Image Processing with MATLAB

Lecture 3: Working with Images in MATLAB

Dr.Eng. Hassan Mohamed

Hassan.hussein@feng.bu.edu.eg

Lecture Contents:

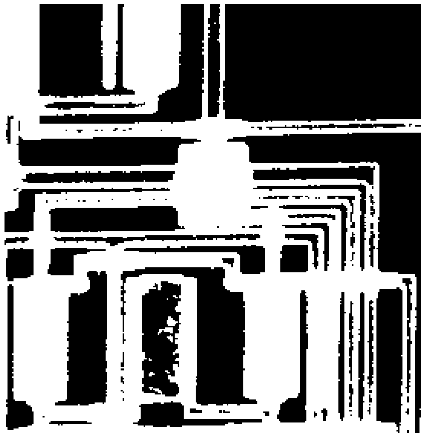
1. Image Types in the Toolbox
2. Converting Between Image Types
3. Converting Between Image Classes
4. Image Resizing
5. Image Rotation
6. Image Cropping
7. Image Arithmetic

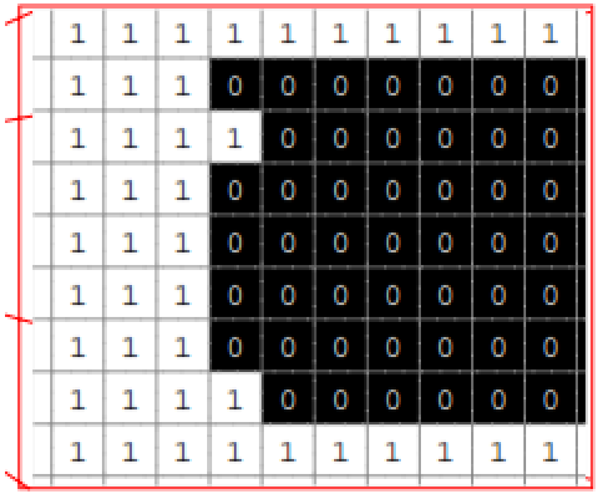
Image Types in the Toolbox:

Image Type	Interpretation
Binary	`Logical` array containing only 0's and 1's, interpreted as black and white, respectively. Also known as a *bilevel* image.
Indexed	Array of class `logical`, `uint8`, `uint16`, `single`, or `double` whose pixel values are direct indices into a colormap. The colormap is an *m*-by-3 array of class `double`. Also known as a *pseudocolor* image. **Note**: For `single` or `double` arrays, integer values range from [1, *p*]. For `logical`, `uint8`, or `uint16` arrays, values range from [0, *p*-1]
Intensity	Array of class `uint8`, `uint16`, `int16`, `single`, or `double` whose pixel values specify intensity values. Also known as a *grayscale* image **Note**: For `single` or `double` arrays, values range from [0, 1]. For `uint8`, values range from [0,255]. For `uint16`, values range from [0, 65535]. For `int16`, values range from [-32768, 32767].
Truecolor	*m*-by-*n*-by-3 array of class `uint8`, `uint16`, `single`, or `double` whose pixel values specify intensity values. Also known as an *RGB* image. **Note**: For `single` or `double` arrays, values range from [0, 1]. For `uint8`, values range from [0, 255]. For uint16, values range from [0, 65535].

Binary Images:

- In a binary image, also known as a bi-level image, each pixel assumes one of only two discrete values: 1 or 0. A binary image is stored as a logical array.





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

Indexed Images:

- An indexed image consists of an array, called X in this documentation, and a colormap matrix, called map. The pixel values in the array are direct indices into a colormap. The colormap matrix is an m-by-3 array of class double containing floating-point values in the range [0,1]. Each row of map specifies the red, green, and blue components of a single color. An indexed image uses direct mapping of pixel values to colormap values. The color of each image pixel is determined by using the corresponding value of X as an index into map.

