دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده مهندسی برق و کامپیوتر

پردازش تصاویر دیجیتال

گزارش تمرین سری سوم-فصل ۴

دانشجو

سید محمد جواد موسوی

استاد کلاس

دکتر حمید سلطانیان‌زاده

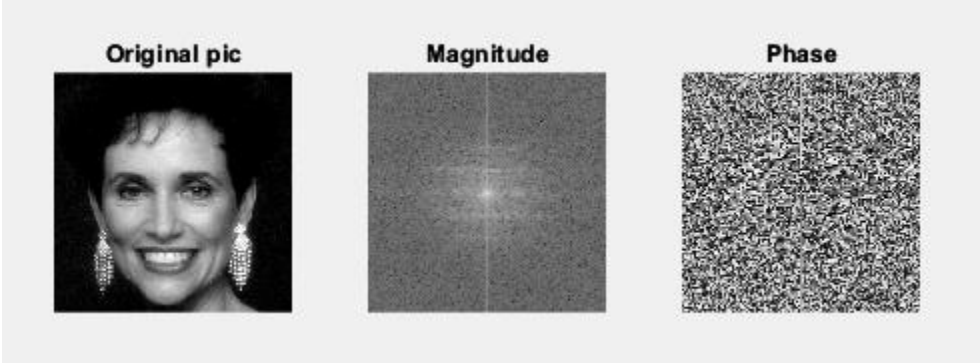
سوال ۲

بخش اول

ابتدا با استفاده از دستور imread تصویر را می خوانیم و سپس با تابع fft۲ فوریه تصویر مورد نظر را محاسبه می‌کنیم. سپس برای انتقال فرکانس های صفر به مرکز از تابع fftshift استفاده شده است. برای به دست آوردن magnitude و phase از ۲تابع abs و angle بهره می بریم. کد متلب گفته شده در پایین نشان داده شده است:

```
 1. %% This exercise explores the characteristics of the Fourier Transform (FT) and implements a
fft2 function in MATLAB
 2. % Part a
 3.
 4. % Load the images "woman.tif" and "parrot.tif"
 5. woman_path = 'images/q2/woman.tif';
 6. parrot_path = 'images/q2/parrot.tif';
 7.
 8. pic_woman = imread(woman_path);
 9. pic_parrot = imread(parrot_path);
١٠.
١١. pic_woman = double(pic_woman);
١٢. pic_parrot = double(pic_parrot);
١٣.
١٤. % Calculate the Discrete Fourier Transform (DFT) of both images
١٥. fft_pic_woman = fft٢(pic_woman);
١٦. fft_pic_parrot = fft٢(pic_parrot);
١٧.
١٨. % Shift zero-frequency component to the cente
١٩. fft_pic_woman=fftshift(fft_pic_woman);
٢٠. fft_pic_parrot=fftshift(fft_pic_parrot);
٢١.
٢٢. % Obtain the magnitude (frequency spectrum) and phase angle of the DFTs
٢٣. mag_woman = abs(fft_pic_woman);
٢٤. phase_woman = angle(fft_pic_woman);
٢٥.
٢٦. mag_parrot = abs(fft_pic_parrot);
٢٧. phase_parrot = angle(fft_pic_parrot);
٢٨.
٢٩. % Plot the results of each image
٣٠. figure(١);
٣١. subplot(١,٣,١), imshow(pic_woman, []), title 'Original pic'
٣٢. subplot(١,٣,٢), imshow(log(double(mag_woman)), []), title 'Magnitude'
٣٣. subplot(١,٣,٣), imshow(phase_woman, []), title 'Phase'
٣٤.
٣٥. figure(٢);
٣٦. subplot(١,٣,١), imshow(pic_parrot,[]), title 'original pic'
٣٧. subplot(١,٣,٢), imshow(log(double(mag_parrot)),[]), title 'Magnitude'
٣٨. subplot(١,٣,٣), imshow(phase_parrot,[]), title 'Phase'
```

نتایج در زیر آورده شده است:

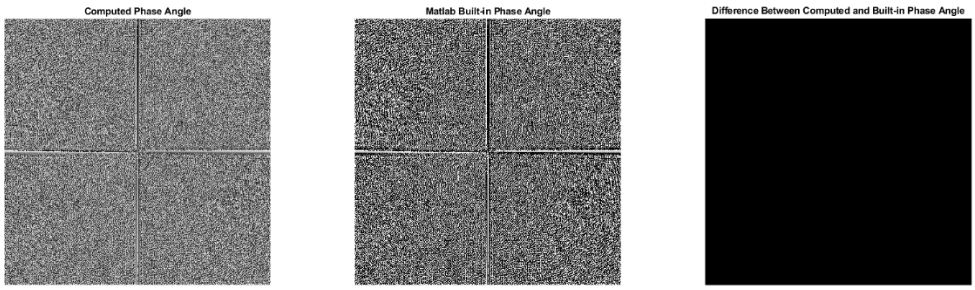


بخش دوم

در این بخش مقدار زاویه تصویر parrot.tif را یکبار با تابع خود متلب و یکبار با فرمول محاسبه به صورت دستی نوشته و مقایسه می کنیم:

```
%% Part b
R_parrot = real(fft_pic_parrot);
I_parrot = imag(fft_pic_parrot);

phi = atan2(I_parrot, R_parrot);
phi_builtin = angle(fft_pic_parrot);
% Display the phase angle
figure(3);
subplot(1,3,1), imshow(phi, []), title('Computed Phase Angle')

% Disply the built-in phase angle
subplot(١,٣,٢), imshow(phi_builtin), title('Matlab Built-in Phase Angle')

% compare the two phase angle
difference = abs(phi - phi_builtin);
subplot(١,٣,٣), imshow(difference, []), title('Difference Between Computed and Built-in
Phase Angle')
```

نتیجه در تصویر زیر آمده است:



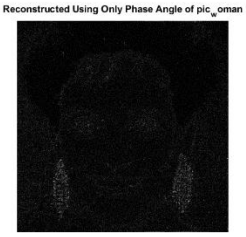
اختلاف بین این دو تابع برابر صفر است که در شکل سمت راست قابل مشاهده است.

بخش سوم

در این بخش با سه روش به ساخت تصویر woman.tifپرداخته شده است. در روش اول باید با استفاده از فاز این تصویر به ساخت تصویر اصلی پرداخته شود. در این حالت مقدار اندازه را برابر یک در نظر میگیریم. در مرحله دوم با استفاده از اندازه و فاز تصویر parrot سعی در ساخت تصویر اصلی داریم. در حالت سوم با استفاده از اندازه ی تصویر parrot و فاز تصویر woman سعی در ساخت تصویر داریم. در این حالت با توجه به تفاوت در ابعاد دو تصویر لازم است تا ابتدا ابعاد دو تصویر برابر شود. با توجه به اینکه تصویر parro دارای سایز ۸۰۰ در ۸۰۰ است و تصویر woman دارای ابعاد ۵۱۲ در ۵۱۲ است ما تصویر woman را به ابعاد ۸۰۰ در ۸۰۰ در آورده این. برای اینکار تصوری ر اصلی را در وسط ماتریس صفر قرار داده و اطراف آنرا صفر در نظر میگیریم. نتایج برای سه حالت اظ طریق کد زیر در ادامه آورده شده است:

```
%% Part c
% Compute the Fourier Transforms
F_woman = fft2(pic_woman);
F_parrot = fft2(pic_parrot);

% Extract magnitude and phase components
magnitude_woman = abs(F_woman);
phase_woman = angle(F_woman);

magnitude_parrot = abs(F_parrot);
```

```
١١. phase_parrot = angle(F_parrot);
١٢.
١٣. % Get sizes
١٤. [rows_w, cols_w] = size(pic_woman);
١٥. [rows_p, cols_p] = size(pic_parrot);
١٦.
١٧. % Create a zero-padded woman image of size ٨٠٠x٨٠٠
١٨. padded_woman = zeros(rows_p, cols_p);
١٩.
٢٠. % Calculate center position to insert the original image
٢١. start_row = floor((rows_p - rows_w) / ٢) + ١;
٢٢. start_col = floor((cols_p - cols_w) / ٢) + ١;
٢٣. end_row = start_row + rows_w - ١;
٢٤. end_col = start_col + cols_w - ١;
٢٥.
٢٦. % Insert the original woman image into the center of the padded image
٢٧. padded_woman(start_row:end_row, start_col:end_col) = pic_woman;
٢٨.
٢٩. % Compute Fourier Transform of the zero-padded woman image
٣٠. F_padded_woman = fft٢(padded_woman);
٣١.
٣٢. % Extract the phase after zero-padding
٣٣. padded_phase_woman = angle(F_padded_woman);
٣٤.
٣٥. % (a) Reconstruction using only the phase angle of the woman image
٣٦. reconstructed_phase_only = ifft٢(exp(١i * phase_woman));
٣٧.
٣٨. % (b) Reconstruction using frequency spectrum + phase of parrot
٣٩. reconstructed_parrot_magnitude_phase = ifft٢(magnitude_parrot .* exp(١i *
phase_parrot));
٤٠.
٤١. % (c) Reconstruction using parrot's magnitude and woman's phase
٤٢. reconstructed_parrot_magnitude_woman_phase = ifft٢(magnitude_parrot .* exp(١i *
padded_phase_woman));
```



مشاهده می شود که با داشتن صرفا زاویه تصویر woman نمیتوان تصویر خوبی از آنرا بازسازی کرد. با داشتن فاز و اندازه ی تصویر parrot میتوان با عکس تبدیل فوریه به تصویر اصلی رسید. از ترکیب فاز تصویر woman و اندازه تصویر parrot نیز تصویر پایین سمت راست تولید شده که تاحدودی اطلاعات تصویر woman بازسازی شده است.

