دانشگاه تهران

دانشکده علوم و فنون نوین

پردازش تصویر دیجیتال

تمرین شماره هشت

نام و نام خانوادگی	فاطمه چیت ساز
شماره دانشجویی	830402092
تاریخ ارسال گزارش	21 آذر 1402

Contents

سوال 1 – انجام فرایند rotate روی تصویر

در این سوال قرار است ما فرایند rotation را روی یک تصویر انجام دهیم و این کار را ابتدا خودمان پیاده سازی کرده و سپس از توابع اماده متلب مانند imrotate استفاده کرده و نتایج را مقایسه کنیم

توضیح کد:

ابتدا برای انجام فرایند تبدیلات نیاز اساسی ما تبدیل فضای تصویر به فضای مختصاتی خودمان است که این امر نیازمند دو تابع مهم
index2coordinate و coordinate2index است

```
% Define functions for coordinate transformations
index2coordinate = @(P, Q, pq) [pq(2)-(Q+1)/2 (P+1)/2-pq(1)];
coordinate2index = @(P, Q, uv) [(P+1)/2-uv(2) uv(1)+(Q+1)/2];
```

در تابع index2coordinate ما ابعاد تصویر خود یعنی P و Q را داریم و میخواهیم تصویر اصلی خود را با فضای مختصاتی جدیدی ببریم که وسط تصویر ما صفر و صفر ما باشد و در واقع به فضای u و v برویم

در تابع coordinate2index دقیقا عکس فرایند بالا انجام میشود و تصویر خود در فضای مختصاتی جدید که با دو بردار u و v نمایش داده میشود را میخواهیم به فضای اصلی خود یعنی p و q ببریم

نکته دومی که در تبدیلات ما مهم است تعریف تابع تبدیل است که اینجا با توجه به این امر که ما میخواهیم فرایند چرخش را انجام دهیم از ماتریس تبدیل زیر استفاده میکنیم

```
% Transformation matrix for rotation
T = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
```

حال که نیازمندی های اصلی را انجام دادیم زمان آن رسیده که فرایند تبدیل را انجام دهیم تابع تبدیل را به صورت زیر تعریف میکنیم به طوری که روش interpolation و محتویات تصویر و مقدار درجه چرخش را از ما میگیرد

```
function compare_rotate_methods(I1, theta, interpolation_method, title_text)
```

در مرحله بعدی ابعاد تصویر خود را به دست آورده و به عنوان P1 وQ1 تعریف میکنیم سپس باید ابعاد تصویر خرخش یافته را بدست آوریم که برای این کار نیاز است که ماتریس تبدل خود را در ابعاد به دست امده ضرب کنیم سپس میتوانیم ابعاد تصویر جدید را با عناوین P2 و Q2 بدست آوریم و همچنین ماتریس تصویر چرخش یافته را با ابعاد به دست آمده بسازیم

```
% Calculate size of the rotated ima
size2 = abs(T) * [P1 Q1]';
P2 = round(size2(1));
Q2 = round(size2(2));

% Initialize rotated image
I2 = zeros(P2, Q2);
```

بعد از این مرحله باید به ازای هر پیکسل در تصویر چرخش یافته مقدار آن را در تصویر اصلی پیدا کرده و آن را در ماتریس I2 مان جایگذاری کنیم که برای این کار ابتدا مختصات p2 و q2 در تصویر چرخش یافته را به فضای u وv میبریم و سپس برای انکه مقدار را در تصویر اصلی بدست اوریم ماتریس ترانهاده ی ماتریس تبدیل خود را در آن ضرب کرده تا u و v در تصویر اصلی به دست آید سپس u و v را به p و q تبدیل کرده حال بدین روش ما آن مقداری که نیاز داریم را در تصویر اصلی پیدا کردیم اما نکته ای که وجود دارد این است که این مقدار یک مقدار اعشاری است و ما نیاز داریم که برای این کار از interpolation ها استفاده کنیم تا با استفاده از پیکسل های اطراف p1 و q1 مقدار تصویر در مختصات p1 و q2 را حدس بزنیم

