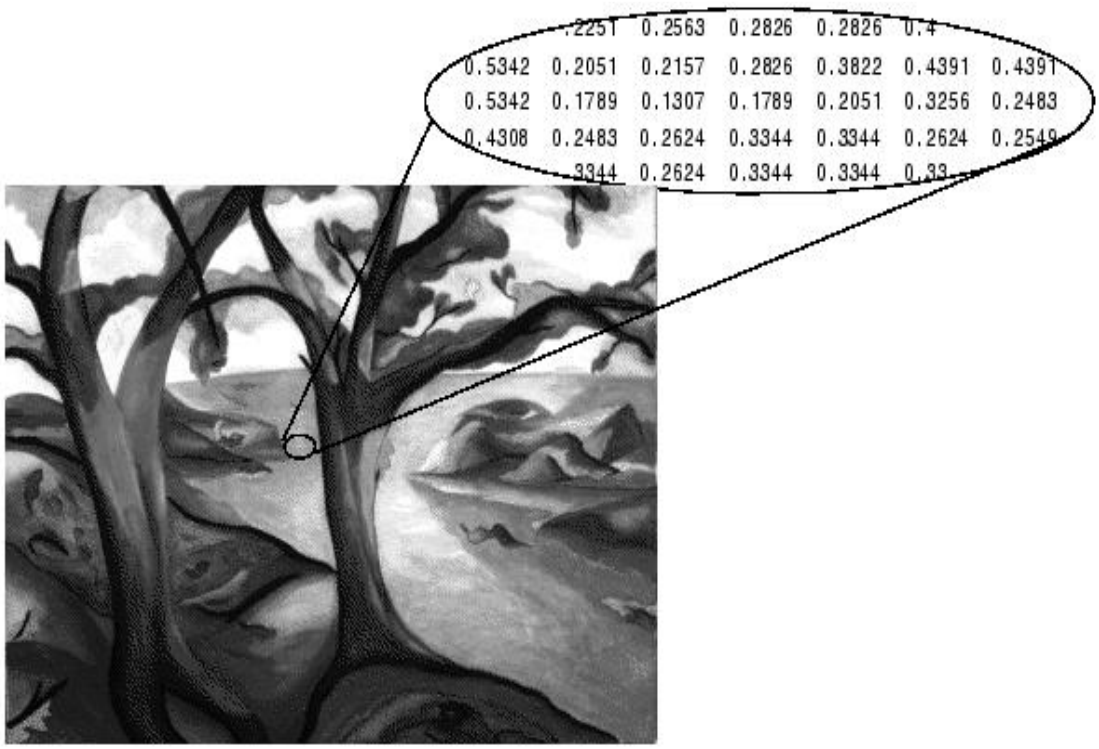
أنواع الصور الرقمية في بيئة الماتلاب

هناك خمس أنواع للصورة الرقمية في بيئة الماتلاب :

١- Grayscale Image : هي صورة تمثل بمصفوفة ببعدين حجمها MxN وعناصرها من نوع double وتقع ضمن المجال [0,1] حيث يمثل 0 اللون الأسود و 1 اللون الأبيض أما القيم الواقعة بينهما تمثل تدرجات اللون الرمادي .

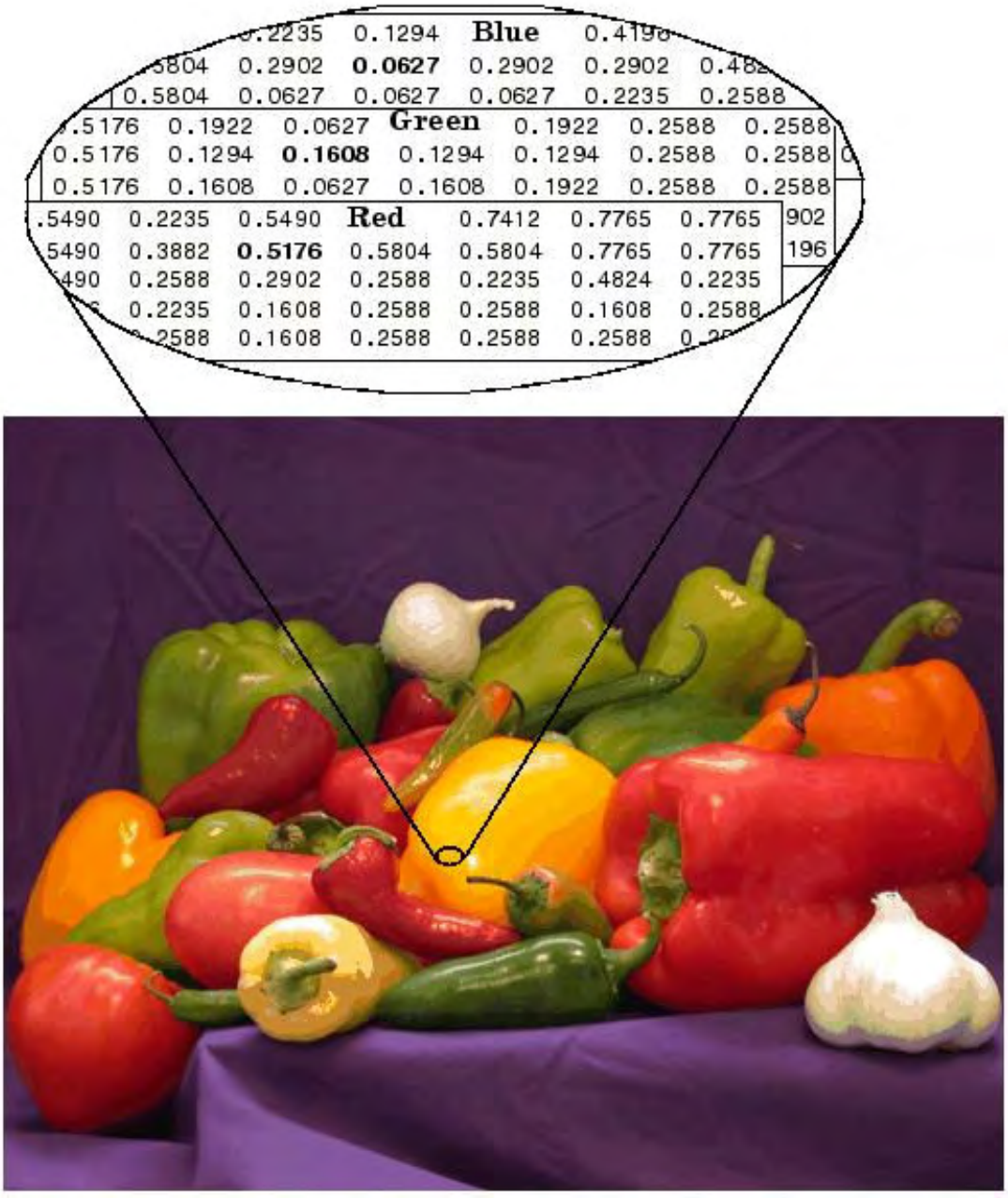
الشكل التالي يوضح هذا النوع من الصور :



نلاحظ أن كل عنصر من المصفوفة يشير إلى لون من الصورة كما نلاحظ أن ألوان الصورة تتدرج من الأبيض إلى الرمادي إلى الأسود .

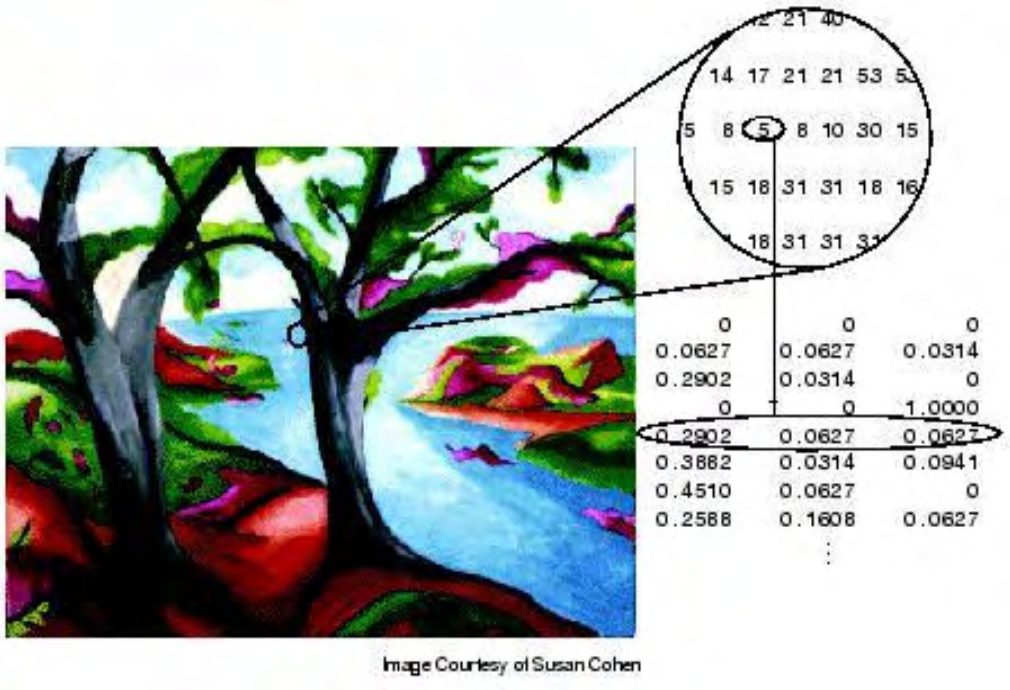
٢- Truecolor RGB Image : هي صورة تمثل بمصفوفة بثلاث أبعاد حجمها MxNx3 وعناصرها من نوع double وتقع ضمن المجال [0,1] وكل بكسل من الصورة ينتج عن دمج المركبة الحمراء والخضراء والزرقاء لإعطاء اللون المناسب حيث أنّ لكل مركبة من المركبات الثلاث مصفوفة ببعدين MxN فالمركبة الحمراء فيها يمثل 0 اللون الأسود و 1 اللون الأحمر وهكذا بالنسبة لبقية المركبات الخضراء والزرقاء وتركيب هذه المركبات الثلاث يعطي الصورة ذات الألوان الحقيقية .

الشكل التالي يوضح هذا النوع من الصور :



٣- Indexed Image : هي صورة تمثل بمصفوفتين مصفوفة الدليل index ببعدين حجمها MxN ومصفوفة خارطة اللون colormap ببعدين Kx3 . حيث تحوي مصفوفة خارطة اللون colormap على جميع الألوان المحتمل وجودها في الصورة وعددها K لون بعدد أسطر مصفوفة خارطة اللون أما الأعمدة الثلاث للمصفوفة فتحوي على مركبات الألوان الحمراء والخضراء والزرقاء . أما مصفوفة الدليل index تحوي بكسلات الصورة التي تشير إلى الألوان في مصفوفة خارطة اللون colormap حيث أن كل بكسل يحمل رقم صحيح يشير إلى سطر من مصفوفة خارطة اللون colormap وهذا السطر يحوي مركبات لون من الألوان.

الشكل التالي يوضح هذا النوع من الصور :



كما في الصورة نلاحظ أن الرقم 5 في مصفوفة الدليل index Matrix يشير إلى السطر الخامس من مصفوفة خارطة اللون Colormap Matrix والذي يحوي على نسب المركبات الحمراء والخضراء والزرقاء والتي تحدد اللون القريب من الأزرق .

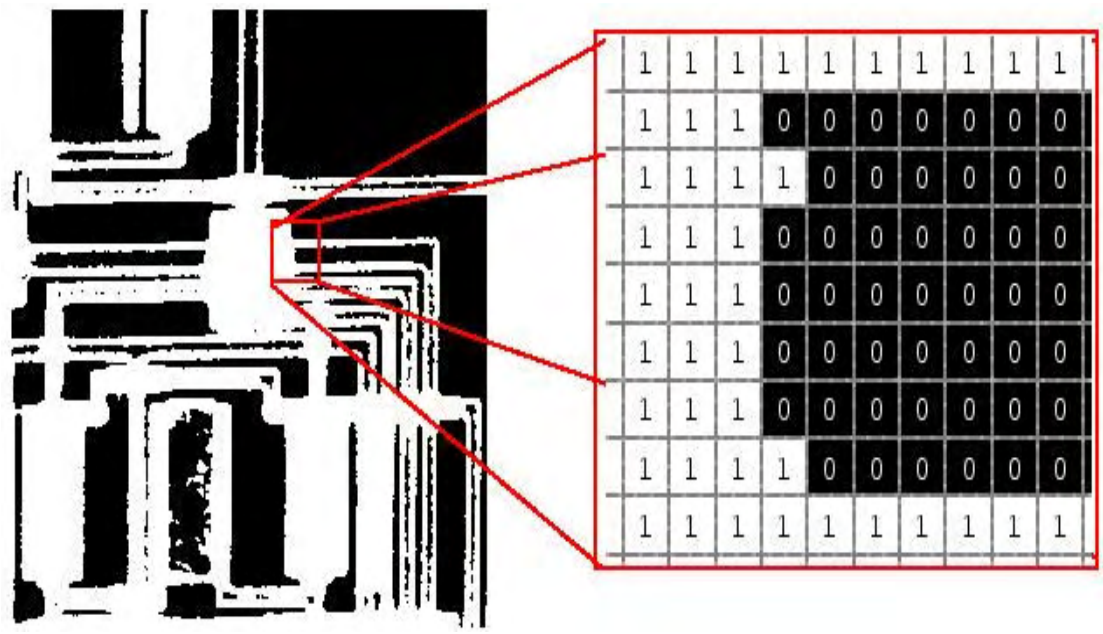
Mycolormap(5,:)=[0.2902 0.0627 0.0627];

&&

Myimage(m,n)=5;

وهذا الدليل 5 يشير إلى اللون القريب من الأزرق الموجود في مصفوفة خارطة اللون .

٤- Binary Image : هي صورة تمثل بمصفوفة ببعدين حجمها MxN وعناصرها من نوع logical أي كل بكسل فيها إما 0 (لون أسود) أو 1 (لون أبيض) .



نلاحظ أن كل عنصر من المصفوفة يشير إلى لون من الصورة كما نلاحظ أن ألوان الصورة هي اللونين الأبيض والأسود فقط .

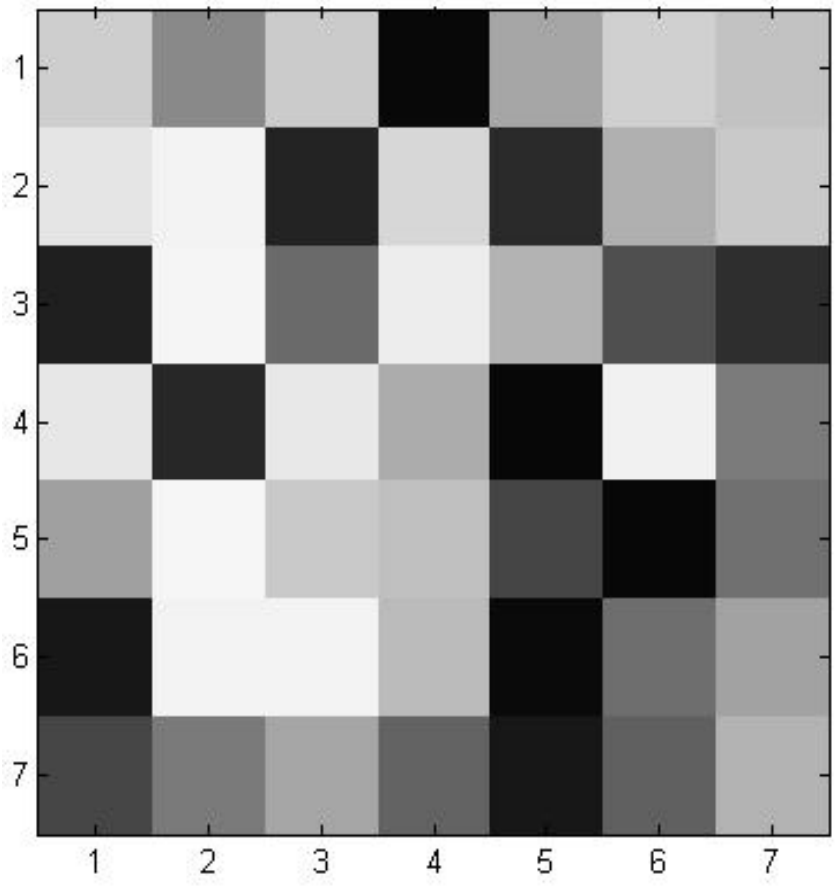
٥- uint8 Image : يستخدم هذا النوع للتقليل من مساحة الذاكرة وللتسريع من عملية معالجة الصورة بدلاً من double Image .

إنشاء صور من أنواع مختلفة :

١- لإنشاء صورة من نوع Grayscale

```
mygray=rand(7,7);
image(mygray*255);
axis image
colormap(gray(256));
```

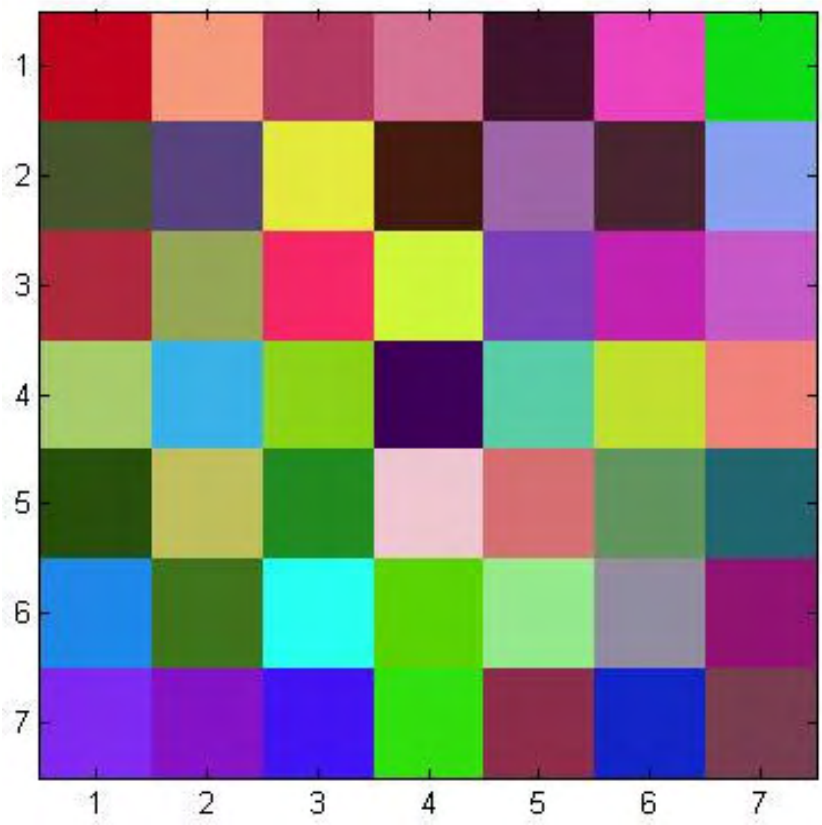
والنتيجة هي :



٢- لإنشاء صورة من نوع Truecolor RGB

```
myrgb(:,:,1)=rand(7,7);

myrgb(:,:,2)=rand(7,7);

myrgb(:,:,3)=rand(7,7);

image(myrgb);

axis image
```

والنتيجة هي :



وإذا أردنا حذف اللونين الأبيض والأسود من الصورة الملونة :

```
image(min(max(myrgb,0),1));

axis image
```

التحويل بين أنواع الصور الرقمية

١- التحويل من grayscale إلى RGB :

```
myrgb = cat(3,mygray,mygray,mygray);
```

مثال :

```
mygray=imread('pout.tif');
imshow(mygray)
myrgb = cat(3,mygray,mygray,mygray);
figure,imshow(myrgb)
```

والنتيجة هي :

من المؤكد أن الصورة RGB لن تكتسب الألوان بمجرد التحويل لكن يمكن إكساب الصورة بعض الألوان كما سنرى في الدروس القادمة .

