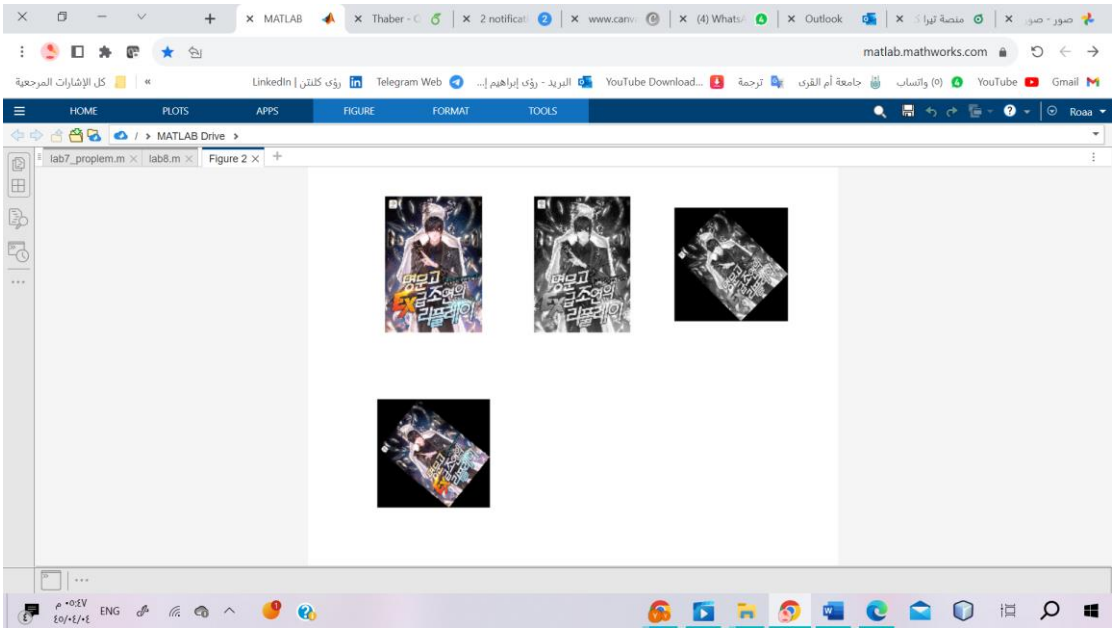
```
%lab 8
I1= imread("/MATLAB Drive/ex-rank-supporting-roles-replay-in-a-prestigious-school.jpg");
I1_gray=rgb2gray(I1);

%To rotate the image è Use imrotate(I,@, method, bbox) ,
%method ('nearest' (default) | 'bilinear' | 'bicubic')
%bbox ('loose' (default) | 'crop')
j= imrotate(I1_gray,45,'bilinear','loose');


tform1 = randomAffine2d(Rotation=[35 55]);
J = imwarp(I1,tform1);
imshow(J);


figure;
subplot(2, 3, 1);
imshow(I1);
subplot(2, 3, 2);
imshow(I1_gray);
subplot(2, 3, 3);
imshow(j);
subplot(2, 3, 4);
imshow(J);
```



```
j = imrotate(I1_gray, 45, 'bilinear', 'loose');
```