در اینجا ما از interp2 استفاده کرده و با توجه به interpolation method فرایند interpolation را انجام میدهیم و مقدار پیکسل p2 و q1 را محاسبه میکنیم

نکته دیگر این است که اگر بعد از اعمال ترانهاده ماتریس تبدیل اگر مختصات ما مختصات پرتی بود یعنی کوچکتر از یک بود یا کلا بیشتر از P و Q بود آن را صفر در نظر کرده و در فرایند interpolation را دخیل نمیکنیم

```
% Loop through each pixel in the rotated image
for p2 = 1:P2
    for q2 = 1:Q2
        % Convert index in the rotated image to coordinates
        uv2 = index2coordinate(P2, Q2, [p2 q2]);

        % Apply inverse transformation to get corresponding coordinates in the original image
        uv1 = Tinv * uv2';

        % Convert coordinates to index in the original image
        pq1 = coordinate2index(P1, Q1, uv1);

        % Check if the index is within the valid range of the original image
        if all(pq1 <= [P1 Q1]) && all(pq1 >= [1 1])
            % Use interp2 for interpolation
            I2(p2, q2) = interp2(I1, pq1(2), pq1(1), interpolation_method, 0);
        end
    end
end
```

حال تصویر I2 بدست آمده تصویر ما بعد از فرایند چرخش است

ما دقیقا همین فرایند را میتوانیم با تابع آماده متلب نیز انجام دهیم

```
% Use imrotate for comparison
I2_imrotate = imrotate(I1, theta, interpolation_method);
```

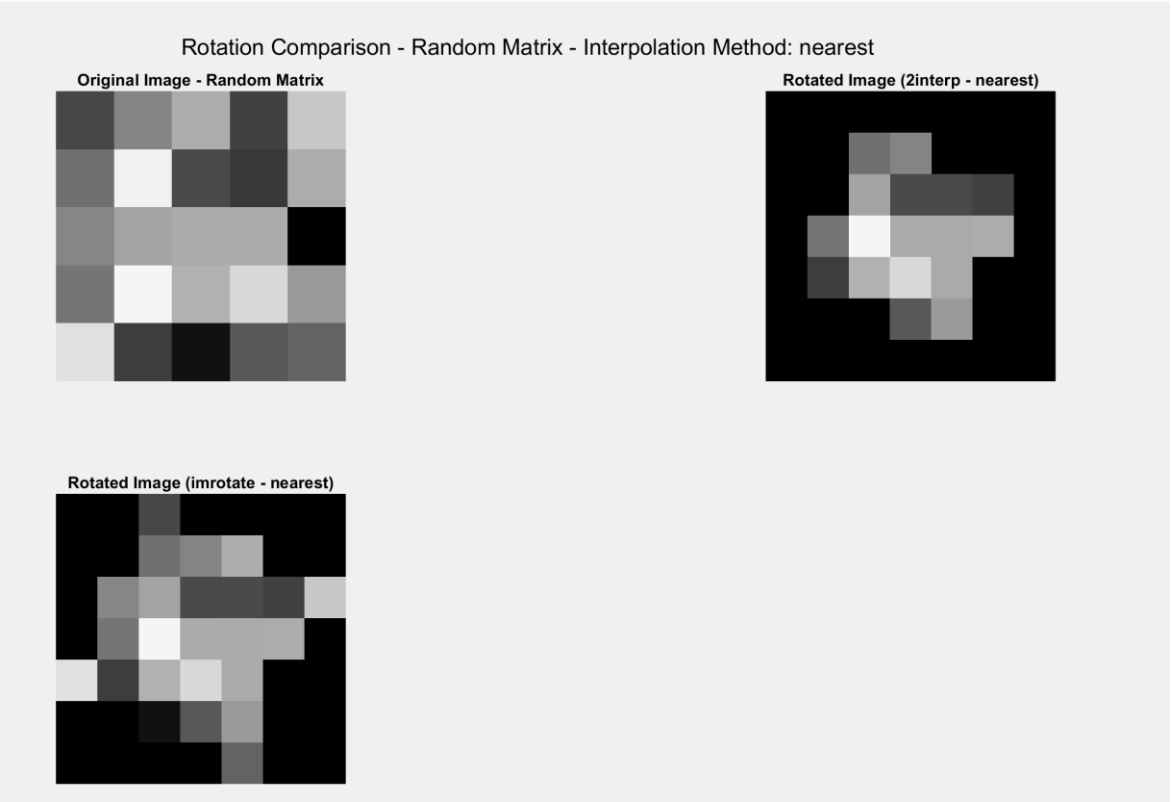
حال میتوانیم به زیبایی توابعی که ساختیم را استفاده کنیم

```
compare_rotate_methods_random_and_image()
```

