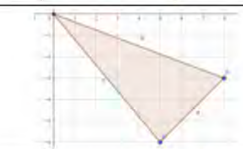
當兩端點在同一個方格時，則 $\lfloor y_2 \rfloor - \lceil y_1 \rceil = -1$ ，即沒通過任何方格，因此這種狀況在設計程式碼的時候要設定值為 0。如右圖: $\lfloor y_2 \rfloor - \lceil y_1 \rceil = 1-2=-1$ ，表示紅色鉛直線段 $\overline{PQ}$ 沒通過任何方格子。



有了這個簡單的概念後，要討論在各種狀況下的通式要如何產生，在一個三角形中的方格會夾在左右兩條鉛直線中，所以只要找到兩條直線重疊共同的部分，就可以進行判斷裡面有幾個方格。如右圖:前兩條鉛直線共同重疊的部份有包含一個方格子，但後面條鉛直線共同重疊的部份沒有包含任何方格子。



因為平面上任意的三角形會隨 $A(x_1,y_1)$ 、 $B(x_2,y_2)$ 決定，所以我們以 A 點為中間進行分割將三角形切成左右兩半部，左半部會利用到 $\overline{OA}$ 和 $\overline{OB}$ 的斜率，右半部則會用到 $\overline{AB}$ 和 $\overline{OB}$ 的斜率，以上皆會是我們在進行計算方格數要考慮的因素，因此我們將所有可能的結果整理出以下的表格(表一)，每種情況都會有一個圖片配合一個通式如下:

三角形在二維平面上之情況	計三角形內數方格子之數學通式
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}k \right\rfloor-\left\lceil \frac{y_2}{x_2}k \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1) \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}k \right\rfloor-\left\lceil \frac{y_2}{x_2}(k+1) \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}k \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}k \right\rfloor-\left\lceil \frac{y_1}{x_1}(k+1) \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1) \right\rceil-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}k \right\rfloor-\left\lceil \frac{y_2}{x_2}(k+1) \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}(k+1) \right\rceil-\left\lceil y_2-\frac{y_2}{x_2}k \right\rceil$
	$\sum_{k=1}^{x_1-1}0-\left\lceil \frac{y_2}{x_2}k \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1) \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}k \right\rfloor-0+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1) \right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}(k+1) \right\rfloor-\left\lceil \frac{y_1}{x_1}k \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}k \right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}(k+1) \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}(k+1) \right\rfloor-\left\lceil \frac{y_2}{x_2}k \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1) \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}(k+1) \right\rfloor-\left\lceil \frac{y_1}{x_1}k \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}k \right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}k \right\rfloor-\left\lceil \frac{y_1}{x_1}k \right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1) \right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k \right\rceil$

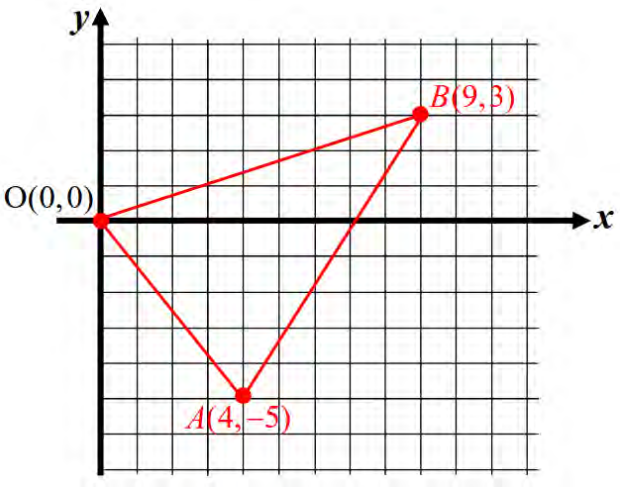
【表一：二維平面上所有可能的三角形與對應的三角形內覆蓋的方格總數之數學通式】

從上表中的十個數學通式之結構中，我們都可以發現有共同的特色，，以 A 點為中間進行分割將三角形切成左右兩半部，都是由左右兩個級數 $\sum$ 加在一起組成的，詳細通式的說明與公式推導，我們將研究的計算過程撰寫至附錄一。

七、將數學的理論程式化，設計一套計算出三角形內覆蓋方格總數之演算法

我們已經分析了在二維平面上，只要固定一點原點 O 與給定兩點座標 A 與 B，便可連成的三角形 ΔOAB ，就可以得出此三角內覆蓋的方格子數之數學通式，我們將其通式放入我們的計算方格子總數之演算法中，我們將研究的過程分以下步驟進行:

(一)在平面上，給定三頂點座標 $O(0,0)$ 、 $A(4,-5)$ 、 $B(9,3)$ ，圍出三角形 ΔOAB 如下圖:



(二)利用 C 語言撰寫計算三角形內的方格子總數之演算法

首先我們宣告四個整數型態變數 x_1、y_1、x_2、y_2，利用函數 scanf( )讓使用者可以輸入 A 點與 B 點的座標，如圖七:

```
printf("請輸入A點座標(x1,y1)與B點座標(x2,y2):   x1  y1  x2  y2\n");
scanf("%d   %d    %d    %d", &x_1,&y_1,&x_2,&y_2);
```

【圖七：使用者可以輸入 A 點與 B 點的座標之示意圖】

接著會判別 O、A、B 三點是否共線，即兩向量 $\overrightarrow{OA}$ 與 $\overrightarrow{OB}$ 之行列式值是否為 0，如果不為 0，則開始計算三角形 OAB 內方格子總數，如圖八:

```
if((x_1*y_2-x_2*y_1)!=0) //OA向量 與 OB向量 所形成的行列式值
{
    result1= sigma_left(x_1-1);
    result2= sigma_right(x_2-x_1-1);
    printf("三角形OAB內方格子總數為:%d\n",result1+result2);
}
else
{
    printf("三頂點OAB圍成面積為0，無法操作\n");
}
```

【圖八：判別 O、A、B 三點是否共線之示意圖】

其中圖八中的兩個函數 sigma_left( )與 sigma_right( )就是在分別計算方格子數，如圖九與圖十，根據上一段討論的數學理論，sigma_left( )計算的範圍是從 1 至 x_1 ，sigma_right( )計算的範圍是從 1 至 $x_2 - x_1 - 1$ ，在這兩個函數中，皆是會先判別 y_1 值，再開始由 y_2 值進行分類，總共會有十個運算通式，依據給定好的 *A* 點與 *B* 點之座標代入計算，最後再將這兩個計算出的結果分別存入整數變數 result1 與 result2，將兩值相加就是三角形內的方格子總數，以上面的三角形 ΔOAB 為例，覆蓋的方格子總數為 15 個，結果顯示於電腦螢幕上如圖十一。

```
int sigma_left(int a)
{
    int k;
    int term= 0;

    for(k=1 ; k<=a ; k++)
    {
        if(y_1>0)
        {
            if(y_2<0)
                term = term + check_func(floor((float)y_1*k/x_1)-ceil((float)y_2*k/x_2));
            if(y_2>=0 && y_1>=y_2)
                term = term + check_func(floor((float)y_1*k/x_1)-ceil((float)y_2*(k+1)/x_2));
            if(y_2>y_1)
            {
                if((float)y_2/x_2 > (float)y_1/x_1) //斜率OB > 斜率OA
                {
                    term = term + check_func(floor((float)y_2*k/x_2)-ceil((float)y_1*(k+1)/x_1));
                }
                else{
                    term = term + check_func(floor((float)y_1*k/x_1)-ceil((float)y_2*(k+1)/x_2));
                }
            }
```

【圖九：設計函數 sigma_left( )中的部份程式碼之示意圖】

```
int sigma_right(int b)
{
    int k;
    int term= 0;

    for(k=1 ; k<=b ; k++)
    {
        if(y_1>0)
        {
            if(y_2<0)
                term = term + check_func(floor(y_2-(float)(y_2-y_1)*k/(x_2-x_1))-ceil(y_2-(float)y_2*(k+1)/x_2));
            if(y_2>=0 && y_1>=y_2)
                term = term + check_func(floor(y_2-(float)(y_2-y_1)*k/(x_2-x_1))-ceil(y_2-(float)y_2*k/x_2));
            if(y_2>y_1)
            {
                if((float)y_2/x_2 > (float)y_1/x_1) //斜率OB > 斜率OA
                {
                    term = term + check_func(floor(y_2-(float)y_2*(k+1)/x_2)-ceil(y_2-(float)(y_2-y_1)*k/(x_2-x_1)));
                }
                else{
                    term = term + check_func(floor(y_2-(float)(y_2-y_1)*(k+1)/(x_2-x_1))-ceil(y_2-(float)y_2*k/x_2));
                }
            }
        }
```

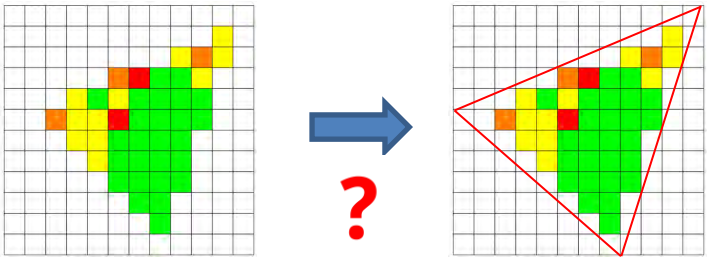
【圖十：設計函數 sigma_right( )中的部份程式碼之示意圖】

```
請輸入A點座標(x1,y1)與B點座標(x2,y2):
4 -5 9 3