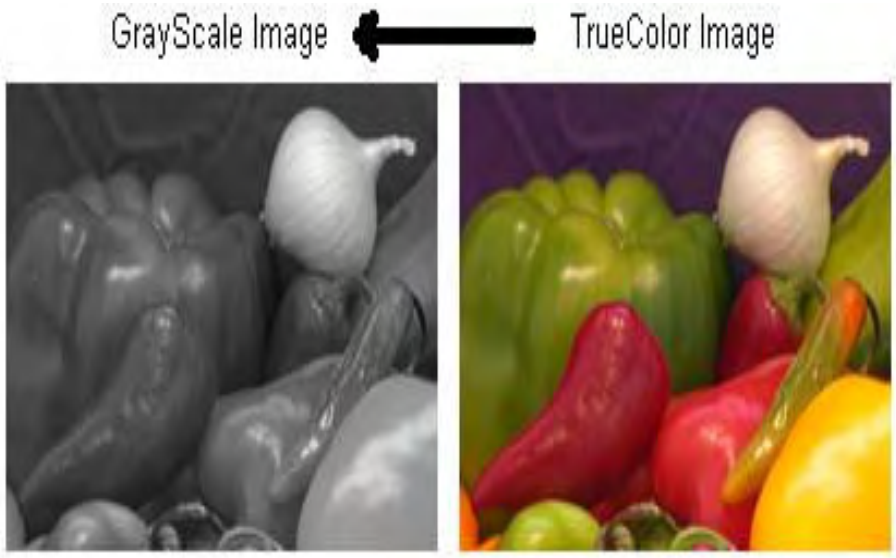
٢- التحويل من RGB إلى grayscale باستخدام القيمة المتوسطة :

```
mygray = mean(myrgb,3);
```

مثال :

```
myrgb=imread('onion.png');
imshow(myrgb)
mygray = round(mean(myrgb,3))/255;
figure,imshow(mygray)
```

والنتيجة هي :



٣- التحويل من RGB إلى grayscale باستخدام الوزن NTSC :

```
mygray = rgb2gray(myrgb);
```

ملاحظة :

- التحويل من RGB إلى Grayscale باستخدام الوزن NTSC يتم على الشكل التالي :

```
mygray =
0.2989* myrgb (:,:,1) + 0.5870* myrgb (:,:,2) + 0.1140* myrgb (:,:,3);
```

مثال :

```
myrgb=imread('onion.png');
imshow(myrgb)
mygray = rgb2gray(myrgb);
figure,imshow(mygray)
```

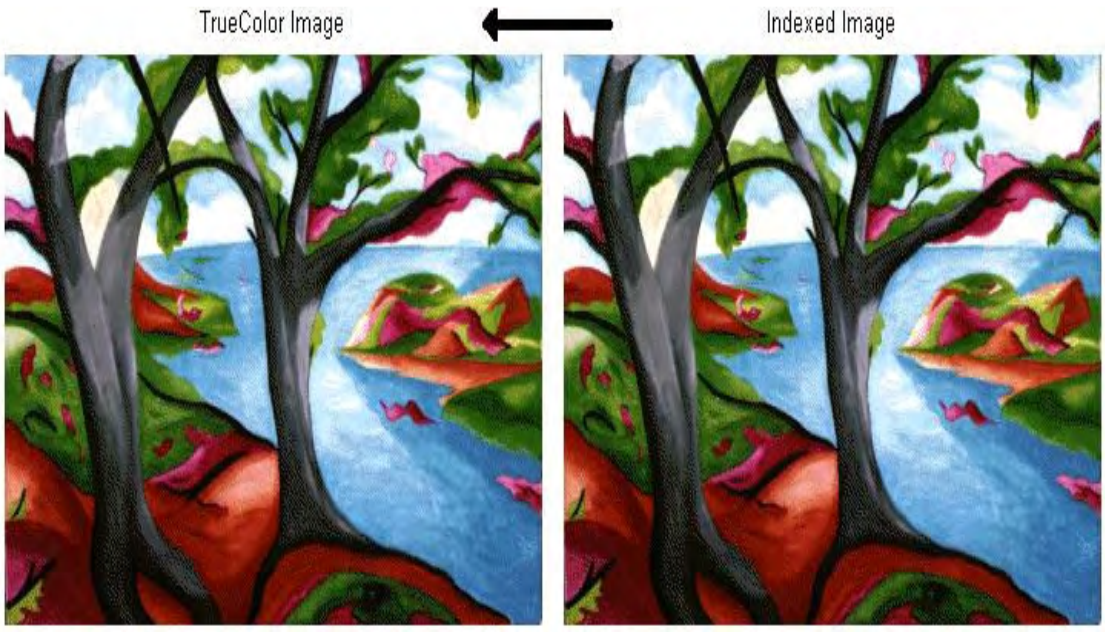
٤- التحويل من Indexed إلى RGB :

```
myrgb = ind2rgb(myindexed,mycolormap);
```

مثال :

```
[myindexed, mycolormap] =imread('trees.tif');
imshow(myindexed, mycolormap)
myrgb = ind2rgb(myindexed,mycolormap);
figure,imshow(myrgb)
```

والنتيجة هي :



نلاحظ أنه لا فرق بين الصورتين لأن كلا الصورتين تملكان نفس نظام الألوان RGB ولكن تختلفان عن بعضهما بطريقة التمثيل في الماتلاب .

٥- التحويل من RGB إلى Indexed باستخدام K لون :

```
[myindexed,mycolormap] = rgb2ind(myrgb,K);
```

مثال :

```
myrgb=imread('peppers.png');
imshow(myrgb)
[myindexed,mycolormap] = rgb2ind(myrgb,256);
figure,imshow(myindexed,mycolormap)
```

والنتيجة هي :

طبعاً في هذا المثال Indexed Image تملك 256 لون تتدرج من الأبيض للأسود في خارطة اللون Colormap الخاصة بالصورة .

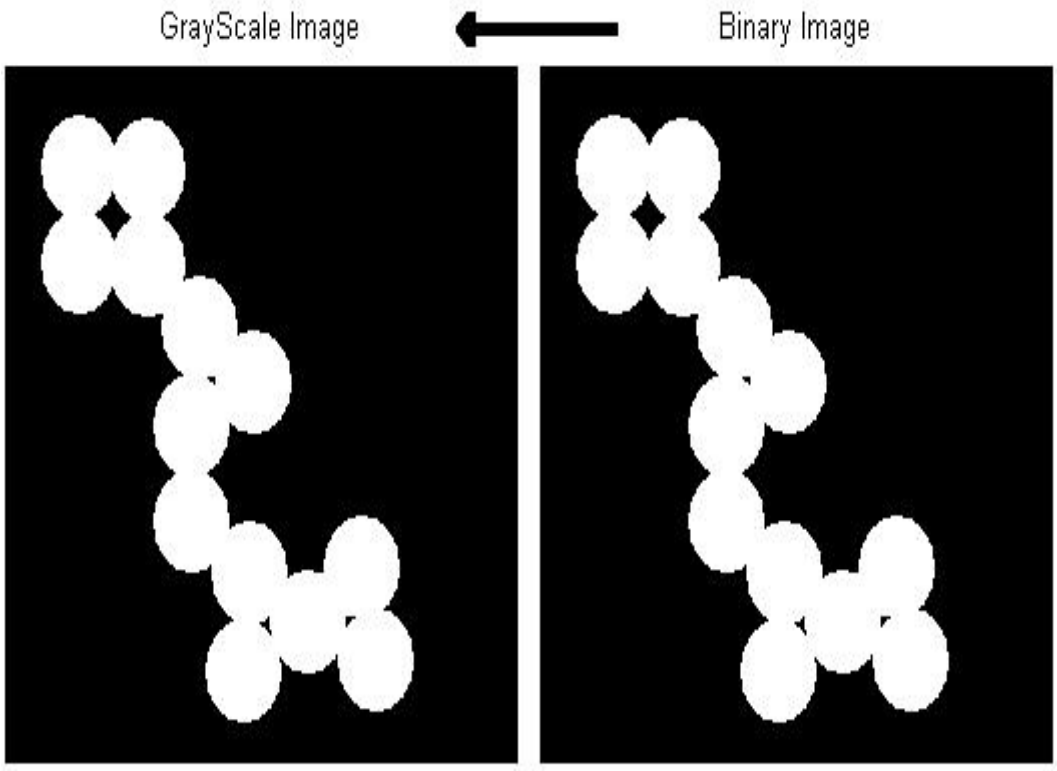
٦- التحويل من Binary إلى Grayscale :

```
mygray = double(mybinary);
```

مثال :

```
mybinary=imread('circles.png');
imshow(mybinary)
mygray = double(mybinary);
figure, imshow(mygray)
```

والنتيجة هي :



نلاحظ أن لا فرق بين الصورتين والصورة الناتجة من نوع GrayScale ويمكن بإجراء بعض التعديلات أن تقبل ألوان رمادية .

٧- التحويل من Grayscale إلى Binary :

```
mybinary = (mygray > a);
```

حيث جميع الألوان الرمادية التي تقع فوق a تتحول إلى لون أبيض والتي تقع تحت a تتحول لون أسود .

مثال :

```
mygray=imread('pout.tif');

imshow(mygray)

level=graythresh(mygray)*256;

mybinary = (mygray > level);
figure , imshow(mybinary)
```

والنتيجة هي :

فتح أو قراءة صورة وعرضها

يتم فتح وقراءة صورة من أي نوع باستخدام التعليمة imread ويتم عرض الصورة باستخدام التعليمة imshow حيث يختلف شكل هاتين التعليمتين بحسب طبيعة الصورة وفق إحدى الحالات التالية :

١ - فتح صورة من الحاسب وعرضها :

لفتح أو قراءة صورة من جهاز الحاسب تستخدم التعليمة imread على الشكل التالي :

```
X = imread(filename,format);

imshow(X)
```

حيث تتم قراءة الصورة عبر المسار filename ذات الامتداد format ومن ثم تخزينها في مصفوفة X .

مثال :

لدينا الصورة التالية موضوعة على القرص D باسم sky ومن نوع jpg :

عندئذ يمكن قراءة الصورة بالتعليمة التالية ثم عرضها :

```
X = imread('D:\sky','jpeg');
imshow(X)
```

٢- فتح صورة من برنامج الماتلاب نفسه وعرضها :

نميز إحدى حالتين :

أ- أن تكون الصورة من نوع Image indexed :

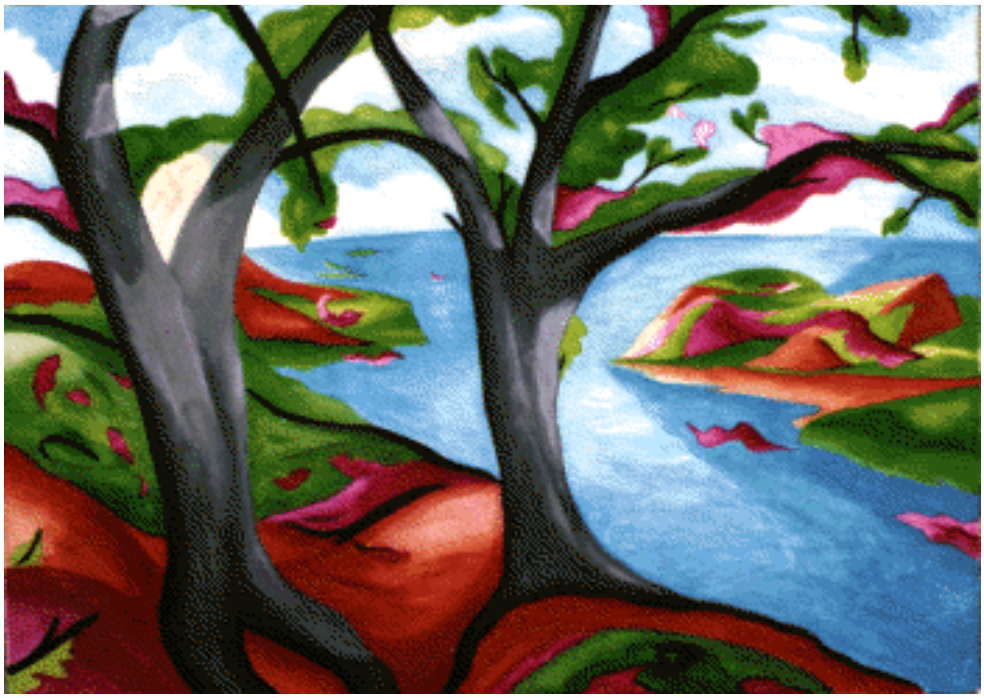
عندئذ يمكن قراءة الصورة وعرضها على الشكل التالي :

```
[X,map]=imread(filename,format);
imshow(X,map)
```

وتكون X مصفوفة الدليل MxN و map مصفوفة خارطة اللون Kx3 .

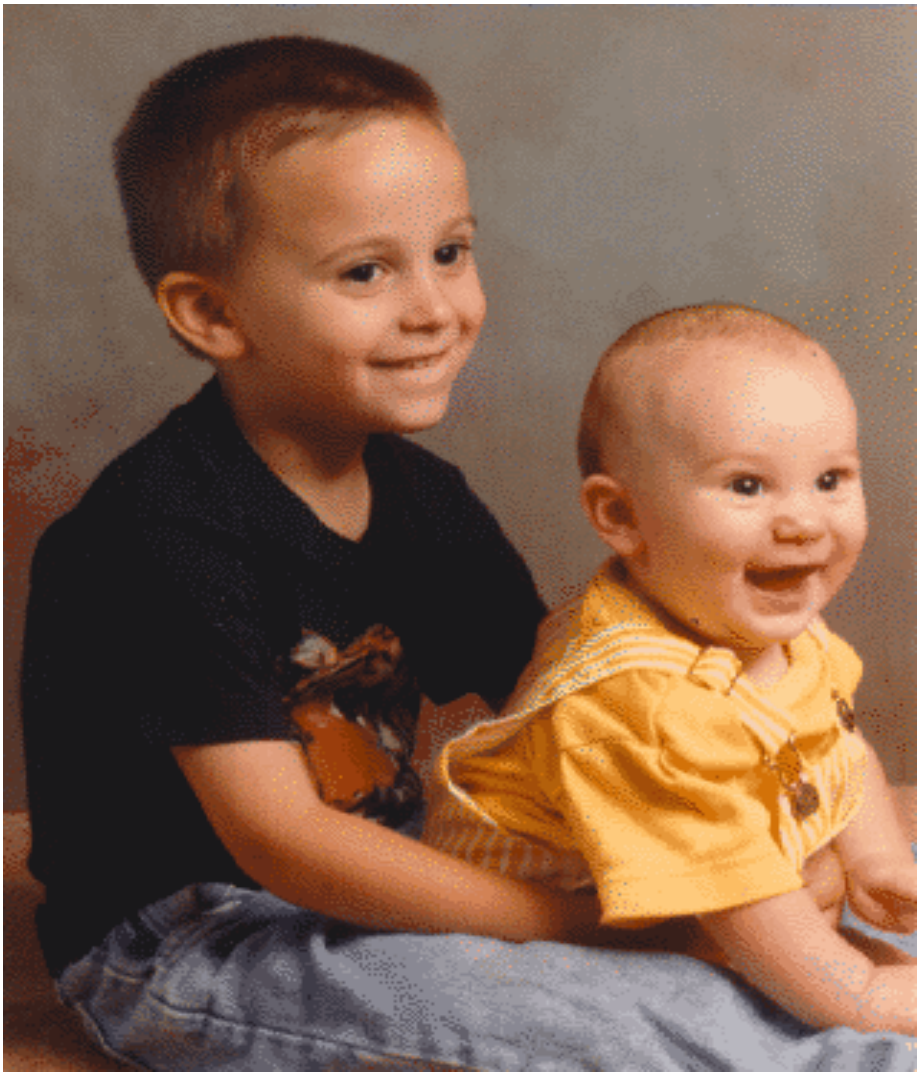
مثال ١ :

```
[X,map] =imread('trees.tif');
imshow(X,map)
```



مثال ٢ :

```
[X,map] =imread('kids.tif');
imshow(X,map)
```



نلاحظ أن الصورة Indexed Image تحتاج لقراءتها وعرضها بارا متر إضافي وهو مصفوفة خارطة اللون Colormap في كلا تعليمتي القراءة والعرض .

ب. أما إذا كانت الصورة من بقية الأنواع :

عندئذ يمكن قراءة الصورة على الشكل التالي :

```
X=imread(filename.format);
```

مثال 1 : صورة من نوع RGB .

```
X = imread('onion.png');
imshow(X)
```

حيث أن X هي مصفوفة أبعادها MxNx3 .

تذكرة :

لاحظ الصورة True Color RGB تحوي على ألوان عديدة وكل لون من هذه الألوان هو مزيج من ثلاث مركبات المركبة الحمراء والخضراء والزرقاء أما قيم عناصر المصفوفة X فهي إما 0 للدلالة على اللون الأسود أو 1 للدلالة على اللون الأحمر للمركبة الحمراء والأخضر للمركبة الخضراء والأزرق للمركبة الزرقاء .

مثال 2 : صورة من نوع Gray Scale .

```
X = imread('pout.tif');
imshow(X)
```

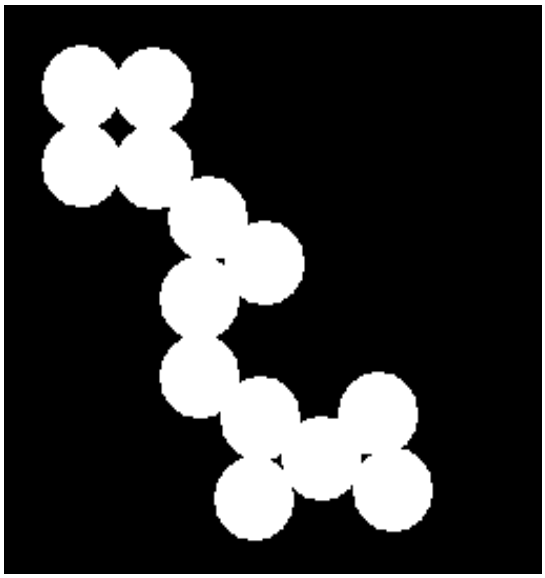
حيث أن X هي مصفوفة أبعادها MxN .