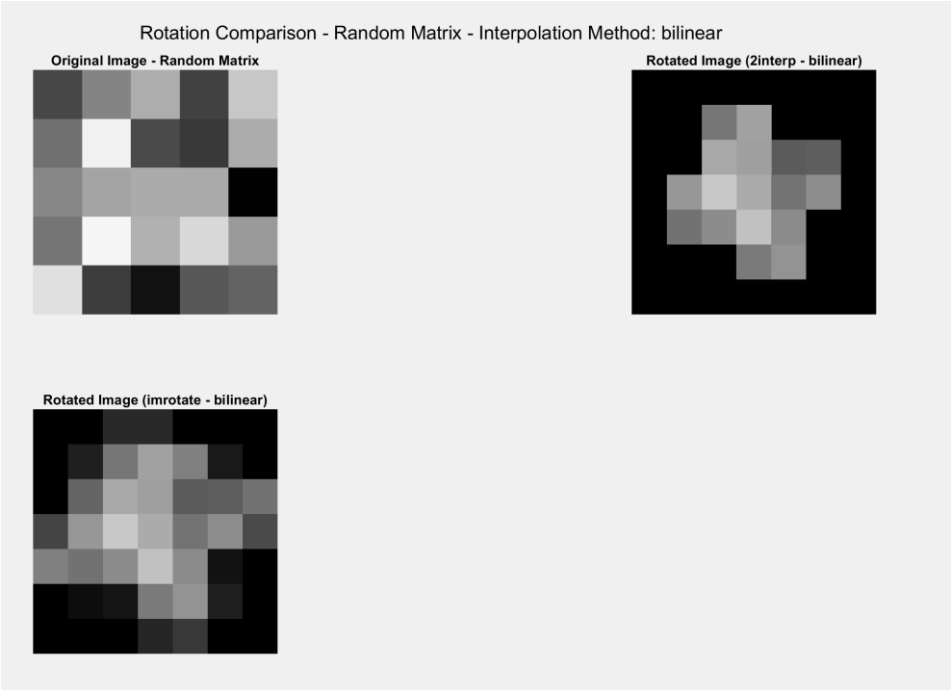
```
function compare_rotate_methods_random_and_image()
    % Create a small random matrix
    random_matrix = rand(5, 5);

    % Set the rotation angle and interpolation methods
    theta = -30;
    interpolation_methods = {'nearest', 'bilinear', 'bicubic'};

    % Loop over interpolation methods for the random matrix
    for i = 1:length(interpolation_methods)
        interpolation_method = interpolation_methods{i};

        % Call the function to compare rotation methods for the random matrix
        compare_rotate_methods(random_matrix, theta, interpolation_method, 'Random Matrix');
    end

    % Load the cameraman image
    cameraman_image = double(imread('cameraman.tif'))/255;

    % Loop over interpolation methods for the cameraman image
    for i = 1:length(interpolation_methods)
        interpolation_method = interpolation_methods{i};

        % Call the function to compare rotation methods for the cameraman image
        compare_rotate_methods(cameraman_image, theta, interpolation_method, 'Cameraman Image');
    end
end
```

ما در اینجا از روش های نزدیکترین و خطی و درجه سه برای cameraman و یک ماتریس پنج در پنج تصادفی و سی درجه چرخش استفاده کرده ایم