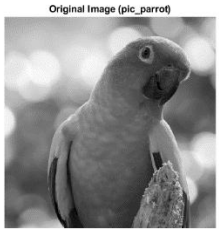
بخش چهارم

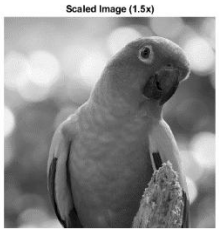
در این بخش ابتدا تصویر parrot با دستور imresize به ابعاد ۱۲۰۰ در ۱۲۰۰ تبدیل شد. برای مقایسه طیف تصویر لازم است تا ابعاد تصاویر برابر بشد. لذا با روش zeropadding یک تصویر ۱۲۰۰ در ۱۲۰۰ ساخته شد که در وسط آن تصویر parrot قرار دارد.

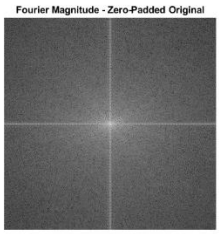
تصویر تغییر سایز داده شده با دستور imresize نیز در زیر آورده شده است:

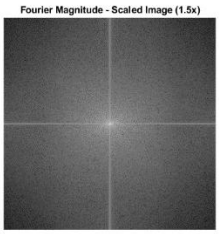
در نهایت از طریق کد زیر به نمایش طیف دو تصویر در دو حالت مختلف پرداخته شده است:

```
%% Part d
% Scale the image by 1.5x
scale_factor = 1.5;
scaled_pic_parrot = imresize(pic_parrot, scale_factor);

% Get new size after scaling
[rows_s, cols_s] = size(scaled_pic_parrot);

% Create a zero-padded version of the original image, centered in the new frame
padded_pic_parrot = zeros(rows_s, cols_s);

start_row = floor((rows_s - rows_p) / ٢) + ١;
start_col = floor((cols_s - cols_p) / ٢) + ١;
end_row = start_row + rows_p - ١;
end_col = start_col + cols_p - ١;

padded_pic_parrot(start_row:end_row, start_col:end_col) = pic_parrot;

% Compute the Fourier Transform of both images
F_original = fft٢(padded_pic_parrot);
F_scaled = fft٢(scaled_pic_parrot);

% Shift for better visualization
F_original_shifted = fftshift(F_original);
F_scaled_shifted = fftshift(F_scaled);

% Compute magnitude spectra (log for better visibility)
magnitude_original = log(١ + abs(F_original_shifted));
magnitude_scaled = log(١ + abs(F_scaled_shifted));
```









با توجه به رابطه زیر بدیهی اســت که وقتی تصــویر را بزرگتر میکنیم در حوزه فرکانس محتویات فرکانس بالا به فرکانسهای پایینتر(مرکز)نزدیکتر می شوند. این باعث می شود که محتوای فرکانس بالا کمتر شده و تصویر نرمتر

شـــود. در حالت عکس اگر تصـــویر کوچکتر شـــود در حوزه فرکانس مقادیر فرکانس پایین به مقادیر فرکانس بالا منتقل می شود و فرکانسهای بالا تقویت می شود و تصویر شارپ تر می شود.

$$f(ax) \longleftrightarrow \frac{1}{|a|}F\left(\frac{u}{a}\right)$$

بخش پنجم

در این بخش به بررسی تاثیر شیفت فرکانسی در جهات مختلف و تاثیر آن بر طیف و زاویه فوریه پرداخته شده است. با توجه به رابطه زیر شیفت در حوزه مکان(زمان) بر طیف تاثیری نمی گذارد ولی فاز را تغییر میدهد.

$$I(x - x_0, y - y_0) \longleftrightarrow F(u, v)e^{-j(2\pi(ux_0+vy_0))}$$

البته حالتهای مختلفی برای شیفت مکانی ممکن است رخ دهد. با توجه به اطلاعات داده شده در صورت سوال، تصویر اصلی دارای ابعاد ۸۰۰ در ۸۰۰ است. حال آنکه شیفت به میزان ۵ درصد به معنی این است که ۴۰ پیکسل به جهت های گفته شده تغییر مکان دهیم. یک استراتژی برای شیفت فرکانسی این است که تصویر اصلی را با روش zeropadding به ابعاد بزرگتری تبدیل کنیم. مثلا فرض کنیم در ۴ جهت تصویر تعداد ۵۰ پیکسل(تعداد پیکسلهای اضافه شده بیشتر از مقدار پیکسلهای شیفت باشد) صفر اضافه کنیم. در این حالت ابعاد تصویر جدید ۹۰۰ در ۹۰۰ شده است. حال اگر در جهتهای مختلف شیفت مکانی داده شود با قطعیت میتوان ثابت کرد که طیف تغییری نمیکند ولی فاز تغییر میکند. استراتژی بعدی این میتواند باشد که شیفت فرکانسی را به صورت چرخشی (circular) انجام دهیم. در این حالت ابعاد تصویر تغییری نمیکند ولی اگر مثلا در جهت محور x از یک سمت چهل پیکسل شیفت بدهیم، در همین راستا و در جهت مخالف همین ۴۰ پیکسل اضافه می شود. استراتژی سوم این است که اگر در یک جهت شیفت مکنی داده شد، در جهت دیگر پیکسل صفر قرار دهیم که این روش با توجه به اینکه اطلاعات تصویر را تغییر میدهد روش مناسبی نیست. در حل این سوال از استراتژی دوم استفاده شده است. از طریق کد زیر ابتدا مقدار شیفت بر حسب پیکسل محاسبه شده و با دستور circshift در سه حالت گفته شده در صورت سوال به شیفت مکانی پرداخته شده است.

```
%% Part e
% Shift parameters (5% of size)
shift_x = round(0.05 * cols_p);
shift_y = round(0.05 * rows_p);

% Apply circular shifting (wrapping around the edges)
shifted_x = circshift(pic_parrot, [0, shift_x]);  % Shift in X direction
shifted_y = circshift(pic_parrot, [shift_y, 0]);  % Shift in Y direction
```

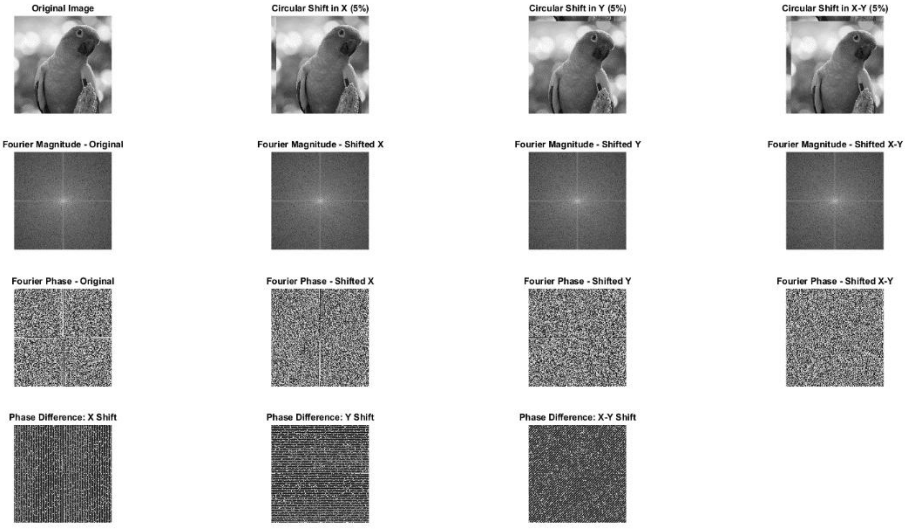
```
9. shifted_xy = circshift(pic_parrot, [shift_y, shift_x]);  % Shift in both X and Y
```

در مرحله بعد به محاسبه فوریه دوبعدی پرداخته شده است:

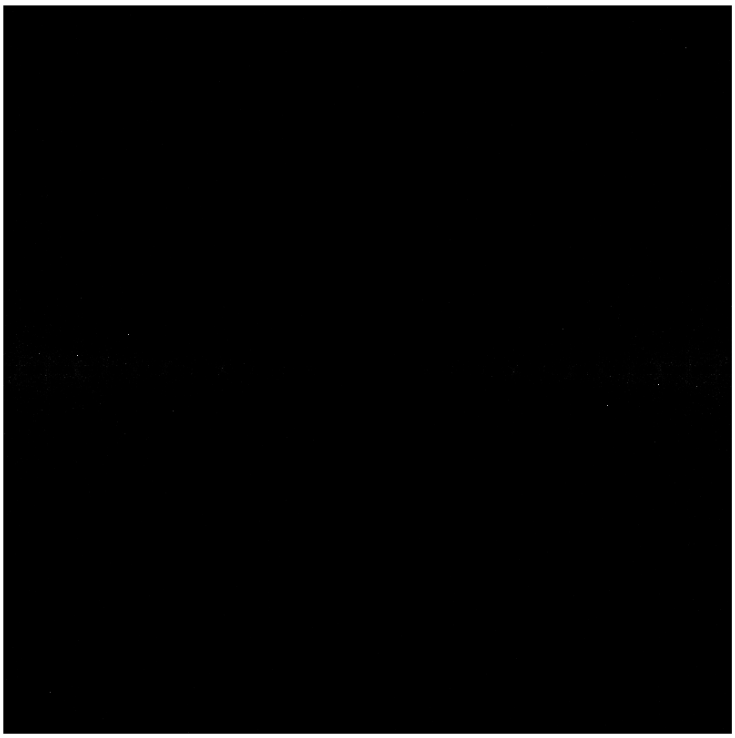
```
1. % Compute Fourier Transform for all images
2. F_original = fft2(pic_parrot);
3. F_shifted_x = fft2(shifted_x);
4. F_shifted_y = fft2(shifted_y);
5. F_shifted_xy = fft2(shifted_xy);
6.
7. % Shift for visualization
8. F_original_shifted = fftshift(F_original);
9. F_shifted_x_shifted = fftshift(F_shifted_x);
١٠. F_shifted_y_shifted = fftshift(F_shifted_y);
١١. F_shifted_xy_shifted = fftshift(F_shifted_xy);
```

در انتها نیز مقادیر طیف و زاویه محاسبه شده و به نمایش گزاشته شده است:

```
1. % Compute magnitude spectra
2. magnitude_original = log(1 + abs(F_original_shifted));
3. magnitude_x = log(1 + abs(F_shifted_x_shifted));
4. magnitude_y = log(1 + abs(F_shifted_y_shifted));
5. magnitude_xy = log(1 + abs(F_shifted_xy_shifted));
6.
7. % Compute phase spectra
8. phase_original = angle(F_original_shifted);
9. phase_x = angle(F_shifted_x_shifted);
١٠. phase_y = angle(F_shifted_y_shifted);
١١. phase_xy = angle(F_shifted_xy_shifted);
```



با توجه به نکات گفته شده در ابتدای توضیحات وقتی عملیات شیفت را به صورت چرخشی انجام می دهیم اگرچه ابعاد تصویر ثابت باقی می ماند ولی تغییراتی در هم فاز و هم طیف ایجاد می شود. البته لازم به ذکر است که تغییرات در طیف بسیار کم است که در زیر اختلاف بین طیف تصویر اصلی و تصویری که به اندازه ۵ درصد در جهت محور x شیفت پیدا کرده را خواهید دید:



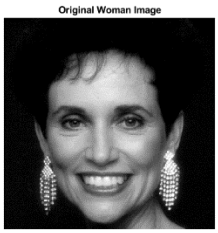
همانطور که در تصویر بالا میبینید اختلاف بین این دو طیف بسیار کم و در حد صفر است.