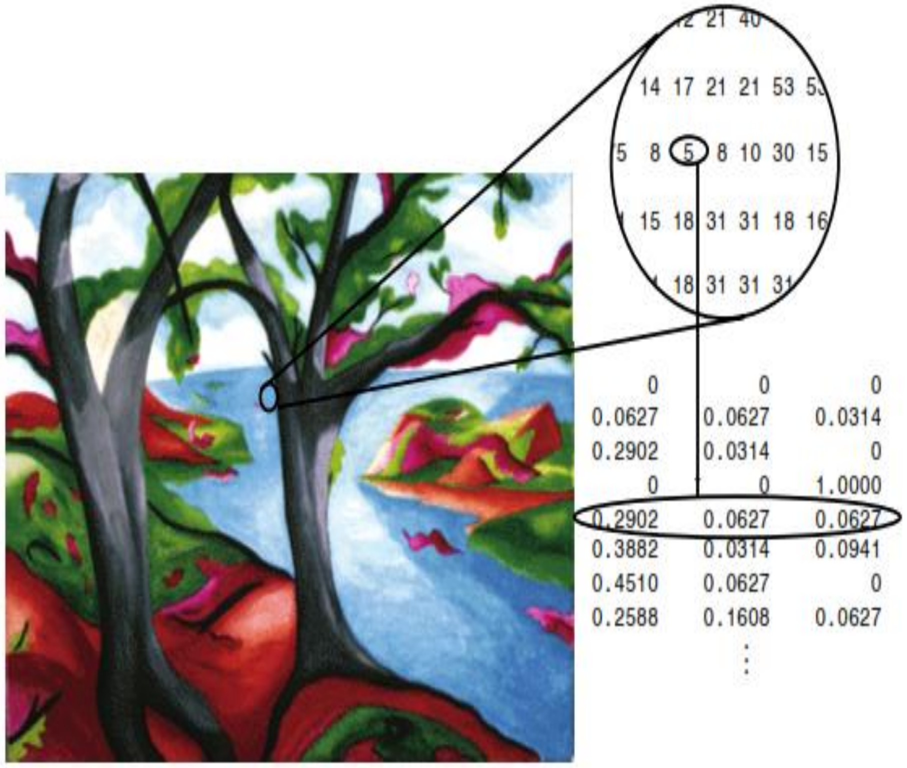
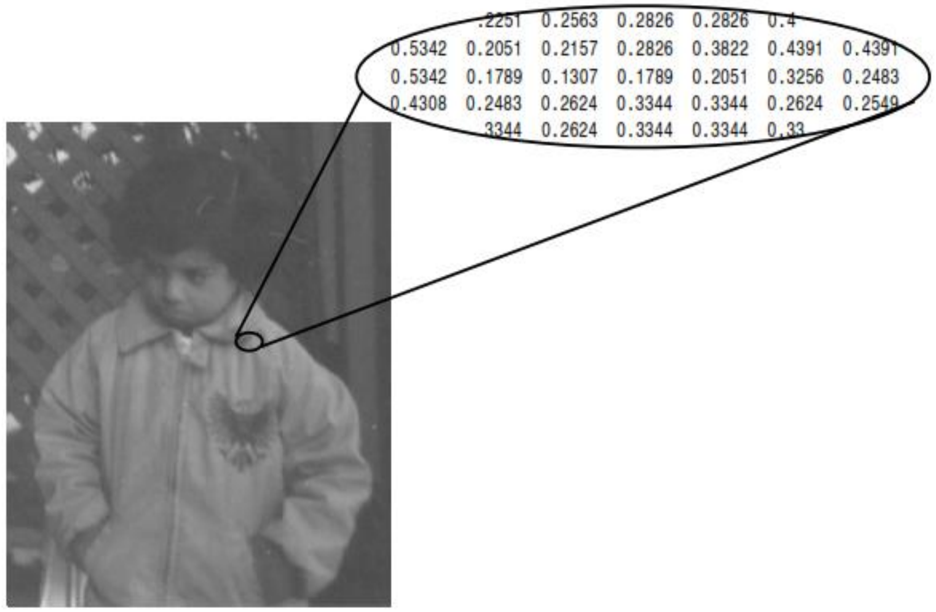
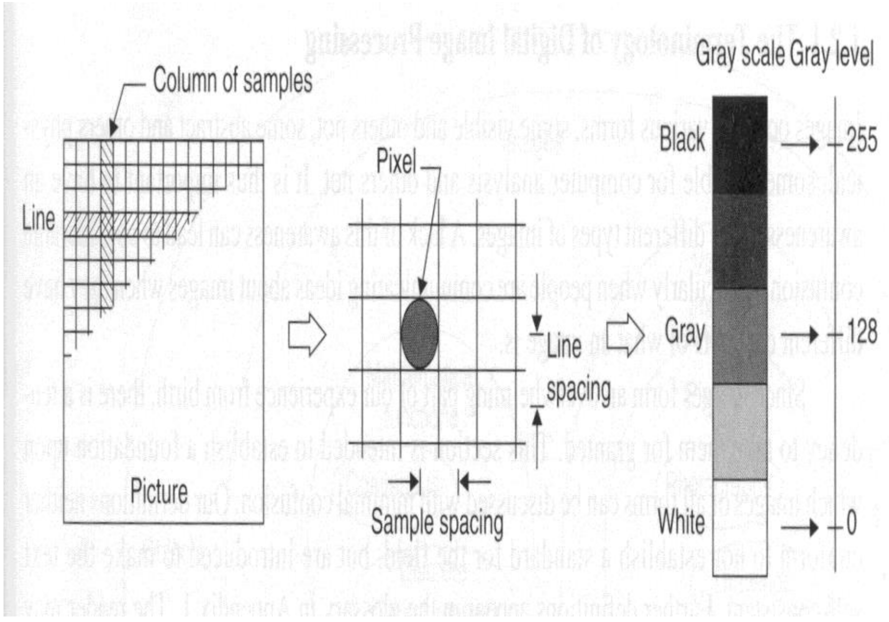