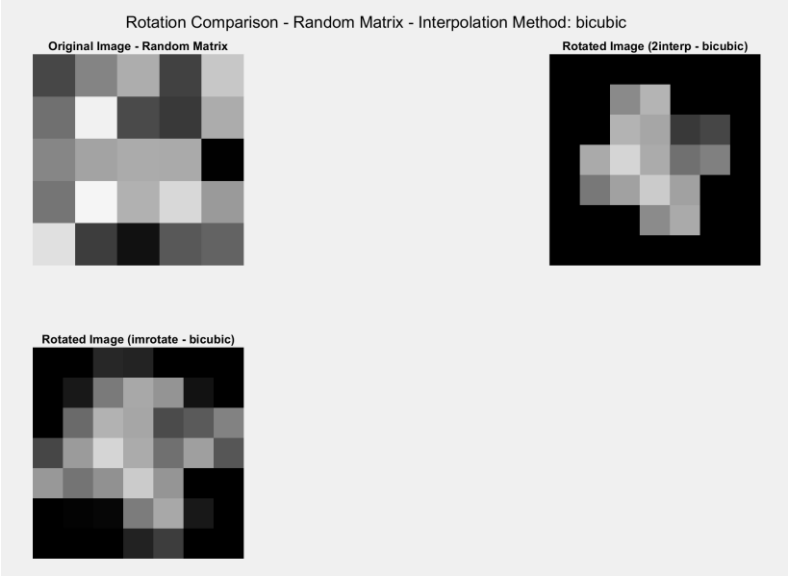
خروجی برای ماتریس تصادفی در روش نزدیکترین



