دانشگاه تهران

دانشکده علوم و فنون نوین

پردازش تصویر دیجیتال

تمرین شماره هفت

نام و نام خانوادگی	فاطمه چیت ساز
شماره دانشجویی	830402092
تاریخ ارسال گزارش	14 آذر 1402

Contents

سوال 1 – اجرای تابع درون یابی 1

سوال 2 – اجرای تابع درون یابی بر روی cameraman 6

سوال 1 – اجرای تابع درون یابی

در این سوال ما قرار است روش های درون یابی را پیاده سازی کرده و هر یک از آنها را با توابع آماده متلب مقایسه کنیم

روش نزدیکترین همسایه :

```
% Get the size of the input matrix
[m, n] = size(input_matrix);

% Define the grid for the original matrix
[XI, YI] = meshgrid(1:scale_factor:n, 1:scale_factor:m); % Increase the resolution for smoot

% Initialize the interpolated matrix
interpolated_matrix = zeros(size(XI));

% Perform interpolation based on the selected type
for i = 1:numel(XI)
    % Calculate the corresponding indices in the original matrix
    sourceRow = YI(i);
    sourceCol = XI(i);

    % Perform interpolation based on the selected type
    switch interpolation_type
        case 'nearest'
            % Nearest-neighbor interpolation
            interpolated_matrix(i) = input_matrix(round(sourceRow), round(sourceCol));
```

در روش درون‌یابی نزدیک‌ترین همسایه (Nearest-neighbor interpolation)، هر نقطه جدید در تصویر درون‌یافته با نزدیک‌ترین نقطه موجود در تصویر اصلی به آن تخصیص می‌یابد

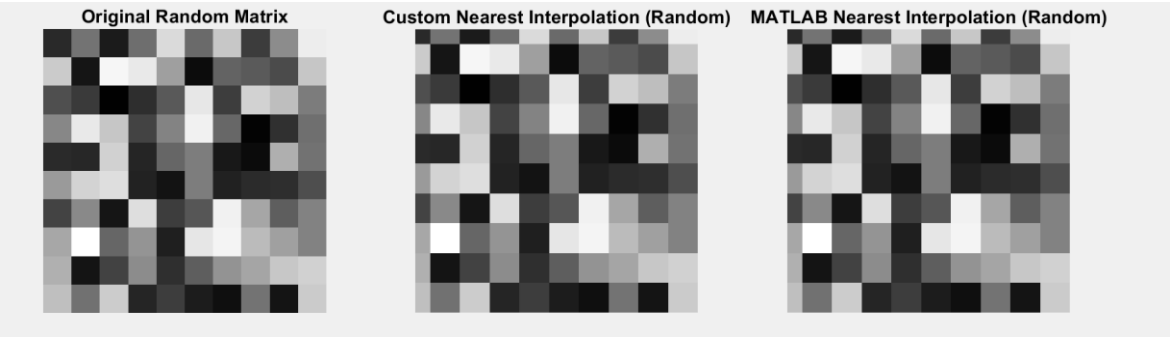
در واقع با توجه به scale factor ما ماتریس جدید و اندازه جدید تصویر خود را میسازیم حال با توجه به موقعیت خود در تصویر نزدیکترین نقطه در تصویر اصلی را پیدا میکنیم که این کار را میتوان به سادگی با round کردن انجام داد حال ماتریس `interpolated_matrix` به ازای هر پیکسل مقدار دهی میشود

همچنین میتوان این کار را با توابع اماده متلب نیز انجام داد

```
[m, n] = size(input_matrix);

% Define the grid for the interpolated matrix
[X, Y] = meshgrid(1:n, 1:m);

% Define the grid for the original matrix
[XI, YI] = meshgrid(1:scale_factor:n, 1:scale_factor:m); % Increase the resolutior

% Perform interpolation based on the selected type
switch interpolation_type
    case 'nearest'
        interpolated_matrix = interp2(X, Y, input_matrix, XI, YI, 'nearest');
```