تذكرة :

لاحظ الصورة Gray Scale تحوي على ألوان تتدرج من الأسود إلى الرمادي بتدريجاته المختلفة إلى الأبيض أما قيم عناصر المصفوفة X فهي إما 0 للدلالة على اللون الأسود أو 1 للدلالة على اللون الأبيض أو بينهما للدلالة على تدريجات اللون الرمادي .

مثال 3 : صورة من نوع Binary

```
X = imread('circles.png');
imshow(X)
```



تذكرة :

لاحظ الصورة الثنائية تحوي على لونين فقط هما اللون الأبيض واللون الأسود أما قيم عناصر المصفوفة X فهي إما 0 للدلالة على اللون الأسود أو 1 للدلالة على اللون الأبيض .

حفظ و طباعة الصورة باسم و امتداد جديدين

بفرض أننا قمنا بمعالجة صورة معينة وأجرينا عليها التغييرات المناسبة ثم أردنا حفظ أو طباعة هذه الصورة على جهاز الحاسب باسم جديد وامتداد جديد نستخدم التعليمة imwrite على الشكل التالي :

imwrite(image,filename)

مثال :

```
X = imread('D:\sky','jpeg');
imshow(X)
imwrite(X,'newsky.bmp')
```

هنا قمنا بطباعة نفس الصورة الموجودة في المسار D إلى المسار التالي :

(المستندات / MATLAB) باسم جديد newsky وامتداد جديد bmp .

الحصول على معلومات عن الصورة

هنا نميز حالتين :

١- الصورة من الحاسب :

يمكن الحصول على معلومات كاملة عن الصورة باستخدام التعليمة Imfinfo :

info=iminfo(filename,format)

حيث يمكن الحصول على العديد من المعلومات وأهمها :

١- مسار ملف الصورة .

٢- حجم الملف .

٣- العرض .

٤- الارتفاع .

٥- الامتداد .

٦- نظام الألوان .

مثال :

info=imfinfo('D:\sky','jpeg')

والنتيجة الظاهرة :

info =

Filename: 'D:\sky.jpg'

FileModDate: '08-١٦:٠٩:٥٢ ٢٠٠٧-ابريل'

FileSize: 575314

Format: 'jpg'

FormatVersion: ''

Width: 1280

Height: 960

BitDepth: 24

ColorType: 'truecolor'

FormatSignature: ''

NumberOfSamples: 3

CodingMethod: 'Huffman'

CodingProcess: 'Sequential'

Comment: {}

Orientation: 1

XResolution: 72

YResolution: 72

ResolutionUnit: 'Inch'

Software: 'ACD Systems Digital Imaging '

DateTime: '2007:03:25 01:04:50 '

YCbCrPositioning: 'Centered'

DigitalCamera: [1x1 struct]

٢- الصورة من برنامج الماتلاب :

مثال :

```
info=imfinfo('cameraman','tif')
```

والنتيجة الظاهرة مماثلة تماماً حيث تظهر نفس المعلومات عن الصورة .

تحويل الصورة إلى صورة ثنائية

يمكن تحويل الصورة أيا كان نوعها إلى صورة ثنائية (أبيض – أسود)
Binary Image باستخدام التعليمة im2bw بالشكل :

```
binary image=im2bw(image,level);
```

حيث image الصورة الأصلية

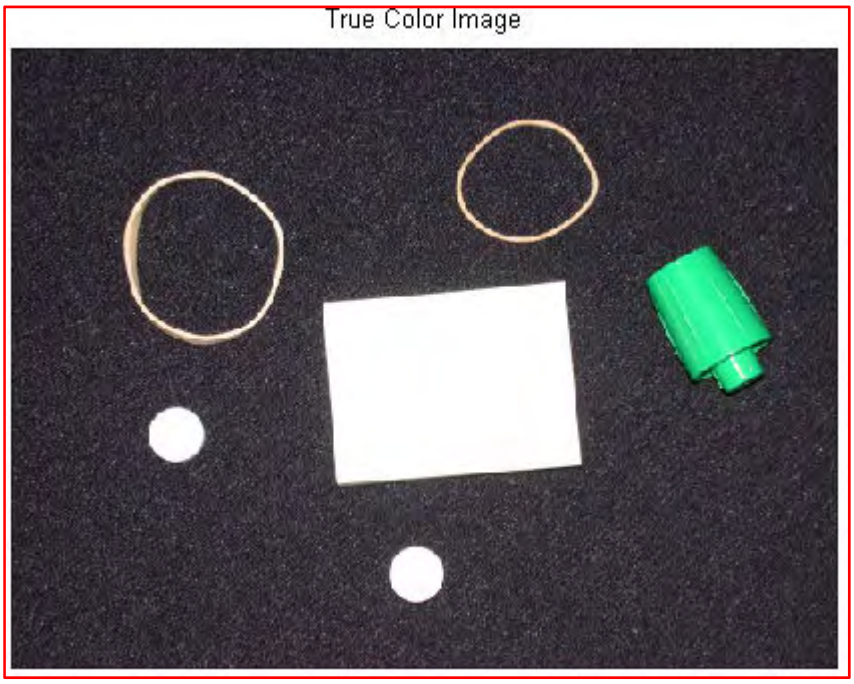
أما level فهي شدة العتبة

تتراوح قيمة شدة العتبة ضمن المجال [0 1] حيث تتحول جميع البكسلات التي تحمل قيمة فوق شدة العتبة level إلى اللون الأبيض أو تحمل القيمة '1' أما البكسلات التي تحمل قيم تحت شدة العتبة level إلى اللون الأسود أو تحمل القيمة '0' .

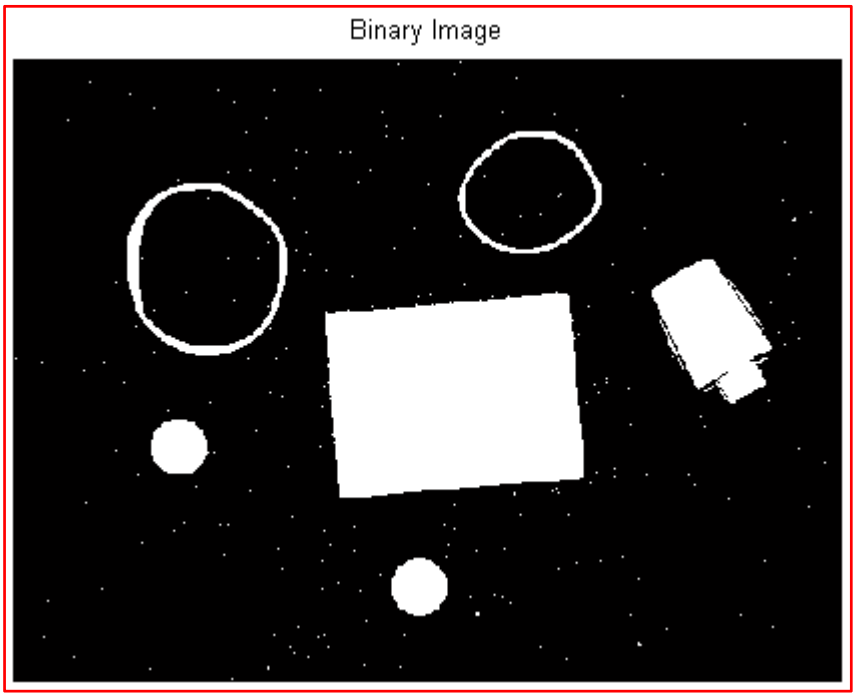
مثال :

```
I=imread('pillsetc.png');
imshow(I)
bw=im2bw(I,0.4);
figure, imshow(bw)
```

الصورة الأصلية :



الصورة الناتجة :



- نلاحظ أن شدة العتبة غير مناسبة بسبب احتواء الصورة الناتجة Binary Image على بعض التشويش (مجموعة منتشرة من النقط) في الصورة الثنائية .

كيف نحصل على شدة العتبة للصورة ؟

- تعبر شدة العتبة عن قيمة كثافة الصورة الطبيعية ولاختيار شدة العتبة بالشكل الأمثلي نستخدم التعليمة graythresh والتي تعطي رقم يقع ضمن المجال [0 , 1] يشير إلى شدة عتبة الصورة على الشكل التالي :

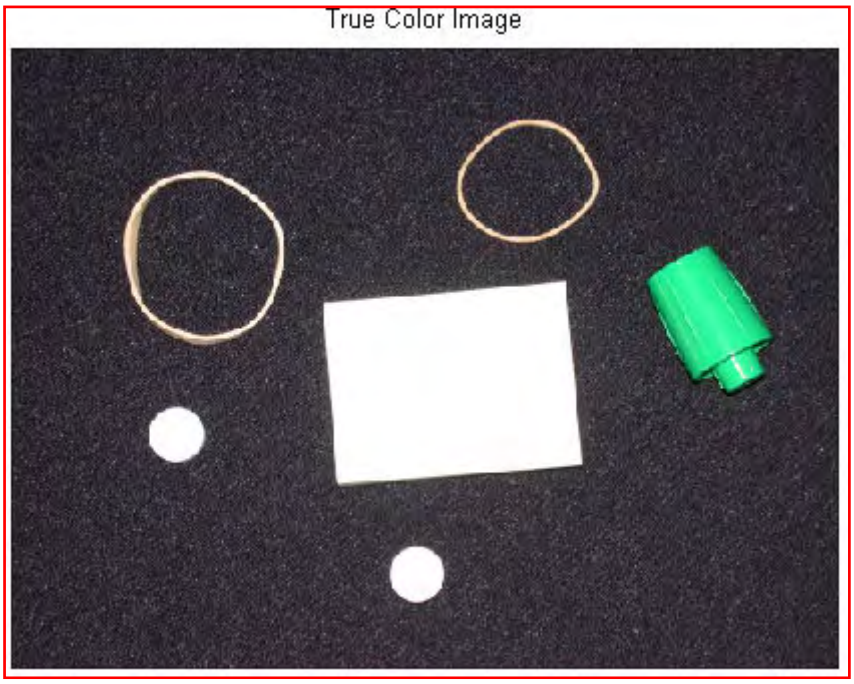
```
level=graythresh(image);
```

وبالتالي يمكن تعديل المثال السابق ليصبح على الشكل التالي :

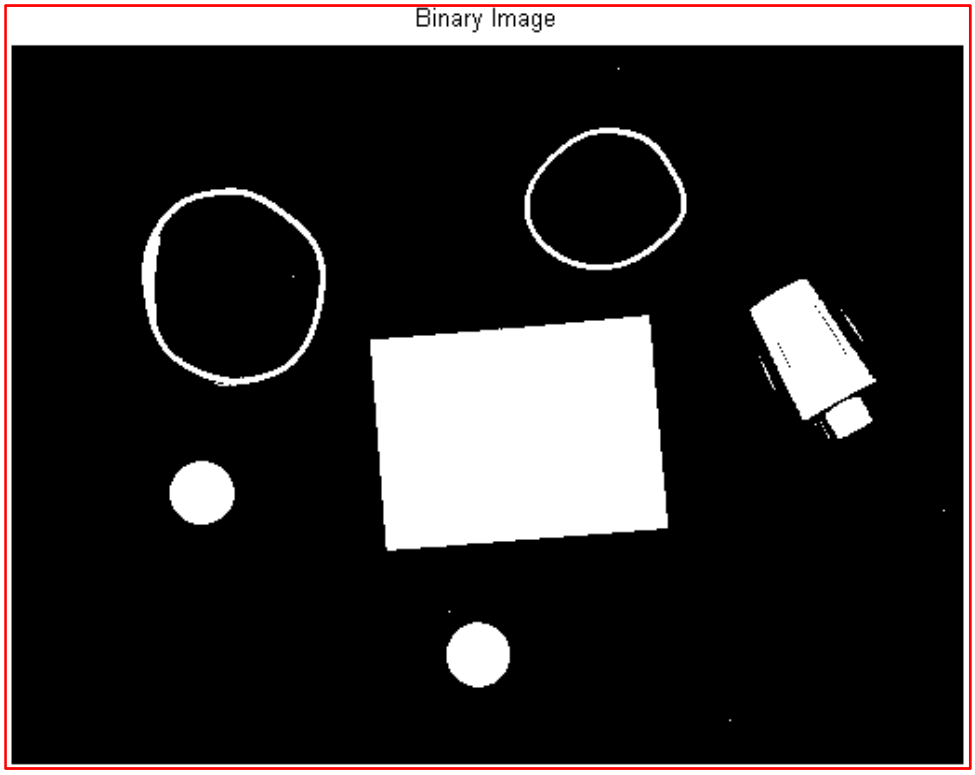
مثال :

```
I=imread('pillsetc.png');
imshow(I)
level=graythresh(I);
bw=im2bw(I,level);
figure, imshow(bw)
```

الصورة الأصلية :



الصورة الناتجة :



تحويل الصورة الثنائية إلى عدد من Objects

ما هو Object ؟

- في الصورة الثنائية كل جزء يحمل اللون الأبيض (أي يحمل الرقم 1) من الصورة وأحيط من جوانبه باللون الأسود (أي أحيطت بالرقم 0) يسمى بـ Object .

- في الماتلاب أصغر Object يتألف من اتصال أربع واحدات كحد أدنى أو من اتصال ثمان واحدات كحد أدنى حسبما نختار .

نتمة الموضوع :

يمكن تحويل الصورة الثنائية Binary Image إلى صورة مؤشرة labeled Image باستخدام التعليمة bwlabel .

ما هي الصورة المؤشرة labeled Image ؟

هي صورة تحوي على اللونين الأبيض والأسود فقط تماماً كالصورة الثنائية إلا أنه كل Object في الصورة (وهو مجموعة واحدات متجمعة في الصورة الثنائية) سيحمل رقم صحيح 1,2,3,4,……, num objects فإذا كان لدينا في الصورة الثنائية 3 Objects عندئذ فإن الصورة المؤشرة عناصرها تحمل الرقم 0 المقابل للون الأسود والرقم 1 المقابل للون الأبيض لأول object والرقم 2 المقابل للون الأبيض لثاني object والرقم 3 المقابل للون الأبيض لثالث object وهكذا .

مثال :

لنفرض أن الصورة الأصلية Binary Image تملك هذه المصفوفة :

$$bw = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

وتكون الصورة المؤشرة Labeled Image :

$$labeled = \begin{bmatrix} 1 & 1 & 0 & 2 & 2 & 0 \\ 1 & 1 & 0 & 2 & 2 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 3 & 0 & 3 \\ 1 & 1 & 0 & 3 & 3 & 3 \\ 1 & 1 & 0 & 3 & 3 & 3 \end{bmatrix}$$

مثال :

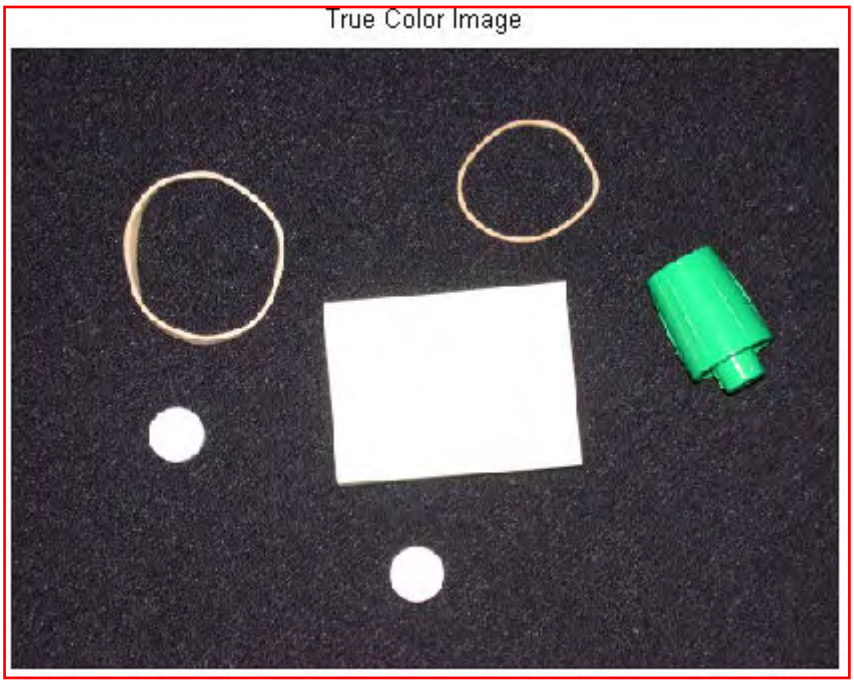
```
I=imread('pillsetc.png');
imshow(I)
level=graythresh(I);
bw=im2bw(I,level);
figure, imshow(bw)
[labeled,numObjects] = bwlabel(bw,4);
figure, imshow(labeled)
pseudo_color = label2rgb(labeled, @spring, 'c', 'shuffle');
figure, imshow(pseudo_color)
numObjects
```

- التعليمة label2rgb تستخدم لتلوين جميع Obejects في الصورة المؤشرة .

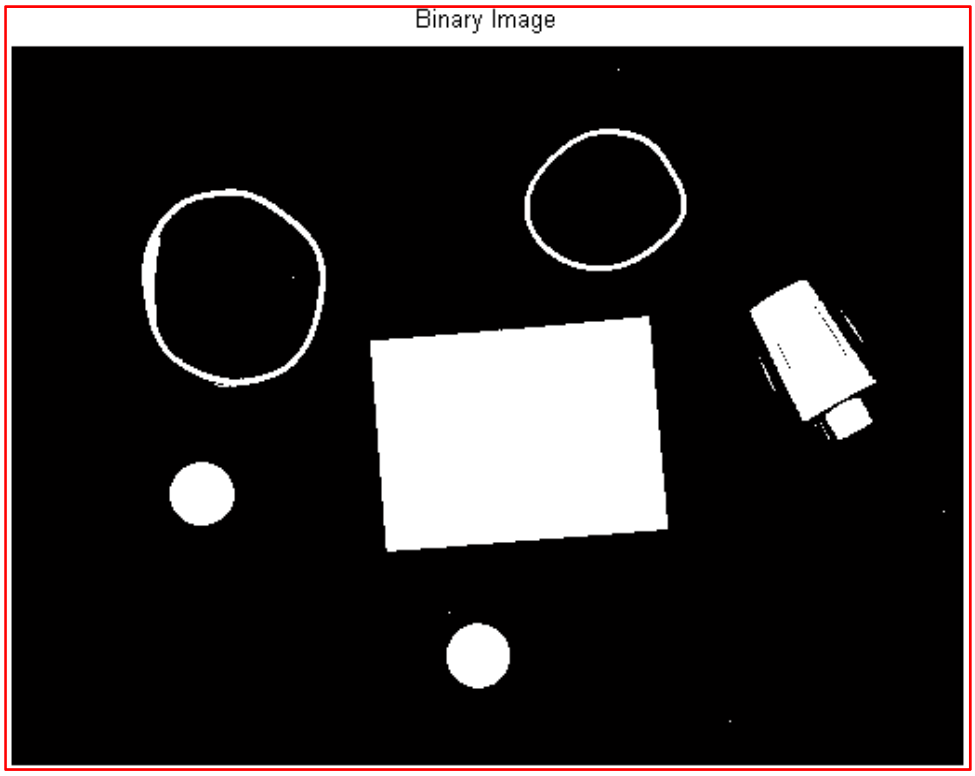
والمتحول numObjects يعطي عدد Objects في الصورة .