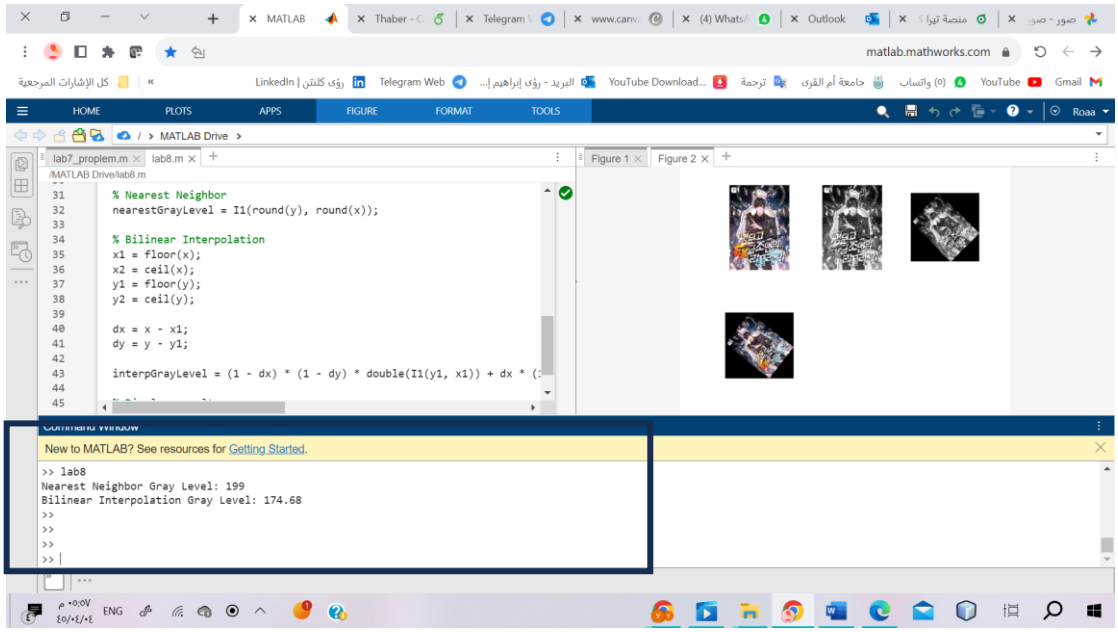
This line rotates the grayscale image I1_gray by 45 degrees counterclockwise using the imrotate() function. The 'bilinear' method is used for interpolation, and the 'loose' option specifies that the output image size should be adjusted to include the entire rotated image. The resulting rotated image is stored in the variable j.

```
tform1 = randomAffine2d('Rotation', [35 55]);

J = imwarp(I1, tform1);

imshow(J);
```

These three lines generate a random affine transformation using the randomAffine2d() function. The 'Rotation' property is set to a range of 35 to 55 degrees. The resulting transformation is stored in the variable tform1. Then, the imwarp() function is used to apply this transformation to the original color image I1, resulting in the transformed image stored in the variable J. Finally, the transformed image J is displayed using the imshow() function.

```
figure;

subplot(2, 3, 1);

imshow(I1);

subplot(2, 3, 2);

imshow(I1_gray);

subplot(2, 3, 3);

imshow(j);

subplot(2, 3, 4);

imshow(J);
```

These lines create a figure window and define a 2x3 grid of subplots. In the first subplot (top left), the original color image I1 is displayed using imshow(). In the second subplot (top middle), the grayscale image I1_gray is displayed. In the third subplot (top right), the rotated image j is displayed. In the fourth subplot (bottom left), the transformed image J is displayed.




```
% Point of interest
x = 43.6;
y = 31.2;

% Nearest Neighbor
nearestGrayLevel = I1(round(y), round(x));

% Bilinear Interpolation
x1 = floor(x);
x2 = ceil(x);
y1 = floor(y);
y2 = ceil(y);

dx = x - x1;
dy = y - y1;

interpGrayLevel = (1 - dx) * (1 - dy) * double(I1(y1, x1)) + dx * (1 - dy)
* double(I1(y1, x2)) + (1 - dx) * dy * double(I1(y2, x1)) + dx * dy *
double(I1(y2, x2));

% Display results
disp(['Nearest Neighbor Gray Level: ' num2str(nearestGrayLevel)]);
disp(['Bilinear Interpolation Gray Level: ' num2str(interpGrayLevel)]);
```

% Point of interest

x = 43.6;

y = 31.2;

These lines define the coordinates of the point of interest (43.6, 31.2) in the image.
The x variable represents the horizontal coordinate, and the y variable represents the vertical coordinate.

% Nearest Neighbor

nearestGrayLevel = I(round(y), round(x));

This line uses the round() function to round the x and y coordinates to the nearest integers. It then accesses the grayscale value of the pixel at the rounded coordinates (round(y), round(x)). The grayscale value is assigned to the variable nearestGrayLevel.