در واقع در اینجا نیز ما با توجه به scale factor خود ماتریس جدید خود را میسازیم و از interp2 برای ساخت تصویر بعد از درون یابی استفاده میکنیم
X و Y اندازه ابتدایی تصویر ما و XI و YI اندازه تصویر بعد از درون یابی است و input matrix نیز تصویر ورودی ماست

خروجی:
در اینجا ما یک ماتریس ده در ده تصادفی را به عنوان تصویر ورودی خود داده ایم و با scale factor =0.5 فرایند درون یابی را بر روی تصویر خود انجام داده و به تصویر 20 در 20 رسیده ایم



روش خطی:
در روش خطی ما باید یک خط را متصور شویم و سپس با توجه به آن خط به هر یک از پیکسل های نزدیک پیکسلی که میخواهیم مقدار آن را حدس بزنیم یک وزنی دهیم و حاصل جمع این موارد مقدار پیکسلی است که ما قصد حدس آن را داریم

```
case 'linear'
    % Bilinear interpolation
    row_floor = floor(sourceRow);
    col_floor = floor(sourceCol);
    row_ceil = min(row_floor + 1, m);
    col_ceil = min(col_floor + 1, n);

    % Interpolate in the row direction
    value_row_floor = input_matrix(row_floor, col_floor) + (input_matrix(row_floor, col_ceil) - input_matrix(row_floor, col_floor)) * (sourceCol - col_floor);
    value_row_ceil = input_matrix(row_ceil, col_floor) + (input_matrix(row_ceil, col_ceil) - input_matrix(row_ceil, col_floor)) * (sourceCol - col_floor);

    % Interpolate in the column direction
    interpolated_matrix(i) = value_row_floor + (value_row_ceil - value_row_floor) * (sourceRow - row_floor);
```

توضیح کد :

row_floor، col_floor، row_ceil، و col_ceil مختصات چهار نقطه اطراف نقطه مورد نظر هستند.

sourceRow و sourceCol مختصات نقطه مورد نظر در تصویر هستند.

m و n ابعاد تصویر ورودی (تعداد سطرها و ستون‌ها) هستند.

row_floor و col_floor به ترتیب معادل عدد پائینترین صحیح از مختصات sourceRow و sourceCol هستند.

row_ceilو col_ceil به ترتیب معادل عدد بزرگ‌ترین صحیح از sourceRow و sourceColهستند. مقدارهای min نیز برای جلوگیری از خطای اعتبارسنجی خارج از محدوده تصویر استفاده شده‌اند.

تفسیر در جهت سطر:

value_row_floorو value_row_ceil با استفاده از تفسیر خطی بین دو نقطهمجاور در سطر محاسبه می‌شوند.

input_matrix(row_floor, col_floor)و

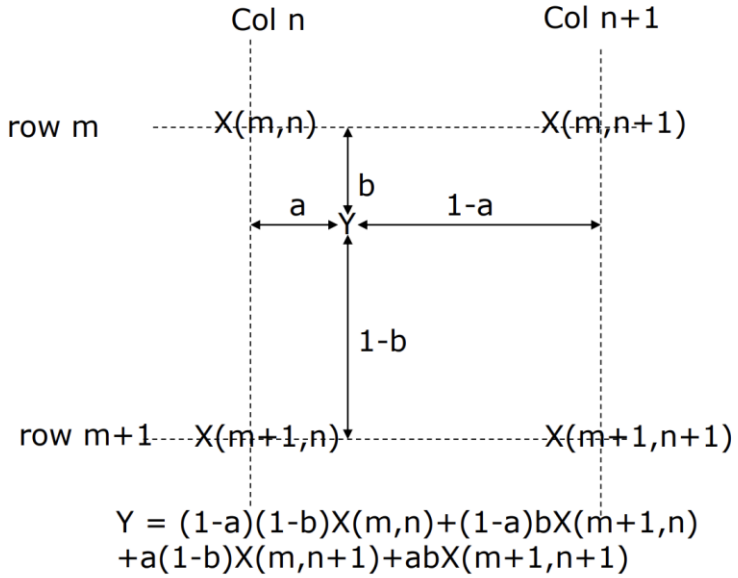
input_matrix(row_floor, col_ceil)به ترتیب مقادیر نقطهٔ پائین و روی سطر row_floor هستند.