وكانت النتيجة كما يلي :

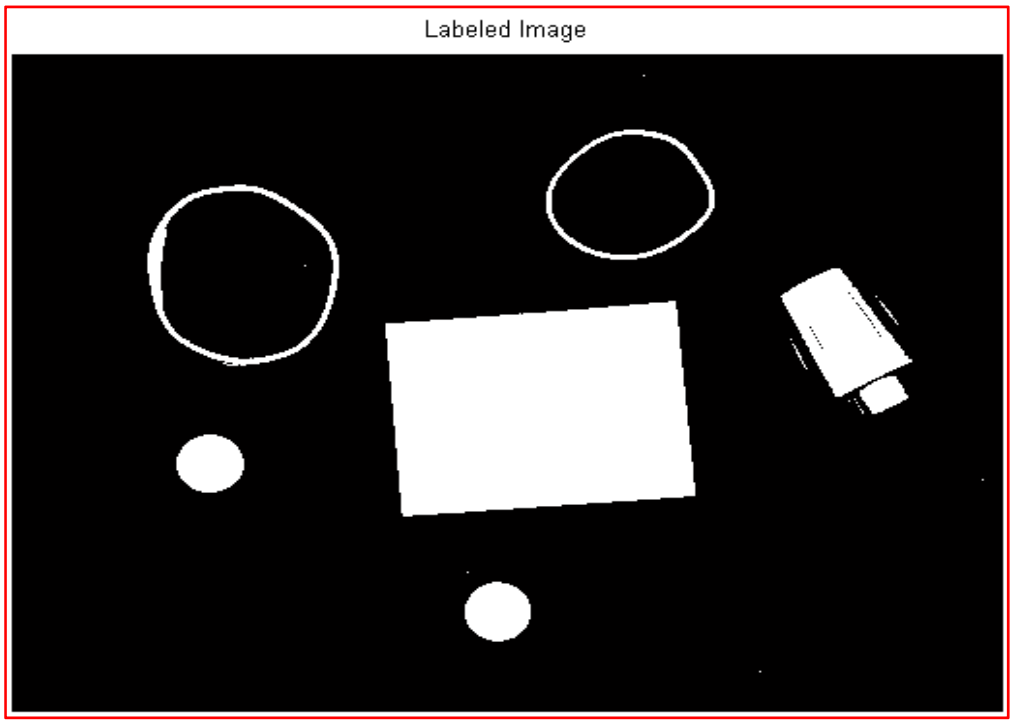
الصورة الأصلية :



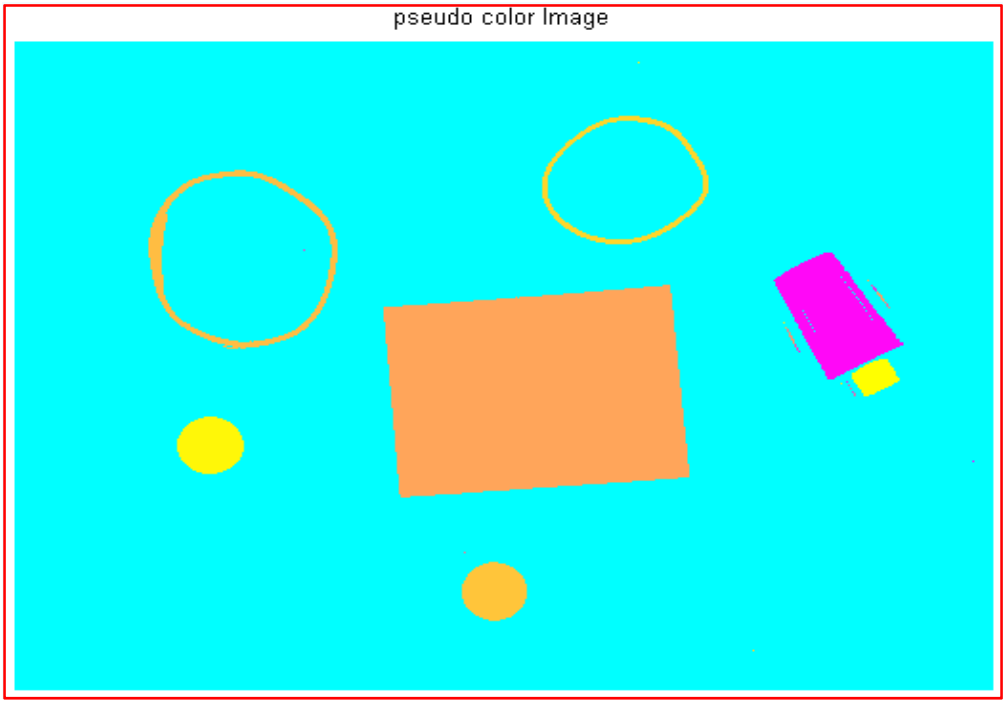
الصورة الثنائية :



الصورة المؤشرة labeled :



والصورة الملونة pseudo_color :



الحصول على خصائص الصورة المؤشرة

يمكن الحصول على قائمة الخواص للصورة المؤشرة labeled Image باستخدام التعليمة regionprops بالشكل :

```
info=regionprops(labeled_image,property_name);
```

لرؤية جميع خواص الصورة المؤشرة في المثال السابق .

مثال :

```
info=regionprops(labeled,'all')
```

والناتج هو

```
info =
32x1 struct array with fields:
    Area
    Centroid
    BoundingBox
    SubarrayIdx
    MajorAxisLength
    MinorAxisLength
    Eccentricity
    Orientation
    ConvexHull
    ConvexImage
    ConvexArea
    Image
```

FilledImage

FilledArea

EulerNumber

Extrema

EquivDiameter

Solidity

Extent

PixelIdxList

PixelList

Perimeter

نلاحظ وجود 32 خاصية .

حيث يمكن الوصول إلى كل خاصية من هذه الخواص كما يلي :

مثلاً إذا أردنا معرفة مساحة الـ Object الأول في المثال السابق :

```
info(1).Area
```

والنتيجة هي :

```
1173
```

أو إذا أردنا تخزين مساحات الـ Object في مصفوفة واحدة وعمود واحد يمكن استخدام التعليمة **cat** للقيام بذلك :

```
Areas=cat(1,info.Area)
```

أو في سطر واحد

```
Areas=cat(2,info.Area)
```

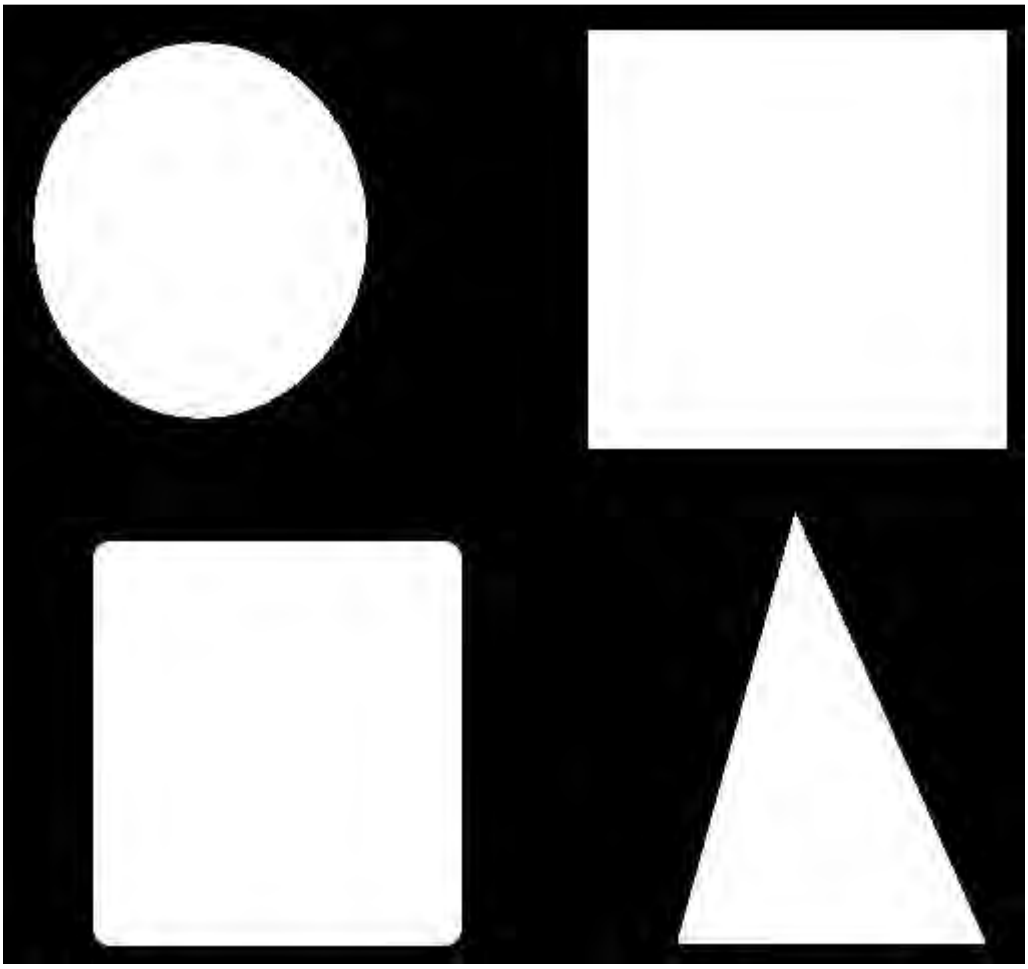
وهذه العملية تيسر الاستفادة من خصائص الصورة بشكل كبير وتنظم العمل أيضاً .

أخيراً :

إن كل خاصية من هذه الخواص تفيد في معالجة الصورة بشكل مباشر من معرفة مساحة كل Object وحدوده والكثير من الخصائص التي سأشرح القسم الأهم منها بإذن الله .

خصائص الصورة المؤشرة

لتكن لدينا الصورة الثنائية التالية :



أولاً قم بنسخ هذه الصورة الثنائية Binary Image إلى السواقة D ثم باستخدام الماتلاب نحولها إلى صورة مؤشرة labeled Image كما شرحنا في الدروس السابقة ونلاحظ أن الصورة تحوي على 4 Objects كل منها يحمل رقم 1,2,3,4 حيث يحمل Object الدائرة الرقم 1 ويحمل Object المربع ذو الحواف الرقم 2 أما Object المربع يحمل الرقم 3 و Object المثلث الرقم 4 حيث تتم عملية الترقيم وفق لعملية مسح الصورة عمود تلو الآخر ويعطى أول Object الرقم 1 والثاني رقم 2 وهكذا .

ويصبح الكود :

```
I=imread('D:\shapes.jpg');
level=graythresh(I);
bw=im2bw(I,level);
[labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
```

طبعاً قبل البدء بشرح خصائص الصورة نلاحظ أن الصورة تتألف من أربعة Objects ومن هذا الكود يعبر المتحول numObjects عن عدد Objects والتعليمة regionprops تعطي خصائص الصورة المؤشرة وهذه الخصائص التي سندرسها تصف كل Object من الصورة واخترت هذه الصورة لقلة عدد Objects مما يؤدي إلى سهولة في الفهم .

حيث سوف نقوم بدراسة خصائص الصورة التالية :

أولاً : مساحة كل Object في الصورة Area

ماذا نقصد بمساحة الـ Object ؟

المساحة هي مجموع بكسلات الصورة الواقعة ضمن الـ Object .

حيث تحسب مساحة الـ Object الأول (الدائرة) :

```
info(1).Area

ans =

        24496
```

مساحة الـ Object الثاني (المربع ذو الحواف) :

```
info(2).Area

ans =

        37100
```

مساحة الـ Object الثالث (المربع) :

```
info(3).Area

ans =

        43681
```

مساحة الـ Object الرابع (المثلث) :

```
info(4).Area

ans =

        16681
```

حيث تقدر مساحة الـ Object بالبكسل .

يمكن جمع المساحات في مصفوفة واحدة باستخدام التعليمة cat

```
Areas=cat(1,info.Area)
```

ثانياً : المركز الهندسي لكل Object في الصورة Centroid

ما المقصود بالمركز الهندسي للـ Object ؟

هو نقطة أو بكسل من الصورة إحداثياته (x , y) وهو عبارة عن مركز ثقل الـ Object فمثلاً المركز الهندسي للدائرة هو مركزها والمركز الهندسي للمربع هو نقطة التقاء قطريه .

المركز الهندسي للـ Object الأول (الدائرة) :

```
info(1).Centroid

ans =

    99    113.5
```

المركز الهندسي للـ Object الثاني (المربع ذو الحواف) :

```
info(2). Centroid

ans =

    137.5    369.5
```

المركز الهندسي للـ Object الثالث (المربع) :

```
info(3). Centroid

ans =

    397    118
```

المركز الهندسي للـ Object الرابع (المثلث) :

```
info(4). Centroid

ans =

    407.9634    397.5332
```

يمكن جمع المراكز الهندسية في مصفوفة واحدة باستخدام التعليمة cat

```
Centroids=cat(1,info.Centroid)
```

ثالثاً : إطار كل Object في الصورة BoundingBox

ما هو الإطار ؟

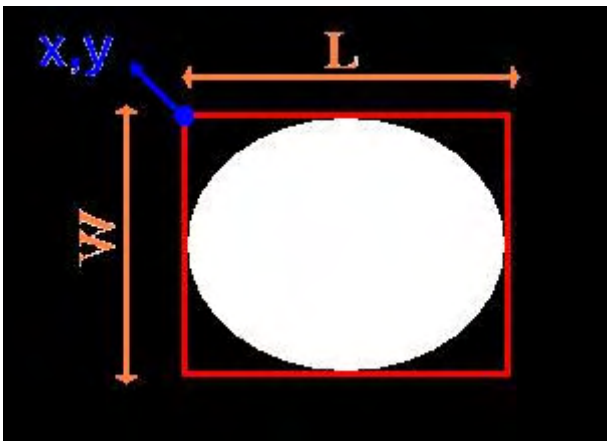
هو أصغر مستطيل يحوي الـ Object .

حيث تعطي هذه الخاصية إحداثيات النقطة العليا اليسارية من إطار Object وطول وعرض Object [x y L W] .

بماذا يفيد الإطار ؟

يفيد كشف الإطار في الكثير من التطبيقات منها عندما نريد التعرف على الأشكال والأحرف العربية أو الانكليزية وغيرها حيث يمكن الاستفادة من نسبة مساحة الـ Object وهي Area والتي تحسب من الخاصة الأولى إلى مساحة الإطار W x L في ذلك .

مثلاً الإطار المحيط بالدائرة له الشكل :



بالعودة إلى الشكل الأولي ومتابعة إيجاد إطار كل شكل من الأشكال :

نجد أن إطار الـ Object الأول (الدائرة) له البارمترات التالية :

```
info(1).BoundingBox

ans =

    15.5000   19.5000  167.0000  188.0000
```

بارمترات إطار الـ Object الثاني (المربع ذو الحواف) :

```
info(2). BoundingBox

ans =

    45.5000  268.5000  184.0000  202.0000
```

بارمترات إطار الـ Object الثالث (المربع) :

```
info(3). BoundingBox

ans =

    292.5000   13.5000  209.0000  209.0000
```

بارمترات إطار الـ Object الرابع (المثلث) :

```
info(4). BoundingBox

ans =

    337.5000  254.5000  153.0000  215.0000
```

يمكن جمع بارمترات الإطارات في مصفوفة واحدة باستخدام التعليمة cat

```
BoundingBoxes=cat(1,info.BoundingBox)
```

رابعاً : رقم Euler لكل Object في الصورة EulerNumber

ما هو رقم Euler ؟

هو رقم يمكننا حسابه من حساب عدد الثقوب في الصورة

رقم Euler للـ Object = ١ – عدد الثقوب ضمن الـ Object

من خلال رقم Euler يمكن الحصول على عدد الثقوب ضمن الـ Object

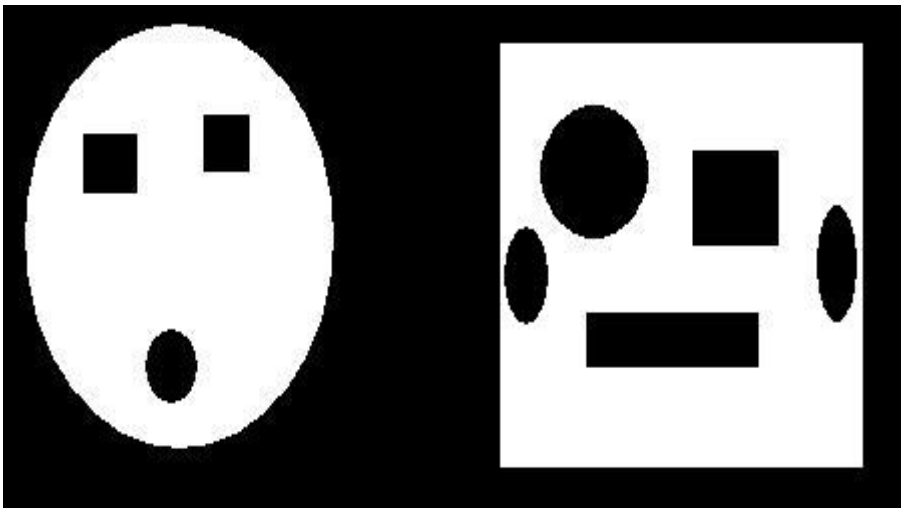
عدد الثقوب ضمن الـ Object = ١ - رقم Euler للـ Object

ما هو الثقب Holes ؟

هو نقيض الـ Object أي هو كل جزء من الصورة الثنائية لونه أسود أحيط من جوانبه باللون الأبيض ويقع الثقب ضمن الـ Object .

مثال :

لتكن لدينا الصورة التالية :



وكود حساب عدد الثقوب في كلا الـ Object :

```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
bw=im2bw(I,level);
 [labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Euler_Number1=info(1).EulerNumber;
NumHoles1=1-Euler_Number1
Euler_Number2=info(2).EulerNumber;
NumHoles2=1-Euler_Number2
```

والنتيجة هي :

```
NumHoles1 =

     3



NumHoles2 =

     5
```

خامساً : مدى الـ Object في الصورة Extent

ما هو المدى ؟

مدى الـ Object هو نسبة مساحة الـ Object إلى مساحة إطار الـ Object .