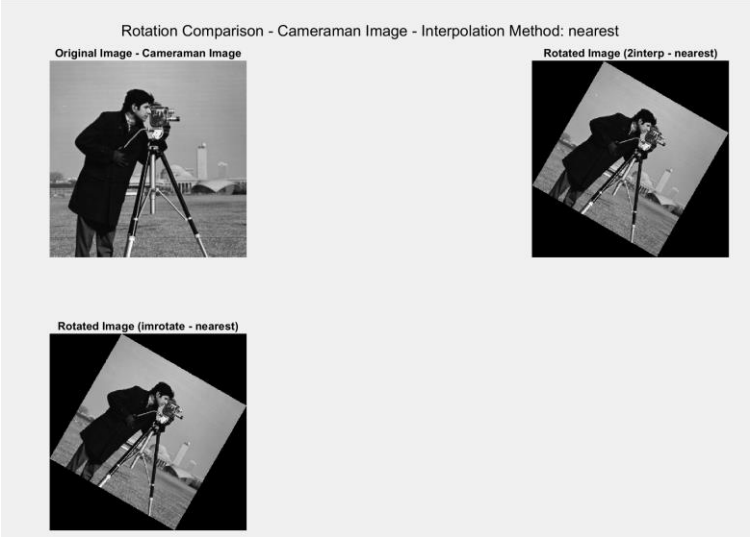
خروجی برای ماتریس تصادفی در روش خطی



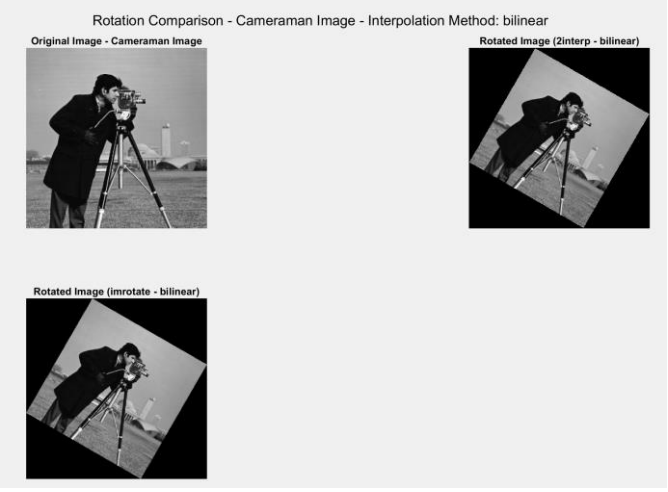
خروجی برای ماتریس تصادفی در روش درجه سه



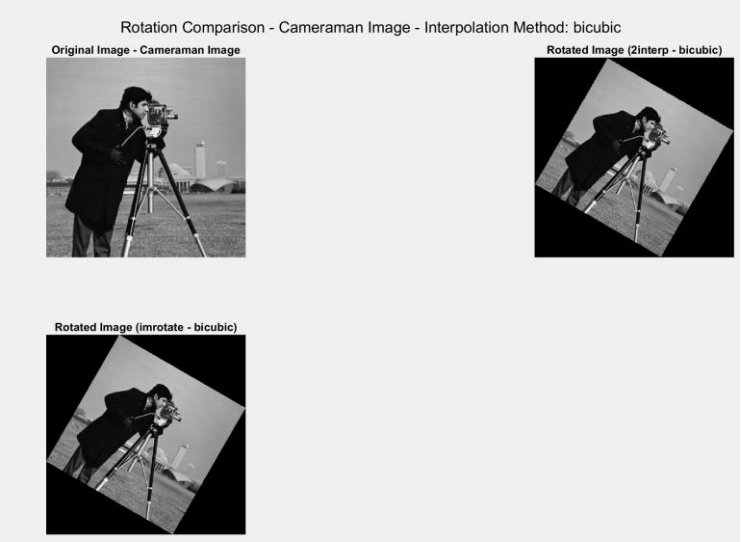
خروجی برای cameramanدر روش نزدیکترین



خروجی برای cameramanدر روش خطی



خروجی برای ماتریس cameraman در روش درجه سه



سوال 2 – انجام فرایند scale روی تصویر

در این قسمت نیز همانند قسمت بالا هستیم و تنها عملیات تغیر و transform ما متفاوت است و اینجا قصد داریم فرایند scale را انجام دهیم بنابراین تنها ماتریس T ما به شکل زیر تغیر پیدا خواهد کرد

```
T = [1/scale_factor, 0; 0, 1/scale_factor];
```

همچنین فرایند scale را میتوانیم با توابع آماده متلب نیز به صورت زیر انجام دهیم

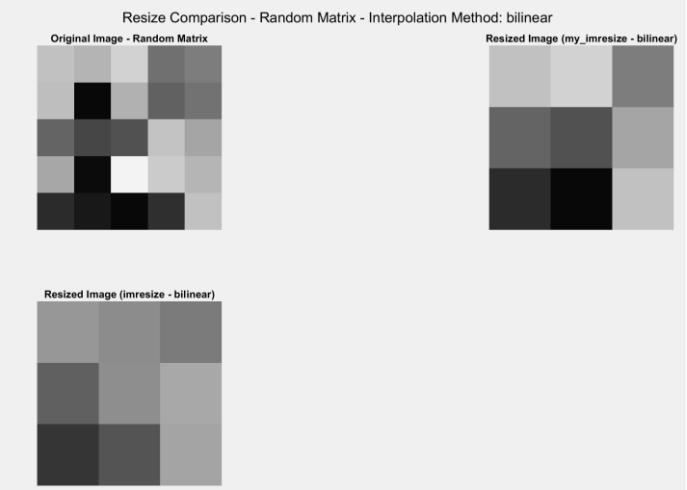
```
% Use imresize for comparison
I2_imresize = imresize(I1, 1/scale_factor, interpolation_method);
```

حال میتوانیم از توابع خود استفاده کرده و در اینجا ما با scale=2 فرایند scale را بر روس ماتریس تصاوفی و cameramanانجام خواهیم داد

