**imread()**

**Purpose:**  
Reads an image file from disk into MATLAB workspace as an array.

**Syntax:**

I = imread('filename');

**Example:**

I = imread('flowers.jpg'); % Reads the image

**Explanation:** I will now be a matrix (MxNx3 for color images).

**imshow()**

**Purpose:**  
Displays an image in a MATLAB figure window.

**Syntax:**

imshow(I);

**Example:**

I = imread('flowers.jpg');

imshow(I); % Shows the image

**Explanation:** Opens a figure window and displays the image.

**imwrite()**

**Purpose:**  
Writes (saves) an image matrix to a file.

**Syntax:**

imwrite(I, 'filename');

**Example:**

I = imread('flowers.jpg');

imwrite(I, 'copy\_flowers.png'); % Saves the image as PNG [\\will](file:///\\will) create one copy of the image.

**rgb2gray()**

**Purpose:**  
Converts a color (RGB) image to grayscale.

**Syntax:**

grayImage = rgb2gray(RGBimage);

**Example:**

I = imread('flowers.jpg');

grayI = rgb2gray(I);

imshow(grayI);

**Explanation:** Displays a grayscale version of the original image.

**imhist()**

**Purpose:**  
Displays the histogram of an image (pixel intensity distribution).

**Syntax:**

imhist(I);

**Example:**

I = imread('flowers.jpg');

grayI = rgb2gray(I);

imhist(grayI); % Shows histogram of grayscale intensities

i**madjust()**

**Purpose:**  
Adjusts image intensity values or contrast.

**Syntax:**

J = imadjust(I);

**Example:**

I = imread('flowers.jpg');

grayI = rgb2gray(I);

J = imadjust(grayI); % Enhance contrast

imshow(J);

**im2bw()**

*(deprecated — replaced by imbinarize in newer MATLAB)*

**Purpose:**  
Converts image to binary (black & white) using a threshold.

**Syntax:**

BW = im2bw(I, level);

level = threshold between 0 and 1.

**Example:**

I = imread('flowers.jpg');

grayI = rgb2gray(I);

BW = im2bw(grayI, 0.5); % Threshold at 0.5

imshow(BW);

**uigetfile()**

**Purpose:**  
Opens a file selection dialog box to let user pick a file.

**Syntax:**

[filename, pathname] = uigetfile('\*.jpg','Select an image');

**Example:**

[filename, pathname] = uigetfile({'\*.jpg;\*.png'}, 'Select an image');

I = imread(fullfile(pathname, filename));

imshow(I);

**imcomplement()**

**Purpose:**  
Computes the complement (inverse) of the image.  
Light pixels become dark and vice versa.

**Syntax:**

J = imcomplement(I);

**Example:**

I = imread('flowers.jpg');

J = imcomplement(I);

imshow(J);

**edge()**

**Purpose:**  
Detects edges in a grayscale or binary image using different methods.

**Syntax:**

BW = edge(I,'method');

'method' can be 'Sobel', 'Canny', 'Prewitt', etc.

**Example:**

I = imread('flowers.jpg');

grayI = rgb2gray(I);

BW = edge(grayI,'Canny'); % Detect edges using Canny

imshow(BW);

**imrotate()**

**Purpose:**  
Rotates an image by a specified angle.

**Syntax:**

J = imrotate(I, angle);

Angle in degrees (positive = counterclockwise).

**Example:**

I = imread('flowers.jpg');

J = imrotate(I, 45); % Rotate 45 degrees counterclockwise

imshow(J);

**Putting it all together:**

% Read image

I = imread('flowers.jpg');

imshow(I); title('Original');

% Convert to grayscale and show histogram

grayI = rgb2gray(I);

figure; imhist(grayI); title('Histogram');

% Adjust contrast

J = imadjust(grayI);

figure; imshow(J); title('Contrast Adjusted');

% Convert to binary

BW = im2bw(grayI,0.5);

figure; imshow(BW); title('Binary Image');

% Edge detection

edges = edge(grayI,'Canny');

figure; imshow(edges); title('Edges');

% Rotate

rotated = imrotate(I,90);

figure; imshow(rotated); title('Rotated 90 degrees');

Would you like me to also draw a **diagram showing how these functions transform the image** step by step (original → grayscale → binary → edges)?