Extent=Area (Object) / Area (BoundingBox)

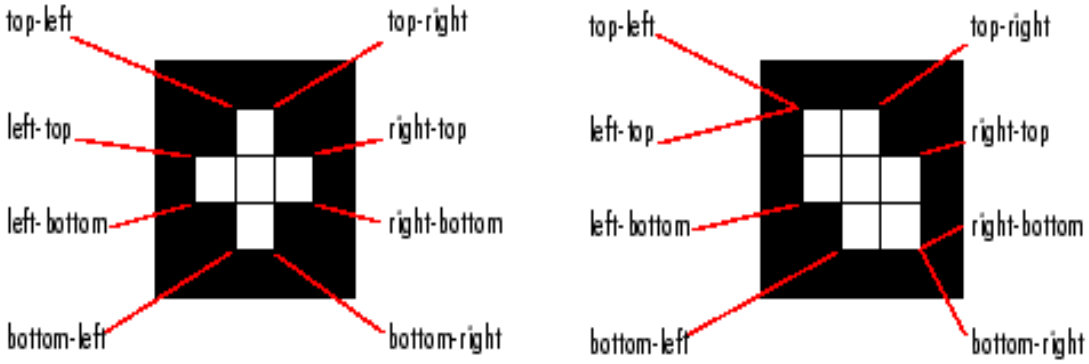
ولحساب المدى لكلا الـ Object :

```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
bw=im2bw(I,level);
 [labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Extent1=info(1).Extent
Extent2=info(2).Extent
```

والنتيجة هي :

```
Extent1=
        0.7129
Extent2=
        0.7678
```

سادساً : زوايا الـ Object في الصورة Extrema



تعطي هذه الخاصية ثمان نقاط تحدد الـ Object من الجهات الأربع

(العليا – اليمينية – السفلى – اليسارية) ومرتبة على الشكل التالي :

top-left

top-right

right-top

right-bottom

bottom-right

bottom-left

left-bottom

left-top

حيث top-right تعني النقطة الأعلى من الـ Object أولاً والأكثر يمينية ثانياً

أما right-top تعني النقطة الأكثر يمينية من الـ Object أولاً والأعلى ثانياً

تفيد هذه الخاصية في تحديد شكل Object من خلال معرفة مواقع النقاط الثمانية .

مثال :

```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
```

```
bw=im2bw(I,level);
 [labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Extrema1=info(1).Extrema
Extrema2=info(2).Extrema
```

والنتيجة هي الحصول على إحداثيات x ,y للنقاط الثمانية :

```
Extrema1 =
   84.5000    8.5000
   92.5000    8.5000
  165.5000   91.5000
  165.5000  103.5000
   92.5000  186.5000
   84.5000  186.5000
   11.5000  103.5000
   11.5000   91.5000
Extrema2 =
  248.5000   16.5000
  429.5000   16.5000
  429.5000   16.5000
  429.5000  194.5000
  429.5000  194.5000
  248.5000  194.5000
  248.5000  194.5000
  248.5000   16.5000
```

سابعاً : مساحة الـ Object بعد ملء الثقوب في الصورة FilledArea

بماذا تفيد هذه الخاصية ؟

تفيد هذه الخاصة معرفة مساحة الـ Object و هو ممتلئ في معرفة مساحة الثقوب داخل الـ Object

مساحة الثقوب ضمن الـ Object = مساحة الـ Objet و هو ممتلئ – مساحة الـ Object بوجود الثقوب .

مثال :

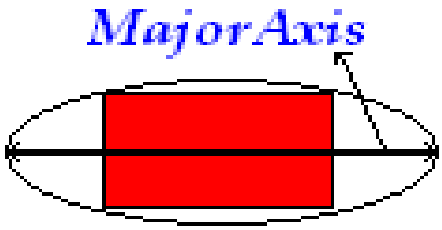
```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
bw=im2bw(I,level);
 [labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Object_Area1=info(1).Area;
Object_Filled_Area1=info(1).FilledArea;
Object_Area2=info(2).Area;
Object_Filled_Area2=info(2).FilledArea;
Holes_Area1= Object_Filled_Area1- Object_Area1
Holes_Area2= Object_Filled_Area2- Object_Area2
```

والنتيجة هي :

```
Holes_Area1 =
     1839
Holes_Area2 =
     7481
```

ثامناً :طول القطر الرئيسي للـ MajorAxisLength Object

إن هذه الخاصية تعطي طول القطر الرئيسي للشكل البيضاوي المحيط بالـ Object مقدرةً بالبكسل والشكل التالي يوضح ذلك :



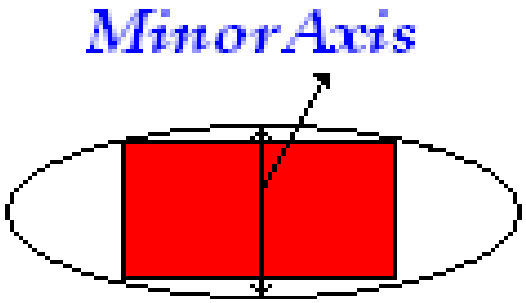

```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
bw=im2bw(I,level);
 [labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Major_Axis1=info(1).MajorAxisLength
Major_Axis2=info(2).MajorAxisLength
```

والنتيجة هي :

```
Major_Axis1 =
  177.8260
Major_Axis2 =
  224.9245
```

تاسعاً :طول القطر الثانوي للـ MinorAxisLength Object

إن هذه الخاصية تعطي طول القطر الثانوي للشكل البيضوي المحيط بالـ Object مقدرةً بالبكسل والشكل التالي يوضح ذلك :



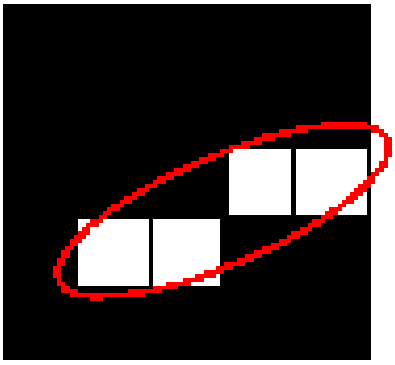

```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
bw=im2bw(I,level);
 [labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Minor_Axis1=info(1).MinorAxisLength
Minor_Axis2=info(2).MinorAxisLength
```

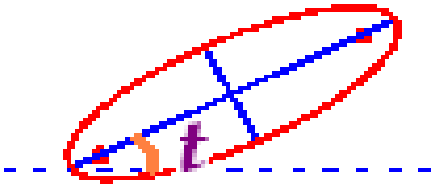
والنتيجة هي :

```
Minor_Axis1 =
  157.4666
Minor_Axis2 =
  214.1390
```

أخيراً الزاوية بين القطر الرئيسي والأفق للـ Orientation Object

إن هذه الخاصية تعطي الزاوية بين القطر الرئيسي والأفق بالـ Object مقدرةً بالدرجات وتتراوح ما بين [90 90-]






```
I=imread('D:\Shapes1.jpg');
level=graythresh(I);
bw=im2bw(I,level);
[labeled,numObjects]=bwlabel(bw,4);
info=regionprops(labeled,'all');
Orientation1=info(1). Orientation
Orientation2=info(2). Orientation
```

والنتيجة هي :

```
Orientation1 =
   89.4732
Orientation2 =
   73.5117
```

تعديل تباين الصورة الرقمية

يمكن تعديل تباين الصورة الرقمية باستخدام التعليمة imadjust والتي تملك عدة أشكال :

الشكل الأول :

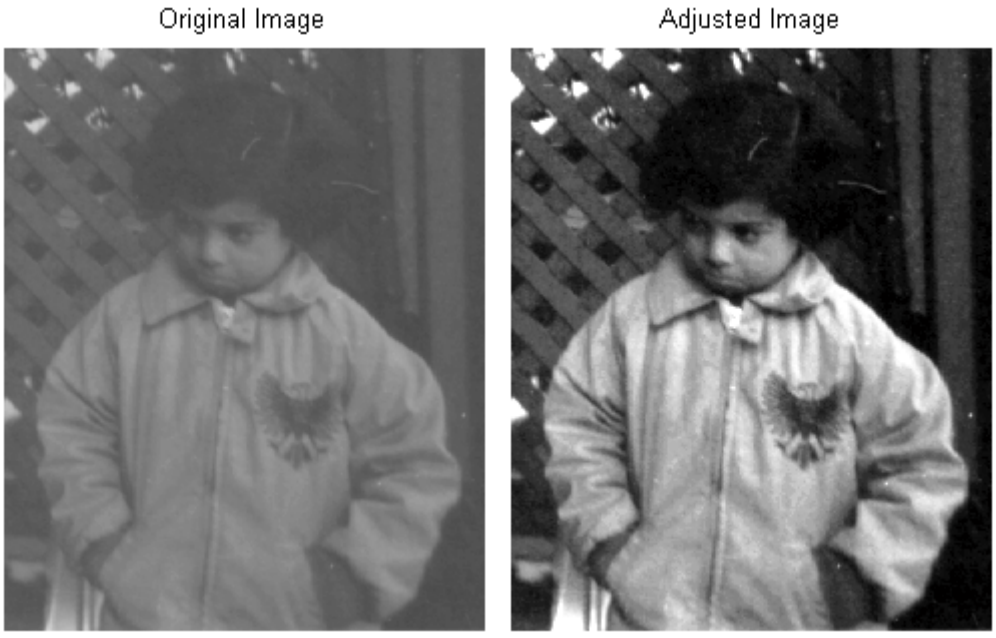
J =imadjust(I);

وبذلك تنتج صورة جديدة J هي نفس الصورة مع زيادة في تباين الصورة من خلال زيادة درجة الإشباع للبيانات بمقدار 1% .

مثال :

```
I=imread('pout.tif');
imshow(I)
J =imadjust(I);
figure,imshow(J)
```

والنتائج كما يلي :



الشكل الثاني :

```
J =imadjust(I,[low_in;high_in],[low_out;high_out]);
```

حيث يتم تعديل تباين الصورة من خلال تعديل جميع البكسلات في الصورة I والتي قيمها تتراوح بين low_in و High_in والتي ستتغير قيمها وتنحصر ما بين low_out و high_out بشكل متقابل أما البكسلات التي تقع قيمها خارج low_in فتصبح قيمها 0 أي تحمل اللون الأسود والبكسلات التي تقع قيمها خارج high_in فتصبح قيمها 1 أي تحمل اللون الأبيض إذا كانت الصورة من نوع grayscale أو الأحمر أو الأخضر أو الأزرق إذا كانت الصورة من نوع RGB .

مثال :

```
I=imread('pout.tif');

imshow(I)

J =imadjust(I,[0.2;0.6],[0.4;0.8]);

figure,imshow(J)
```

والنتائج كما يلي :

ويمكن اختيار القيم [low_in;high_in] أو القيم [low_out;high_out] كـ [ ] للدلالة على أنها تحمل القيم الافتراضية [0 1] .

مثال :

```
I=imread('pout.tif');

imshow(I)

J =imadjust(I,[0.2;0.7],[ ]);

figure,imshow(J)
```

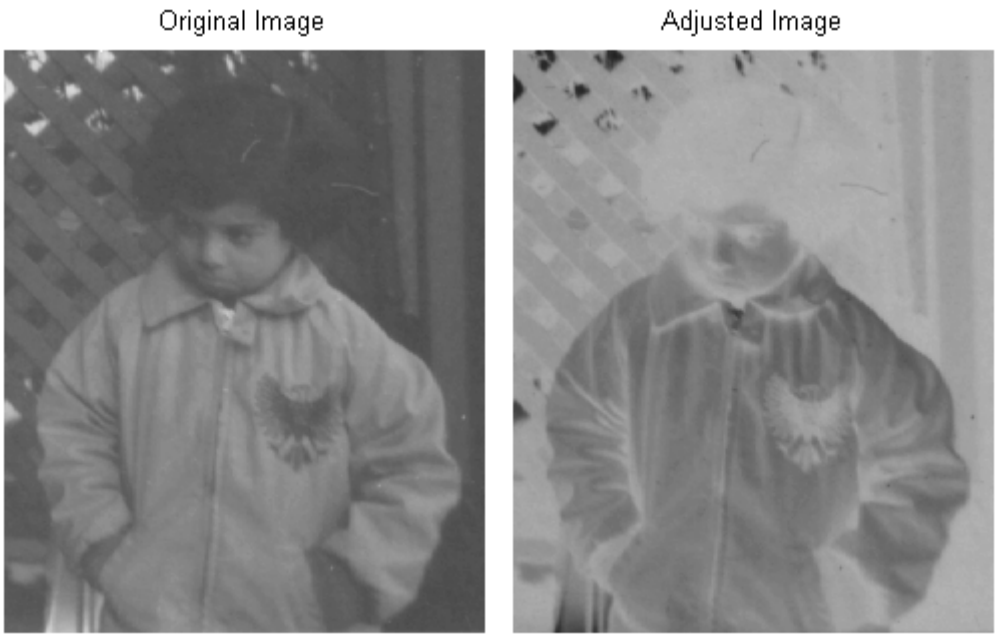
والنتائج كما يلي :

كما يمكن الحصول على Negative من الصورة نجعل Low_out > High_out .

مثال :

```
I=imread('pout.tif');
imshow(I)
J1=imadjust(I,[ ],[1;0]);
figure,imshow(J1)
```

والنتائج كما يلي :



الأمر لا يختلف بالنسبة للصورة الملونة :

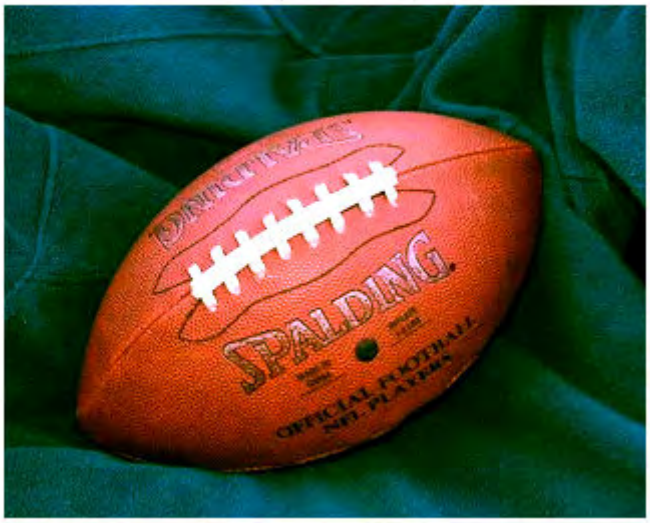
مثال :

```
I=imread('football.jpg');
imshow(I)
J1=imadjust(I,[0.2 0.1 0.3;0.6 0.8 0.7],[0.3 0.3 0.2;0.8 0.9 0.5]);
figure,imshow(J1)
J2=imadjust(I,[0.2 0.1 0.3;0.6 0.8 0.7],[]);
figure,imshow(J2)
J3=imadjust(I,[],[1 1 1;0 0 0]);
figure,imshow(J3)
```

والنتائج كما يلي :

J1 Adjusted Image

J2 Adjusted Image



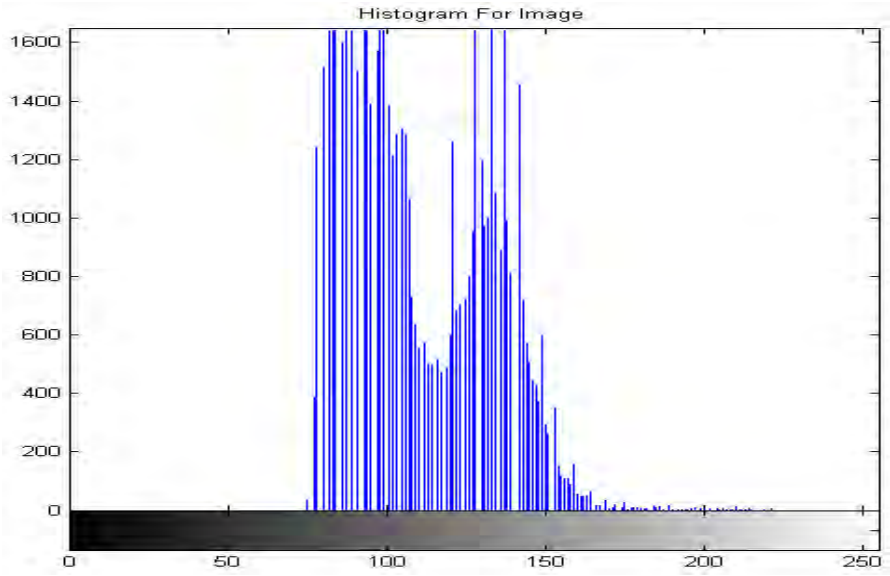
J3 Negative Image

تحسين تباين الصورة

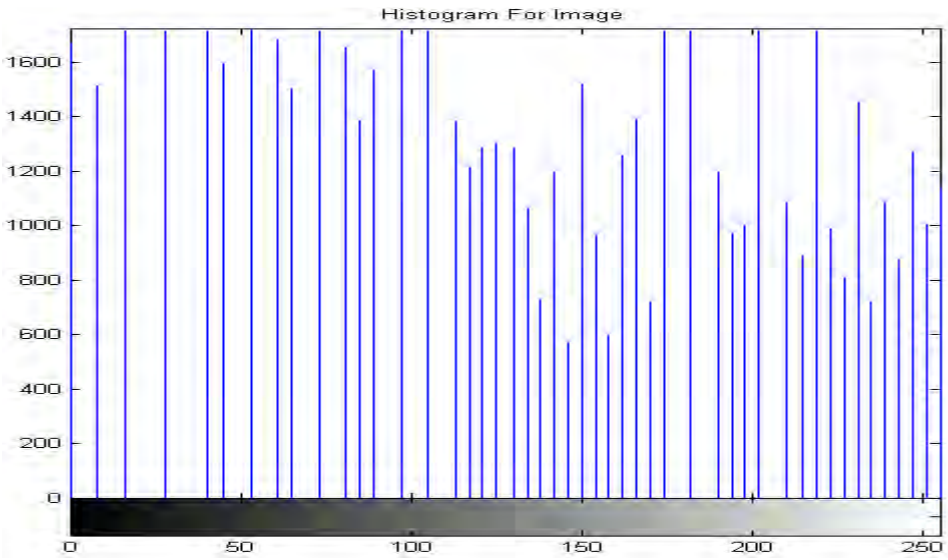
يمكن تحسين تباين الصورة باستخدام التعليمة histeq كما يلي :

```
I = imread('pout.tif');
imshow(I)
figure, imhist(I)
I2 = histeq(I);
figure, imshow(I2)
figure, imhist(I2)
```

والنتائج كما يلي :



Adjusted Image



اقتطاع خلفية الصورة

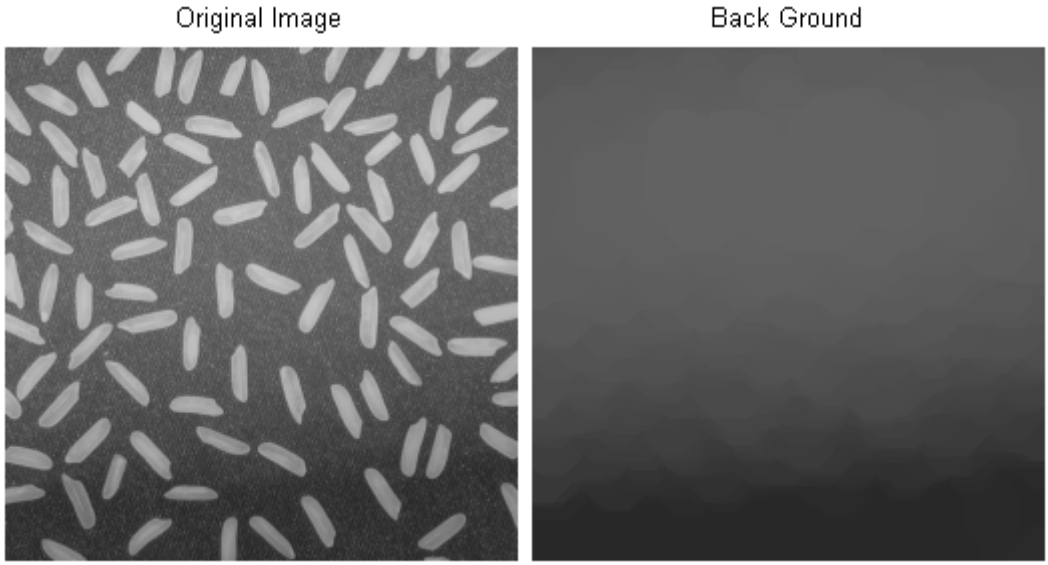
يمكن اقتطاع خلفية الصورة باستخدام التعليمة imopen كما في المثال التالي :

مثال :

```
I=imread('rice.png');
imshow(I)
background = imopen(I,strel('disk',15));
figure, imshow(background)
```

التعليمة streal تنشئ مصفوفة أصفار وواحدات تكون الواحدات فيها على شكل قرص نصف قطره 15 والتعليمة imopen تقوم بفتح الصورة بشكل صرفي متناسب مع عنصر التركيب streal .