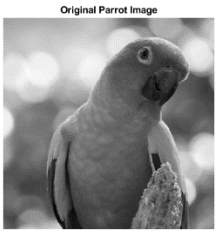
بخش ششم

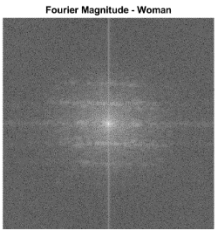
در این بخش، هدف بررسی تأثیر مؤلفه‌های پایین‌فرکانس (Low-Frequency Components) بر ساختار کلی تصویر است. برای این منظور، ابتدا تصاویر woman و parrot بارگذاری و بررسی شد که اندازه‌ی آن‌ها برابر باشد. در صورت نیاز، تصویر طوطی به ابعاد تصویر زن تغییر اندازه داده شد تا قابلیت مقایسه و ترکیب داشته باشند. سپس هر دو تصویر به نوع double تبدیل شده و با استفاده از تبدیل فوریه دوبعدی به فضای فرکانسی انتقال یافتند. همچنین با اعمال fftshift، فرکانس‌های پایین به مرکز طیف منتقل شدند. در مرحله بعد، یک ناحیه مربعی با اندازه ۱۲۸×۱۲۸ پیکسل در مرکز طیف فرکانسی مشخص شد که نمایانگر مؤلفه‌های پایین‌فرکانس تصویر است.

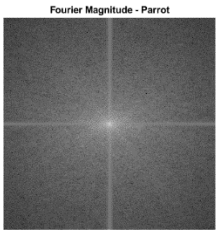
این ناحیه از تصویر طوطی گرفته شده و جایگزین مؤلفه‌های متناظر در تصویر زن شد. این جایگزینی باعث می‌شود که ساختار کلی و محتوای پایه‌ای تصویر (مانند نورپردازی و ترکیب کلی سطوح خاکستری) از تصویر طوطی گرفته شود، در حالی که جزئیات دقیق‌تر تصویر زن حفظ می‌گردند. سپس با استفاده از ifftshift و ifft۲، تصویر ترکیب‌شده به فضای مکانی بازگردانده شد و بخش حقیقی آن نمایش داده شد. در نهایت، شش تصویر در کنار هم نمایش داده شدند که شامل تصویر اصلی زن، تصویر اصلی طوطی، طیف فرکانسی هر دو تصویر، طیف ترکیبی، و تصویر بازسازی‌شده نهایی است.

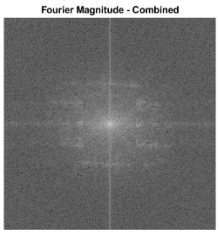
```
%% Part f
% Ensure both images have the same size
[rows_w, cols_w] = size(pic_woman);
[rows_p, cols_p] = size(pic_parrot);

if rows_w ~= rows_p || cols_w ~= cols_p
    % Resize the parrot image to match woman image size
    pic_parrot = imresize(pic_parrot, [rows_w, cols_w]);
end

% Convert to double
pic_woman = double(pic_woman);
pic_parrot = double(pic_parrot);

% Compute the Fourier Transform of both images
F_woman = fft۲(pic_woman);
F_parrot = fft۲(pic_parrot);

% Shift zero frequency components to the center
F_woman_shifted = fftshift(F_woman);
F_parrot_shifted = fftshift(F_parrot);

center_x = floor(rows_w/۲);
center_y = floor(cols_w/۲);
patch_size = ۱۲۸;
half_patch = floor(patch_size/۲);

% Replace lower half frequency components
F_combined = F_woman_shifted;
F_combined(center_x-half_patch:center_x+half_patch, center_y-
half_patch:center_y+half_patch) = F_parrot_shifted(center_x-half_patch:center_x+half_patch,
center_y-half_patch:center_y+half_patch);

% Shift back the modified Fourier Transform
F_combined_unshifted = ifftshift(F_combined);

% Compute the inverse Fourier Transform
reconstructed_image = ifft۲(F_combined_unshifted);
reconstructed_image = abs(reconstructed_image);
```

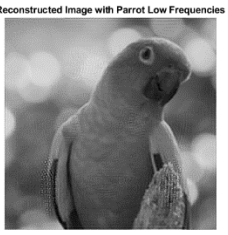










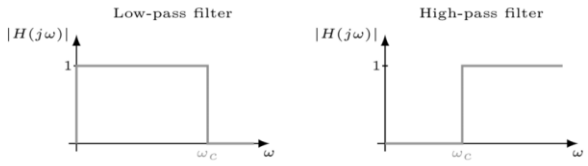


نتایج نشان می‌دهند که مؤلفه‌های پایین‌فرکانس به شدت بر ساختار کلی، روشنایی، و ترکیب کلی تصویر تأثیرگذار هستند. با وجود اینکه لبه‌ها و جزئیات دقیق همچنان از تصویر زن گرفته شده‌اند، اما ظاهر کلی تصویر بازسازی‌شده تحت تأثیر تصویر طوطی قرار گرفته است.

سوال ۳

در این سوال به تعریف تابعی می پردازیم که تصویر و نوع فیلتر و آرگومانهای آنرا گرفته و در نهایت تصویر فیلتر شده را به نمایش می گذارد. در ابتدا لازم است تا به نکات زیر توجه شود. فیلتر بالا گذر دستگاهی است که فرکانس‌های بالاتر از مقدار خاصی را می‌گذراند و فرکانس‌های پایین‌تر از آن را عبور نمی‌دهد (تضعیف می‌کند). در مقابل فیلتر پایین گذر نوعی فیلتر است که سیگنال‌های با بسامدی کمتر از یک بسامد مشخص را عبور می‌دهد.

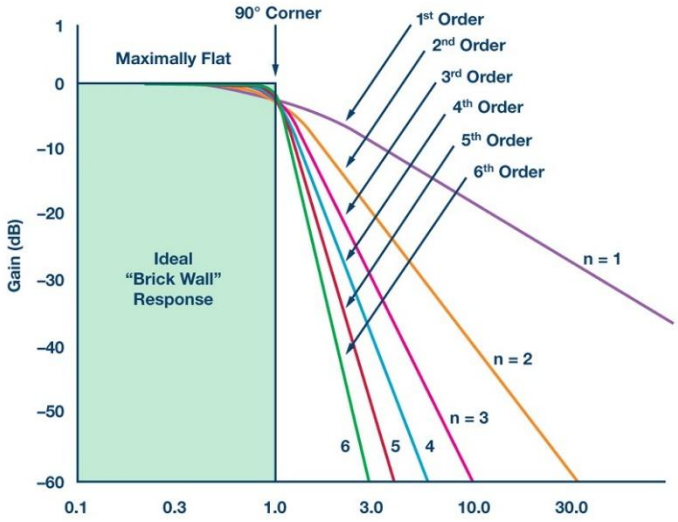
فیلتر ایده آل



رابطه تابع تبدیل این فیلتر(بالاگذر) به صورت زیر است:

$$H(u,v)=\begin{cases}0 & if \quad D(u,v)\le D_\circ\\ 1 & if \quad D(u,v)>D_\circ\end{cases}$$

فیلتر Butterworth



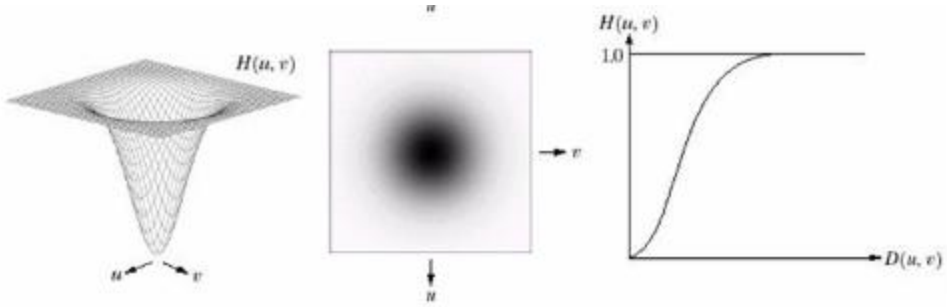
رابطه تابع تبدیل این فیلتر(بالاگذر) به قرار زیر است:

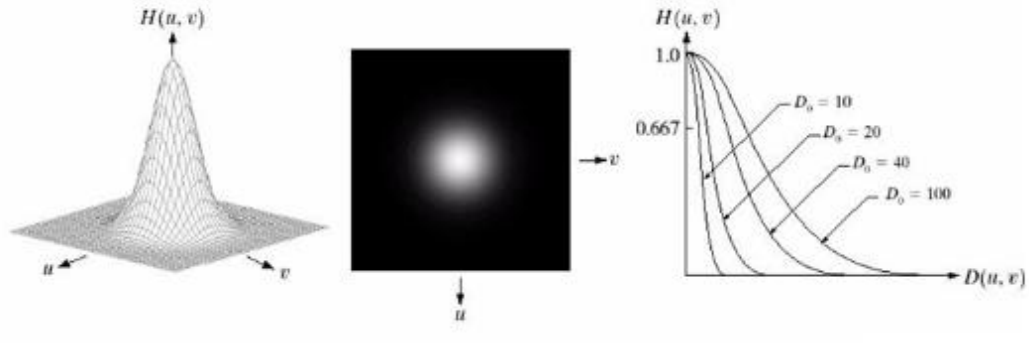
$$H(u,v)=\frac{1}{1+\left[D_\circ / D(u,v)\right]^{2n}}$$

فیلتر Gaussian

رابطه تابع تبدیل این فیلتر(بالاگذر) به قرار زیر است:

$$H(u,v)=1-e^{-D^2(u,v)/2D_\circ{}^2}$$





کد زیر برای پیاده سازی تابع استفاده شده است:

```
 1. function [filtered_image, filter_response] = filter_function(image, filter_type, kernel_type,
D0, n)
 2.     % Convert image to grayscale and double precision if necessary
 3.     if size(image, 3) == 3
 4.         image = rgb2gray(image);
 5.     end
 6.     image = double(image);
 7.
 8.     % Get image size
 9.     [rows, cols] = size(image);
١٠.      u = -floor(rows/٢):floor(rows/٢)-١;
١١.      v = -floor(cols/٢):floor(cols/٢)-١;
١٢.      [U, V] = meshgrid(v, u);
١٣.      D = sqrt(U.^٢ + V.^٢); % Distance from the center
١٤.
١٥.      % Select the appropriate filter
١٦.      switch lower(kernel_type)
١٧.          case 'ideal'
١٨.              H = double(D <= D٠);
١٩.          case 'butterworth'
٢٠.              H = ١ ./ (١ + (D ./ D٠).^(٢*n)); % Butterworth filter of order n
٢١.          case 'gaussian'
٢٢.              H = exp(-(D.^٢) / (٢ * D٠^٢)); % Gaussian LPF
٢٣.          otherwise
٢٤.              error('Invalid kernel type. Choose "Ideal", "Butterworth", or
"Gaussian".');
٢٥.      end
٢٦.
٢٧.      % Convert Low-Pass to High-Pass if needed
٢٨.      if strcmpi(filter_type, 'hpf')
٢٩.          H = ١ - H; % HPF is complement of LPF
٣٠.      elseif ~strcmpi(filter_type, 'lpf')
```

```
        error('Invalid filter type. Choose "LPF" (Low-Pass Filter) or "HPF" (High-Pass Filter).');
    end

    % Compute Fourier Transform of the image
    F_image = fft2(image);
    F_shifted = fftshift(F_image);

    % Apply filter in the frequency domain
    F_filtered = F_shifted .* H;

    % Compute the inverse Fourier Transform
    F_unshifted = ifftshift(F_filtered);
    filtered_image = real(ifft2(F_unshifted));

    % Return the filter response for visualization
    filter_response = H;
end
```

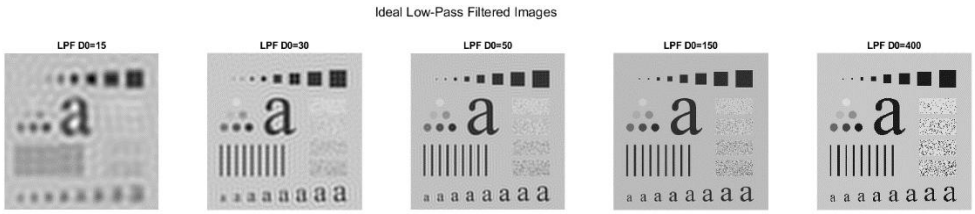
با توجه به صورت سوال اقدام به اعمال فیلتر های مختلف روی تصویر داده شده میکنیم:

در بخش اول فیلترهای ایده آل پایین گذر و بالا گذر را برای مقادیر فرکانس قطع مختلف بر روی تصویر داده شده اعمال کرده و نتایج را نمایش میدهیم. کد زیر مربوط به پیاده سازی این مرحله است:

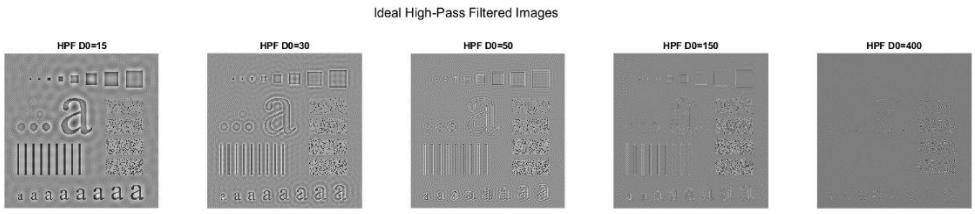
```
% Read the image
image_path = 'images/q3/char.tif';
char_img = imread(image_path);

% Define cutoff frequencies
D0_values = [15, 30, 50, 150, 400];
n = 2; % Butterworth order

%% Ideal filter
% Initialize figures
figure;
sgtitle('Ideal Low-Pass Filtered Images');

figure;
sgtitle('Ideal High-Pass Filtered Images');

figure;
sgtitle('Frequency Responses - Ideal Filters');

% Apply Ideal Filters
for i = 1:length(D0_values)
    D0 = D0_values(i);

    % Apply Ideal Low-Pass Filter
    [filtered_lpf, response_lpf] = filter_function(char_img, 'lpf', 'ideal', D0, n);

    % Apply Ideal High-Pass Filter
    [filtered_hpf, response_hpf] = filter_function(char_img, 'hpf', 'ideal', D0, n);

    % Display LPF results
    figure(1);
    subplot(2, length(D0_values), i);
    imshow(filtered_lpf, []);
    title(['LPF D0=', num2str(D0)]);

```

```
    % Display HPF results
    figure(2);
    subplot(2, length(D0_values), i);
    imshow(filtered_hpf, []);
    title(['HPF D0=', num2str(D0)]);

    % Display frequency response
    figure(3);
    subplot(2, length(D0_values), i);
    imshow(response_lpf, []);
    title(['LPF D0=', num2str(D0)]);

    subplot(2, length(D0_values), i + length(D0_values));
    imshow(response_hpf, []);
    title(['HPF D0=', num2str(D0)]);
end
```

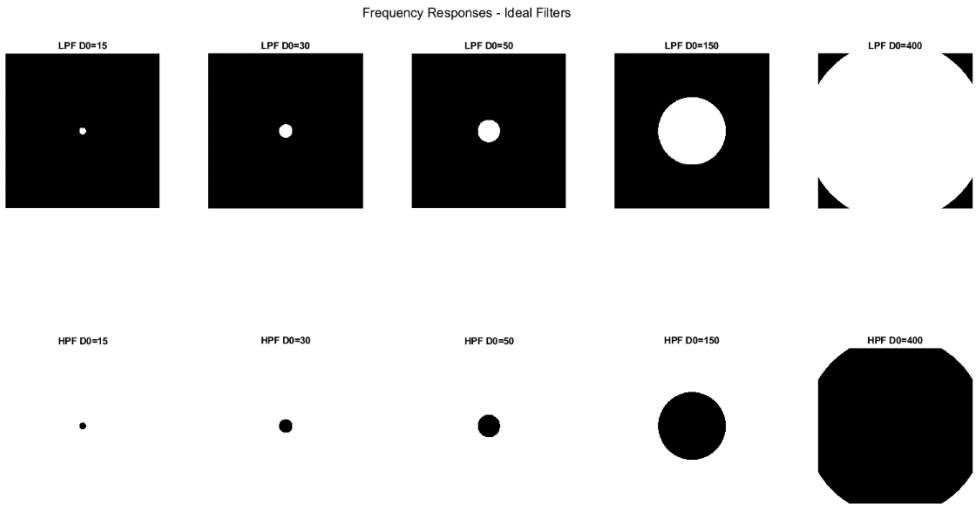
نتایج اعمال فیلتر پایین گذر ایده آل به صورت زیر است:



نتایج اعمال فیلتر بالا گذر ایده آل نیر به قرار زیر است:



و در نهایت پاسخ فرکانسی هر فیلتر(ماسک مورد استفاده) به قرار زیر است:

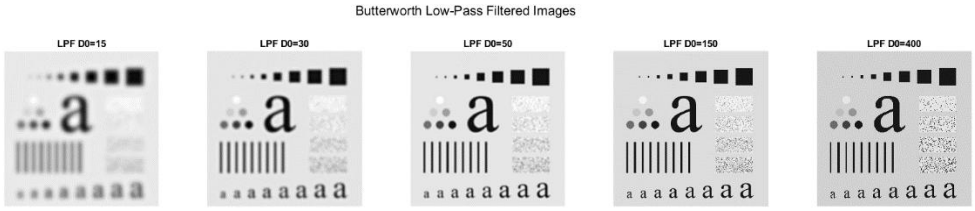


همین مراحل را یکبار دیگر برای فیلتر باتروورث انجام می دهیم:

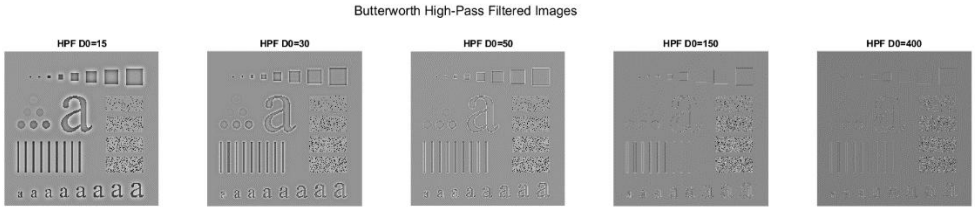
```
%% Butterworth Filter
figure;
sgtitle('Butterworth Low-Pass Filtered Images');

figure;
sgtitle('Butterworth High-Pass Filtered Images');

figure;
sgtitle('Frequency Responses - Butterworth Filters');

for i = 1:length(D0_values)
    D0 = D0_values(i);

    % Apply Butterworth Low-Pass Filter
    [filtered_lpf, response_lpf] = filter_function(char_img, 'lpf', 'butterworth', D0, n);

    % Apply Butterworth High-Pass Filter
    [filtered_hpf, response_hpf] = filter_function(char_img, 'hpf', 'butterworth', D0, n);

    % Display LPF results
    figure(3);
    subplot(2, length(D0_values), i);
    imshow(filtered_lpf, []);
    title(['LPF D0=', num2str(D0)]);

    % Display HPF results
    figure(4);
    subplot(2, length(D0_values), i);
    imshow(filtered_hpf, []);
    title(['HPF D0=', num2str(D0)]);

    % Display frequency response
    figure(5);
    subplot(2, length(D0_values), i);
    imshow(response_lpf, []);
    title(['LPF D0=', num2str(D0)]);

```

```
    subplot(۲, length(D۰_values), i + length(D۰_values));
    imshow(response_hpf, []);
    title(['HPF D۰=', num۲str(D۰)]);
end
```

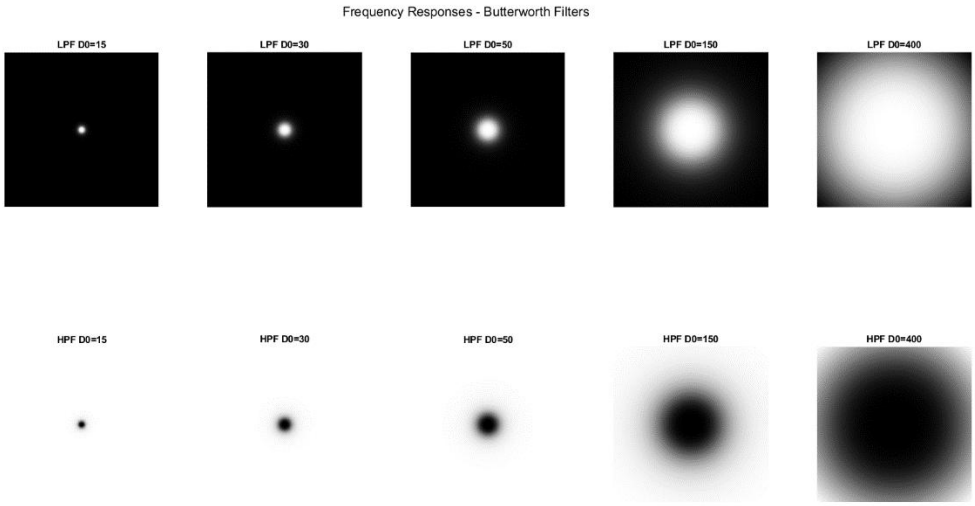
نتایج حاصل از اعمال فیلتر پایین گذر باتروورث به قرار زیر است:



نتایج برای فیلتر بالاگذر از همین نوع نیز در زیر آورده شده است:



و در نهایت پاسخ فرکانسی برای هر فیلتر در تصویر زیر قابل نمایش است:

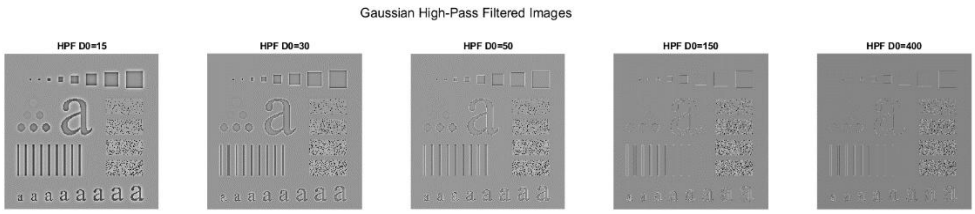


در آخر همین مراحل را برای فیلتر گوسی تکرار می کنیم.

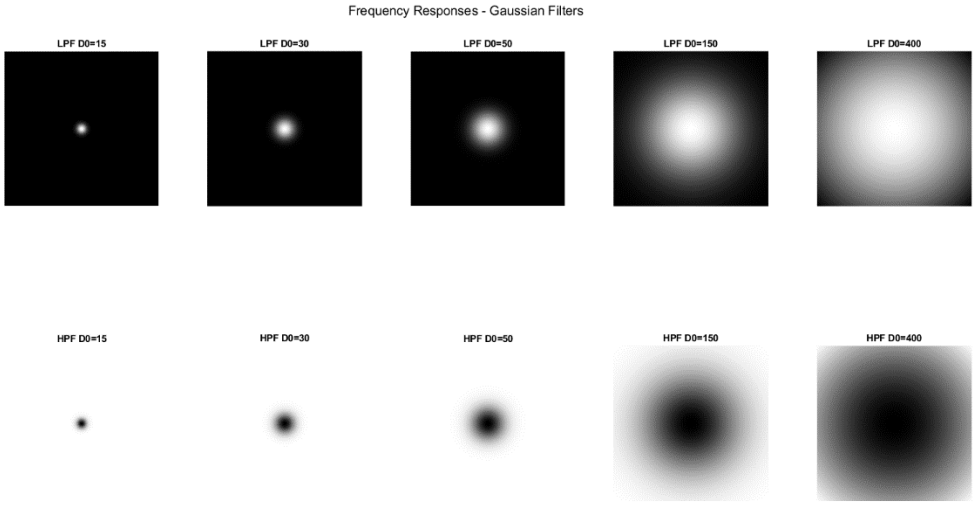
```
%% Gaussian Filter
figure;
sgtitle('Gaussian Low-Pass Filtered Images');

figure;
sgtitle('Gaussian High-Pass Filtered Images');

figure;
sgtitle('Frequency Responses - Gaussian Filters');

for i = 1:length(D0_values)
    D0 = D0_values(i);

    % Apply Gaussian Low-Pass Filter
    [filtered_lpf, response_lpf] = filter_function(char_img, 'lpf', 'gaussian', D0, n);

    % Apply Gaussian High-Pass Filter
    [filtered_hpf, response_hpf] = filter_function(char_img, 'hpf', 'gaussian', D0, n);

    % Display LPF results
    figure(7);
    subplot(2, length(D0_values), i);
    imshow(filtered_lpf, []);
    title(['LPF D0=', num2str(D0)]);

    % Display HPF results
    figure(8);
    subplot(2, length(D0_values), i);
    imshow(filtered_hpf, []);
    title(['HPF D0=', num2str(D0)]);

    % Display frequency response
    figure(9);
    subplot(2, length(D0_values), i);
    imshow(response_lpf, []);
    title(['LPF D0=', num2str(D0)]);

    subplot(2, length(D0_values), i + length(D0_values));
    imshow(response_hpf, []);
    title(['HPF D0=', num2str(D0)]);
end
```

نتایج برای فیلتر پایین گذر به قرار زیر است:

برای فیلتر بالاگذر گوسی داریم:



و در نهایت پاسخ فرکانسی برای فیلترهای اعمالی به قرار زیر است:



سوال ۴

بخش اول

در این بخش، هدف بهبود وضوح تصویر اشعه ایکس قفسه سینه با استفاده از فیلتر تأکید بر فرکانس‌های بالا است. این نوع فیلتر برای تقویت جزئیات و لبه‌هایی که به دلیل تاری ناشی از محدودیت‌های نوری اشعه ایکس از بین رفته‌اند، بسیار کاربردی است. در ابتدا تصویر **chest.tif** بارگذاری و بررسی شد که در صورت رنگی بودن، به تصویر خاکستری تبدیل شود. سپس تصویر برای پردازش به نوع **double** تبدیل شد تا در محاسبات فوریه دقت کافی وجود داشته باشد. پس از آن، با استفاده از **fft۲** تبدیل فوریه دوبعدی روی تصویر اعمال شد و برای قرار گرفتن مؤلفه‌های کم‌فرکانس در مرکز، از **fftshift** استفاده شد.

در گام بعد، یک فیلتر بالاگذر گاوسی طراحی شد. ابتدا شبکه فرکانسی با مختصات **u** و **v** ایجاد شد و با انتقال مرکز آن‌ها به صفر، فاصله فرکانسی **D** برای هر نقطه محاسبه گردید. سپس با استفاده از این فاصله، فیلتر بالاگذر گاوسی ساخته شد که مؤلفه‌های پایین‌فرکانس را تضعیف کرده و فرکانس‌های بالا را حفظ می‌کند. سپس با تعریف ضرایب **k۱ = ۰,۵** و **k۲ = ۰,۷۵**، فیلتر تأکید بر فرکانس بالا ایجاد شد که ترکیبی از تصویر اصلی و فیلتر بالاگذر گاوسی است. این ترکیب باعث می‌شود نه‌تنها لبه‌ها تقویت شوند، بلکه جزئیات نیز به طور قابل کنترل حفظ یا برجسته شوند. در ادامه، با استفاده از **ifftshift** و سپس **ifft۲** تصویر از حوزه فرکانس به حوزه مکان بازگردانده شد و فقط بخش حقیقی آن برای نمایش استفاده گردید.