% Bilinear Interpolation

x1 = floor(x);

x2 = ceil(x);

y1 = floor(y);

y2 = ceil(y);

These lines calculate the four surrounding pixel coordinates (x1, y1), (x1, y2), (x2, y1), and (x2, y2) based on the floor and ceil functions. The floor() function returns the largest integer less than or equal to x or y, while the ceil() function returns the smallest integer greater than or equal to x or y.

dx = x - x1;

dy = y - y1;

These lines calculate the fractional parts dx and dy by subtracting the rounded coordinates x1 and y1 from the original x and y coordinates, respectively. These fractional parts indicate the relative distances between the point of interest and the surrounding pixels.

interpGrayLevel = (1 - dx) * (1 - dy) * double(I(y1, x1)) + dx * (1 - dy) * double(I(y1, x2)) + (1 - dx) * dy * double(I(y2, x1)) + dx * dy * double(I(y2, x2));

This line performs bilinear interpolation to estimate the gray level at the point of interest. It calculates the weighted average of the grayscale values of the four surrounding pixels. The grayscale values are multiplied by the corresponding weights (1 - dx) * (1 - dy), dx * (1 - dy), (1 - dx) * dy, and dx * dy, respectively. The double() function converts the grayscale values to double type for accurate calculations. The resulting interpolated gray level is assigned to the variable interpGrayLevel.