والنتائج كما يلي :



جعل خلفية الصورة سوداء تماماً

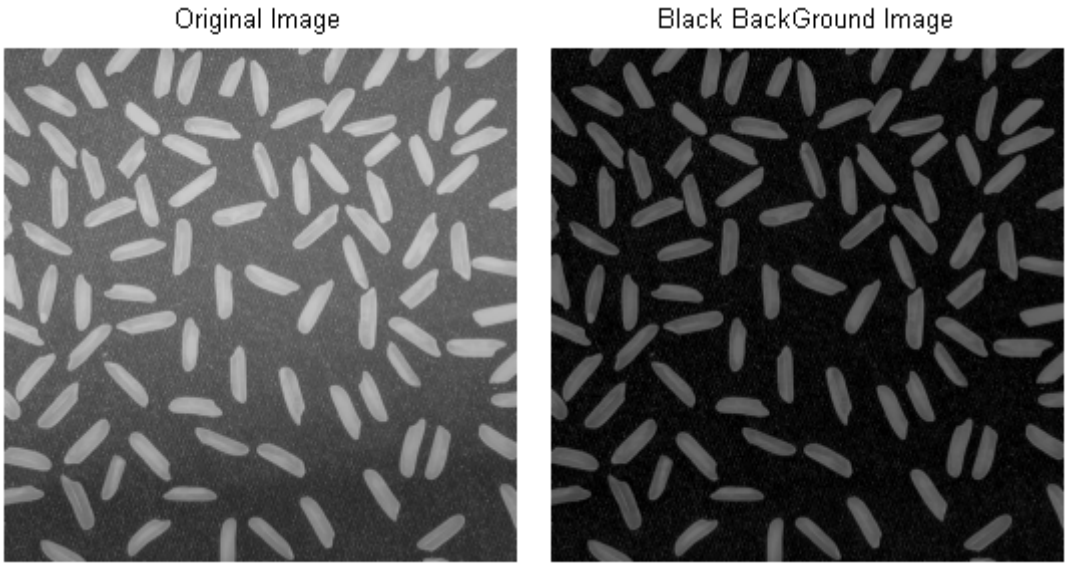
بعد اقتطاع خلفية الصورة كما في المثال في الموضوع السابق يمكن طرح الخلفية من الصورة الأصلية فنحصل على صورة فيها خلفية سوداء تماماً .

مثال :

```
I=imread('rice.png');

imshow(I)

background = imopen(I,strel('disk',15));

I2=imsubtract(I,background);

figure, imshow(I2)
```

حيث تقوم التعليمة imsubtract بطرح الصورتين من بعضهما وستشرح هذه التعليمة لاحقاً .

والنتائج كما يلي :



تغيير حجم الصورة (تكبير – تصغير)

يمكن تغيير حجم الصورة ثنائية البعد سواء كانت من نوع grayscale أو RGB أو Binary باستخدام التعليمة imresize وفق أحد الأشكال التالية :

الشكل الأول :

J=imresize(I,Scale)

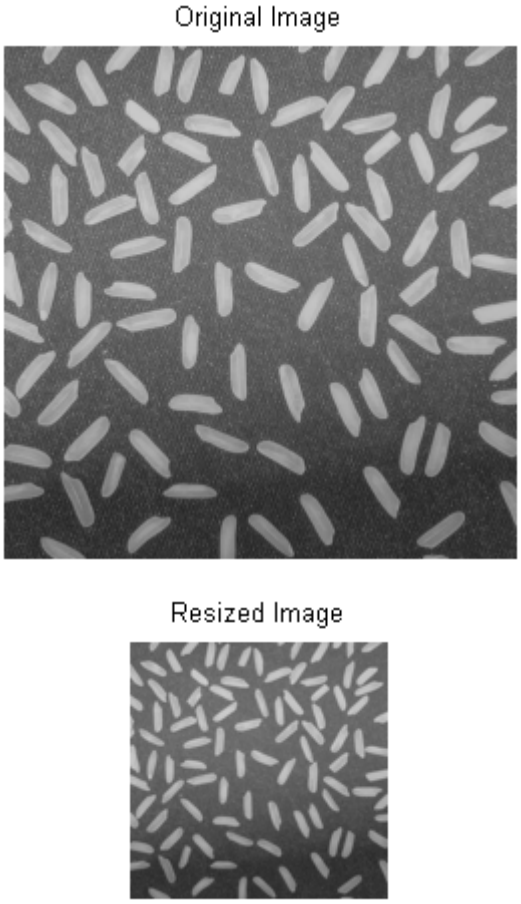
حيث I الصورة قبل إعادة التحجيم و Scale نسبة إعادة التحجيم فإذا كانت اكبر من الواحد يزداد حجم الصورة أما إذا كانت اصغر من الواحد ينقص حجم الصورة .

مثال :

```
I = imread('rice.png');

J = imresize(I, 0.5);

Orginal_size = size(I)

After_size = size(J)

figure, imshow(I), figure, imshow(J)
```

والنتائج كما يلي :

```
Orginal_size =

   256   256

After_size =

   128   128
```



الشكل الثاني :

J=imresize(I,[nrows ncol])

في هذا الشكل يمكننا إعادة تحجيم الصورة إلى الحجم الذي نرغب به

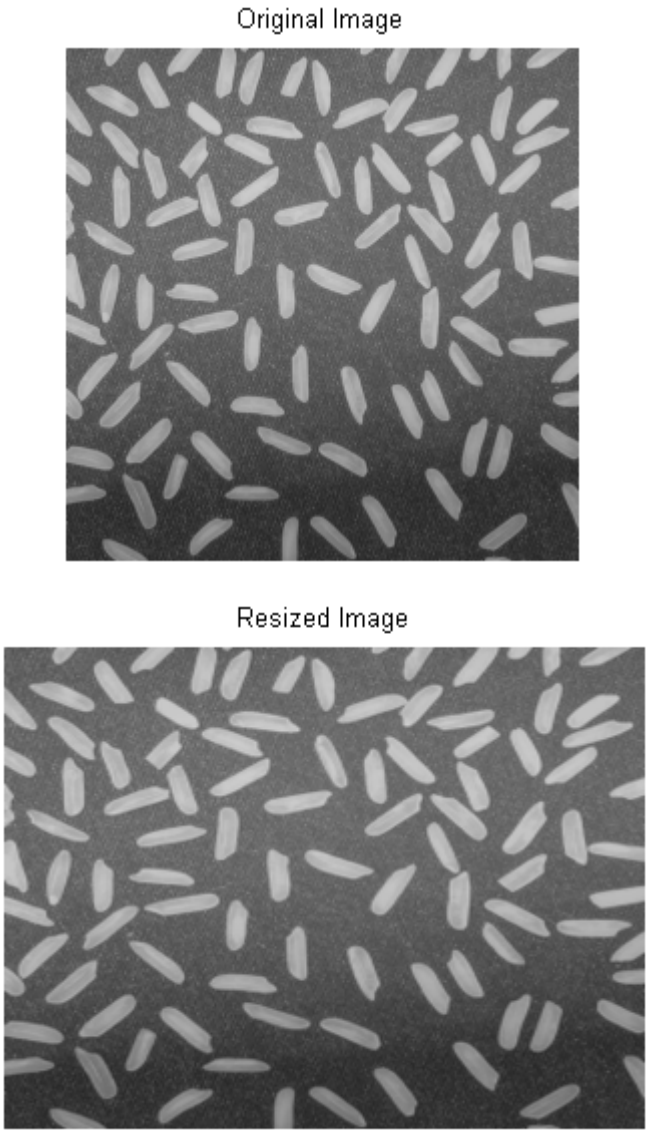
مثال :

```
I = imread('rice.png');

J = imresize(I,[240 320]);

Orginal_size = size(I)

After_size = size(J)

figure, imshow(I), figure, imshow(J)
```

والنتائج كما يلي :

```
Orginal_size =

   256   256

After_size =

   240   320
```

والصور



الشكل الثالث :

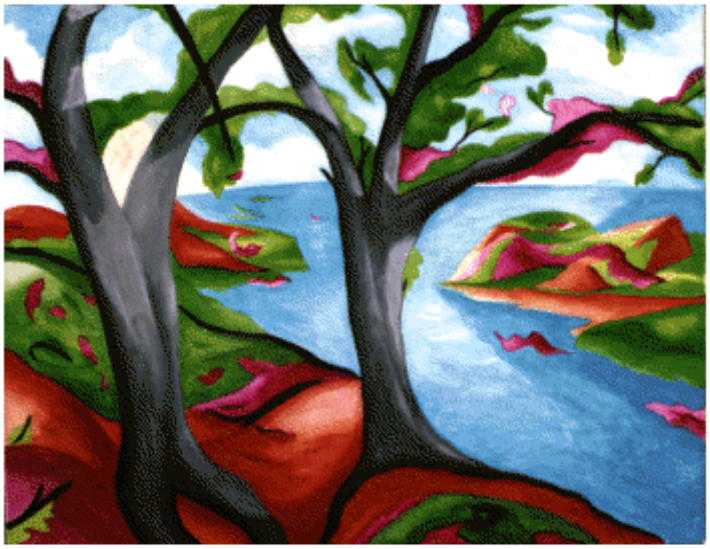
وهو خاص بالصور من نوع GrayScale :

```
 [X, map] = imread('trees.tif');
imshow(X, map)
 [Y, newmap] = imresize(X, map, 1.5);
figure,imshow(Y, newmap)
Orginal_size = size(X)
After_size = size(Y)
```

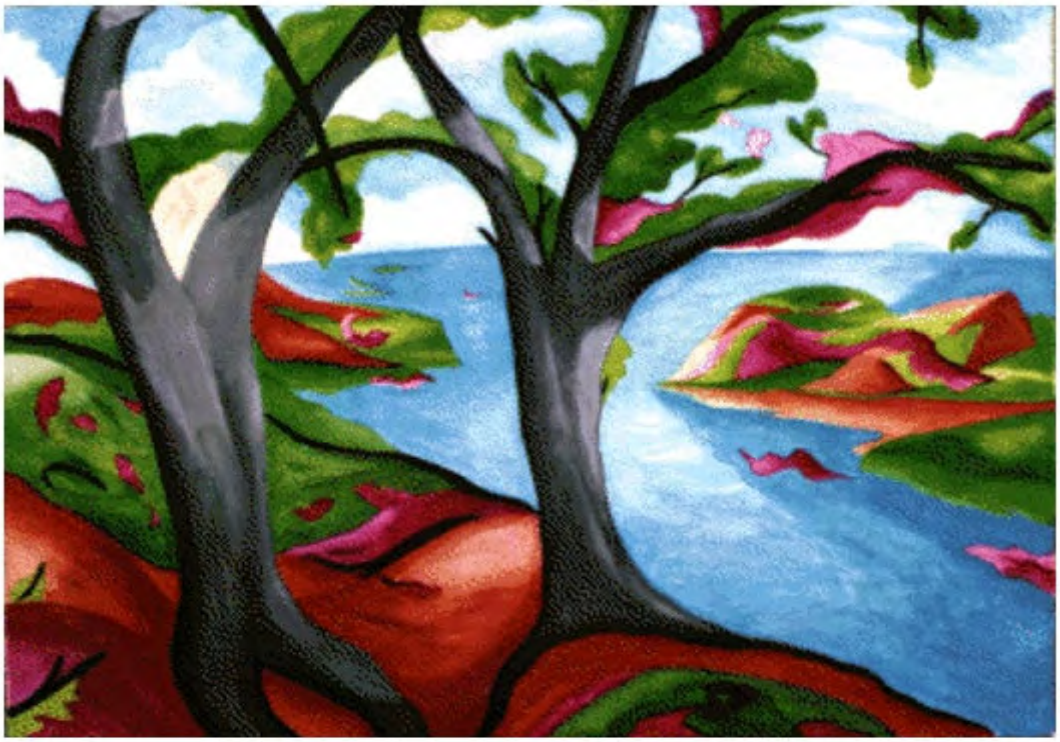
والنتائج كما يلي :

```
Orginal_size =
   258   350

After_size =
   387   525
```

Original Image



Resized Image



تدوير الصورة بزاوية معينة

يمكن تدوير الصورة بزاوية معينة مقدرة بالدرجات باستخدام التعليمة imrotate :

```
J=imrotate(I,Angle,Method,bbox);
```

حيث I الصورة المراد تدويرها و Angle زاوية الدوران مقدرة بالدرجات فإذا كانت Angle > 0 عندئذ الدوران يتم بعكس عقارب الساعة والعكس بالعكس .

أما Method (اختيارية) يمكن أن تأخذ إحدى القيم التالية :

. 'nearest','bilinear','bicubic'

أما bbox (اختيارية) تفيد تحديد حجم الصورة الناتجة فإذا أردنا أن يكون حجم الصورة الناتجة نفس حجم الصورة الأصلية نكتب 'crop' أما افتراضياً حجم الصورة الناتجة اكبر من حجم الصورة الأصلية نكتب 'loose' .

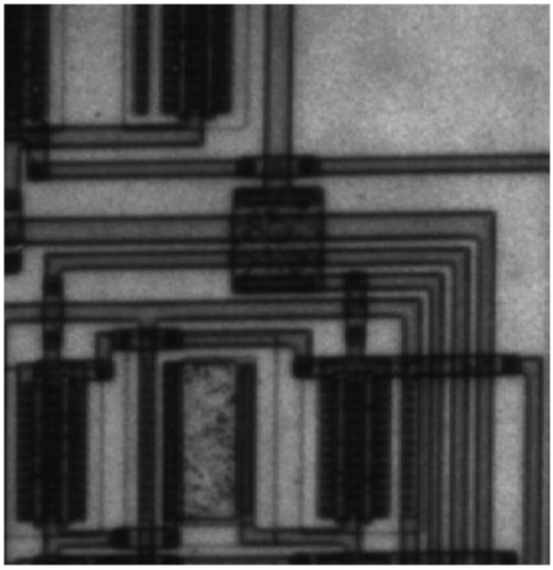
مثال ١ :

```
I = imread('circuit.tif');
J = imrotate(I,45,'bilinear');
Orginal_size = size(I)
After_size = size(J)
imshow(I)
figure, imshow(J)
```

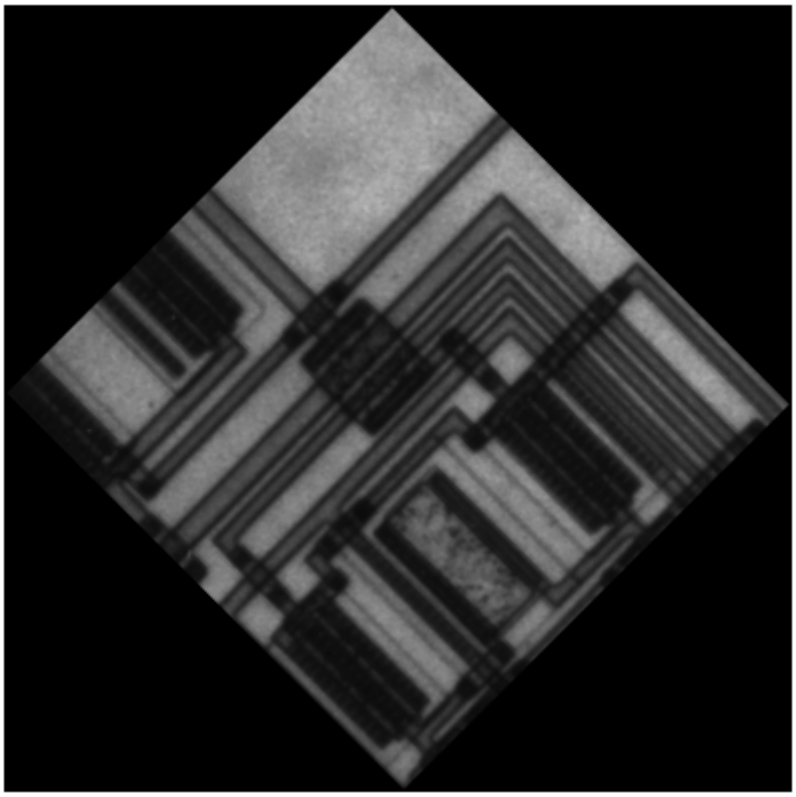
والنتائج كما يلي :

```
Orginal_size =
  280   272
After_size =
  393   393
```

Original Image



Rotated Image



أما أردنا أن يكون للصورة الناتجة نفس حجم الصورة الأصلية لكن نلاحظ اقتطاع جزء من الصورة .

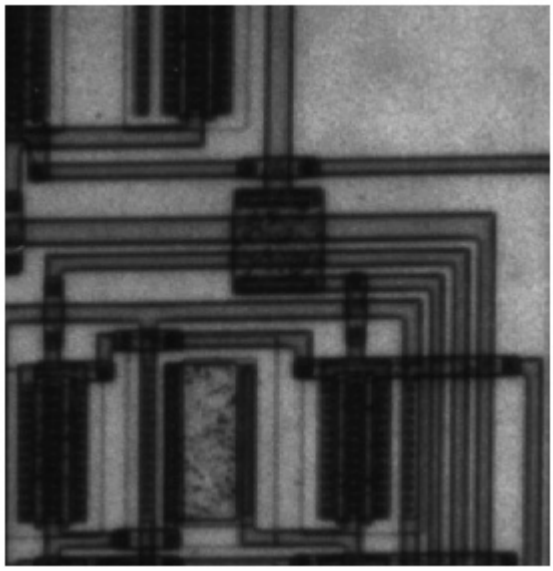
مثال ٢ :

```
I = imread('circuit.tif');
J = imrotate(I, 45,'bilinear','crop');
Orginal_size = size(I)
After_size = size(J)
imshow(I)
figure, imshow(J)
```

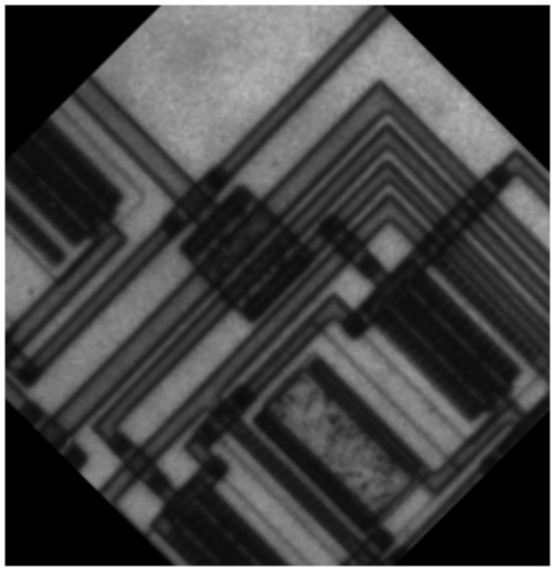
والنتائج كما يلي :

```
Orginal_size =
   280   272
After_size =
   280   272
```

Original Image



Rotated Image



اقتطاع جزء من الصورة

يمكن اقتطاع جزء من صورة وإنشاء صورة جديدة من المقطع الجديد .