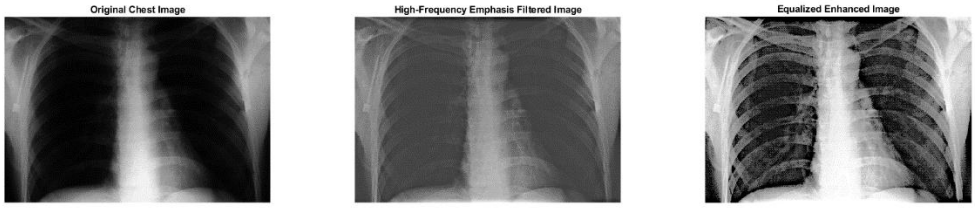
```
%% Part a
% Load the chest image
chest_img = imread('images/q4/chest.tif');

% Convert to grayscale if the image is not already
if size(chest_img, 3) == 3
    chest_img = rgb2gray(chest_img);
end

%  Convert the image to double for processing
chest_img = double(chest_img);

% Apply Fourier Transform to the image
F = fft٢(chest_img);

% Shift the zero-frequency component to the center
F_shifted = fftshift(F);

% Create the Gaussian high-pass filter
[M, N] = size(chest_img);
[u, v] = meshgrid(٠:N-١, ٠:M-١);
u = u - floor(N/٢);
v = v - floor(M/٢);

% Compute the frequency distance
D = sqrt(u.^٢ + v.^٢);

% Set the standard deviation (sigma) for the Gaussian filter
sigma = ١٠;

% Gaussian High-pass filter
H_hp = ١ - exp(- (D.^٢) / (٢ * sigma^٢));

% Step ٦: Define the constants k١ and k٢
k١ = ٠,٥;
k٢ = ٠,٧٥;

% High-Frequency Emphasis Filtering
G_shifted = (k١ + k٢ * H_hp) .* F_shifted;

% Inverse Fourier Transform to get the filtered image
G = ifftshift(G_shifted);  % Reverse the shift before performing inverse FFT
g = real(ifft٢(G));

% Display the results
figure;
subplot(١, ٣, ١);
imshow(chest_img, []);
```

```
title('Original Chest Image');

subplot(۱, ۳, ۲);
imshow(g, []);
title('High-Frequency Emphasis Filtered Image');
```

بخش دوم

در این بخش، هدف بهبود بیشتر تصویر فیلتر شده با استفاده از همسان‌سازی هیستوگرام **(HE)** است. همان‌طور که در بخش قبل مشاهده شد، فیلتر تأکید بر فرکانس‌های بالا توانست جزئیات و لبه‌های تصویر را تقویت کند، اما مقدار شدت روشنایی تصویر حاصل معمولاً در یک بازه‌ی محدود و نسبتاً باریک متمرکز می‌شود. این تمرکز باعث می‌شود تصویر نهایی کنتراست ضعیفی داشته باشد و بخشی از اطلاعات بصری آن به‌خوبی قابل مشاهده نباشد . برای رفع این مشکل، از تکنیک همسان‌سازی هیستوگرام استفاده شده است که با گسترش محدوده‌ی شدت روشنایی و توزیع یکنواخت‌تر سطوح خاکستری، کنتراست تصویر را بهبود می‌بخشد. در این کد، ابتدا تصویر حاصل از **ifft۲(G)** به نوع **uint۸** تبدیل شده تا قابل استفاده در تابع **histeq** باشد. سپس خروجی حاصل، به‌عنوان تصویر نهایی همسان‌سازی‌شده نمایش داده شد.

با مشاهده نتیجه، می‌توان دید که تصویر نهایی نسبت به تصویر اولیه‌ی فیلتر شده، وضوح و کنتراست بیشتری دارد و جزئیات آن به‌مراتب بهتر دیده می‌شوند. بنابراین ترکیب فیلتر تأکید بر فرکانس بالا و همسان‌سازی هیستوگرام یک روش مؤثر برای افزایش کیفیت و وضوح تصاویر پزشکی مانند اشعه ایکس محسوب می‌شود.

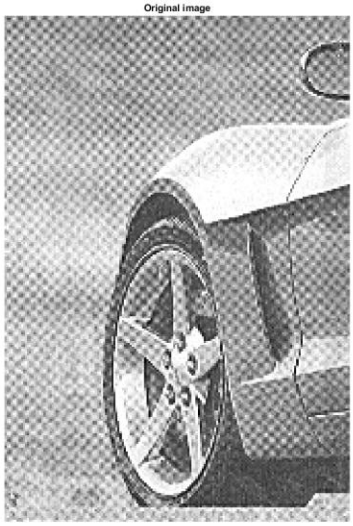


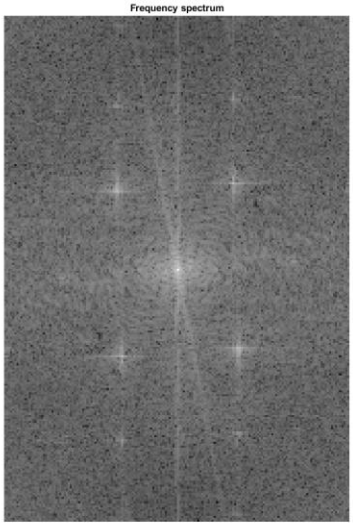
سوال ۵

بخش اول

ابتدا برای نمایش تصویر اصلی و طیف آن از دستور زیر استفاده شده است:

```
%% Part a
% Load the image
image_path = 'images/q5/carmoire.tif';
carmoire_img = imread(image_path);

% convert to grayscale
if size(carmoire_img, 3) == 3
    carmoire_img = rgb2gray(carmoire_img);
end

% compute FT
F = fft٢(carmoire_img);
F_shifted = fftshift(F);

figure(١);
subplot(١, ٢, ١);
imshow(carmoire_img, []);
title('Original image');

magnitude_spectrum = log(١ + abs(F_shifted));
subplot(١, ٢, ٢);
imshow(magnitude_spectrum, []);
title('Frequency spectrum');
```





بخش دوم

الگوی موآره یک آرتیفکت بصری ناخواسته است که زمانی رخ می‌دهد که دو الگوی منظم و متناوب روی هم قرار بگیرند و باعث ایجاد امواج یا اعوجاج‌های دوره‌ای در تصویر شوند. این الگو می‌تواند وضوح تصویر را کاهش دهد و یک اثر تداخل ناخواسته ایجاد کند. برای حذف این نویز از فیلتر ناچ استفاده شده است. ابتدا مختصات نقاط را از روی تصویر پیدا کرده و بعد به اعمال فیلتر ناچ میپردازیم. با توجه به اینکه تبدیل فوریه سینوسی محض که یک تابع متناوب است، جفتی از ضربه های متقارن و مزدوج است لذا در تصویر بالا سمت راست(طیف) ۸ نقطه روشن می بینیم. لذا در ادامه به حذف آنها می پردازیم:

ابتدا تابعی برای اعمال فیلتر ناچ می نویسیم:

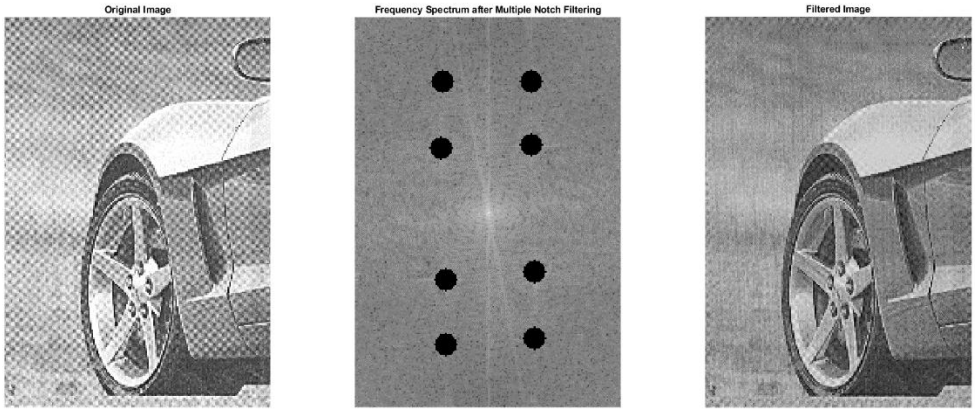
```
function filtered_F = apply_multiple_notch_filters(F_shifted, F_size, notches, radius)
    cols = F_size(1);
    rows = F_size(2);
    notch_filter = ones(cols, rows);
    for i = 1:size(notches, 1)
        notch_u = notches(i, 1);
        notch_v = notches(i, 2);

        for u = 1:cols
             for v = ١:rows
                 if (u - notch_u)^٢ + (v - notch_v)^٢ <= radius^٢
                     notch_filter(u, v) = ٠;
                 end
             end
         end
     end

     filtered_F = F_shifted .* notch_filter;
end
```

این تابع طیف تصویر و ابعاد آن و یک لیست از نقاطی که قرار است فیلتر اعمال بشود و شعاع مسایگی فیلتر را گرفته و بعد فیلتر را بر تصویر اعمال میکند. نقاط مد نظر به قرار زیر است:

```
١١٢   ٤١
١١٢   ٨١
 56   41
 55   83
 58  166
١١٤  ١٦١
 58  207
١١٤  ٢٠٣
```

```
%% Part b
notch_points = [41, 112; 81, 112; 41, 56; 83, 55; 166, 58;161, 114; 207, 58;203, 114];
radius = 7;

```

```
 5. notch_filtered_F = apply_multiple_notch_filters(F_shifted, size(F_shifted), notch_points,
radius);
 6.
 7. % Inverse Fourier Transform to obtain the filtered image
 8. F_ift = ifftshift(notch_filtered_F); % Shift back
 9. filtered_image = ifft2(F_ift); % Inverse FFT to get the image in spatial domain
١٠.
١١. figure;
١٢. subplot(١, ٣, ١);
١٣. imshow(carmoire_img, []);
١٤. title('Original Image');
١٥.
١٦. subplot(١, ٣, ٢);
١٧. imshow(log(١ + abs(notch_filtered_F)), []);
١٨. title('Frequency Spectrum after Multiple Notch Filtering');
١٩.
٢٠. subplot(١, ٣, ٣);
٢١. imshow(abs(filtered_image), []);
٢٢. title('Filtered Image');
```