~~~~~~利用三角形覆蓋演算法結果~~~~~~
三角形OAB內方格子總數為:15
```

【圖十一：實際輸入兩點座標 *A* 與 *B*，透過演算法跑出結果之示意圖】

八、找尋「最佳」的覆蓋方格子之三角形，應用於分析南投縣 AED 位置網格分佈圖

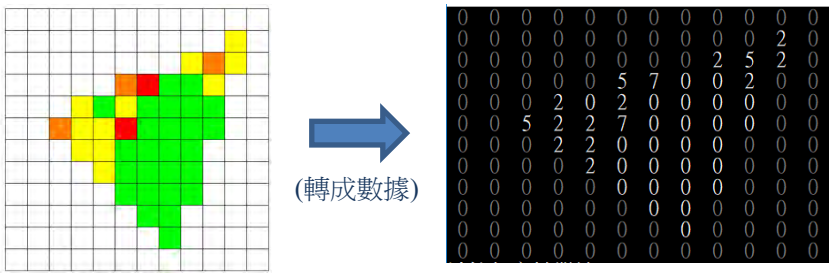
在此我們要先解釋「**最佳的覆蓋方格子之三角形**」，以南投縣 AED 位置分佈網格圖為例:



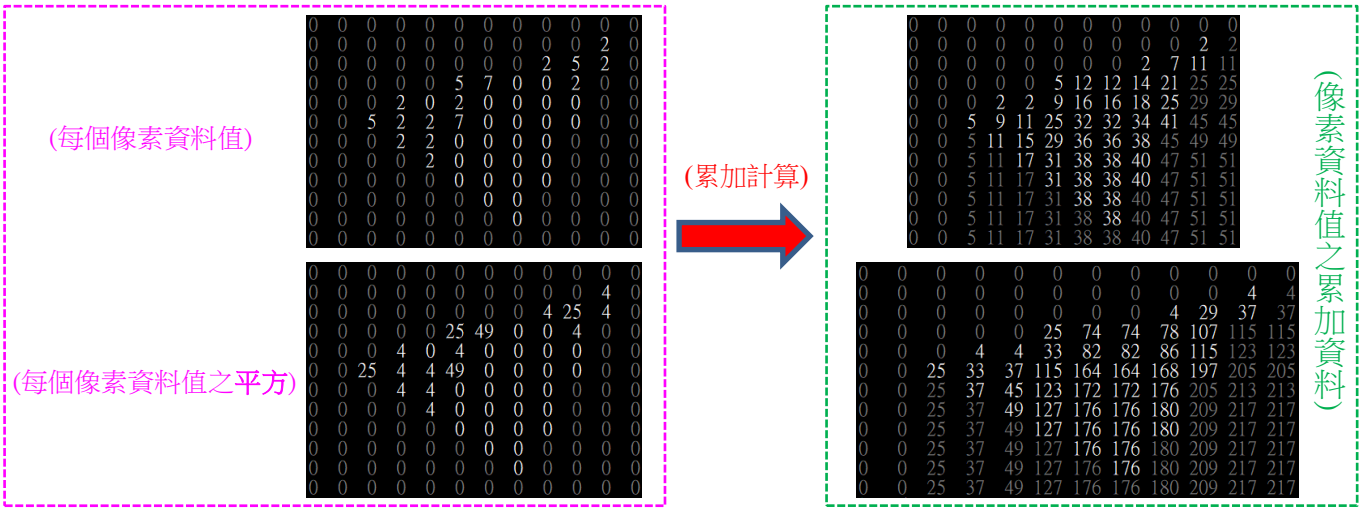
我們希望可以找到一個三角形，非常貼近 AED 位置的分佈圖，這時非必要討論的白色區域方格數會是最少的，進而更準確地覆蓋住 AED 位置的分佈，並將這樣的三角形的頂點座標化，有了頂點的座標，就可以代入我們設計的演算法，快速地計算出三角形內覆蓋的方格總數，因此我們要結合「**利用積分影像計算影像的區域變異量**」與「**三角形內覆蓋方格數之演算法**」來找尋最佳的覆蓋方格子之三角形，我們將研究的過程分為以下步驟進行:

(一)將影像數值轉化成一組數據:

首先我們先撰寫程式，先將南投 AED 網格分布點陣圖轉化成一組數據，其中紅色設定為 7，橘色設定為 5，黃色設定為 2，綠色設定為 0，白色也設定為 0，轉化的過程如下圖:



(二)利用積分影像原理分別計算像素資料值之累加資料:

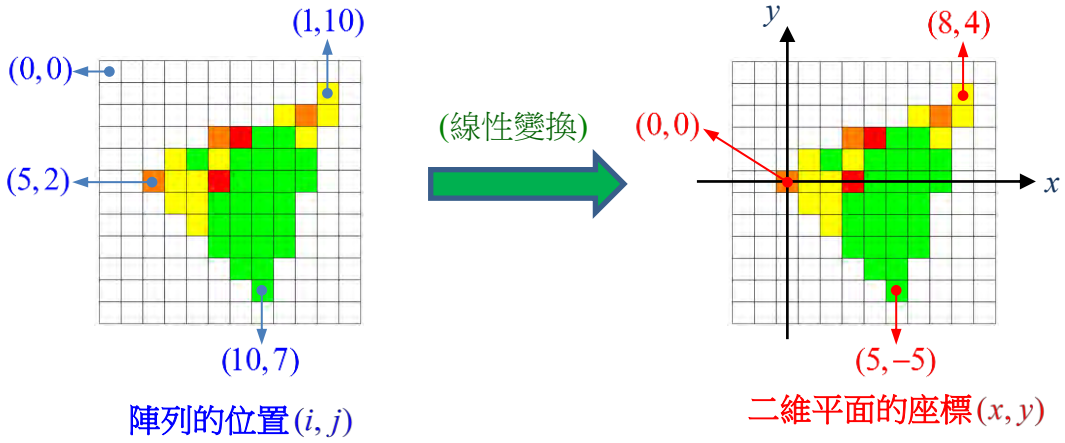


(三)利用數學上所學的線性變換，將陣列的位置 (i,j) 轉化成二維平面的座標 (x,y)：

以南投 AED 網格分佈圖為例，每個方格的中心都會是一陣列的位置 (i,j)，最左上的位置以 $(0,0)$ 開始，因此最左邊分佈的橘色方格的陣列位置為 $(5,2)$，最上方的黃色方格的陣列位置為 $(1,10)$，最下方的綠色方格的陣列位置為 $(10,7)$，接著我們利用數學的線性變換之旋轉矩陣 $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$，找出陣列位置 (i,j) 與二維平面上座標 (x,y) 的關係，以矩陣運算的方式呈現如下：

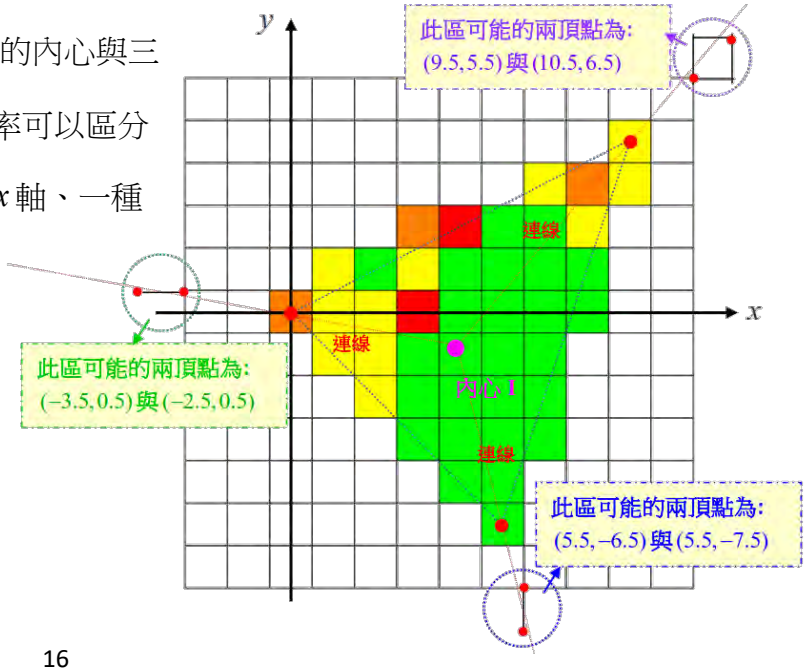
$$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} i \\ j \end{pmatrix}+\begin{pmatrix} -2 \\ 5 \end{pmatrix}=\begin{pmatrix} x \\ y \end{pmatrix}$$

將三個陣列的位置 $(5,2)$、$(10,7)$ 與 $(1,10)$ 代入上式可得二維平面上的三個對應的座標為 $(0,0)$、$(5,-5)$ 與 $(8,4)$，如圖所示：

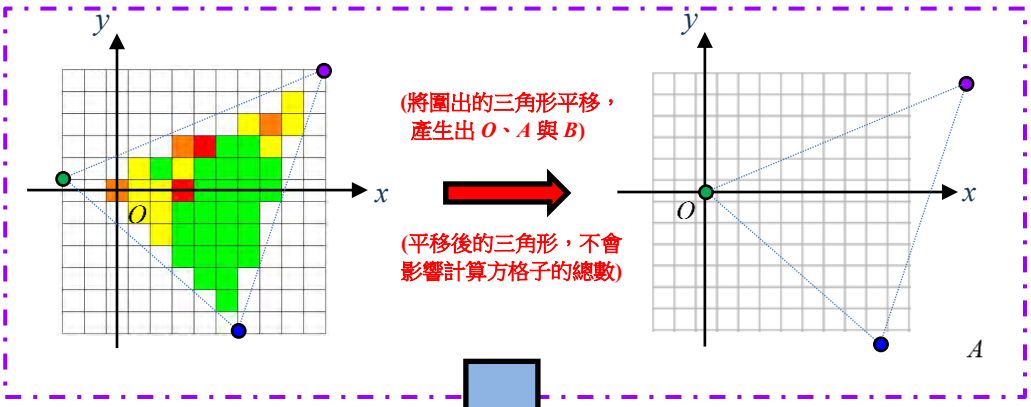


(四)決定「最佳」的覆蓋方格子之三角形的三頂點之研究：

有了以上的步驟後，在二維的平面上，我們開始研究如何建構可能的三角形來覆蓋南投 AED 網格分佈，首先以三個頂點座標 $(0,0)$、$(5,-5)$ 與 $(8,4)$ 所圍出的三角形出發，利用此三角形的內心與三頂點的連線，此時得出的三連線，依據斜率可以區分為三種狀況來比較，一種是較接近於平形 x 軸、一種是較接近於平形 y 軸、另外一種斜的歸類在與斜率是 ± 1 的線。在分類之後，觀察連線與它的斜率對應到的直線所交的位置，最靠近的兩個頂點就會作為設計可能覆蓋三角形的三頂點，如右圖所示：



分別從過這三個區域裡各選一個座標點，即可圍出一個**可能完全覆蓋 AED 分佈的三角形**，因此共有 $2\times2\times2$ 個可能的三角形，接下來要從 8 個三角形中選出一個最佳的三角形，使用「**積分影像計算影像的區域變異量**」與「**三角形內覆蓋方格數之演算法**」來進行處理，過程如下圖:



將這 8 組的座標 A 與座標 B 分別輸入至「**三角形內覆蓋方格數之演算法**」，計算出對應的三角形覆蓋的方格子總數，分別為 $N_1, N_2 \cdots N_8$

利用「積分影像」計算的兩組像素值的累加資料: $\sum x_i = 51$ **與** $\sum x_i^2 = 217$

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	2	2
0	0	0	0	0	0	0	0	2	7	11	11
0	0	0	0	0	5	12	12	14	21	25	25
0	0	0	2	2	9	16	16	18	25	29	29
0	0	5	9	11	25	32	32	34	41	45	45
0	0	5	11	15	29	36	36	38	45	49	49
0	0	5	11	17	31	38	38	40	47	51	51
0	0	5	11	17	31	38	38	40	47	51	51
0	0	5	11	17	31	38	38	40	47	51	51
0	0	5	11	17	31	38	38	40	47	51	51
0	0	5	11	17	31	38	38	40	47	51	51

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	4	4
0	0	0	0	0	0	0	0	4	29	37	37
0	0	0	0	0	25	74	74	78	107	115	115
0	0	0	4	4	33	82	82	86	115	123	123
0	0	25	33	37	115	164	164	168	197	205	205
0	0	25	37	45	123	172	172	176	205	213	213
0	0	25	37	49	127	176	176	180	209	217	217
0	0	25	37	49	127	176	176	180	209	217	217
0	0	25	37	49	127	176	176	180	209	217	217
0	0	25	37	49	127	176	176	180	209	217	217
0	0	25	37	49	127	176	176	180	209	217	217

利用三個數值:**方格子總數** $N_k\,(k=1,2,\cdots,8)$、$\sum x_i = 51$ **與** $\sum x_i^2 = 217$，代入變異數公式如下:

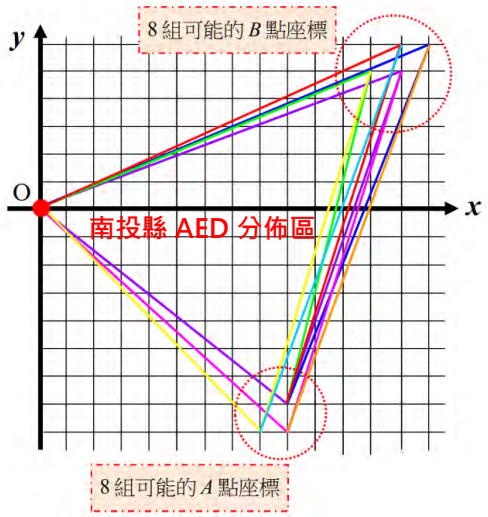
$$\sigma_k^{\,2} = \text{像素值平方和的平均} - \text{像素值平均的} = \frac{217}{N_k} - \left(\frac{51}{N_k}\right)^2, k = 1,2,\cdots,8$$

因為我們會有 8 個**可能完全覆蓋 AED 分佈的三角形**，所以會計算出對應的 8 個變異數，在上式中可以發現在分子 217 與 51 固定下，N_k 值會決定變異數的大小，如果 N_k 越大，表示三角形覆蓋到的白色方格越多，此時變異數會越小；相反地，如果 N_k 越小，表示三角形覆蓋到的白色方格越少，此時變異數會越大，因此我們會取變異數最大的三角形，即為**最佳的覆蓋方格子之三角形！**我們將計算與討論的結果，撰寫至我們的研究結果裡。

伍、 研究結果

一、找出最佳的覆蓋方格子之三角形:

(1)我們將這八組可能的三角形的三頂點平移至產生出 O、A、B 如下圖:



(2)再把這 8 組的座標 A 與座標 B 分別輸入至「**三角形內覆蓋方格總數之演算法**」，我們將計算結果整理如下表:

$O(0,0)$ $A(9,-7)$ $B(13,5)$	$O(0,0)$ $A(9,-7)$ $B(14,6)$	$O(0,0)$ $A(9,-8)$ $B(13,5)$	$O(0,0)$ $A(9,-8)$ $B(14,6)$
格子數N_1 : 47 變異數σ_1 = 3.3496	格子數N_2 : 52 變異數σ_2 = 3.2112	格子數N_3 : 51 變異數σ_3 = 3.2549	格子數N_4 : 58 變異數σ_4 = 2.9682
$O(0,0)$ $A(8,-7)$ $B(12,5)$	$O(0,0)$ $A(8,-7)$ $B(13,6)$	$O(0,0)$ $A(8,-8)$ $B(12,5)$	$O(0,0)$ $A(8,-8)$ $B(13,6)$
格子數N_5 : 42 變異數σ_5 = 3.6922	格子數N_6 : 46 變異數σ_6 = 3.4882	格子數N_7 : 49 變異數σ_7 = 3.3453	格子數N_8 : 55 變異數σ_8 = 3.0856

從上表可以發現當取到 O、A 與 B 三個頂點座標分別為 $(0,0)$ 、 $(8,-7)$ 與 $(12,5)$ 時，可以得知格子數 N_5 值最小，表示覆蓋到非必要討論的白色區域方格數會是最少的，此時對應的變異數 σ_5 值最大，因此我們會取變異數最大的三角形，此三角形為**最佳的覆蓋方格子之三角形!**

二、將南投 AED 網格分布點陣圖匯入設計好的演算法中進行計算:(完整程式碼於附錄二)

測試的點陣圖 BMP 資料

尺寸:320×320

檔名: nantou_aed_distribution.bmp

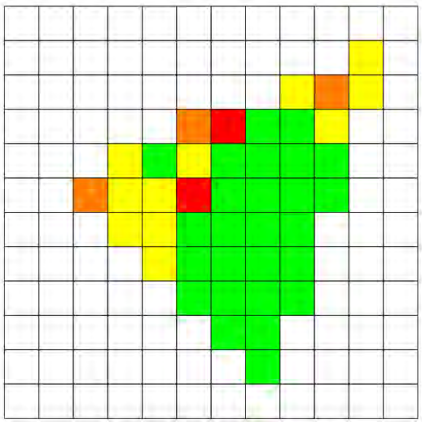
顏色意義:

紅色的方格表示 AED 裝置分佈非常密集。

橘色的方格表示次多。

黃色表示少許。

綠色方格表示無 AED 裝置的位置分佈之地。



網格化後的南投縣 AED 位置分佈圖

利用三角形內的方格子總數之演算法跑出的結果

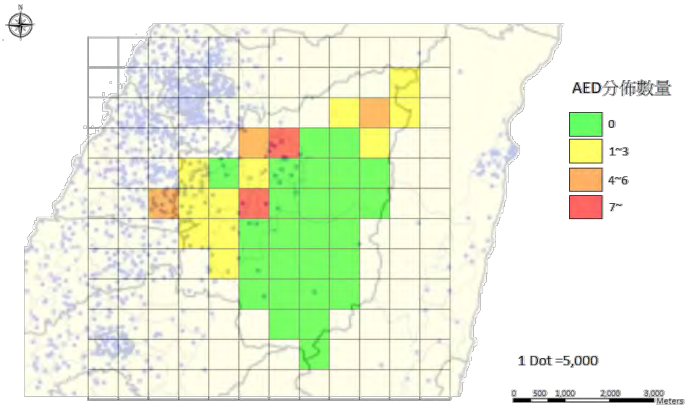
使用者可分別輸入最佳三角形的座標 $A(8,-7)$ 與 $B(12,5)$，經演算法計算後可得以下結果:

由各顏色的分佈統計比例可知，綠色方格子所佔的比例最高，表示這一區中，無 AED 裝置的位置分佈偏高，應該要多增設 AED 裝置，讓民眾馬上就能反應並尋找到 AED 裝置，以備不時之需，期望掌握 4 至 6 分鐘「黃金救命時間」，提高患者救活率，以達到醫療資源分配的均勻。