الشكل الأول :

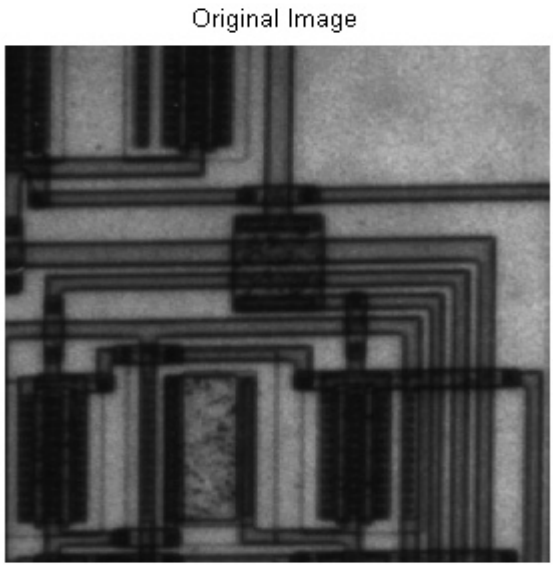
```
J=imcrop(I);
```

هنا تنشأ الصورة المقتطعة J يدوياً حيث يتوجب عليك تحديد الجزء من الصورة الذي تريد اقتطاعه يدوياً .

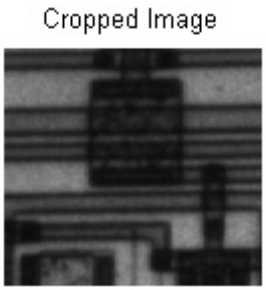
مثال :

```
I = imread('circuit.tif');

I2 = imcrop(I);

imshow(I), figure, imshow(I2)
```

والنتائج كما يلي :



وبعد اقتطاع الصورة يدويا نجد :



الشكل الثاني :

J=imcrop(I,rect);

هنا تنشأ الصورة المقتطعة J آلياً بتحديد [xmin ymin Length Width] نقطة البداية العليا اليسارية ذات الإحداثيات xmin , ymin وطول المستطيل Length وعرضه Width .

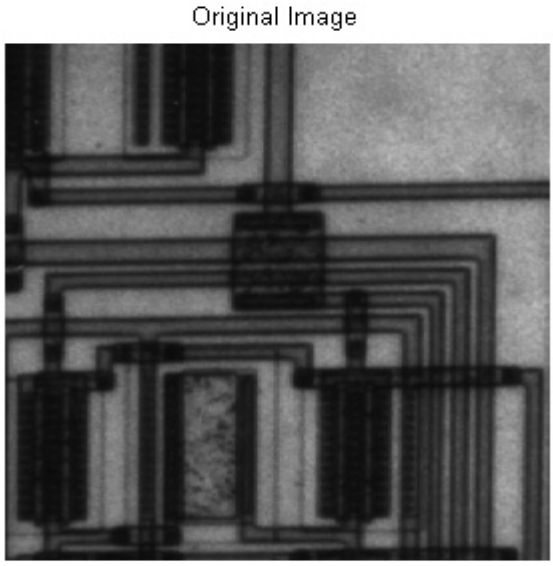
مثال :

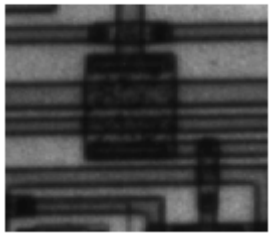
I = imread('circuit.tif');

I2 = imcrop(I,[75 68 130 112]);

imshow(I), figure, imshow(I2)

والنتائج كما يلي :





العمليات على الصور الثنائية

١ـ اختيار عدة Objects من صورة ثنائية bwselect :

يمكن اختيار عدة Objects من صورة ثنائية وتخزينها في صورة أخرى تحوي فقط على Objects التي تم اختيارها باستخدام التعليمة bwselect وفق شكلين :

الشكل الأول :

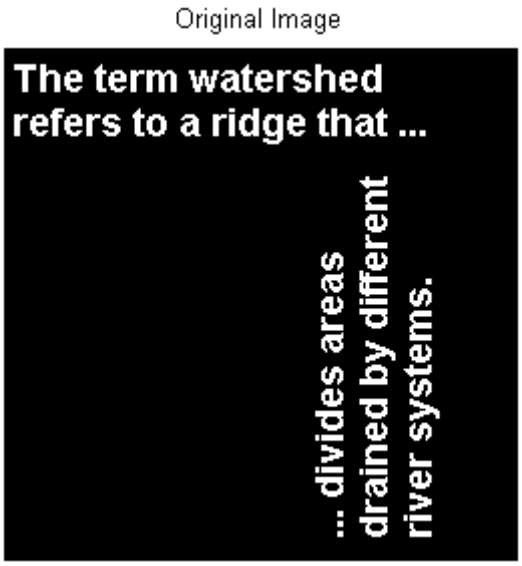
```
bw2=bwselet(bw1);
```

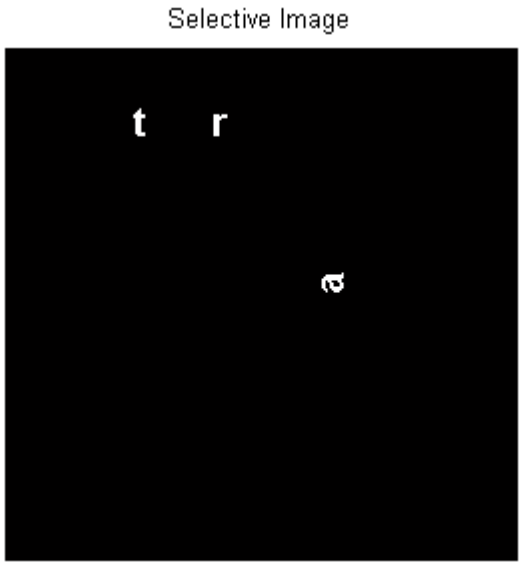
يتم اختيار Objects من الصورة bw يدوياً بالضغط مرة واحدة على Object وبعد الانتهاء من الاختيار نضغط Enter .

مثال :

```
bw1 = imread('text.png');
bw2=bwselect(bw1);
imshow(bw1)
figure , imshow(bw2)
```

والنتائج كما يلي :





الشكل الثاني :

bw2=bwselet(bw1,c,r,n);

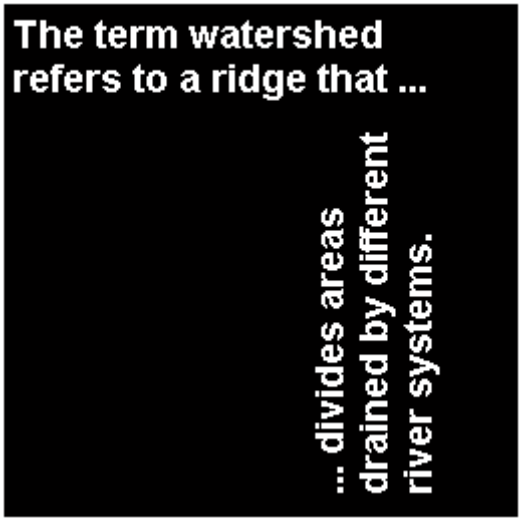
يتم اختيار Objects من خلال تحديد إحداثيات نقطة واحدة من كل Object وبالتالي نحدد عدة نقاط حيث c تحوي الإحداثيات الأفقية x و r تحوي الإحداثيات الشاقولية y أما n تأخذ إحدى القيمتين 8 , 4 وهي تعبر أن أصغر Object مؤلف من تجمع 4 أو 8 بكسلات متجمعة .

مثال :

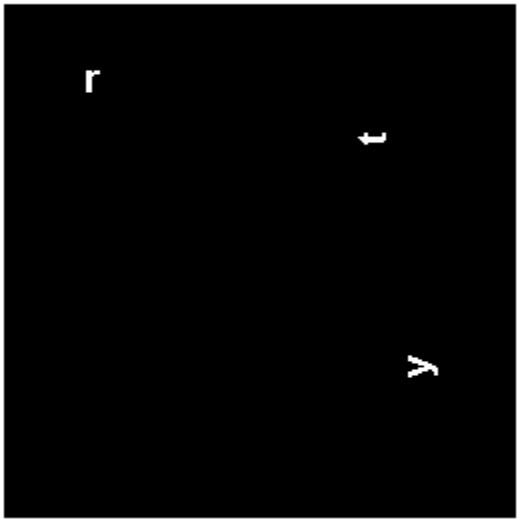
```
bw1 = imread('text.png');

c = [43 185 212];

r = [38 68 181];

bw2=bwselect(bw1,c,r,4);

imshow(bw1)

figure, imshow(bw2)
```

والنتائج كما يلي :

Original Image



Selective Image



٢ـ مساحة جميع Objects في الصورة الثنائية bwarea:

يمكن الحصول على مساحة جميع Objects في الصورة الثنائية Binary image باستخدام التعليمة bwarea وفق الشكل التالي :

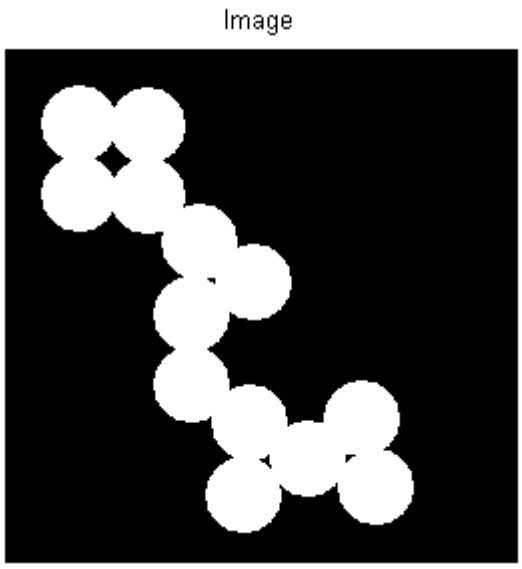
```
Total_Area=bwarea(bw)
```

مثال :

```
bw= imread('circles.png');
imshow(bw);
Total_Area=bwarea(bw)
```

والنتائج كما يلي :

```
Total_Area =
  1.4187e+004
```



٣- حذف Objects من الصورة الثنائية ذات مساحة أقل من N بكسل :

يمكن حذف Objects من الصورة الثنائية ذات مساحة ثنائية أقل من N بكسل والصورة الجديدة تحوي جميع Objects ذات المساحة أكبر من N بكسل باستخدام التعليمة bwareaopen وفق الشكل :

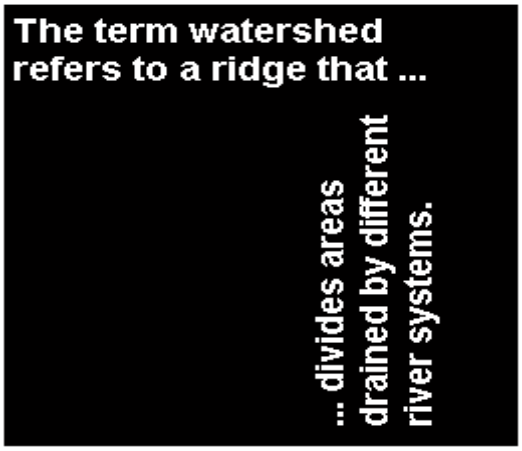
bw2=bwareaopen(bw1,N)

حيث N المساحة التي تحتها يحذف الـ Object من الصورة .

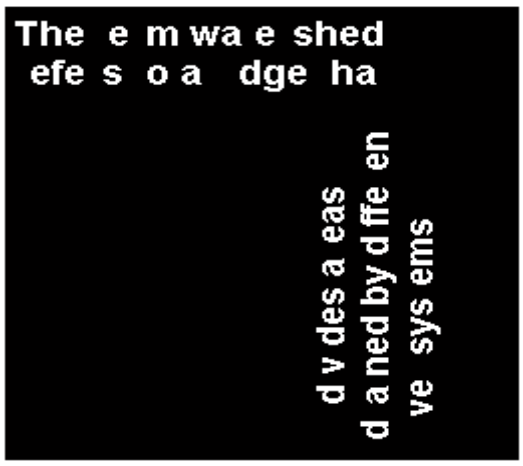
```
bw1 = imread('text.png');

imshow(bw1)

bw2 =bwareaopen(bw1,50);

figure , imshow(bw2)
```

والنتائج كما يلي :

Original Image



Omitted Image



٤ـ تحديد المسافة بين كل بكسل والبكسل الأقرب غير المساوي للصفر :

يمكن الحصول على مصفوفة المسافة للمصفوفة الثنائية Binary Image باستخدام التعليمة bwdist وفق الشكل :

[D,L]=bwdist (bw ,method)

حيث D مصفوفة المسافة هي مصفوفة كل عنصر فيها هو المسافة بين العنصر المقابل و اقرب عنصر يساوي الواحد له في المصفوفة الثنائية .

أما L مصفوفة تحوي الدليل الخطي للعنصر الأقرب المساوي للواحد من الصورة الثنائية .

تعطى المسافة بحسب طريقة الحساب :

chessboard : Distance = max(|x1-x2| , |y1-y2|) ;

cityblock : Distance = |x1-x2| + |y1-y2| ;

euclidean : Distance = sqrt((x1-x2).^2+(y1-y2).^2) ;

quasi-euclidean :

Distance = |x1-x2| + (sqrt(2) -1) |y1-y2| ; if |x1-x2| > |y1-y2|

Distance = (sqrt(2) -1) |x1-x2| + |y1-y2| ; otherwise

افتراضياً تحسب المسافة وفق طريقة Euclidean .

مثال :

```
bw = zeros(5,5); bw(2,2) = 1; bw(4,4) = 1;
bw

[D,L] = bwdist(bw);

D

L
```

والنتائج كما يلي :

bw =

0	0	0	0	0
0	1	0	0	0
0	0	0	0	0
0	0	0	1	0
0	0	0	0	0

D =

1.4142	1.0000	1.4142	2.2361	3.1623
1.0000	0	1.0000	2.0000	2.2361
1.4142	1.0000	1.4142	1.0000	1.4142
2.2361	2.0000	1.0000	0	1.0000
3.1623	2.2361	1.4142	1.0000	1.4142

L =

7	7	7	7	7
7	7	7	7	19
7	7	7	19	19
7	7	19	19	19
7	19	19	19	19

٥- إيجاد حواف Objects في صورة ثنائية bwperim :

يمكن إيجاد حواف Objects في الصورة الثنائية باستخدام التعليمة bwperim وفق الشكل :

```
bw2=bwperim(bw1,conn)
```

حيث bw1 الصورة الأصلية أما conn هو 4 أو 8 وهو أقل مساحة للـ Object .

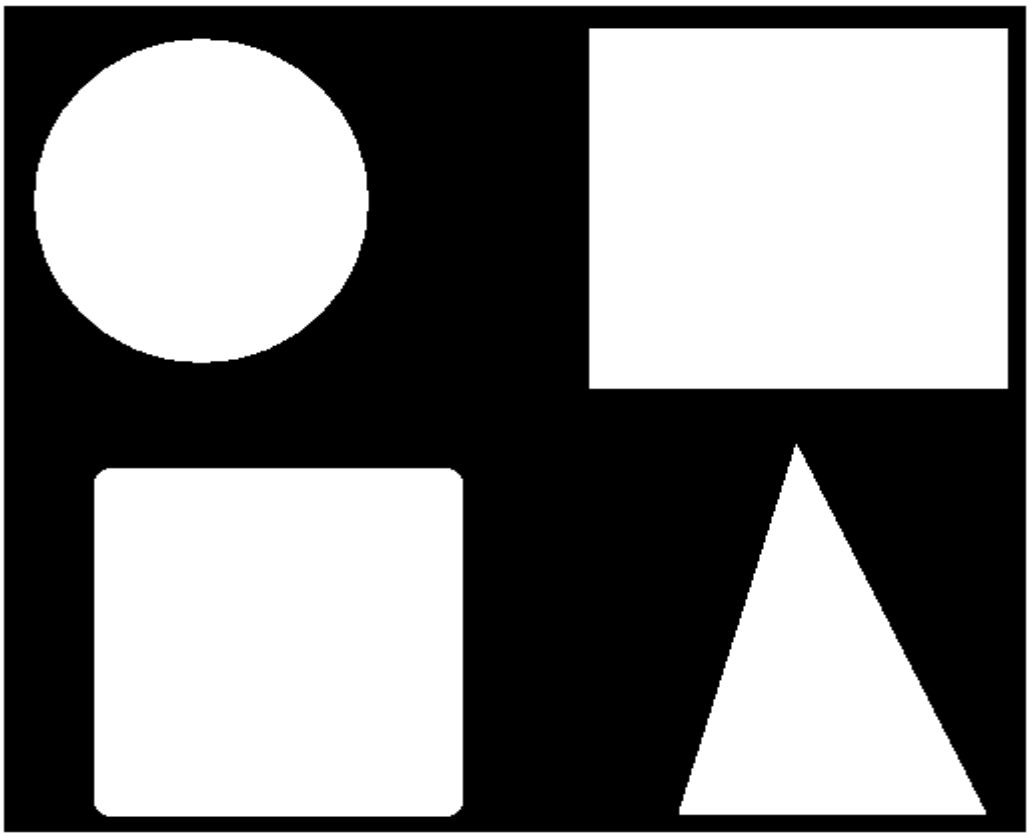
و bw2 صورة ثنائية تظهر حواف كل Object وكل Hole في الصورة الأصلية .