در ابتدا، تصویر اصلی دارای الگوی موآره است که به‌صورت موج‌های ناخواسته در برخی قسمت‌های تصویر، به‌خصوص در اطراف چرخ خودرو، دیده می‌شود. این الگو باعث کاهش وضوح تصویر شده و یک اعوجاج نامطلوب ایجاد کرده است. سپس، طیف فرکانسی تصویر پس از اعمال فیلتر ناچ نمایش داده شده که در آن، برخی نقاط سیاه‌رنگ مشاهده می‌شوند. این نقاط نشان‌دهنده حذف فرکانس‌هایی هستند که در ایجاد نویز موآره نقش داشته‌اند. این حذف مؤثر باعث کاهش تأثیر نویز در تصویر نهایی می‌شود. در نهایت، تصویر فیلتر شده نشان داده شده که در آن، الگوی موآره به‌طور چشمگیری کاهش یافته و تصویر واضح‌تر از قبل شده است. در این تصویر، جزئیات اطراف چرخ خودرو بهتر دیده می‌شوند و سایر بخش‌های تصویر، مانند بدنه خودرو و پس‌زمینه، تقریباً

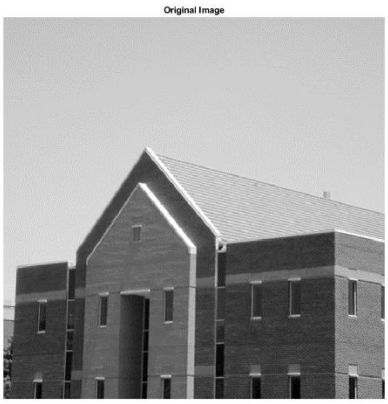
بدون تغییر باقی مانده‌اند. البته ممکن است کمی تاری در لبه‌های تصویر ایجاد شده باشد، اما در مجموع، فیلتر ناچ عملکرد مؤثری در حذف نویز موآره داشته است. این روش باعث بهبود کیفیت تصویر شده و برای کاربردهایی مانند بازیابی تصویر و کنترل کیفیت در پردازش تصویر دیجیتال بسیار مفید است.

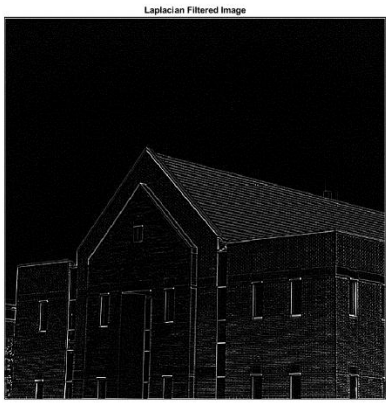
سوال ۶

بخش اول

در این بخش، هدف اعمال فیلتر لاپلاسین روی تصویر ورودی به منظور برجسته‌سازی لبه‌ها و ساختارهای پرجزئیات است. ابتدا تصویر **bld.tif** بارگذاری شد و بررسی گردید که در صورت رنگی بودن، به سطح خاکستری تبدیل شود تا پردازش روی یک کانال انجام گیرد. سپس یک کرنل لاپلاسین با ابعاد ۳×۳ تعریف شد که با مقادیر مشخص شده قادر است لبه‌های تصویر را با دقت بالا استخراج کند. به جای استفاده از تابع **imfilter**، از تابع **conv۲** برای اعمال کانولوشن دوبعدی بین کرنل و تصویر استفاده شد. از آنجایی که فیلتر لاپلاسین می‌تواند مقادیری خارج از محدوده مجاز تصویر (۰ تا ۲۵۵) تولید کند، مقادیر خروجی با استفاده از تابع **min** و **max** در این بازه بریده شده و به نوع داده‌ای **uint۸** تبدیل شد تا نمایش صحیحی از تصویر فراهم گردد. در نهایت، تصویر اصلی و تصویر فیلترشده به‌صورت کنار هم نمایش داده شدند تا تأثیر فیلتر در تقویت لبه‌ها به‌وضوح قابل مشاهده باشد. این فرآیند نشان می‌دهد که فیلتر لاپلاسین با تأکید بر تغییرات ناگهانی شدت روشنایی، برای آشکارسازی مرزها و ساختارهای پرجزئیات در تصویر بسیار مؤثر است. کد مربوط به این بخش در زیر آمده است:

```
1. %% Part a
2. % Load the image
3. img = imread('images/q6/bld.tif');
4. if size(img, 3) == 3  % Convert only if the image is RGB
5.     img = rgb2gray(img);
6. end
7.
8. % Define the Laplacian kernel
9. laplacian_kernel = [-1 -1 -1; -1 8 -1; -1 -1 -1];
١٠.
١١. % Apply the Laplacian filter using convolution
١٢. %filtered_img = imfilter(double(img), laplacian_kernel, 'replicate');
١٣. filtered_img = conv٢(laplacian_kernel, img);
١٤. filtered_img_clip = uint٨(min(max(filtered_img, ٠), ٢٥٥));
١٥.
١٦. % Display the original and filtered images
١٧. figure(١);
١٨. subplot(١,٢,١); imshow(img); title('Original Image');
١٩. subplot(١,٢,٢); imshow(filtered_img_clip, []); title('Laplacian Filtered Image');
```



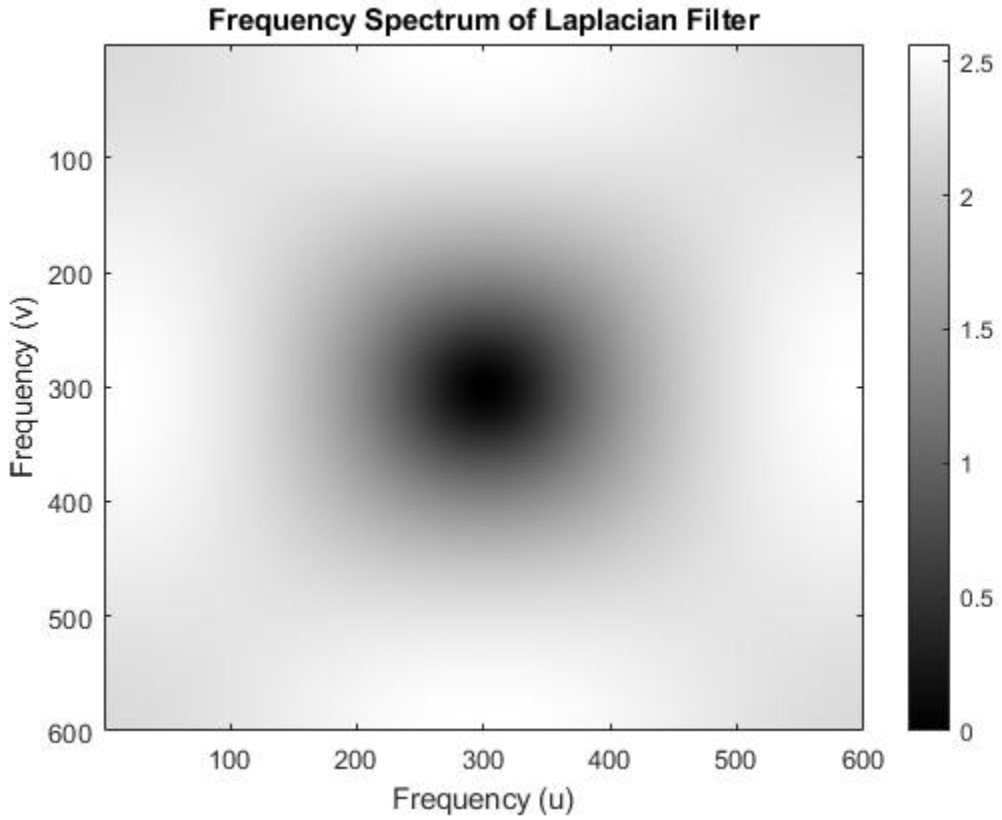


بخش دوم

ابتدا تبدیل فوریه دوبعدی برای کرنل لاپلاسین انجام شد. برای اینکه بتوان تابع تبدیل را در تبدیل فوریه تصویر ضرب کرد، تبدیل فوریه با اندازه بزرگ‌تر (۶۰۰×۶۰۰) انجام شد. سپس با استفاده از تابع **fftshift**، مؤلفه‌های فرکانسی با فرکانس صفر به مرکز منتقل شدند، تا نمایش طیف به‌صورت متقارن و قابل درک‌تر باشد. از آنجایی که مقدارهای موجود در فضای فرکانس دامنه‌ی وسیعی دارند و مقادیر بالا می‌توانند مقادیر کوچکتر را تحت‌الشعاع قرار دهند، از تبدیل لگاریتمی برای نمایش بهتر جزئیات استفاده شد. در نهایت، با استفاده از تابع **imagesc** طیف فرکانسی به‌صورت تصویری نمایش داده شد. از این تحلیل مشخص می‌شود که فیلتر لاپلاسین یک فیلتر **high-pass** است که مؤلفه‌های فرکانسی بالا را تقویت می‌کند و مؤلفه‌های پایین‌تر (مربوط به نواحی صاف تصویر) را تضعیف می‌نماید.

```
%% Part b
% Compute the 2D Fourier Transform of the kernel
fft_kernel = fft2(laplacian_kernel, 600, 600);

% Shift the zero-frequency component to the center
fft_shifted = fftshift(abs(fft_kernel));

% Convert to logarithmic scale for better visualization
fft_magnitude = log(1 + fft_shifted);

% Plot the frequency spectrum
figure(٢);
imagesc(fft_magnitude);
colormap('gray'); colorbar;
title('Frequency Spectrum of Laplacian Filter');
```

```
١٦. xlabel('Frequency (u)');
١٧. ylabel('Frequency (v)');
```