Image Courtesy of Susan Cohen

Intensity Images:

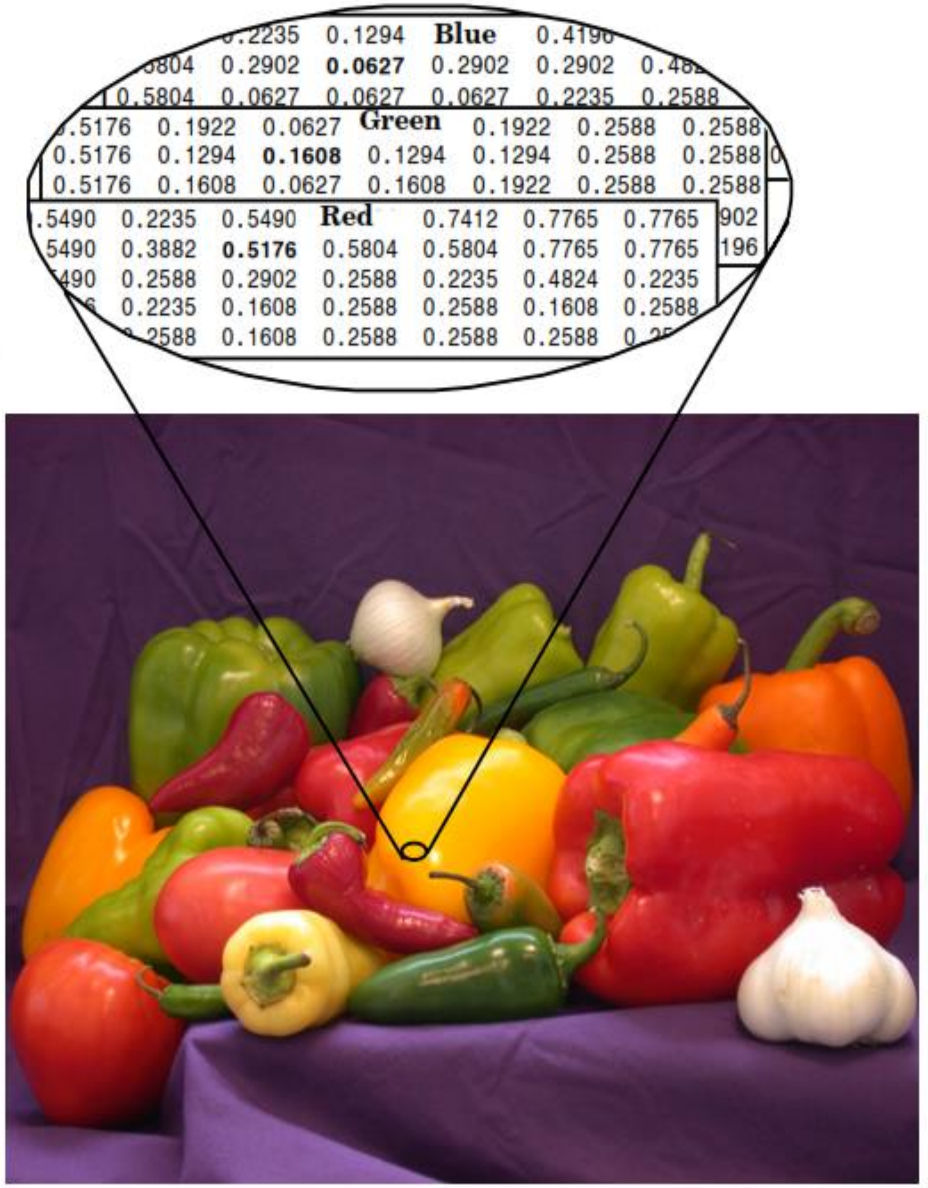
- An intensity image, also known as a grayscale image, is a data matrix, I, whose values represent intensities within some range. MATLAB stores an intensity image as a individual matrix, with each element of the matrix corresponding to one image pixel. The matrix can be of class uint8, uint16, int16, single, or double. While intensity images are rarely saved with a colormap, MATLAB uses a colormap to display them.