تفسیر در جهت ستون:

interpolated_matrix(i)مقدار نهایی تفسیر دوخطی برای نقطهٔ i از تصویر خروجی است.

مقدار value_row_floor و value_row_ceil به کمک تفسیر خطی بین دو نقطه مجاور در ستون محاسبه می‌شود.

فرمول فرایند محاسبه:

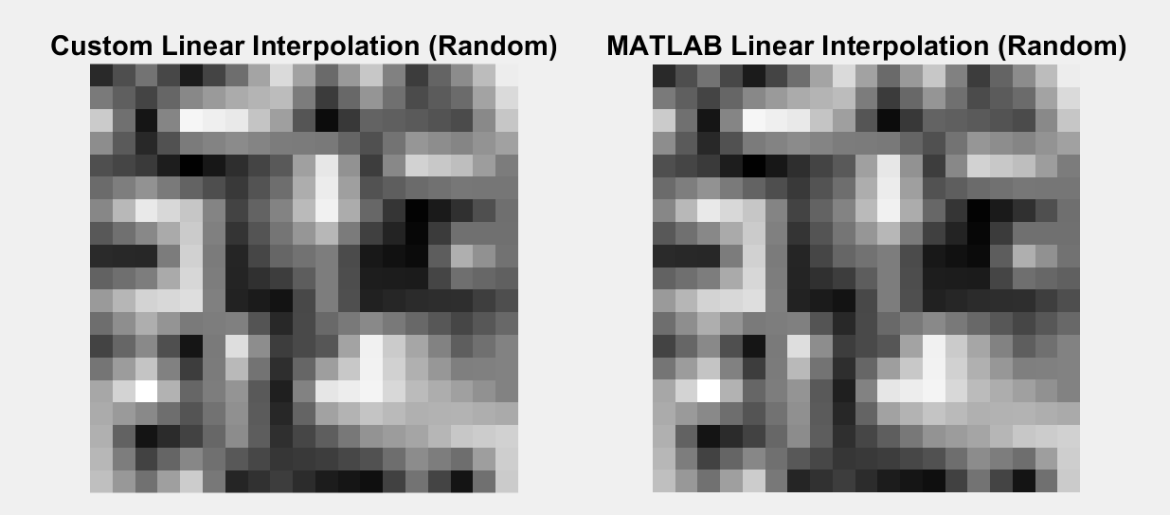


همچنین با تابع interp2 نیز میتوان فرایند محاسبه درون یابی خطی را انجام داد

```
case 'linear'
    interpolated_matrix = interp2(X, Y, input_matrix, XI, YI, 'linear');
```

خروجی کد:

تصویر ده در ده ما به تصویر بیست در بیست زیر تبدیل میشود



روش درجه سوم :

در این روش ما باید یک تابع درجه سوم به شکل زیر بسازیم

$$R_c(x) = \begin{cases} A_1|x|^3 + B_1|x|^2 + C_1|x| + D_1 & \text{for } 0 \le |x| \le 1 \\ A_2|x|^3 + B_2|x|^2 + C_2|x| + D_2 & \text{for } 1 < |x| \le 2 \end{cases}$$

که در آن شروط زیر برقرار است

1. $R_c(x) = 1$ at $x = 0$, and $R_c(x) = 0$ at $x = 1, 2$.

2. The first-order derivative $R'_c(x) = 0$ at $x = 0, 1, 2$.

بنابراین نتیجه میشود که

$$R_c(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & \text{for } 0 \le |x| \le 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & \text{for } 1 < |x| \le 2 \end{cases}$$

حال ما میتوانیم این rc را در متلب بسازیم

```
u = @(x) double(x >= 0);

% Define the cubic interpolation function with a=0.5
a = 0.5;
Rc = @(x) ((a + 2) * abs(x).^3 - (a + 3) * x.^2 + 1) .* (u(x + 1) - u(x - 1)) + ...
    (a * abs(x).^3 - 5 * a * x.^2 + 8 * a * abs(x) - 4 * a) .* (u(x - 1) - u(x - 2) + u(x + 2) - u(x + 1));
```