```
請輸入A點座標(x1,y1)與B點座標(x2,y2):
8 -7 12 5

~~~~~~利用演算法計算~~~~~~
三角形OAB內方格子總數為:42


~~~~~~方格的分佈概況~~~~~~
紅色方格子數:2
橘色方格子數:3
黃色方格子數:11
綠色方格子數:26

~~~~~~各顏色分布的比例~~~~~~
紅色方格子數佔的比例:0.047619
橘色方格子數佔的比例:0.071429
黃色方格子數佔的比例:0.261905
綠色方格子數佔的比例:0.619048
請按任意鍵繼續 . . .
```

南投縣 AED 位置分佈圖與南投縣人口分佈點狀圖之疊圖

在人口點狀圖中，每個單位點表示 5000 人，在進行疊圖過後，我們可以看到有許多的綠色格子是代表沒有裝置的，代表某些區域有人口但是沒有裝置，其他的地方有些是有裝置但是沒有人口，所以需要重新規劃 AED 裝置的放置情況。



陸、 討論

一、設計利用方格子反推出可完全包含它們的三角形演算法

我們這次的研究主要是將 AED 分佈圖放置二維平面上，給定三個頂點的座標，把座標上的數值代入數方格子之通式後，計算出方格子數量，就可以分析 AED 分佈的狀況；但若以相反的概念，在平面上如果給定一些有顏色的方格子，則可以深入探究要如何用最有效率的方式框住所有顏色的方格子之三角形，是一個值得我們日後討論與研究的問題。

二、將計算方格數量之演算法延伸與推廣至 *n* 邊形

在生活中的幾何圖形有很多種，並不一定皆是三角形，因此若能設計出一個通式，可以在當邊長數量為任意正整數 *n* 時，內部所包含的方格子數量，撰寫出泛用度更高的 *n* 邊形之演算法；另外如討論(一)的想法，方格子也可用其他種類的多邊形，在日後我們在探討幾何圖形覆蓋時，可以設計出更有效率的計數之演算法。

柒、 結論

我們利用了計算三角形內方格子數量的通式，製作了可以計算方格數量的演算法，克服了電腦數方格時需要龐大的計算，只需要利用數學通式做出的演算法，節省一格一格數的時間，使運作的反應時間更短，除此之外本研究也提供出如何將理論層面轉化成實際應用層面之研究方法，期待將此轉化的模式作成未來更實用且有效率的演算的技術，應用至其他領域，解決其它更多種類的問題。

捌、 參考資料與附錄

一、許志農、黃森山教授。龍騰版高中數學課本第三冊第二章「直線與圓」。

二、陳國川教授。龍騰版高中地理課本第一冊第五章「地理資訊系統」。

三、劉洲溶、蘇淵源、翟家甫。全華版高中資訊科技概論課本第五章「電腦與問題解決」。

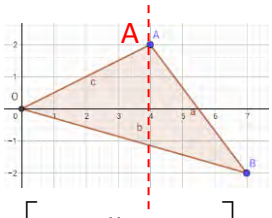
四、政府資料開放平台。2017 年 12 月。取自 http://data.gov.tw/

附錄一：三角形內覆蓋的方格總數之數學通式推導

一、當 $y_1 > 0$ 之情況下:

(一) $y_2 < 0$ 的條件下:

如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為負；右半部斜率乘積為正，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數} = \sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}k \right\rfloor - \left\lceil \frac{y_2}{x_2}k \right\rceil + \sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2 - \frac{y_2-y_1}{x_2-x_1}k \right\rfloor - \left\lceil y_2 - \frac{y_2}{x_2}(k+1) \right\rceil$$

(二) $0 \le y_2 \le y_1$ 的條件下:

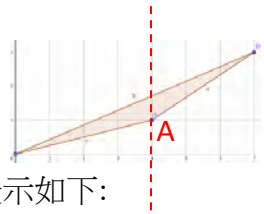
如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為正；右半部斜率乘積為正，因此可以由數學式表示如下:

$$\text{三角形內覆蓋方格總數} = \sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}k \right\rfloor - \left\lceil \frac{y_2}{x_2}(k+1) \right\rceil + \sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2 - \frac{y_2-y_1}{x_2-x_1}k \right\rfloor - \left\lceil y_2 - \frac{y_2}{x_2}k \right\rceil$$

(三) $y_1 < y_2$ 的條件下:

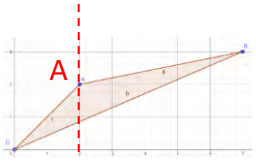
在這種狀況中，可能因為 $\overline{OB}$ 和 $\overline{OA}$ 的斜率大小，影響 y 值較大的點在不同的直線上，所以可以分成兩種狀況:

1. 當 $m_{OB} > m_{OA}$ 時，如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為正；右半部斜率乘積為正，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數} = \sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}k \right\rfloor - \left\lceil \frac{y_1}{x_1}(k+1) \right\rceil + \sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2 - \frac{y_2}{x_2}(k+1) \right\rfloor - \left\lceil y_2 - \frac{y_2-y_1}{x_2-x_1}k \right\rceil$$

2. 當 $m_{OA} > m_{OB}$ 時，如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為正；右半部斜率乘積為正，因此可以由數學式表示如下:

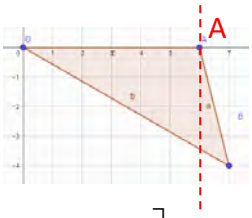


$$\text{三角形內覆蓋方格總數} = \sum_{k=1}^{x_1-1}\left\lfloor \frac{y_1}{x_1}k \right\rfloor - \left\lceil \frac{y_2}{x_2}(k+1) \right\rceil + \sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2 - \frac{y_2-y_1}{x_2-x_1}(k+1) \right\rfloor - \left\lceil y_2 - \frac{y_2}{x_2}k \right\rceil$$

二、當 $y_1=0$ 之情況下:

(一) $y_2<0$ 的條件下

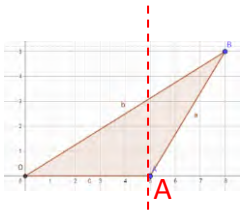
如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為零；右半部斜率乘積為正，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數}=\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}k\right\rfloor+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rceil$$

(二) $y_2>0$ 的條件下

如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為零；右半部斜率乘積為正，因此可以由數學式表示如下:



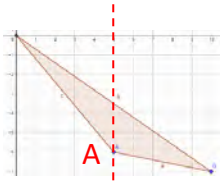
$$\text{三角形內覆蓋方格總數}=\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}k\right\rfloor-0+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rceil$$

三、當 $y_1<0$ 之情況下:

(一) $y_2<y_1$ 的條件下

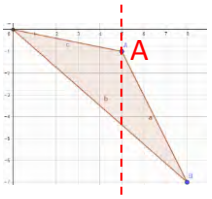
在這種狀況中，可能因為 $\overline{OB}$ 和 $\overline{OA}$ 的斜率大小，影響 y 值較大的點在不同的直線上，所以可以分成兩種狀況:

1.當 $m_{OB}>m_{OA}$ 時，如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為正；右半部斜率乘積為正，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數}=\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil\frac{y_1}{x_1}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}k\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}(k+1)\right\rceil$$

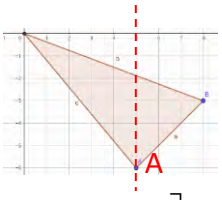
2.當 $m_{OA}>m_{OB}$ 時，如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為正；右半部斜率乘積為正，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數}=\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_1}{x_1}(k+1)\right\rfloor-\left\lceil\frac{y_2}{x_2}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1)\right\rceil$$

(二) $0 \geq y_2 \geq y_1$ 的條件下

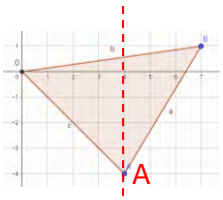
如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為正；右半部斜率乘積為負，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數} = \sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}(k+1) \right\rfloor - \left\lceil \frac{y_1}{x_1}k \right\rceil + \sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2 - \frac{y_2}{x_2}k \right\rfloor - \left\lceil y_2 - \frac{y_2-y_1}{x_2-x_1}k \right\rceil$$

(三) $y_2 > 0$ 的條件下

如右圖所示，以通過 A 點作鉛直線，左半部斜率乘積為負；右半部斜率乘積為正，因此可以由數學式表示如下:



$$\text{三角形內覆蓋方格總數} = \sum_{k=1}^{x_1-1}\left\lfloor \frac{y_2}{x_2}k \right\rfloor - \left\lceil \frac{y_1}{x_1}k \right\rceil + \sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2 - \frac{y_2}{x_2}(k+1) \right\rfloor - \left\lceil y_2 - \frac{y_2-y_1}{x_2-x_1}k \right\rceil$$

附錄二:完整程式碼

```
#include <stdio.h>
#include<stdlib.h>
#include <windows.h>
#include<math.h>

int x_1,y_1,x_2,y_2;
int OutputImage[320*320*3];
int block[12][12][3];
int output[12][12];
int integral[12][12] ={0};
int output2[12][12];
int integral2[12][12] ={0};
int check_func(int);
int sigma_right(int);
int sigma_left(int);
int square(int);

struct RGB
{
   unsigned char R;
   unsigned char G;
   unsigned char B;
};

struct HSV
{
   double H;
   double S;
   double V;
};

struct RGB data;
struct HSV value;

int main()
{
   FILE *image_input;
   int i, j, k;
   int header[54];
   const int row = 320;
   const int col = 320;
   int color[4]={0,0,0,0};//[0]:GREEN [1]:YELLOW [2]:ORANGE [3]:RED
   int result1=0,result2=0;

   HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);

   printf("請輸入A點座標(x1,y1)與B點座標(x2,y2):\n");
   scanf("%d   %d    %d    %d", &x_1,&y_1,&x_2,&y_2);
   printf("\n");

   if((x_1*y_2-x_2*y_1)!=0)//OA向量與OB向量所形成的行列式值
   {
      result1= sigma_left(x_1-1);
      result2= sigma_right(x_2-x_1-1);
   }
   else
   {
      printf("三頂點OAB圍成面積為0，無法操作\n");
   }

if ((image_input =
fopen("nantou_aed_distribution.bmp","rb")) == NULL)
{
     printf( "\n Can't open file %s","test.bmp");
}
else
{
   for(i=0;i<54;i++)
   {
      header[i] = fgetc(image_input);
   }

   for(i=0; i<(row*col*3); i++)
   {
      OutputImage[i] = fgetc(image_input);
   }

   fclose(image_input);

   for(i=0; i<12; i++)
   {
      for(j=0; j<12; j++)
      {
         for(k=0; k<3; k++)
         {
            block[i][j][k]=
            OutputImage[(320*13*(25-2*(i+1))
            +13*(2*(j+1)-1))*3+k];
         }
      }
   }

   SetConsoleTextAttribute(hConsole, 15);
   printf("南投縣AED網格圖的像素值:\n");
   for(i=0; i<12; i++)
   {
      for(j=0; j<12; j++)
      {
         data.R = block[i][j][2];
         data.G = block[i][j][1];
         data.B = block[i][j][0];
         value = RGBToHSV(data);

         if(value.S !=0)
         {
            if(value.H ==0) //red
            {
               color[3]++;
               output[i][j] = 7;
               SetConsoleTextAttribute(hConsole, 15);
               printf("%2d ",output[i][j]);
            }
            else if(value.H < 60 && value.H > 20 )//orange
            {
               color[2]++;
               output[i][j] = 5;
               SetConsoleTextAttribute(hConsole, 15);
               printf("%2d ",output[i][j]);
            }
```

```
        else if(value.H == 60){/yellow
           color[1]++;
           output[i][j] = 2;
           SetConsoleTextAttribute(hConsole, 15);
           printf("%2d ",output[i][j]);
        }else if(value.H == 120){//green
           color[0]++;
           output[i][j] = 0;
           SetConsoleTextAttribute(hConsole, 15);
           printf("%2d ",output[i][j]);
        }else{
           output[i][j] = 0;//white
           SetConsoleTextAttribute(hConsole,8);
           printf("%2d ",output[i][j]);
        }
     }
     printf("\n");
  }

  printf("\n\n");
  SetConsoleTextAttribute(hConsole, 15);
  printf("南投縣AED網格圖的像素累加值:\n");

  integral[0][0] = output[0][0];

for(i=0; i<12; i++)
{
  for(j=0; j<12; j++)
  {
     if(i==0 && j>0){
     integral[0][j] = integral[0][j-1] + output[0][j];
     }else if(j==0 && i>0){
     integral[i][0] = integral[i-1][0] + output[i][0];
     }else if(i>0 && j>0){
     integral[i][j] = integral[i-1][j] +
     integral[i][j-1] - integral[i-1][j-1]
     +output[i][j];}
  }
}

  for(i=0;i<12;i++)
  {
     for (j=0;j<12;j++)
     {
        data.R = block[i][j][2];
        data.G = block[i][j][1];
        data.B = block[i][j][0];
        value = RGBToHSV(data);

        if(value.S !=0)
        {
           SetConsoleTextAttribute(hConsole, 15);
           printf("%2d ",integral[i][j]);
        }else{
           SetConsoleTextAttribute(hConsole,8);
           printf("%2d ",integral[i][j]);
        }
     }
     printf("\n");
  }
}
```

```
  printf("\n\n");
  SetConsoleTextAttribute(hConsole, 15);
  printf("南投縣AED網格圖的像素值平方:\n");

  for(i=0;i<12;i++)
  {
     for (j=0;j<12;j++)
     {
     output2[i][j]=square(output[i][j]);
     }
  }

  for(i=0;i<12;i++)
  {
     for (j=0;j<12;j++)
     {
        data.R = block[i][j][2];
        data.G = block[i][j][1];
        data.B = block[i][j][0];
        value = RGBToHSV(data);

        if(value.S !=0)
        {
           SetConsoleTextAttribute(hConsole, 15);
           printf("%2d ",output2[i][j]);
        }else{
           integral[i][j] = 0;//white
           SetConsoleTextAttribute(hConsole,8);
           printf("%2d ",output2[i][j]);
        }
     }
     printf("\n");
  }

  integral2[0][0] = output2[0][0];

  for(i=0;i<12;i++)
  {
     for (j=0;j<12;j++)
     {
        if(i==0 && j>0){
        integral2[0][j]=integral2[0][j-1]+output2[0][j];
        }else if(j==0 && i>0){
        integral2[i][0]=integral2[i-1][0]+output2[i][0];
        }else if(i>0 && j>0){
        integral2[i][j] = integral2[i-1][j] +
        integral2[i][j-1] - integral2[i-1][j-1]
        +output2[i][j];}
     }
  }

  printf("\n\n");
  SetConsoleTextAttribute(hConsole, 15);
  printf("南投縣AED網格圖的像素平方累加值:\n");
```

```
    for(i=0;i<12;i++)
    {
       for (j=0;j<12;j++)
       {
          data.R = block[i][j][2];
          data.G = block[i][j][1];
          data.B = block[i][j][0];
          value = RGBToHSV(data);

          if(value.S !=0)
          {
             SetConsoleTextAttribute(hConsole, 15);
             printf("%3d ",integral2[i][j]);
          }else{
             SetConsoleTextAttribute(hConsole,8);
             printf("%3d ",integral2[i][j]);
          }
       }
       printf("\n");
    }

    printf("\n\n");
    SetConsoleTextAttribute(hConsole, 15);
    printf("\n~~~~~~利用三角形覆蓋演算法結果~~~~~~\n");
    printf("三角形OAB內方格子總數
    為:%d\n\n",result1+result2);

    printf("\n~~~~~~方格的分佈概況~~~~~~\n");
    printf("紅色方格子數:%d\n",color[3]);
    printf("橘色方格子數:%d\n",color[2]);
    printf("黃色方格子數:%d\n",color[1]);
    printf("綠色方格子數:%d\n",color[0]);

    printf("\n~~~~~~各顏色分布的比例~~~~~~\n");
    printf("紅色方格子數佔的比
    例:%f\n",(float)color[3]/(result1+result2));
    printf("橘色方格子數佔的比
    例:%f\n",(float)color[2]/(result1+result2));
    printf("黃色方格子數佔的比
    例:%f\n",(float)color[1]/(result1+result2));
    printf("綠色方格子數佔的比
    例:%f\n",(float)color[0]/(result1+result2));

    system("pause");
}

static double Min(double a, double b) {
     return a <= b ? a : b;
}

static double Max(double a, double b) {
     return a >= b ? a : b;
}
```

```
struct HSV RGBToHSV(struct RGB rgb) {
   double delta, min;
   double h = 0, s, v;

   min = Min(Min(rgb.R, rgb.G), rgb.B);
   v = Max(Max(rgb.R, rgb.G), rgb.B);
   delta = v - min;

   if (v == 0.0)
      s = 0;
   else
      s = delta / v;

   if (s == 0)
   h = 0.0;
   else
   {
      if (rgb.R == v)
         h = (rgb.G - rgb.B) / delta;
      else if (rgb.G == v)
         h = 2 + (rgb.B - rgb.R) / delta;
      else if (rgb.B == v)
         h = 4 + (rgb.R - rgb.G) / delta;

         h *= 60;

         if (h < 0.0)
         h = h + 360;
   }

   struct HSV hsv;
   hsv.H = h;
   hsv.S = s;
   hsv.V = v / 255;

   return hsv;
}

int square(int k)
{
   return k*k;
}

int check_func(int x) //check:"x>=1"
{
   if(x>=1)
   {
      return x;
   }
   else
   {
      return 0;
   }
}
```

```
int sigma_left(int a)
{
   int k;
   int term= 0;

   for(k=1 ; k<=a ; k++)
   {
      if(y_1>0)
      {
         if(y_2<0)
            term = term + check_func(floor((float)y_1*k/x_1)-ceil((float)y_2*k/x_2));
         if(y_2>=0 && y_1>=y_2)
            term = term + check_func(floor((float)y_1*k/x_1)-ceil((float)y_2*(k+1)/x_2));
         if(y_2>y_1)
         {
            if((float)y_2/x_2 > (float)y_1/x_1) //斜率OB > 斜率OA
            {
                  term = term + check_func(floor((float)y_2*k/x_2)-ceil((float)y_1*(k+1)/x_1));
            }
            else{
                  term = term + check_func(floor((float)y_1*k/x_1)-ceil((float)y_2*(k+1)/x_2));
            }
         }
      }

      if(y_1==0)
      {
         if(y_2<0)
         {
            term = term + check_func(0-ceil((float)y_2*k/x_2));
         }
         else{
            term = term + check_func(floor((float)y_2*k/x_2)-0);
         }
      }

      if(y_1<0)
      {
         if(y_2<y_1)
         {
            if((float)y_2/x_2 > (float)y_1/x_1) //斜率OB > 斜率OA
            {
                  term = term + check_func(floor((float)y_2*(k+1)/x_2)-ceil((float)y_1*k/x_1));
            }
            else{
                  term = term + check_func(floor((float)y_1*(k+1)/x_1)-ceil((float)y_2*k/x_2));
            }
         }
         if(0>=y_2 && y_2>=y_1)
            term = term + check_func(floor((float)y_2*(k+1)/x_2)-ceil((float)y_1*k/x_1));
         if(y_2>0)
            term = term + check_func(floor((float)y_2*k/x_2)-ceil((float)y_1*k/x_1));

      }
   }

   return term;
}
```

```
int sigma_right(int b)
{
   int k;
   int term= 0;

   for(k=1 ; k<=b ; k++)
   {
      if(y_1>0)
      {
         if(y_2<0)
            term = term + check_func(floor(y_2-(float)(y_2-y_1)*k/(x_2-x_1))-ceil(y_2-(float)y_2*(k+1)/x_2));
         if(y_2>=0 && y_1>=y_2)
            term = term + check_func(floor(y_2-(float)(y_2-y_1)*k/(x_2-x_1))-ceil(y_2-(float)y_2*k/x_2));
         if(y_2>y_1)
         {
            if((float)y_2/x_2 > (float)y_1/x_1) //斜率OB > 斜率OA
            {
                  term = term + check_func(floor(y_2-(float)y_2*(k+1)/x_2)-ceil(y_2-(float)(y_2-y_1)*k/(x_2-x_1)));
            }
            else{
                  term = term + check_func(floor(y_2-(float)(y_2-y_1)*(k+1)/(x_2-x_1))-ceil(y_2-(float)y_2*k/x_2));
            }
         }
      }

      if(y_1==0)
      {
         if(y_2<0)
         {
            term = term + check_func(floor(y_2-(float)(y_2-y_1)*k/(x_2-x_1))-ceil(y_2-(float)y_2*(k+1)/x_2));
         }
         else{
            term = term + check_func(floor(y_2-(float)y_2*(k+1)/x_2)-ceil(y_2-(float)(y_2-y_1)*k/(x_2-x_1)));
         }
      }

      if(y_1<0)
      {
         if(y_2<y_1)
         {
            if((float)y_2/x_2 > (float)y_1/x_1) //斜率OB > 斜率OA
            {
                  term = term + check_func(floor(y_2-(float)y_2*k/x_2)-ceil(y_2-(float)(y_2-y_1)*(k+1)/(x_2-x_1)));
            }
            else{
                  term = term + check_func(floor(y_2-(float)(y_2-y_1)*k/(x_2-x_1))-ceil(y_2-(float)y_2*(k+1)/x_2));
            }
         }

         if(0>=y_2 && y_2>=y_1)
            term = term + check_func(floor(y_2-(float)y_2*k/x_2)-ceil(y_2-(float)(y_2-y_1)*k/(x_2-x_1)));
         if(y_2>0)
            term = term + check_func(floor(y_2-(float)y_2*(k+1)/x_2)-ceil(y_2-(float)(y_2-y_1)*k/(x_2-x_1)));
      }
   }

   return term;
}
```