Truecolor Images:

- A truecolor image, also known as an RGB image, is stored in MATLAB as an m-by-n-by-3 data array that defines red, green, and blue color components for each individual pixel. Truecolor images do not use a colormap. The color of each pixel is determined by the combination of the red, green, and blue intensities stored in each color plane at the pixel's location.

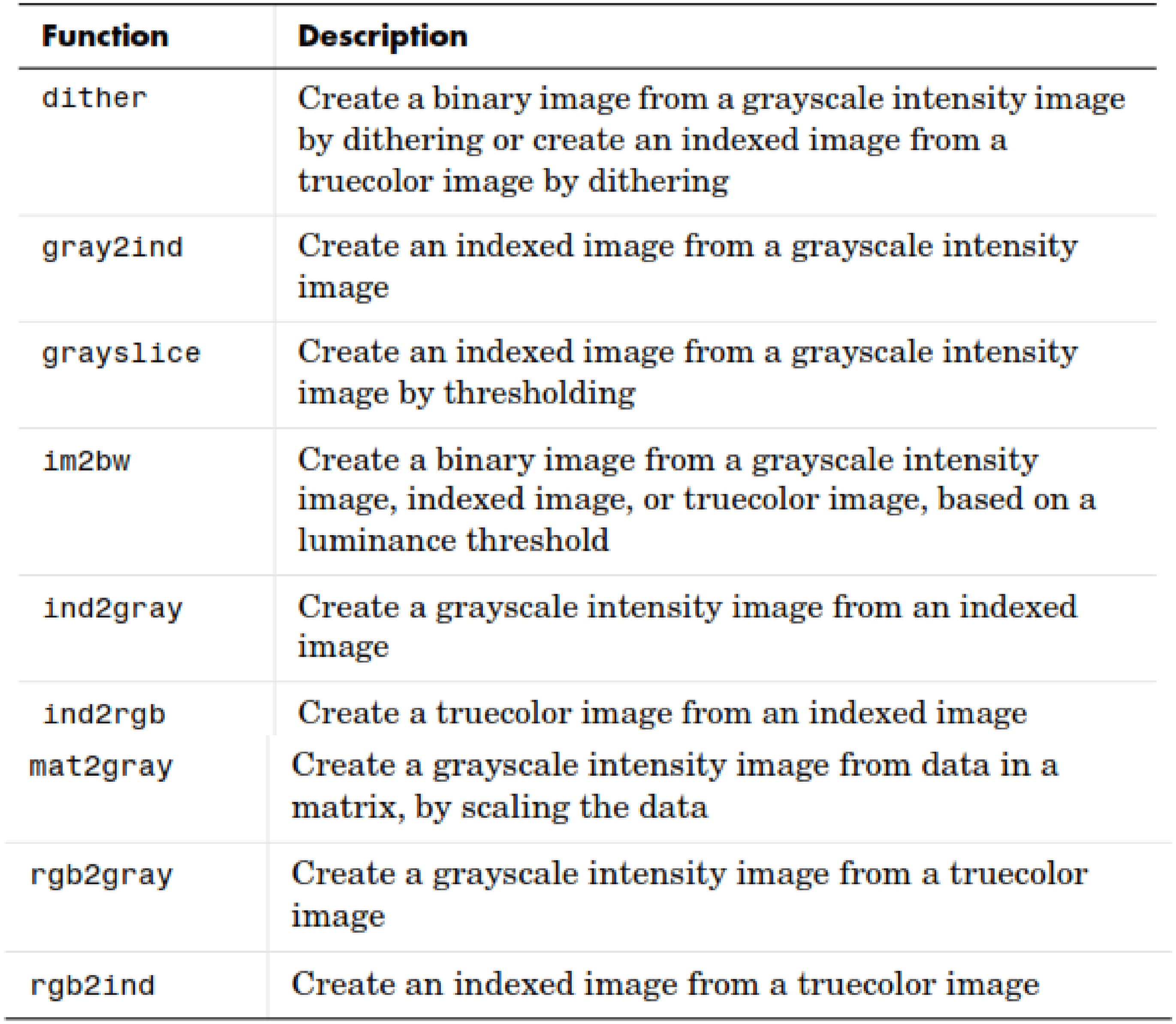


```
.2235   0.1294   Blue   0.4196
3804   0.2902   0.0627   0.2902   0.2902   0.482
0.5804   0.0627   0.0627   0.0627   0.2235   0.2588
.5176   0.1922   0.0627   Green   0.1922   0.2588   0.2588
0.5176   0.1294   0.1608   0.1294   0.1294   0.2588   0.2588
0.5176   0.1608   0.0627   0.1608   0.1922   0.2588   0.2588
.5490   0.2235   0.5490   Red   0.7412   0.7765   0.7765   902
5490   0.3882   0.5176   0.5804   0.5804   0.7765   0.7765   196
490   0.2588   0.2902   0.2588   0.2235   0.4824   0.2235
0.2235   0.1608   0.2588   0.2588   0.1608   0.2588
2588   0.1608   0.2588   0.2588   0.2588
```