همچنین برای سادگی فرض میکنیم a ثابت و برابر 0.5 است

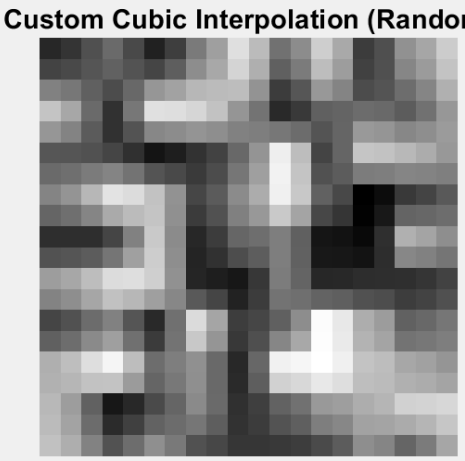
حال فرایند درون یابی را با یافتن همسایگان هر پیکسل و تشکیل ماتریس دو در دو میتوان محاسبه کرد

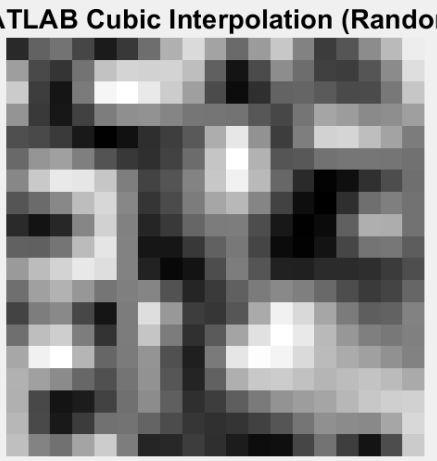
```
new_rows = scaleFactor * rows;
new_cols = scaleFactor * cols;

% Create new grid
new_x = linspace(1, cols, new_cols);
new_y = linspace(1, rows, new_rows);

% Initialize the new image
interpolated_img = zeros(new_rows, new_cols);

% Perform cubic interpolation
for i = 1:new_rows
    for j = 1:new_cols
        % Find the indices of the nearest neighbors
        x_idx = floor(new_x(j));
        y_idx = floor(new_y(i));

        % Clip indices to stay within the matrix bounds
        x_idx = max(1, min(cols - 1, x_idx));
        y_idx = max(1, min(rows - 1, y_idx));

        % Get the four nearest neighbors
        neighbors = input_matrix(y_idx:y_idx + 1, x_idx:x_idx + 1);

        % Perform cubic interpolation using Rc
        x_offset = (new_x(j) - x_idx) / scaleFactor;
        y_offset = (new_y(i) - y_idx) / scaleFactor;
        interpolated_value = Rc(x_offset) * Rc(y_offset) * neighbors(1, 1) + ...
                             Rc(x_offset) * (1 - Rc(y_offset)) * neighbors(2, 1) + ...
                             (1 - Rc(x_offset)) * Rc(y_offset) * neighbors(1, 2) + ...
                             (1 - Rc(x_offset)) * (1 - Rc(y_offset)) * neighbors(2, 2);

        % Update the new image
        interpolated_img(i, j) = interpolated_value;
```

همچنین همین فرایند را میتوان با توابع آمده متلب نیز انجام داد

```
case 'cubic'
    interpolated_matrix = interp2(X, Y, input_matrix, XI, YI, 'cubic');
```

خروجی: بعد از اعمال درون یابی درجه سه بر روی تصویر خواهیم دید





سوال 2 – اجرای تابع درون یابی بر روی cameraman

در این قسمت از سوال از ما خواسته شده تصویر cameraman ک 256* 256 است را با روش های بالا به 512* 512 تبدیل کنیم که برای این مار لازم است تنها از توابعی که در بخش قبل تعریف کردیم استفاده کرده و بین هر دو پیکسل یک پیکسل جدی متصور شویم و بدین گونه توصیر را دو برابر کنیم

خروجی :