【評語】052504

本研究使用政府資料開放平臺網提供的 AED 分佈位置資料，將資料導入地理資訊系統，基於實際 AED 網格化之分佈情況，隨不同台灣行政區域形狀，將數學上所學之幾何圖形和數學原理設計覆蓋方格總數的演算法，來探討是否需要再某些區域增設 AED 裝置。該研究包含演算法設計、實作、分析。該團隊對該問題有詳細之研究以及分析，更進一步解決問題。

作品海報

壹、研究動機

健護課提到AED裝置在我們生活當中扮演了相當重要的醫療角色，若能夠在發生猝死時取得AED，我們便可以在實施CPR的時候同時進行AED急救，大大縮減了過去等待醫護人員到達才施予電擊除顫的時間差，而越早進行電擊、則病人搶救成功的機率也越大。

根據政府資料開放平臺網提供的AED位置資訊顯示，其實台灣的AED裝置之分佈並不平均，將這些資料匯入地理資訊系統(QGIS)，AED位置會呈現在網格圖上，則我們觀察到分佈位置的網格圖可以依據台灣行政區域的形狀(如:三角形)，利用幾何圖形來覆蓋它並計算在幾何圖形內最多覆蓋的方格子數，即可探討AED的分佈之方格子數，最後利用電腦程式語言C設計覆蓋方格子數的演算法，給出一個有效且正確的計數方格子數之方法。期待透過撰寫程式來改善AED裝置不均的分佈，藉以達到降低該類傷病患到院前死亡率之目標。

貳、研究目的

透過數學理論與電腦演算法的結合，製作出可以利用三頂點座標計算方格數量的演算法來解決問題，其中我們利用QGIS系統將取得的資料製作成網格圖，之後將影像資料放入演算法轉換成數據，再利用積分影像原理跟統計學的變異數來把它進行統計，最後嘗試找出三角形內部方格總數的數學通式與如何找出最佳覆蓋方格子之三角形三頂點的方法，將研究的理論程式化，藉此統計AED分佈的狀況，來改善AED裝置不均的分佈之情況。

參、研究設備及器材

一、程式語言C與編譯程式語言系統

C語言是一種被廣泛使用電腦程式語言，從硬體、軟體的設計到韌體、系統的開發；Microsoft Visual C++ 2010 Express是一套由微軟公司開發的免費整合開環境，我們作為這次研究撰寫程式的工具。

二、數學軟體GeoGebra

GeoGebra 是一套動態數學教育軟體，它不但包含動態幾何尺規作圖、幾何轉換等功能，還加入函數繪圖、代數運算和基本微積分功能。我們用來繪製二維平面中三頂點連的三角形。

三、數學運算軟體Mathematica

Mathematica 是一套整合數字以及符號運算的數學工具軟體，廣泛使用不同的領域上，本研究藉由Mathematica高速運算特色，來驗證我們的數學通式。

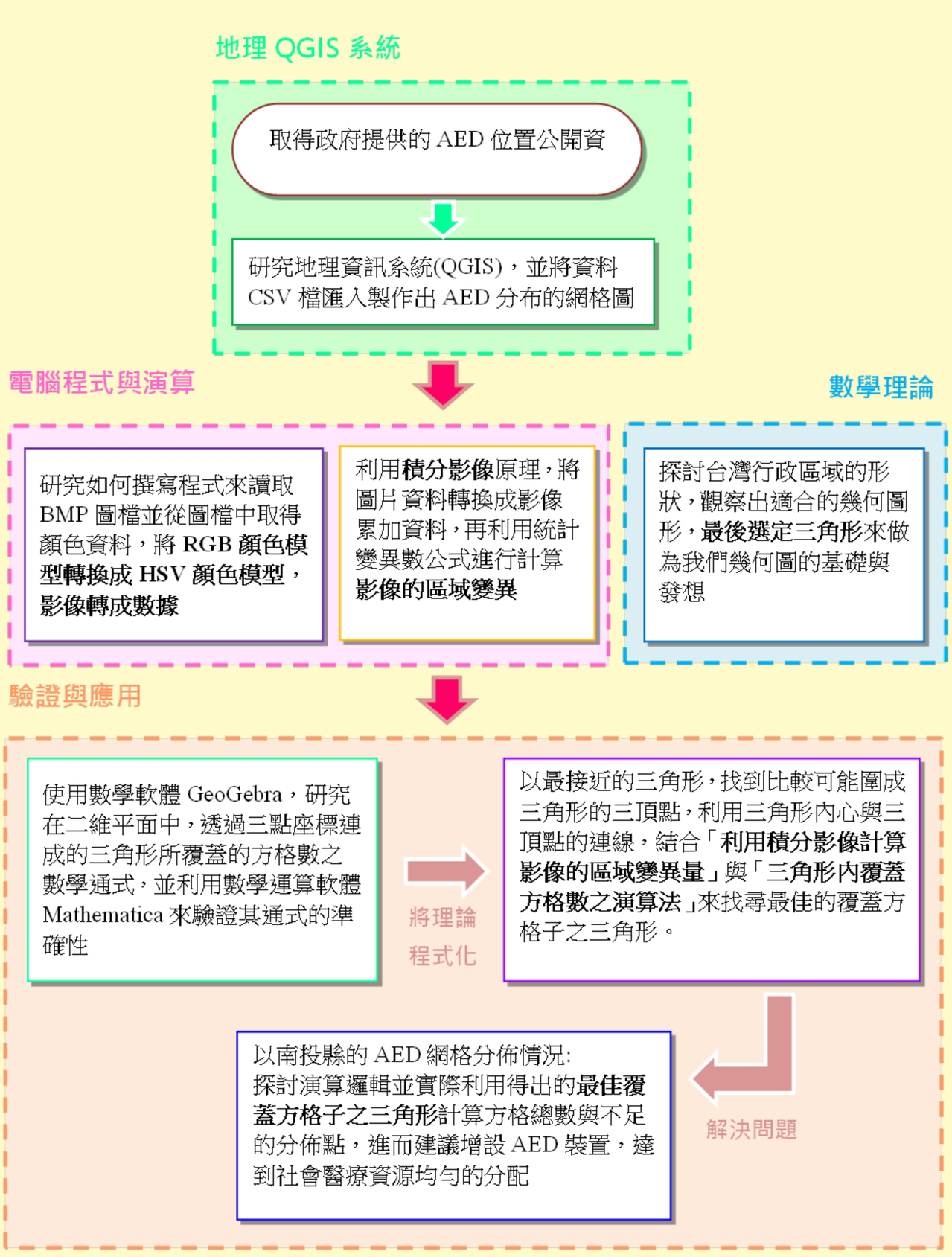
四、地理資訊系統(QGIS)