Converting Between Image Types:

- You might need to convert an image from one type to another. For example, if you want to filter a color image that is stored as an indexed image, you should first convert it to truecolor format. When you apply the filter to the truecolor image, MATLAB filters the intensity values in the image, as is appropriate. If you attempt to filter the indexed image, MATLAB simply applies the filter to the indices in the indexed image matrix, and the results might not be meaningful.

Converting Between Image Types:



Function	Description
`dither`	Create a binary image from a grayscale intensity image by dithering or create an indexed image from a truecolor image by dithering
`gray2ind`	Create an indexed image from a grayscale intensity image
`grayslice`	Create an indexed image from a grayscale intensity image by thresholding
`im2bw`	Create a binary image from a grayscale intensity image, indexed image, or truecolor image, based on a luminance threshold
`ind2gray`	Create a grayscale intensity image from an indexed image
`ind2rgb`	Create a truecolor image from an indexed image
`mat2gray`	Create a grayscale intensity image from data in a matrix, by scaling the data
`rgb2gray`	Create a grayscale intensity image from a truecolor image
`rgb2ind`	Create an indexed image from a truecolor image

Converting Between Image Classes:

- You can convert uint8 and uint16 image data to double using the MATLAB double function. However, converting between classes changes the way MATLAB and the toolbox interpret the image data. If you want the resulting array to be interpreted properly as image data, you need to rescale or offset the data when you convert it. For easier conversion of classes, use one of these toolbox functions: im2uint8, im2uint16, im2int16, im2single,orim2double. These functions automatically handle the rescaling and offsetting of the original data of any image class. For example, this command converts a double-precision RGB image with data in the range [0,1] to a uint8 RGB image with data in the range [0,255].
- **RGB2 = im2uint8(RGB1);**

Converting Between Image Classes:

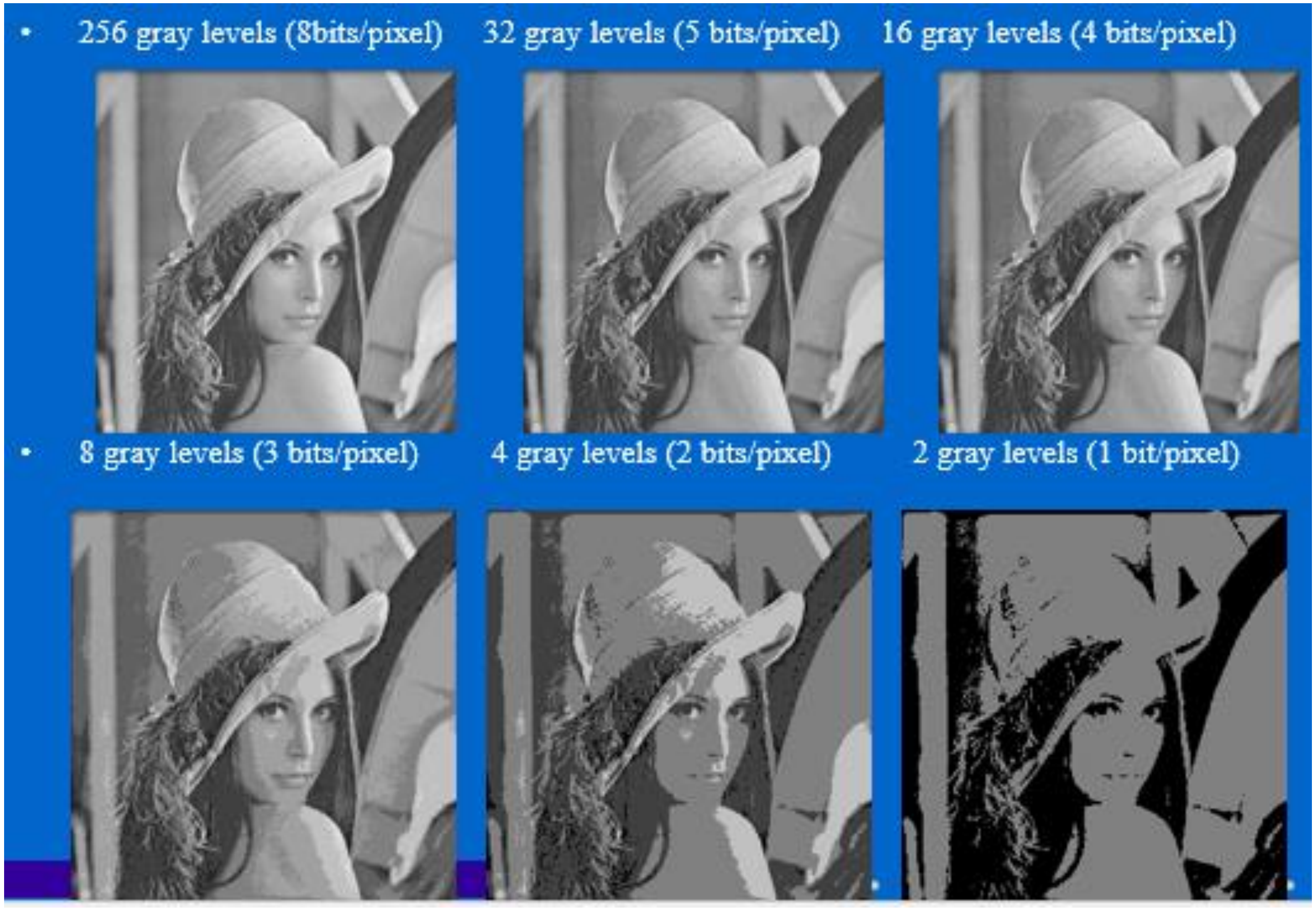

256 gray levels (8bits/pixel)
32 gray levels (5 bits/pixel)
16 gray levels (4 bits/pixel)
8 gray levels (3 bits/pixel)
4 gray levels (2 bits/pixel)
2 gray levels (1 bit/pixel)

Image Resizing:

- To change the size of an image, use the *imresize* function. Using *imresize*, you can:
- Specify the size of the output image
- Specify the interpolation method used
- **Using the Magnification Factor**
- I = imread('circuit.tif');
- J = imresize(I,1.25);
- imshow(I)
- figure, imshow(J)

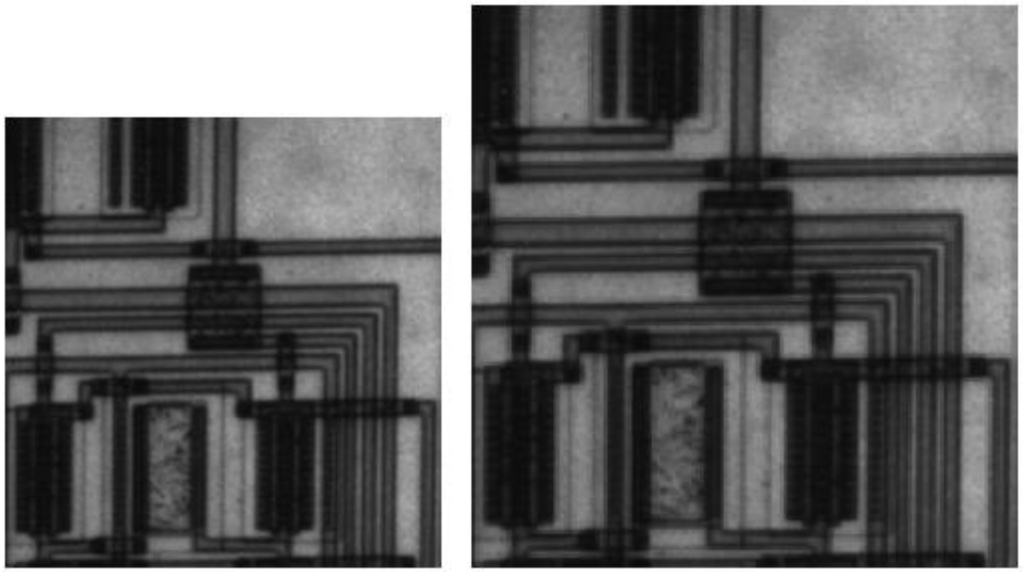


Image Courtesy of Steve Decker and Shujaat Nadeem

Image Resizing:

- **Specifying the Size of the Output Image**
- I = imread('circuit.tif');
- Y = imresize(X,[100 150])
- imshow(Y)
- figure, imshow(I)

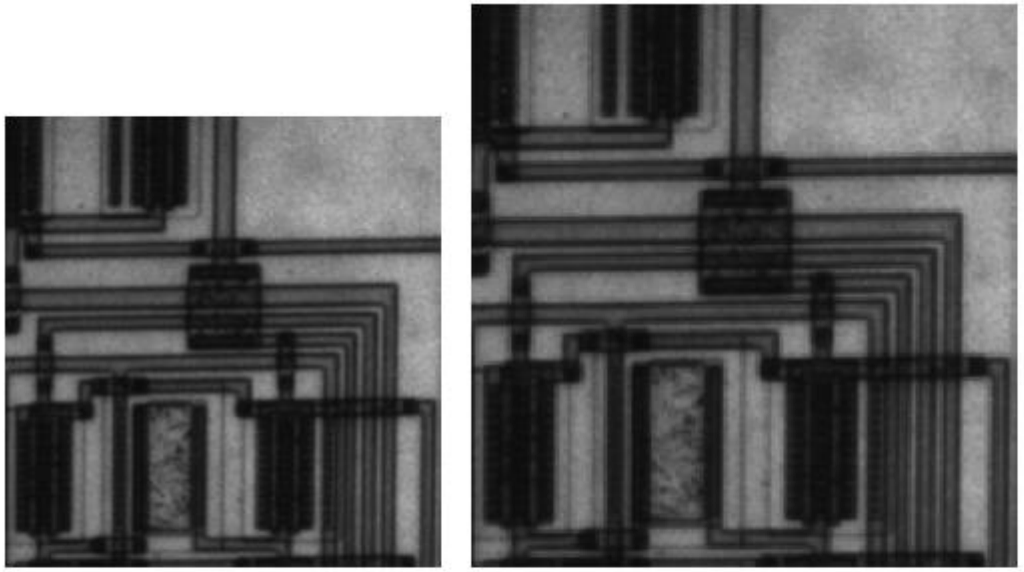
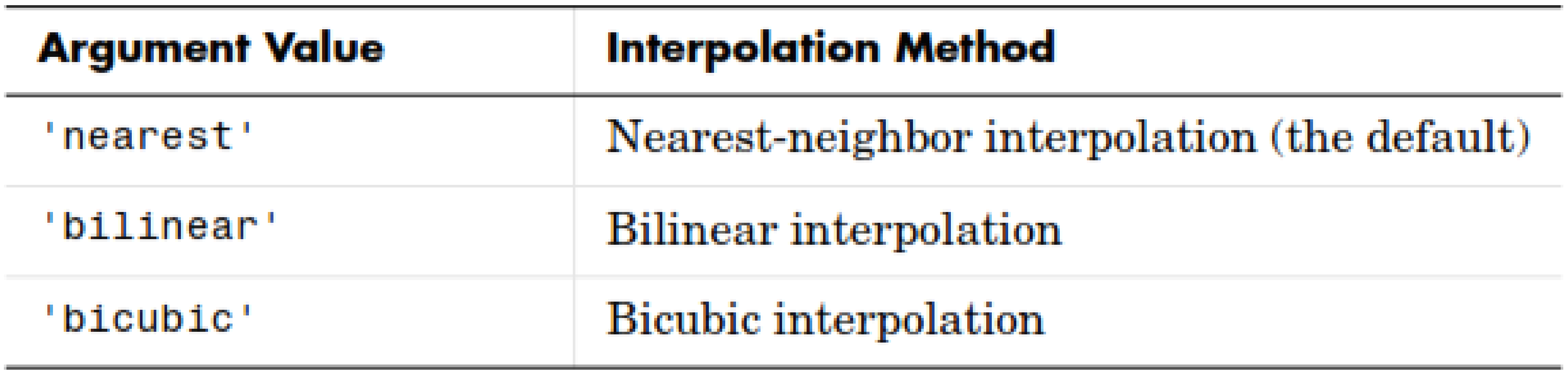


Image Courtesy of Steve Decker and Shujaat Nadeem

Image Resizing:

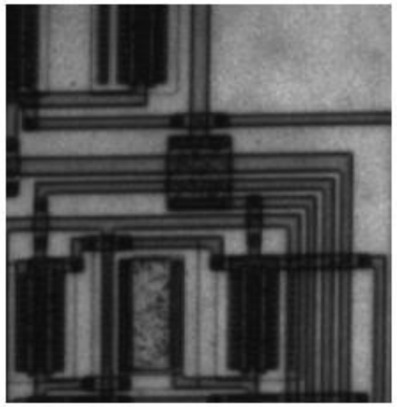
- **Specifying the Interpolation Method**
- I = imread('circuit.tif');
- Y = imresize(X,[100 150],'bilinear')
- imshow(Y)
- figure, imshow(I)