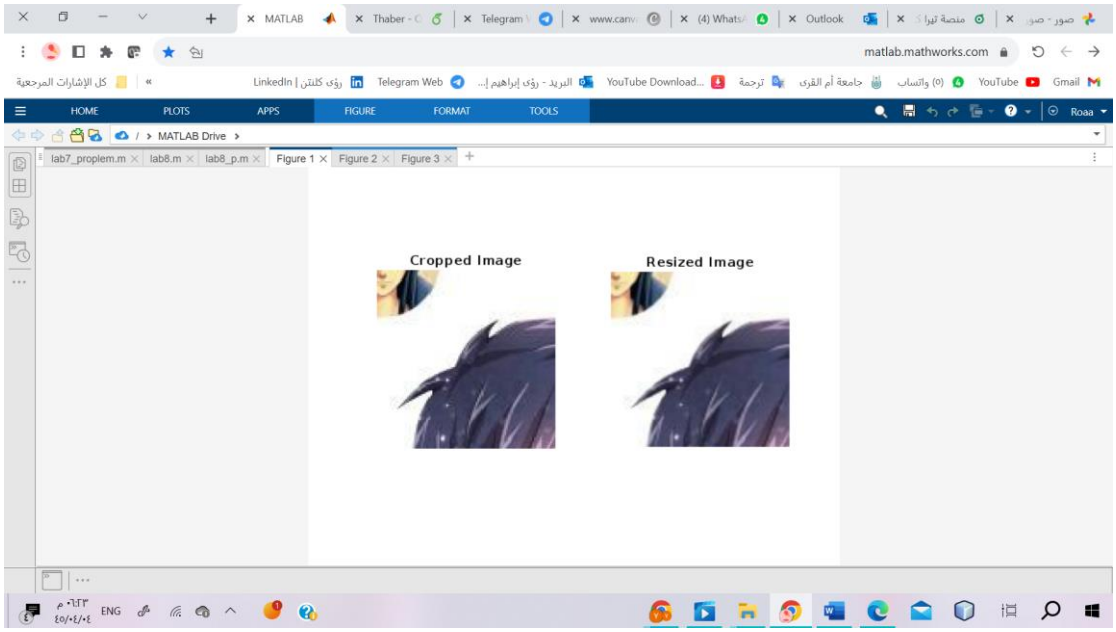
% Display results

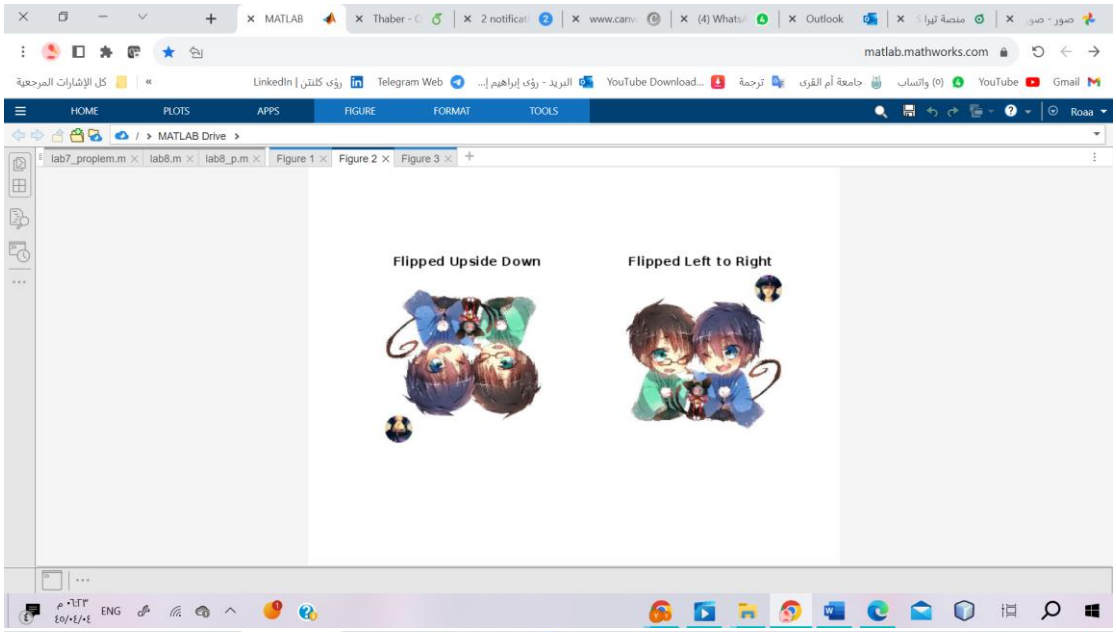
disp(['Nearest Neighbor Gray Level: ' num2str(nearestGrayLevel)]);

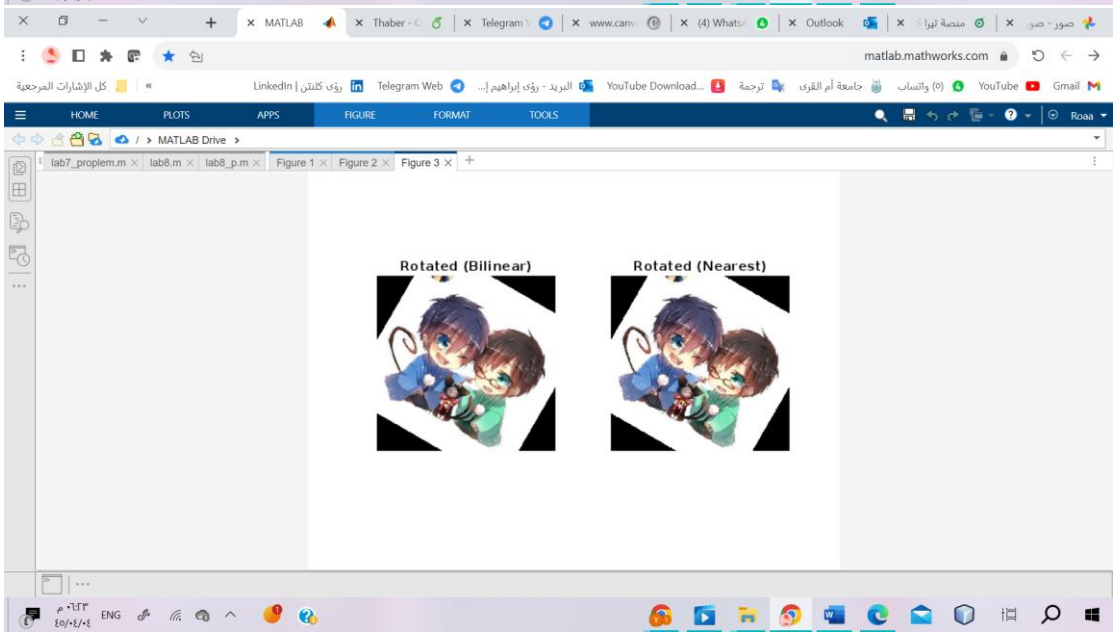
disp(['Bilinear Interpolation Gray Level: ' num2str(interpGrayLevel)]);

These lines display the gray levels obtained from nearest neighbor and bilinear interpolation. The disp() function is used to show the results as strings, which include the gray level values concatenated with the corresponding method names.