原稱Quantum GIS，是一個開放且免費的地理資訊系統，處理許多的地圖文件，包括地形圖、分布圖與多重疊圖等，我們利用它來處理AED公開資料。

五、筆電與桌上型電腦

肆、研究過程或方法

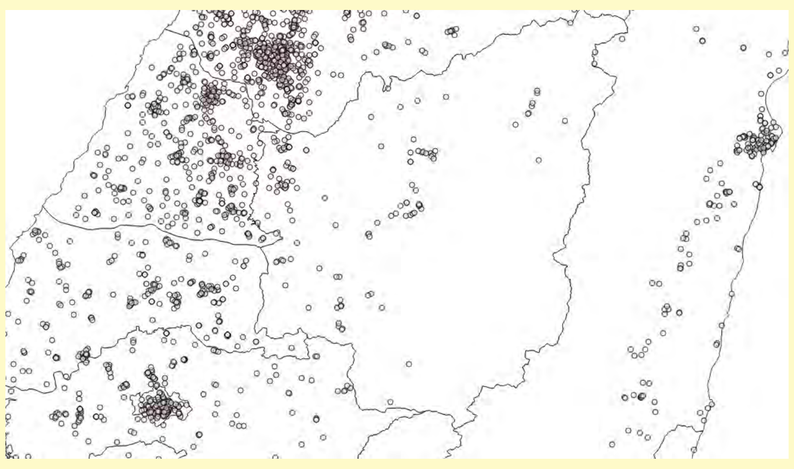
一、 研究流程



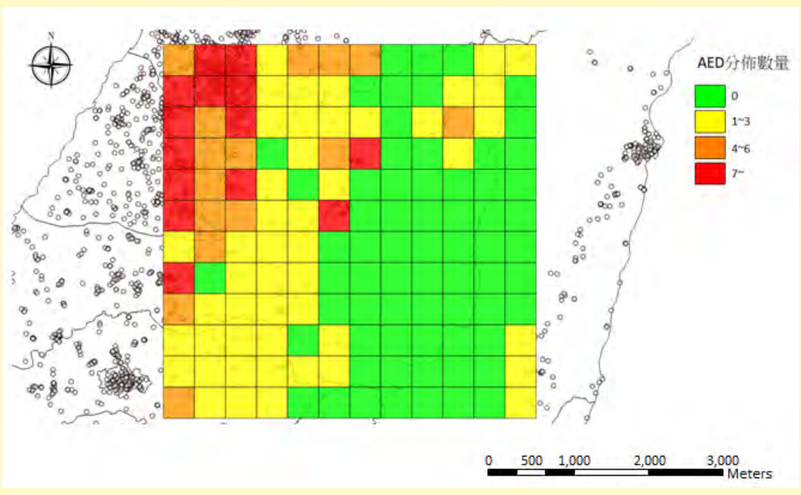
【圖一：研究流程圖】

二、利用QGIS系統繪製AED分佈網格圖

首先我們先取得政府資料開放平臺網提供的AED位置公開資料，資料檔為CSV檔，將所有的AED位置與台灣的行政區圖進行疊圖，可以清楚觀察到從北至南的行政區域中，AED裝置的分佈是不均勻地，甚至在台灣偏中間的位置，如:南投山區，幾乎是沒有的，因此我們必須重新規劃AED裝置的分佈，以確保民眾在需要時，可以即時享有醫療的資源。我們選擇先以台灣中部地區「南投縣」作為基礎，開始進行製做AED裝置的分佈點狀圖(圖二）。接著再利用QGIS系統內的網格系統進行網格化（圖三），將一個方格子覆蓋AED分佈點之密集程度作為分類的依據，轉化成顏色，以顏色區分每個地方不同的密集程度依序為「紅 > 橙 > 黃 > 綠」。



【圖二：台灣AED裝置的分佈之示意圖】



【圖三：台灣AED裝置之網格化示意圖】

三、撰寫程式讀取點陣圖檔與顏色分類的研究

BMP檔通常包含四個資訊：**檔案標頭**、**檔案資訊標頭**、**調色板**與**點陣圖資料**，其中著重在第四個資訊「**點陣圖資料**」，因為一張點陣圖實際的像素資料會存放在此。接著我們嘗試利用C語言來撰寫程式來讀取南投縣AED位置的分佈網格化之點陣圖與研究顏色的分類，我們分以下步驟來介紹:

(一)撰寫C語言來讀取南投縣AED位置的分佈之點陣圖

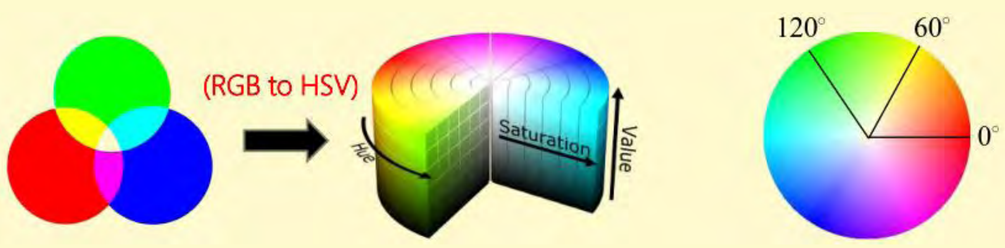
主要是利用**三層巢狀迴圈**來存取點陣圖的像素資料(圖四)，共存取144個位置中**RGB的數值**。

```
for(i=0; i<12; i++)
{
    for(j=0; j<12; j++)
    {
        for(k=0; k<3; k++)
        {
            block[i][j][k]= OutputImage[(320*13*(25-2*(i+1)) +13*(2*(j+1)-1))*3+k];
        }
    }
}
```

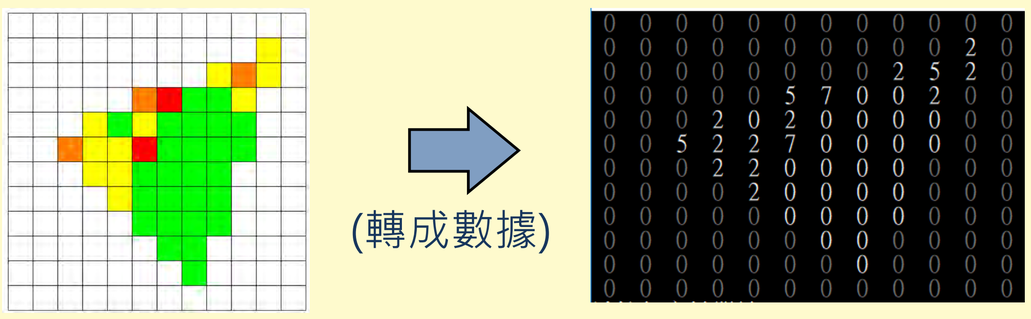
【圖四:宣告陣列陣列block[12][12][3]與利用迴圈存取RGB 】

(二)方格子顏色分類之研究:RGB模型轉成 HSV模型

為了將讀取出來的每一個方格中的RGB值作顏色標準的分類，我們將RGB轉換到HSV顏色模型，HSV模型空間由三個部分組成:**色調(Hue)**、**飽和度(Saturation)**、**值(Value)**，在HSV模型中，我們主要是**利用「色調」的特性**，因為它將顏色對應到角度，角度範圍為0度到360度如下:



我們撰寫程式，宣告兩個資料結構RGB與HSV並對應的變數data與value，來存取每個方格的RGB資料與HSV資料，再利用設計好的函數RGBToHSV(data)，進行兩個顏色模型的轉換。轉換後再依據**範圍為0至120度，可以得到四種顏色:紅色、橘色、黃色與綠色，對應的數值分別設定**為7、5、2與0，讀取影像資料並將之轉化成一組數據後，接下來開始對這組數據做分析與研究。利用函數print( )呈現在電腦螢幕上，轉化的過程如下圖:



四、應用「積分影像」計算「影像的區域變異量」

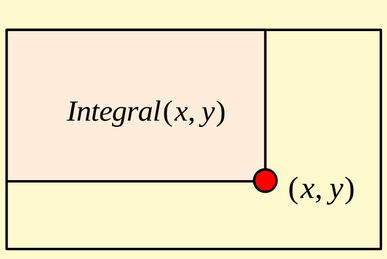
積分影像是一個能快速且有效率地計算網格或矩形區域內影像資料累加的方式，可將原始影像轉換成積分影像，藉由透過積分影像的數值或特徵，分析影像特色或取得有用的資料，我們以簡單的步驟來介紹積分影像的計算原理與計算區域變異量如下:

(一)積分影像的基本原理

考慮原始灰階影像的每個**像素點值image(i,j)**，其中i與j值範圍由影像大小決定，積分影像定義數學式子表示如下:

$$Integral(x,y)=\sum_{i=0}^{x}\sum_{j=0}^{y}image(i,j)$$

以粉紅色區域表示從影像左上角進行像素值累加至位置(x ,y)之總和，以圖示意如下:



(二)撰寫程式將原影像資料快速轉化成積分影像資料

接著將積分影像的運算原理寫成演算法，主要是利用雙層迴圈，計算出**累加資料**，程式碼設計如下圖:

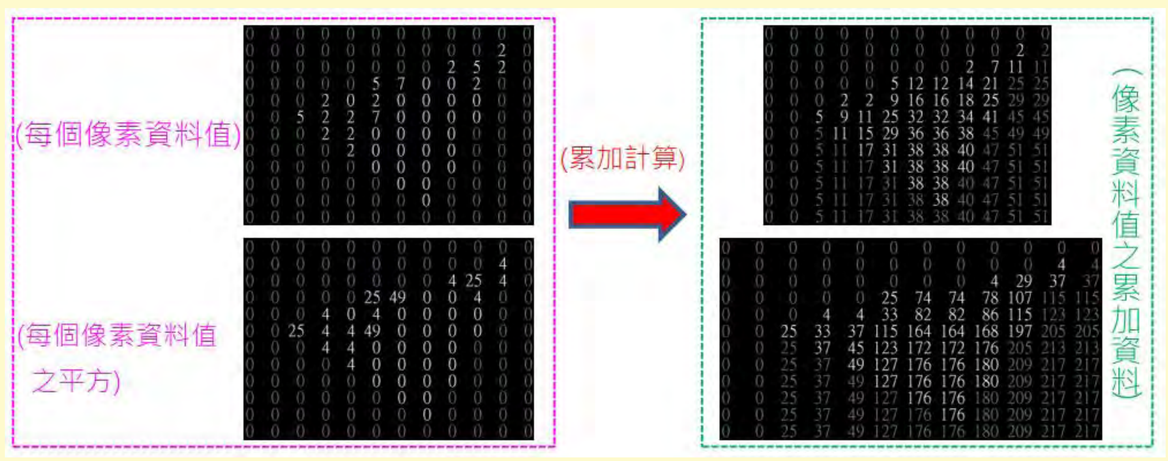
```
for(i=0;i<5;i++)
{
    for (j=0;j<5;j++)
    {
        if(i==0 && j>0){
            integral[0][j] = integral[0][j-1] + image[0][j];
        }else if(j==0 && i>0){
            integral[i][0] = integral[i-1][0] + image[i][0];
        }else if(i>0 && j>0){
            integral[i][j] = integral[i-1][j] + integral[i][j-1] - integral[i-1][j-1] +image[i][j];
        }
    }
}
```