Argument Value	Interpolation Method
`'nearest'`	Nearest-neighbor interpolation (the default)
`'bilinear'`	Bilinear interpolation
`'bicubic'`	Bicubic interpolation

Image Rotation:

- To rotate an image, use the imrotate function. imrotate accepts two primary arguments:
- The image to be rotated
- The rotation angle
- **I = imread('circuit.tif');**
- **J = imrotate(I,35,'bilinear');**
- **imshow(I)**
- **figure, imshow(J)**



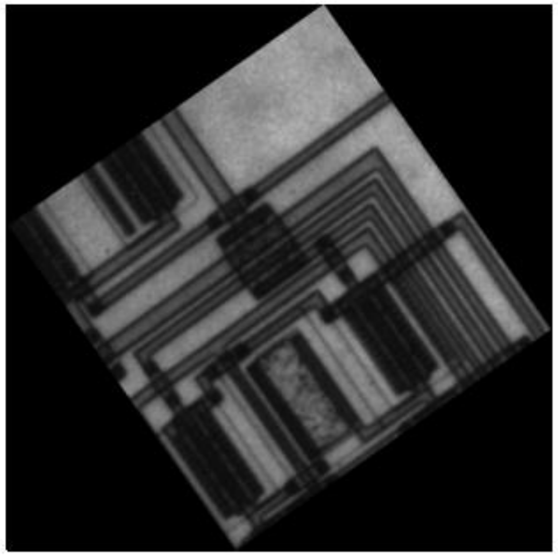


Image Cropping:

- To extract a rectangular portion of an image, use the imcrop function. *Imcrop* accepts two primary arguments:
- The image to be cropped
- The coordinates of a rectangle that defines the crop area
- **imshow circuit.tif**
- **I = imcrop;**
- **imshow(I);**

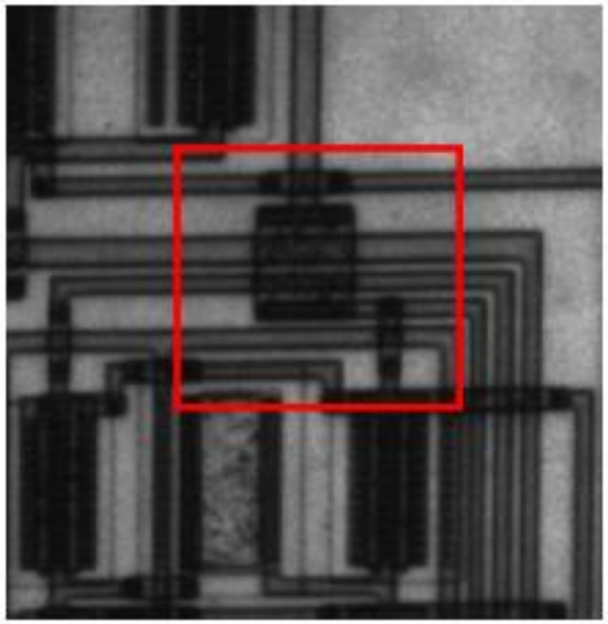


Image Arithmetic:

- Image arithmetic is the implementation of standard arithmetic operations, such as addition, subtraction, multiplication, and division, on images. Image arithmetic has many uses in image processing both as a preliminary step in more complex operations and by itself. For example, image subtraction can be used to detect differences between two or more images of the same scene or object.

Image Arithmetic :

- a=imread ('2ndyear.jpg');
- a=imread ('3rdyear.jpg');
- b=imread ('3rdyear2.jpg');
- c=b(200:967,1:1199,:);
- imshow(c)
- d=[a b];
- imshow(d)
- e=[a; b];
- imshow(e)
- f=imread ('3rdyear3.jpg');
- imshow(f)
- g = imsubtract(a,f);
- imshow(g)
- h = imcomplement(a);
- imshow(h)
- i = imlincomb(1,a,1,f);
- imshow(i)
- j=fliplr(a);
- imshow(j)
- k=flipud(a);
- imshow(k)
- X=a(:,:,1);
- imshow(X)

Supplementary files:

- MATLAB Tutorial:

http://www.mathworks.com/products/matlab/matlab_tutorial.html

- MATLAB documentation:

http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml

Dr.Eng. Hassan Mohamed