رؤى إبراهيم كلنتن 441004834 شعبة 1 نظري








```
% a. Crop the image, then resize it (bilinear) to be the same size as the
original image and show both images

I = imread('/MATLAB
Drive/8dd58e1aa720db90851b2440adfb57af5e449b91_hq.jpg');
I_cropped = imcrop(I, [50, 50, 100, 100]); % Example crop parameters,
adjust as needed
I_resized = imresize(I_cropped, size(I(:, :, 1)), 'bilinear'); % Corrected
resizing syntax
figure;
subplot(1, 2, 1);
imshow(I_cropped);
title('Cropped Image');
subplot(1, 2, 2);
imshow(I_resized);
title('Resized Image');

% b. Flip the image upside down and from left to right and show both
results
```

```
I_flipped_ud = flipud(I);
I_flipped_lr = fliplr(I);
figure;
subplot(1, 2, 1);
imshow(I_flipped_ud);
title('Flipped Upside Down');
subplot(1, 2, 2);
imshow(I_flipped_lr);
title('Flipped Left to Right');

% c. Rotate the image 30 degrees clockwise direction (try bilinear and
nearest options) then compare the results
I_rot_bilinear = imrotate(I, -30, 'bilinear', 'crop');
I_rot_nearest = imrotate(I, -30, 'nearest', 'crop');
figure;
subplot(1, 2, 1);
imshow(I_rot_bilinear);
title('Rotated (Bilinear)');
subplot(1, 2, 2);
imshow(I_rot_nearest);
title('Rotated (Nearest)');
```

I_cropped = imcrop(I, [50, 50, 100, 100]);

- This line crops the image I using the imcrop function. The [50, 50, 100, 100] argument specifies the crop rectangle as [x, y, width, height]. You can adjust these values to change the cropping region.

matlab

Copy

I_resized = imresize(I_cropped, size(I(:, :, 1)), 'bilinear');

- This line resizes the cropped image I_cropped to be the same size as the original image I. The size(I(:, :, 1)) argument retrieves the size of the original image in the RGB channels. The 'bilinear' option specifies the interpolation method used for resizing.

matlab

Copy

figure;

subplot(1, 2, 1);

imshow(I_cropped);

title('Cropped Image');

subplot(1, 2, 2);

imshow(I_resized);

title('Resized Image');

- These lines create a figure with two subplots. The first subplot displays the cropped image using imshow, and the second subplot displays the resized image.
  The title function adds titles to the subplots.

matlab

Copy

```
I_flipped_ud = flipud(I);

I_flipped_lr = fliplr(I);
```

- These lines flip the image I upside down and from left to right using
  the flipud and fliplr functions, respectively. The flipped images are assigned to the variables I_flipped_ud and I_flipped_lr.

matlab

Copy

```
figure;

subplot(1, 2, 1);

imshow(I_flipped_ud);

title('Flipped Upside Down');

subplot(1, 2, 2);

imshow(I_flipped_lr);

title('Flipped Left to Right');
```

- These lines create a new figure with two subplots. The first subplot displays the image flipped upside down, and the second subplot displays the image flipped from left to right. The title function adds titles to the subplots.

matlab

Copy

```
I_rot_bilinear = imrotate(I, -30, 'bilinear', 'crop');

I_rot_nearest = imrotate(I, -30, 'nearest', 'crop');
```

- These lines rotate the image I 30 degrees clockwise. The first line uses
  the 'bilinear' interpolation method, and the second line uses
  the 'nearest' interpolation method. The rotated images are assigned to the variables I_rot_bilinear and I_rot_nearest.

matlab

Copy

```
figure;

subplot(1, 2, 1);
```

```
imshow(I_rot_bilinear);
title('Rotated (Bilinear)');
subplot(1, 2, 2);
imshow(I_rot_nearest);
title('Rotated (Nearest)');
```

- These lines create a new figure with two subplots. The first subplot displays the image rotated using bilinear interpolation, and the second subplot displays the image rotated using nearest neighbor interpolation. The title function adds titles to the subplots.