(三)應用積分影像的原理計算「影像的區域變異量」

我們利用統計學上常用於分析的**統計量「平均數 μ」與「變異數 σ^2」來定義「影像的區域變異量」**，若給定一組n筆影像的每個像素點值 x_i，因此 μ值為所有像素點值和之平均值，σ^2值可以解釋為所有像素點值與平均值差的平方和之平均，其中 σ^2值可化簡如下:

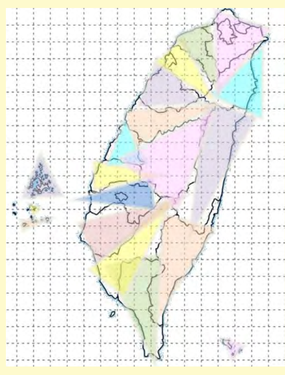
$$\sigma^2=\frac{\sum_{i=1}^{n}x_i^{\,2}}{n}-\mu^2 = \text{像素值平方和的平均} - \text{像素值平均的平方}$$

將**變異數σ^2定義為「影像的區域變異量」**，像素值平方和也可以由「積分影像」的運算原理快速地求出，並撰寫程式計算出南投縣AED位置的分佈的**兩種累積資料「像素資料和」與「像素資料平方和」**，如下圖:



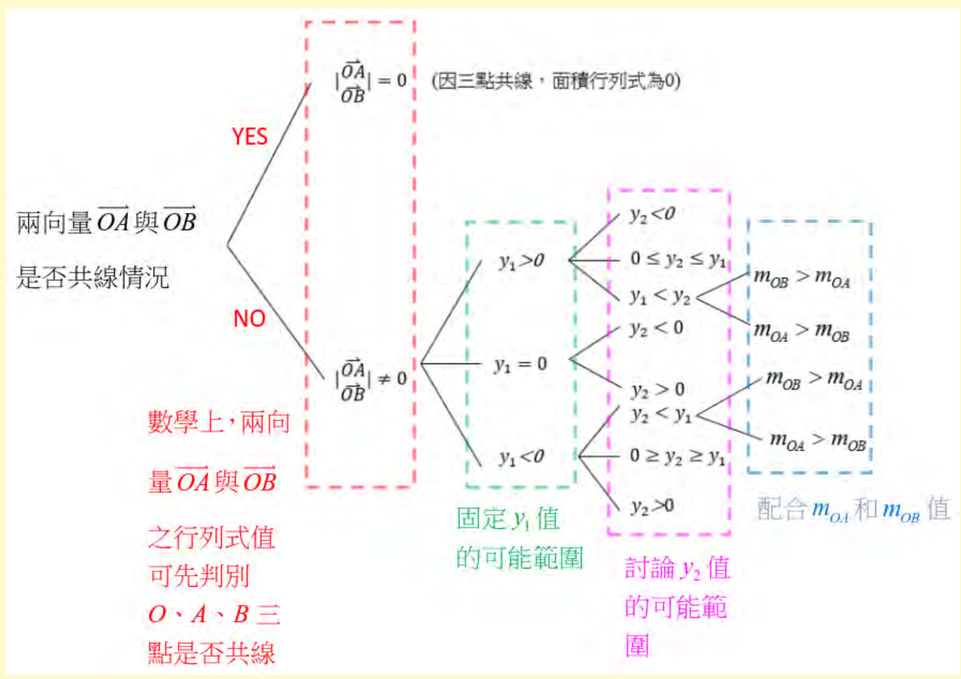
五、從「三角形」發想，探討台灣行政區形狀

我們觀察台灣每個行政區的形狀，因為每個行政區的形狀皆不一致，所以就算我們有網格過後的AED位置分佈圖，也很難在單一的行政區裡計算具有AED裝置的方格，因此我們研究最適合的圖形來覆蓋每個行政區，如右圖，**最後選用三角形**做為我們幾何圖的發想。

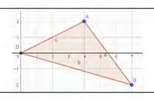
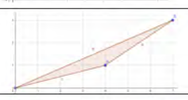
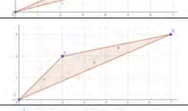
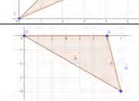
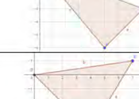


六、研究在二維平面上，推導出透過三點座標連成的三角形內覆蓋的方格數之數學通式

由原點O出發，考慮 $A(x_1,y_1)$、$B(x_2,y_2)$，其中 $0<x_1<x_2$，將這三點圍出的三角形區域定義為 ΔOAB，為了設計三角形內數方格之演算法，**我們必須要考慮平面上所有可能發生的三角形之三頂點情況**，總共會有對應的十個數學通式，我們將所有的結果之流程討論整理如下:



將討論所有三角形與對應的計算三角形內方格數之通式結果以表格呈現如下(詳細的通式推導於附錄一)並將**數學理論程式化，撰寫三角形內覆蓋方格數之演算法**(詳細的程式碼於附錄二)

三角形在二維平面上之情況	計三角形內數方格子之數學通式
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_1}{x_1}k\right\rfloor-\left\lceil\frac{y_2}{x_2}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1)\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_1}{x_1}k\right\rfloor-\left\lceil\frac{y_2}{x_2}(k+1)\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}k\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}k\right\rfloor-\left\lceil\frac{y_1}{x_1}(k+1)\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_1}{x_1}k\right\rfloor-\left\lceil\frac{y_2}{x_2}(k+1)\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}(k+1)\right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}k\right\rceil$
	$\sum_{k=1}^{x_1-1}0-\left\lceil\frac{y_2}{x_2}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1)\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}k\right\rfloor-0+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil\frac{y_1}{x_1}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}k\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}(k+1)\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_1}{x_1}(k+1)\right\rfloor-\left\lceil\frac{y_2}{x_2}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rfloor-\left\lceil y_2-\frac{y_2}{x_2}(k+1)\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil\frac{y_1}{x_1}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}k\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rceil$
	$\sum_{k=1}^{x_1-1}\left\lfloor\frac{y_2}{x_2}k\right\rfloor-\left\lceil\frac{y_1}{x_1}k\right\rceil+\sum_{k=1}^{x_2-x_1-1}\left\lfloor y_2-\frac{y_2}{x_2}(k+1)\right\rfloor-\left\lceil y_2-\frac{y_2-y_1}{x_2-x_1}k\right\rceil$

七、找尋「最佳」覆蓋方格子之三角形，應用於分析南投縣AED位置網格分佈圖

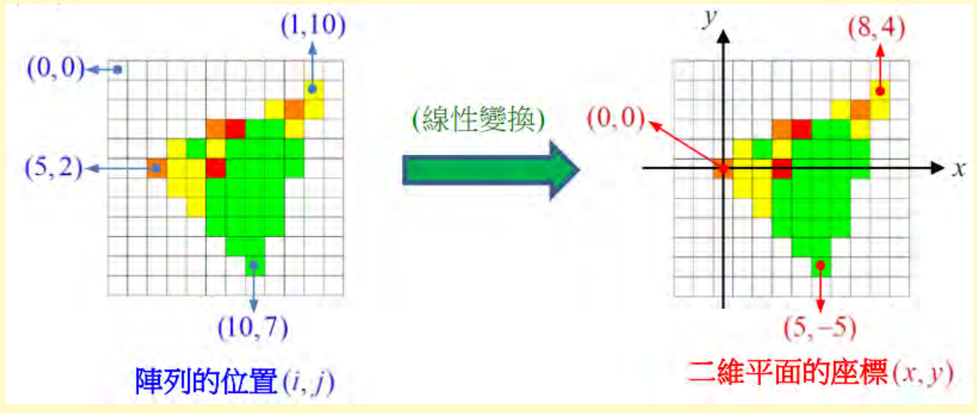
為了找到一個準確地覆蓋AED位置分佈之三角形，即非必要計算的白色方格數會是最少的，我們將研究的過程分為以下步驟進行:

(一)陣列的位置(i, j)轉化成二維平面的座標(x, y)

我們利用數學的線性變換之旋轉矩陣，以矩陣運算的方式呈現陣列位置與二維平面座標的關係如下:

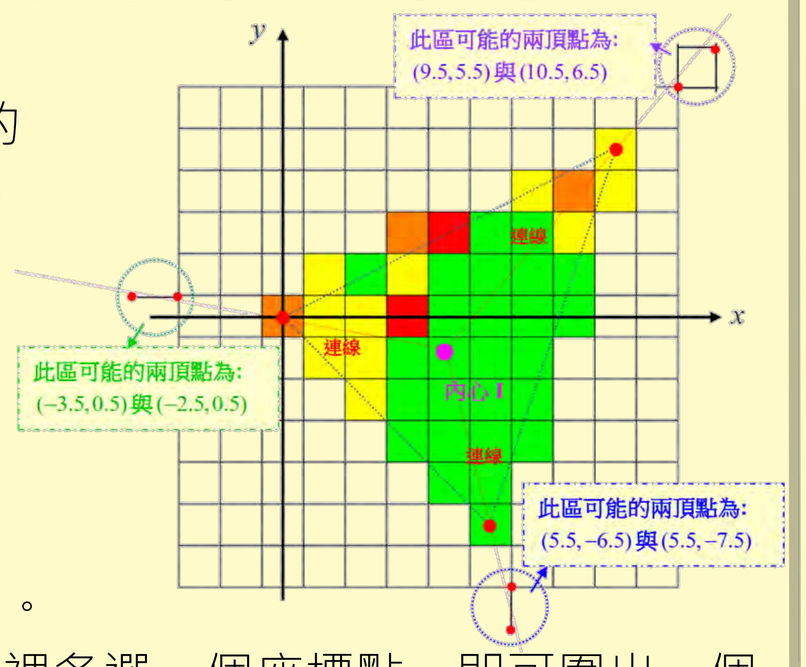
$$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} i \\ j \end{pmatrix}+\begin{pmatrix} -2 \\ 5 \end{pmatrix}=\begin{pmatrix} x \\ y \end{pmatrix}$$

將三個陣列的位置代入上式可得二維平面上的三個對應的座標如下圖所示:

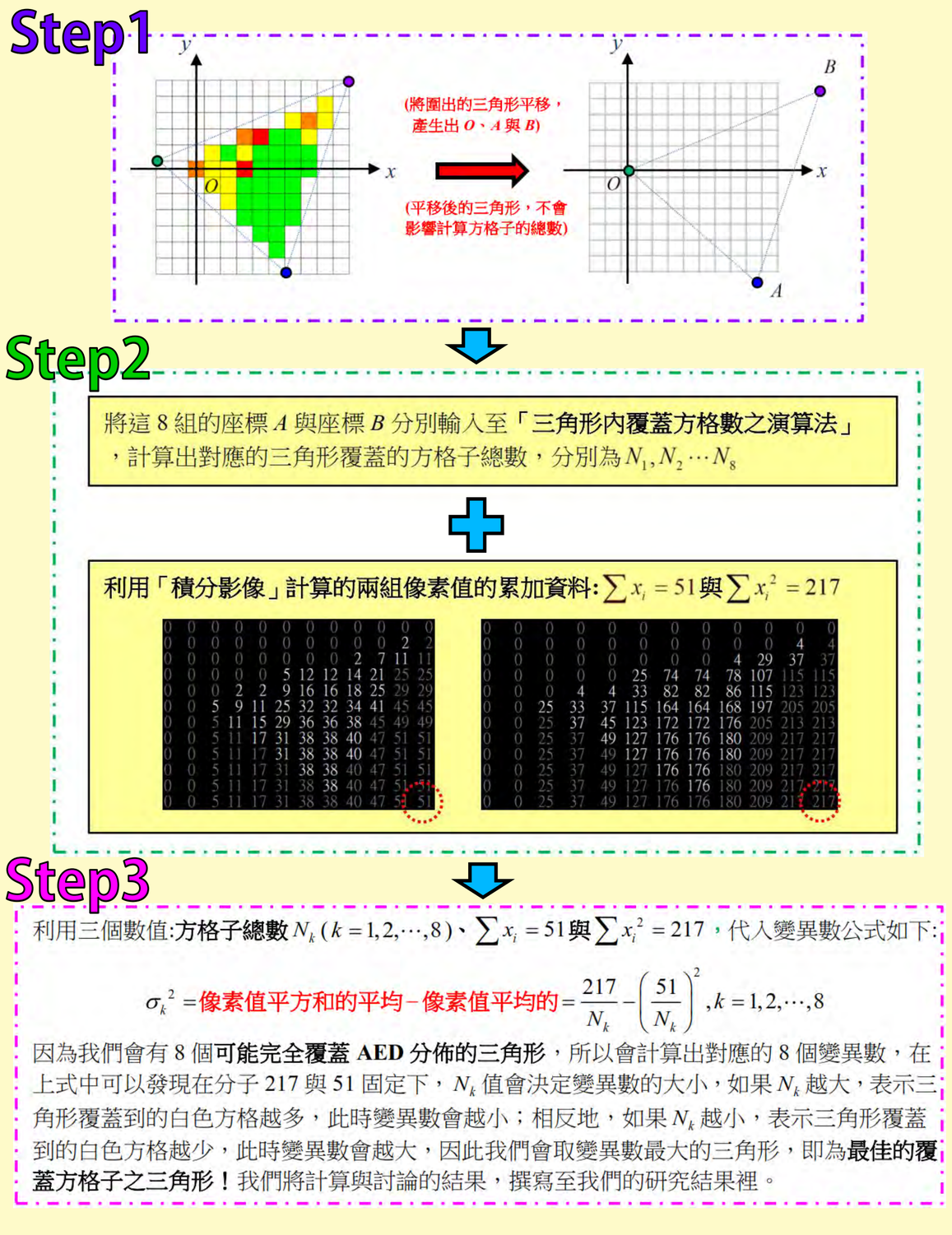


(二) 決定「最佳」覆蓋方格子之三角形的三頂點

以三個頂點座標(0,0)(5,-5)與(8,4)所圍出的三角形出發，利用此三角形的內心與三頂點的連線，觀察連線與它的斜率對應到的直線所交的位置，最靠近的兩個頂點就會作為設計可能覆蓋三角形三頂點，如右圖。



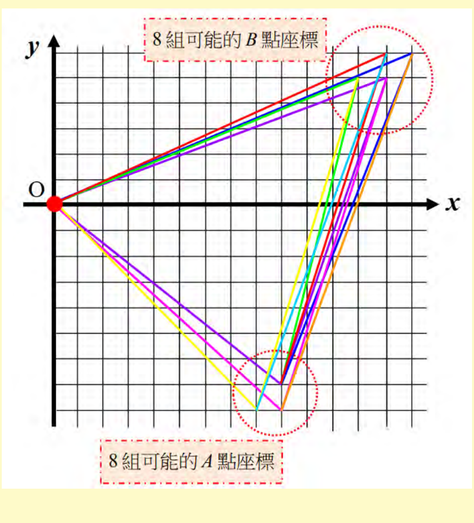
分別從過這三個區域裡各選一個座標點，即可圍出一個可能完全覆蓋AED分佈的三角形，因此共有8個可能的三角形，接下來再從8個三角形中選出一個最佳的三角形，**結合「積分影像計算影像的區域變異量」與「三角形內覆蓋方格數之演算法」**來進行處理，過程如下圖:



伍、研究結果

一、找出最佳的覆蓋方格子之三角形

我們將這八組可能的三角形的三頂點平移至原點產生出O、A、B 如右圖，再把這8組的座標A與座標B分別輸入至「三角形內覆蓋方格總數之演算法」，我們將計算的結果整理如下表。



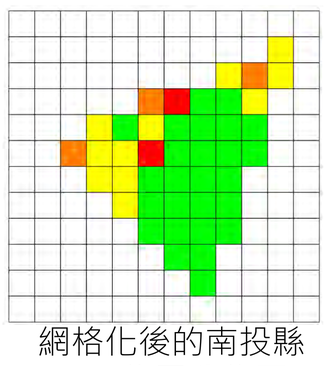
$O(0,0)\ A(9,-7)\ B(13,5)$	$O(0,0)\ A(9,-7)\ B(14,6)$	$O(0,0)\ A(9,-8)\ B(13,5)$	$O(0,0)\ A(9,-8)\ B(14,6)$
格子數N_1 : 47 變異數σ_1 = 3.3496	格子數N_2 : 52 變異數σ_2 = 3.2112	格子數N_3 : 51 變異數σ_3 = 3.2549	格子數N_4 : 58 變異數σ_4 = 2.9682
$O(0,0)\ A(8,-7)\ B(12,5)$	$O(0,0)\ A(8,-7)\ B(13,6)$	$O(0,0)\ A(8,-8)\ B(12,5)$	$O(0,0)\ A(8,-8)\ B(13,6)$
格子數N_5 : 42 變異數σ_5 = 3.6922	格子數N_6 : 46 變異數σ_6 = 3.4882	格子數N_7 : 49 變異數σ_7 = 3.3453	格子數N_8 : 55 變異數σ_8 = 3.0856

從上表可發現當O(0,0)、A(8,-7)與B(12,5)時，可知格子數N_5值最小，表示覆蓋到非必要計算的白色區域方格數會是最少的，此時對應的變異數σ_5值最大，此三角形為最佳覆蓋方格子之三角形。

二、將南投AED網格分佈點陣圖匯入演算法

測試的點陣圖BMP資料

尺寸: 320×320
檔名: nantou_aed_distribution.bmp
顏色意義:
紅色方格表示AED裝置分佈非常密集。
橘色方格表示次多。
黃色方格表示少許。
綠色方格表示無AED裝置的位置分佈之地。


網格化後的南投縣AED位置分佈圖

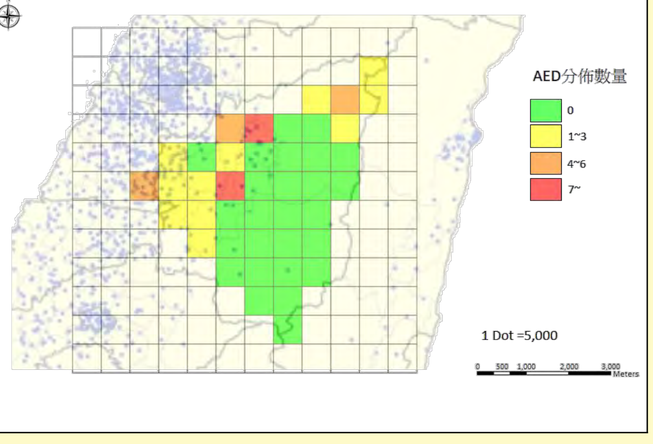
匯入三角形內方格子數之演算法跑出的結果

使用者可分別輸入最佳三角形的座標A(8,-7)與B(12,5)，經演算法計算後可得以下結果:
由各顏色的分佈比例可知，綠色方格子所佔的比例最高，表示這一區中，無AED裝置的位置分佈偏高，應該要多增設AED裝置，讓民眾馬上就能反應並尋找到AED裝置，以備不時之需，期望掌握4至6分鐘「黃金救命」，提高患者救活率，以達到醫療資源分配的均勻。

```
請輸入A點座標(x1,y1)與B點座標(x2,y2):
8 -7 12 5

~~~~~~利用演算法計算~~~~~~
三角形OAB內方格子總數為:42


~~~~~~方格的分佈概況~~~~~~
紅色方格子數:2
橘色方格子數:3
黃色方格子數:11
綠色方格子數:26

~~~~~~各顏色分布的比例~~~~~~
紅色方格子數佔的比例:0.047619
橘色方格子數佔的比例:0.071429
黃色方格子數佔的比例:0.261905
綠色方格子數佔的比例:0.619048
請按任意鍵繼續 . . .
```

南投縣AED位置分佈圖與人口分佈點狀圖之疊圖

在南投縣人口點狀圖中，每個單位點表示5000人，在進行疊圖過後，我們可以看到有許多的綠色格子是代表沒有裝置的，代表某些區域有人口但是沒有裝置，其他的地方有些是有裝置但是沒有人口，所以需要重新規劃AED裝置的放置情況。



陸、討論

一、設計用方格子反推完全包含它們的三角形演算法

我們這次的研究主要是將AED分佈圖放置二維平面上，給定三個頂點的座標，代入演算通式後，就可分析AED分佈的狀況；但若以相反的概念，在平面上如果給定有顏色的方格子，則可以探究要如何用最有效率的方式框住所有顏色的方格子之三角形，是一個值得我們日後討論與研究的問題。

二、將計算方格數量之演算法延伸與推廣至n邊形

在生活中的幾何圖形有很多種，並非皆是三角形，未來可研究撰寫出泛用度更高的n邊形內部所包含的方格子數量之演算法；另外方格子也可用其他種類的多邊形，日後我們在探討幾何圖形覆蓋時，可以設計出更有效率的計數演算法。

柒、結論

我們結合地理QGIS系統、三角形內方格數量的演算法、統計學的變異數與積分影像原理，設計出一套最佳覆蓋方格數量的方法，進而準確地統計AED分佈的狀況，克服了電腦數方格時需要龐大的計算，除此之外本研究也提供出如何將**理論層面轉化成實際應用層面**之研究方法，期待將此轉化的模式作成未來更實用且有效率的演算的技術，應用至其他領域，解決其它更多種類的問題。

捌、參考資料

一、許志農、黃森山教授。龍騰版高中數學課本第三冊第二章「直線與圓」。
二、陳國川教授。龍騰版高中地理課本第一冊第五章「地理資訊系統」。
三、劉洲溶、蘇淵源、翟家甫。全華版高中資訊科技概論課本第五章「電腦與問題解決」。
四、政府資料開放平台。2017年12月。取自http://data.gov.tw