مثال :

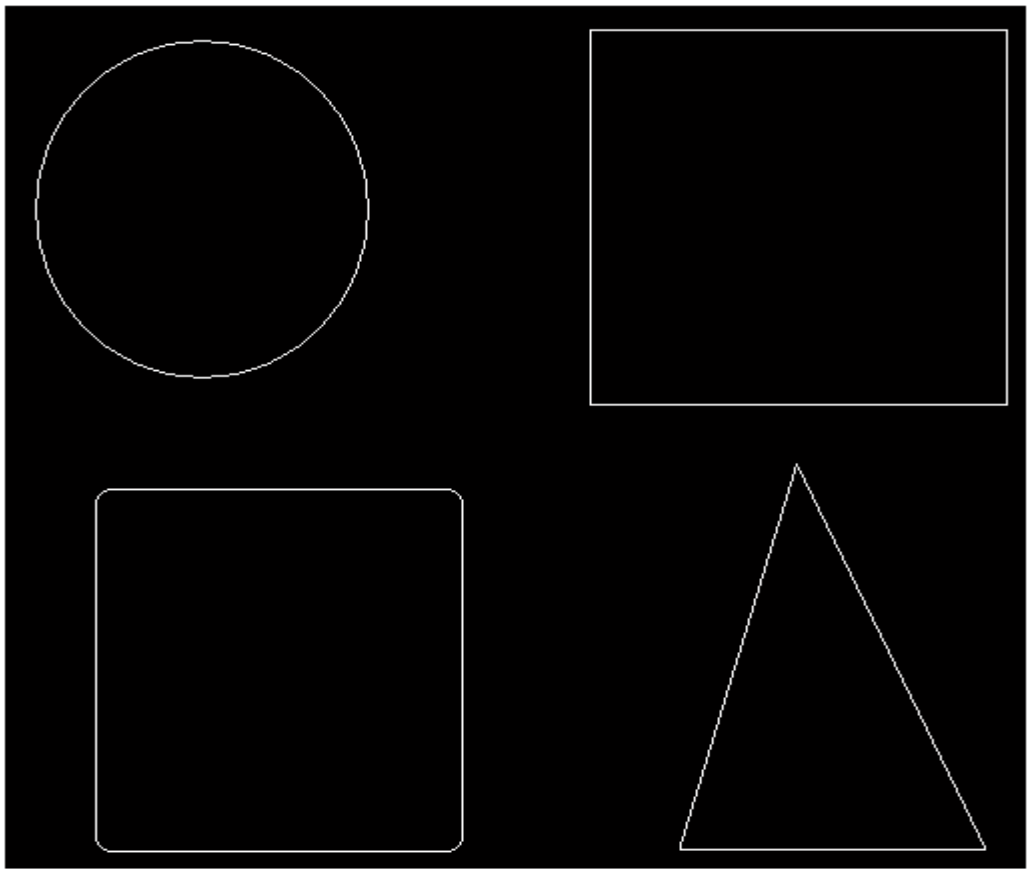
```
I1=imread('D:\Shapes.jpg');
level=graythresh(I1)
bw1=im2bw(I1);
imshow(bw1)
bw2=bwperim(bw1,4);
figure , imshow(bw2)
```

والنتائج كما يلي :

Original Image



Perimed Image



٦- رقم Euler للصورة الثنائية bweuler :

للحصول على رقم Euler للصورة كاملة نستخدم التعليمة bweuler :

رقم Euler للصورة = عدد Objects في الصورة – عدد الثقوب في الصورة

ومنه

عدد الثقوب في الصورة = عدد Objects في الصورة – رقم Euler للصورة

```
I1=imread('D:\Shapes1.jpg');
level=graythresh(I1)
bw1=im2bw(I1,level);
[labeled,numObjects]=bwlabel(bw1,4);
info=regionprops(labeled,'all');
Euler_Number=bweuler(bw1,4);
NumHoles=numObjects-Euler_Number
```

والنتائج كما يلي :

```
level =
   0.4843
NumHoles =
    8
```

مشروح سابقاً في درس شرح خواص الصورة الثنائية Regionprops .

٧- ترقيم Objects والحصول على صورة مؤشرة bwlabel :

مشروح سابقاً .

العمليات الحسابية على الصور الرقمية

١- جمع (دمج) صورتين رقميتين imadd :

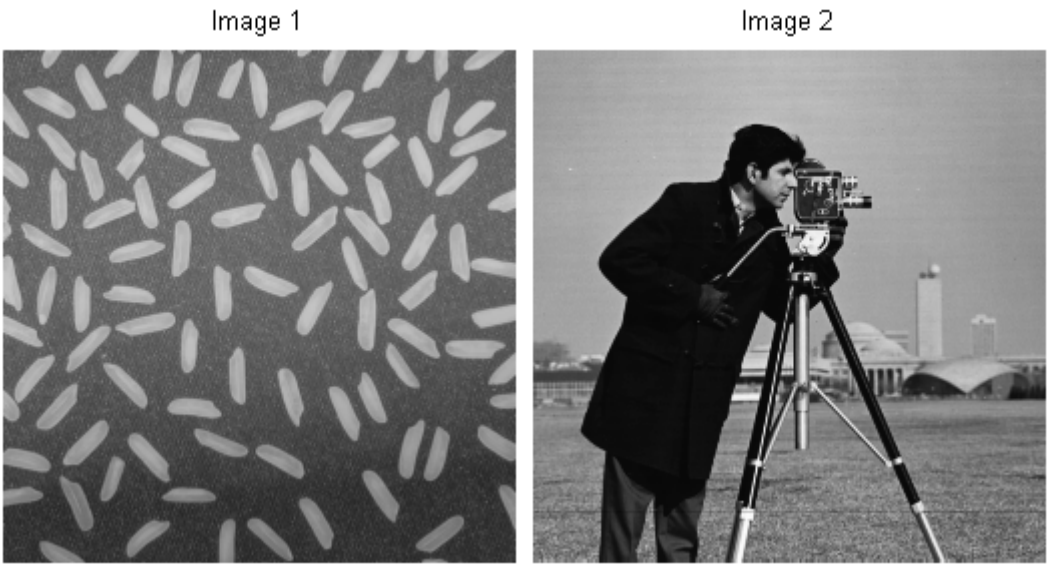
يمكن جمع صورتين أي دمجهما في صورة واحدها من خلال التعليمة imadd على الشكل :

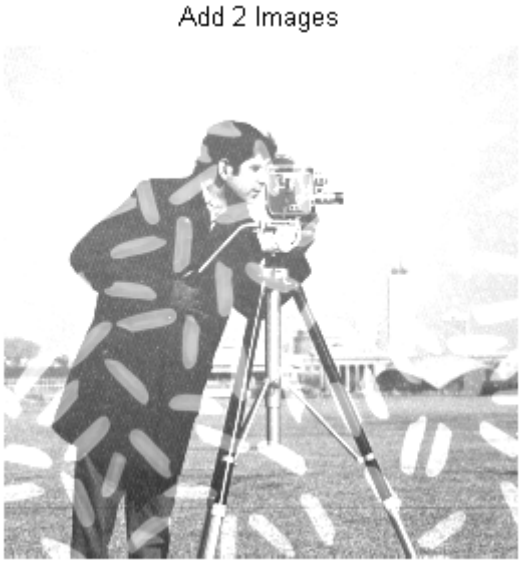
```
K=imadd(I,J) ;
```

مثال :

```
I = imread('rice.png');
imshow(I)
J = imread('cameraman.tif');
figure , imshow(J)
K=imadd(I,J);
figure, imshow(K)
```

والنتائج كما يلي :

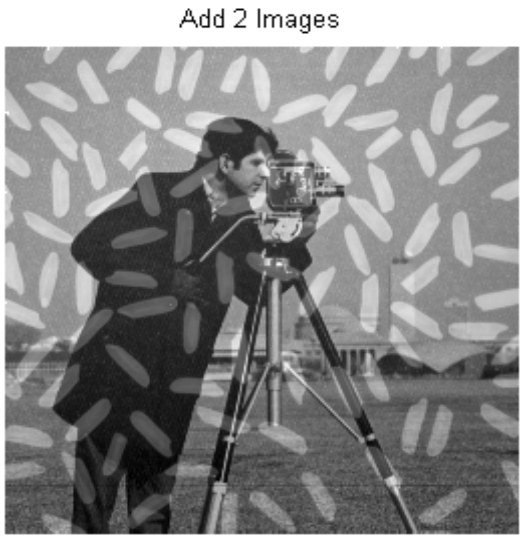




نحن نعلم أن الألوان قيمها تتراوح بين 0 (اللون الأسود) و 255 (اللون الأبيض) لكن هناك مشكلة عند جمع قيمتين ويكون الناتج يزيد عن 255 فإنه يقص إلى 255 لأن نوع الصورة uint8 مما يؤدي إلى ضياع بعض الألوان حيث تذهب إلى اللون الأبيض لذلك يفضل توسيع مجال الألوان إلى [0 512] أي التحويل إلى نوع uint16 .

مثال :

```
I = imread('rice.png');

J = imread('cameraman.tif');

K = imadd(I,J,'uint16');

imshow(K,[])
```

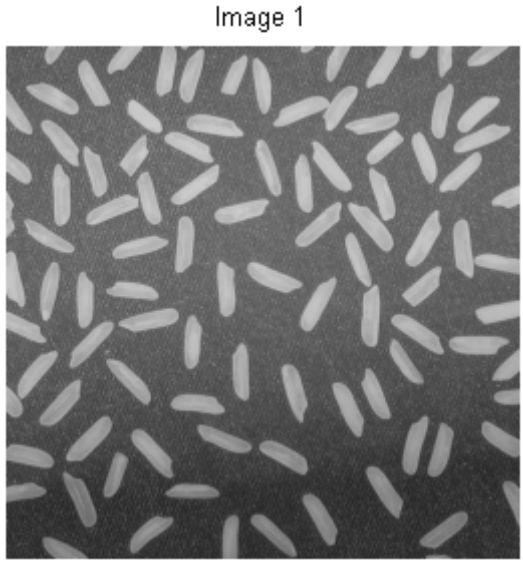


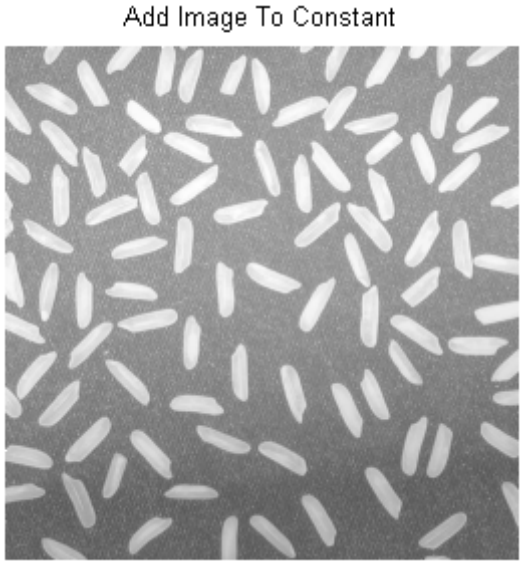
أو يمكن إضافة عدد ثابت إلى الصورة (تفتيح الصورة) .

مثال :

```
I = imread('rice.png');
imshow(I)
J = imadd(I,50);
figure , imshow(J)
```

والنتائج كما يلي :





٢- طرح صورة رقمية من أخرى imsubtract :

يمكن طرح صورة رقمية من أخرى من خلال التعليمة imsubtract على الشكل التالي :

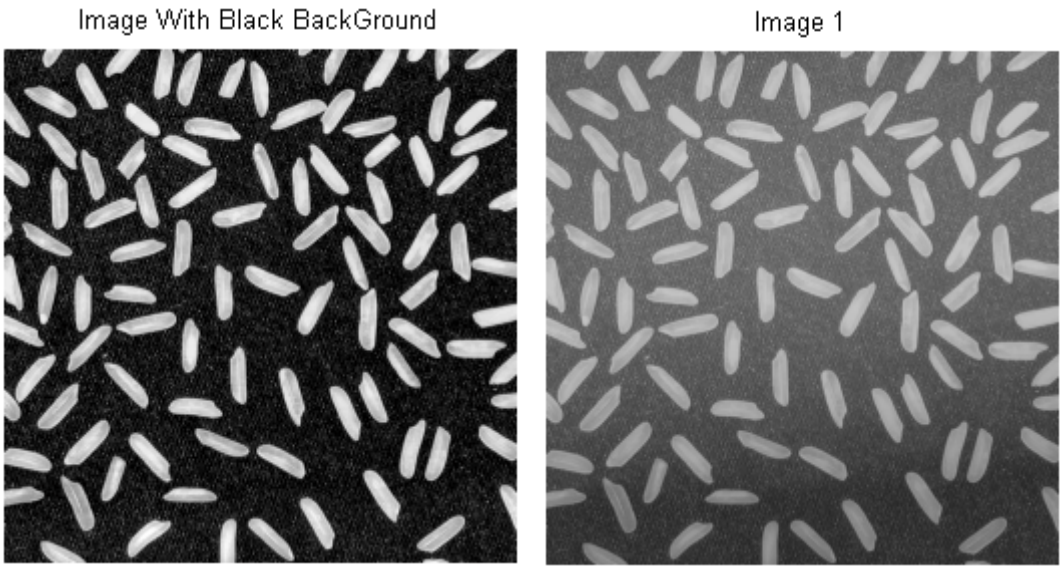
```
K=imsubtract(I,J) ;
```

غالباً من غير المفيد طرح صورة رقمية من أخرى لكن من الممكن طرح صورة رقمية من خلفية نفس الصورة لجعل الخلفية سوداء تماماً كما ذكرنا سابقاً

مثال :

```
I = imread('rice.png');

background = imopen(I,strel('disk',15));

Ip = imsubtract(I,background);

imshow(Ip,[])
```

والنتائج كما يلي :

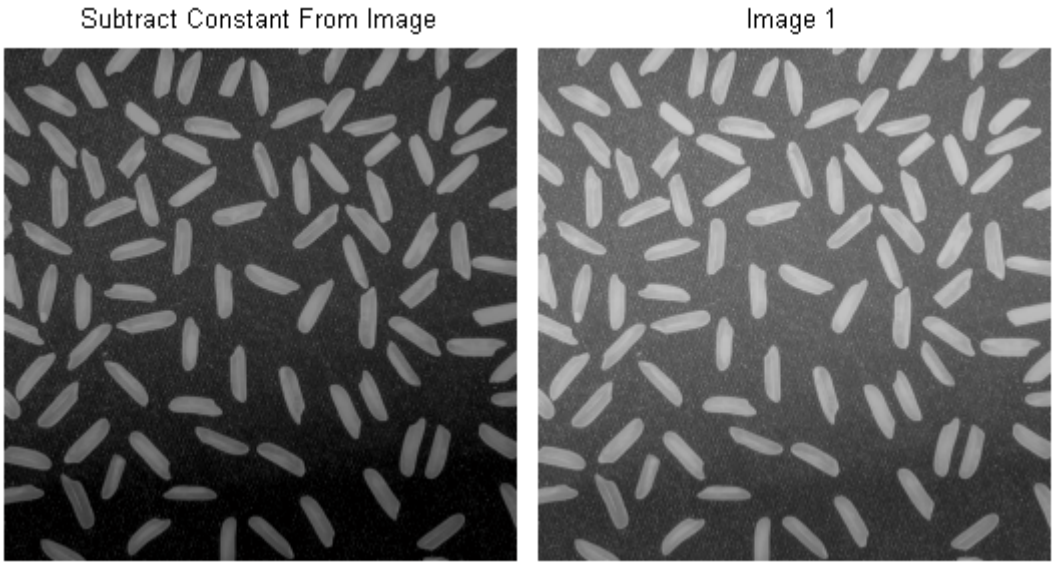


أو يمكن طرح عدد ثابت من الصورة (تعتيم الصورة) .

مثال :

```
I = imread('rice.png');
Iq = imsubtract(I,50);
figure, imshow(I), figure, imshow(Iq)
```

والنتائج كما يلي :



٣- ضرب صورتين رقميتين ببعضهما immultiply :

يمكن ضرب صورة رقمية بأخرى من خلال التعليمة immultiply على الشكل التالي :

```
K= immultiply (I,J) ;
```

حيث يمكن ضرب صورة رقمية بعدد ثابت .

مثال :

```
I = imread('moon.tif');

imshow(I)

J = immultiply(I,1.2);

figure , imshow(J)
```

والنتائج كما يلي :

٤ـ قسمة صورتين رقميتين من بعضهما imdivide :

يمكن قسمة صورة رقمية من أخرى من خلال التعليمة imdivide على الشكل التالي :

K= imdivide (I,J) ;

حيث يمكن قسمة صورة رقمية على عدد ثابت .

مثال :

```
I = imread('moon.tif');

imshow(I)

J = imdivide(I,1.5);

figure , imshow(J)
```

والنتائج كما يلي :

٥- متمم صورة رقمية أو ما يسمى Negative :

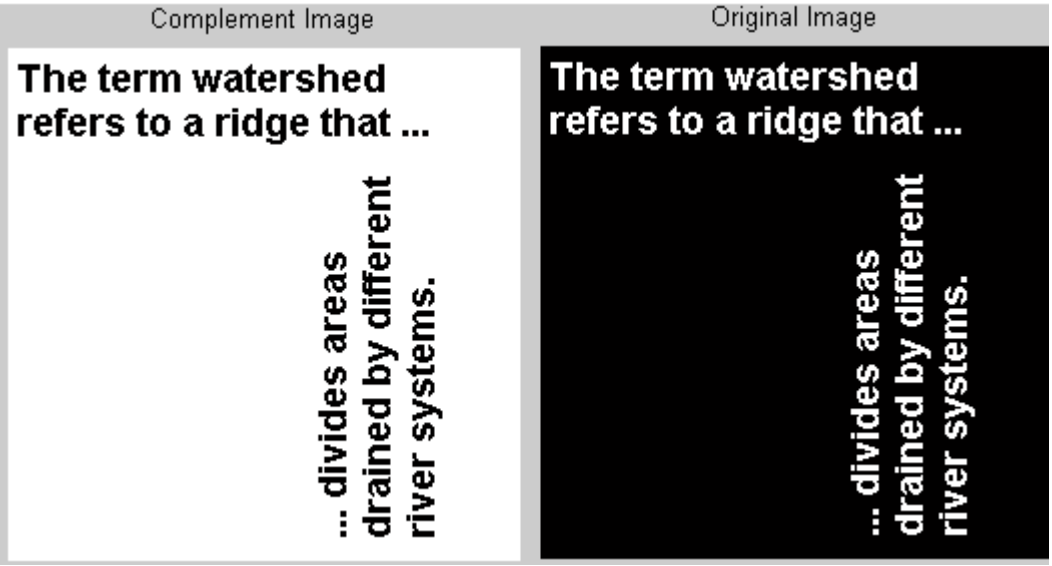
يمكن الحصول على Negative من الصورة باستخدام التعليمة imcomplement على الشكل التالي :

```
J= imcomplement(I);
```

مثال :

```
bw = imread('text.png');
imshow(bw)
bw2 = imcomplement(bw);
figure , imshow(bw2)
```

والنتائج كما يلي :



٦- الجمع الخطي للصور الرقمية imlincomb :

يمكن القيام بالجمع الخطي لعدة صور رقمية بعد تحديد عامل الضرب الخاص بكل صورة من خلال التعليمة imlincomb على الشكل التالي :

Z = imlincomb(K1,A1,K2,A2,...,Kn,An)

حيث K1,K2,…., Kn عوامل الضرب

و A1,A2,….,An الصور المراد جمعها

والصورة الناتجة هي :

Z=K1*A1+K2*A2+…….+Kn*An

والصورة ذات عامل الضرب الأكبر هي الصورة الأكثر أهمية .

مثال :

```
I = imread('rice.png');
J = imread('cameraman.tif');
K=imlincomb(0.2,I,0.8,J);
imshow(K)
```

أي الصورة الجديدة تحوي 20% من I و 80% من J .

والنتائج كما يلي :