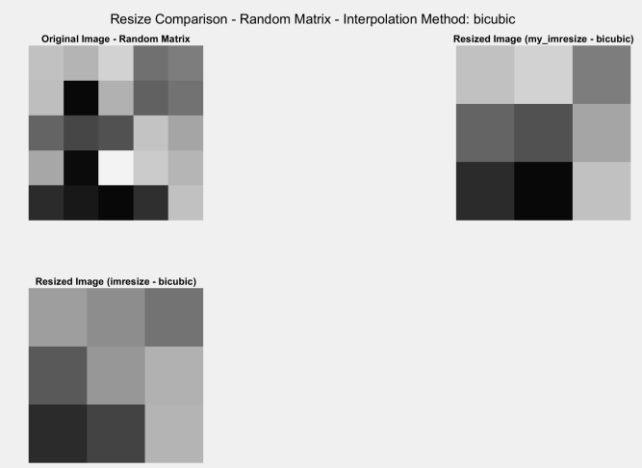
```
function compare_resize_methods_random_and_image()
    % Create a small random matrix
    random_matrix = rand(5, 5);

    % Set the scaling factor and interpolation methods
    scale_factor = 2;
    interpolation_methods = {'nearest', 'bilinear', 'bicubic'};

    % Loop over interpolation methods for the random matrix
    for i = 1:length(interpolation_methods)
        interpolation_method = interpolation_methods{i};

        % Call the function to compare resize methods for the random matrix
        compare_resize_method(random_matrix, scale_factor, interpolation_method, 'Random Matrix');
    end

    % Load the cameraman image
    cameraman_image = double(imread('cameraman.tif')) / 255;

    % Loop over interpolation methods for the cameraman image
    for i = 1:length(interpolation_methods)
        interpolation_method = interpolation_methods{i};

        % Call the function to compare resize methods for the cameraman image
        compare_resize_method(cameraman_image, scale_factor, interpolation_method, 'Cameraman Image');
    end
end
```

خروجی برای ماتریس تصادفی در روش نزدیکترین

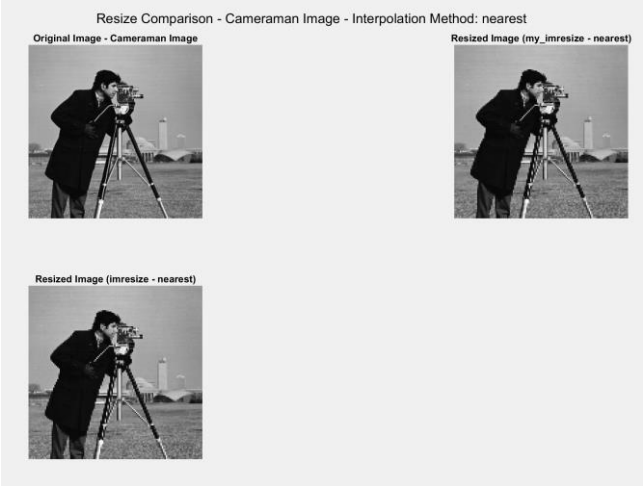
خروجی برای ماتریس تصادفی در روش خطی



خروجی برای ماتریس تصادفی در روش درجه سه

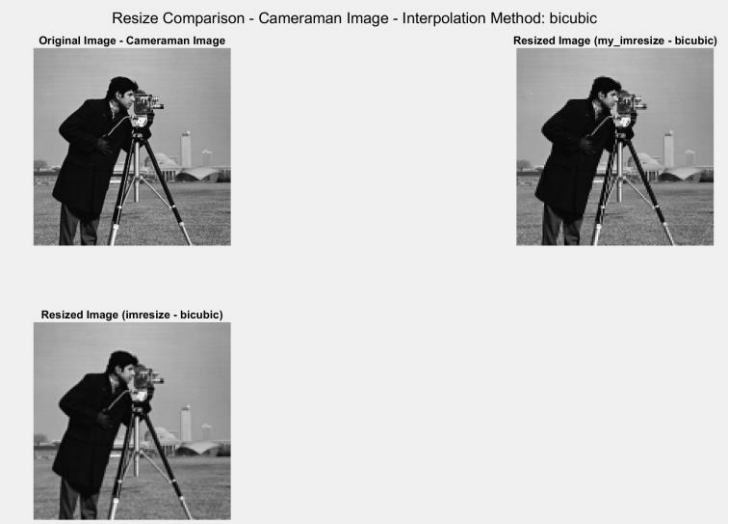


خروجی برای cameramanدر روش نزدیکترین



خروجی برای cameramanدر روش خطی

خروجی برای ماتریس cameraman در روش درجه سه


Resize Comparison - Cameraman Image - Interpolation Method: bicubic
Original Image - Cameraman Image
Resized Image (my_imresize - bicubic)
Resized Image (imresize - bicubic)