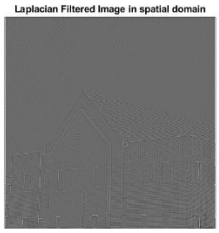


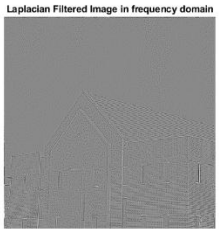
بخش سوم

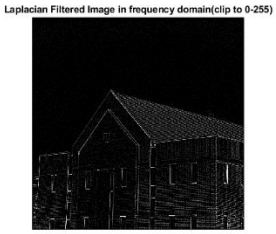
در این بخش، فیلتر لاپلاسین به‌جای اعمال مستقیم در حوزه مکانی، به‌صورت ضرب در فضای فرکانسی روی تصویر اعمال شد. ابتدا برای تطبیق ابعاد کرنل لاپلاسین با تصویر، این کرنل با استفاده از **padarray** به ابعاد تصویر (۶۰۰×۶۰۰) توسعه داده شد، به‌طوری‌که مقادیر اضافی صفر در انتهای سطرها و ستون‌ها اضافه شدند. سپس، تبدیل فوریه دوبعدی **(fft۲)** روی کرنل و تصویر انجام شد و با **fftshift**، مؤلفه‌های فرکانسی به مرکز منتقل شدند. در فضای فرکانسی، اعمال فیلتر لاپلاسین با ضرب ساده‌ی طیف تصویر و طیف فیلتر انجام گرفت. نتیجه‌ی حاصل، با تبدیل فوریه معکوس **(ifft۲)** به فضای مکانی بازگردانده شد. از آنجایی که ممکن است مقادیر خروجی خارج از محدوده‌ی قابل نمایش تصویر (۰ تا ۲۵۵) باشند، نسخه‌ای از تصویر نیز با استفاده از **min**و **max**در این بازه بریده **(clip)** و به **uint۸** تبدیل شد. در شکل‌های حاصل، چهار تصویر نمایش داده شد که امکان مقایسه مستقیم بین نتایج این دو روش را فراهم می‌کند. با وجود تفاوت‌های عددی جزئی، نتایج بصری حاصل از هر دو روش مشابه هستند و تأکید فیلتر بر لبه‌ها و تغییرات شدید شدت روشنایی را نشان می‌دهند.

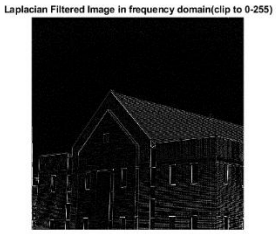
این مسئله تأییدی بر اصل مهمی در پردازش سیگنال‌هاست که اعمال کانولوشن در حوزه مکان معادل ضرب در حوزه فرکانس است.

```
%% Part c
pad_mask = padarray(laplacian_kernel,[597 597],'post');
mask_fft=fftshift(fft2(pad_mask));
img_fft=fftshift(fft2(img));

img_frequency_filtered=ifft2(ifftshift(mask_fft .* img_fft));
img_frequency_filtered_clip = uint8(min(max(img_frequency_filtered, 0), 255));

figure(3);
subplot(٢, ٢, ١);
imshow(filtered_img, []); title('Laplacian Filtered Image in spatial domain');

subplot(٢, ٢, ٢);
imshow(img_frequency_filtered, []); title('Laplacian Filtered Image in frequency
domain');

subplot(٢, ٢, ٣);
imshow(filtered_img_clip, []); title('Laplacian Filtered Image in frequency domain(clip
to 0-255)');

subplot(٢, ٢, ٤);
imshow(img_frequency_filtered_clip, []); title('Laplacian Filtered Image in frequency
domain(clip to 0-255)');
```









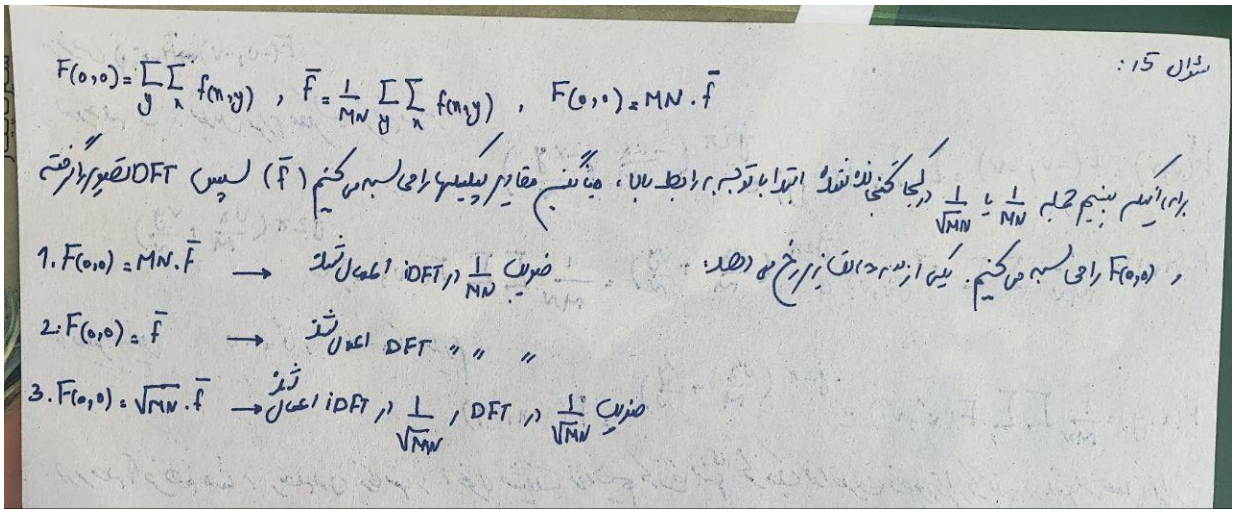
بخش چهارم

همان‌طور که در خروجی‌ها مشاهده شد، نتایج بصری حاصل از هر دو روش بسیار به یکدیگر نزدیک هستند، به‌ویژه زمانی که پدینگ کرنل در حوزه فرکانس به‌درستی انجام شده و جابه‌جایی‌های لازم با **fftshift** و

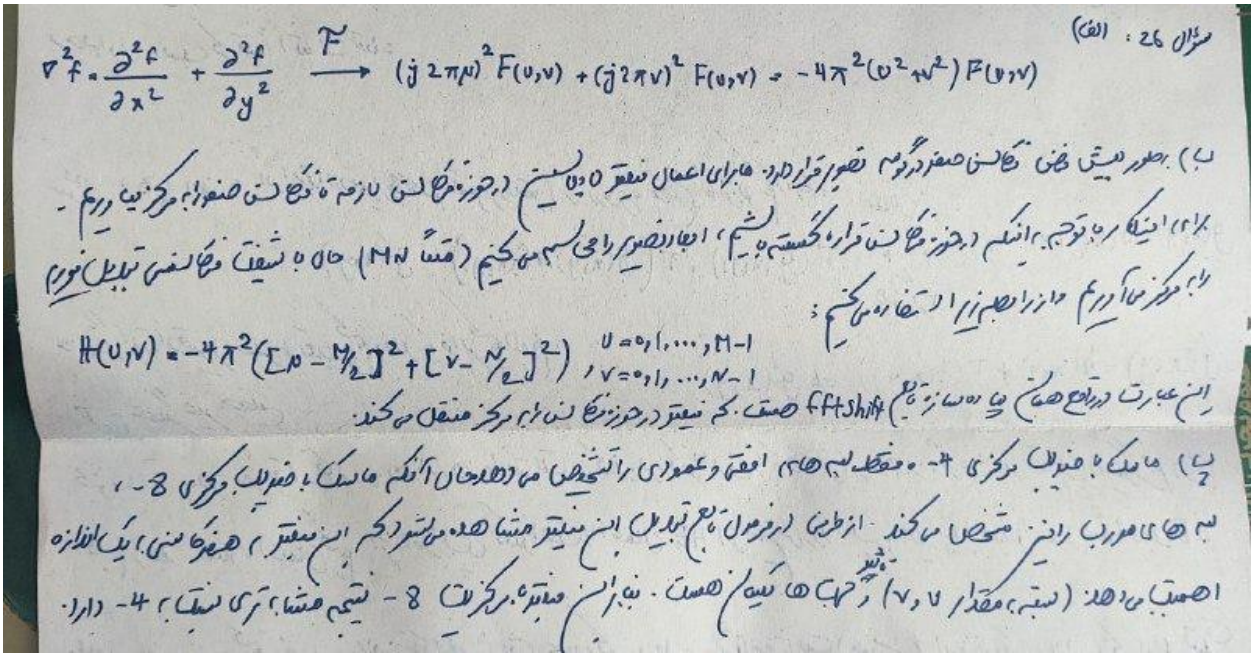
ifftshift صورت گرفته باشد. در هر دو روش، لبه‌ها و نواحی دارای تغییر شدت شدید به‌خوبی تقویت شده و نواحی یکنواخت تضعیف شده‌اند. با این حال، تفاوت‌هایی نیز میان دو روش وجود دارد. یکی از مهم‌ترین تفاوت‌ها مربوط به دقت عددی است؛ در روش فرکانسی ممکن است به دلیل محاسبات فوریه و معکوس آن، مقادیر موهومی بسیار کوچکی تولید شود که معمولاً با تابع **()real** حذف می‌شوند. از سوی دیگر، در فیلتر مکانی، نحوه‌ی برخورد با لبه‌های تصویر (مثلاً استفاده از پدینگ صفر، تکرار مقادیر یا تقارن) بر نتیجه تأثیر می‌گذارد، در حالی که در حوزه فرکانس، کانولوشن به‌صورت دوری در نظر گرفته می‌شود و این می‌تواند باعث ایجاد اعوجاج در اطراف لبه‌های تصویر شود، مگر آن‌که پدینگ مناسب اعمال شود. از نظر سرعت نیز باید گفت که در فیلترهایی با اندازه کوچک مانند لاپلاسین ۳×۳، روش مکانی سریع‌تر و ساده‌تر است؛ اما زمانی که اندازه کرنل بزرگ‌تر شده یا چند فیلتر به‌طور همزمان اعمال می‌شوند، روش فرکانسی می‌تواند از نظر محاسباتی بهینه‌تر باشد

سوال ۷

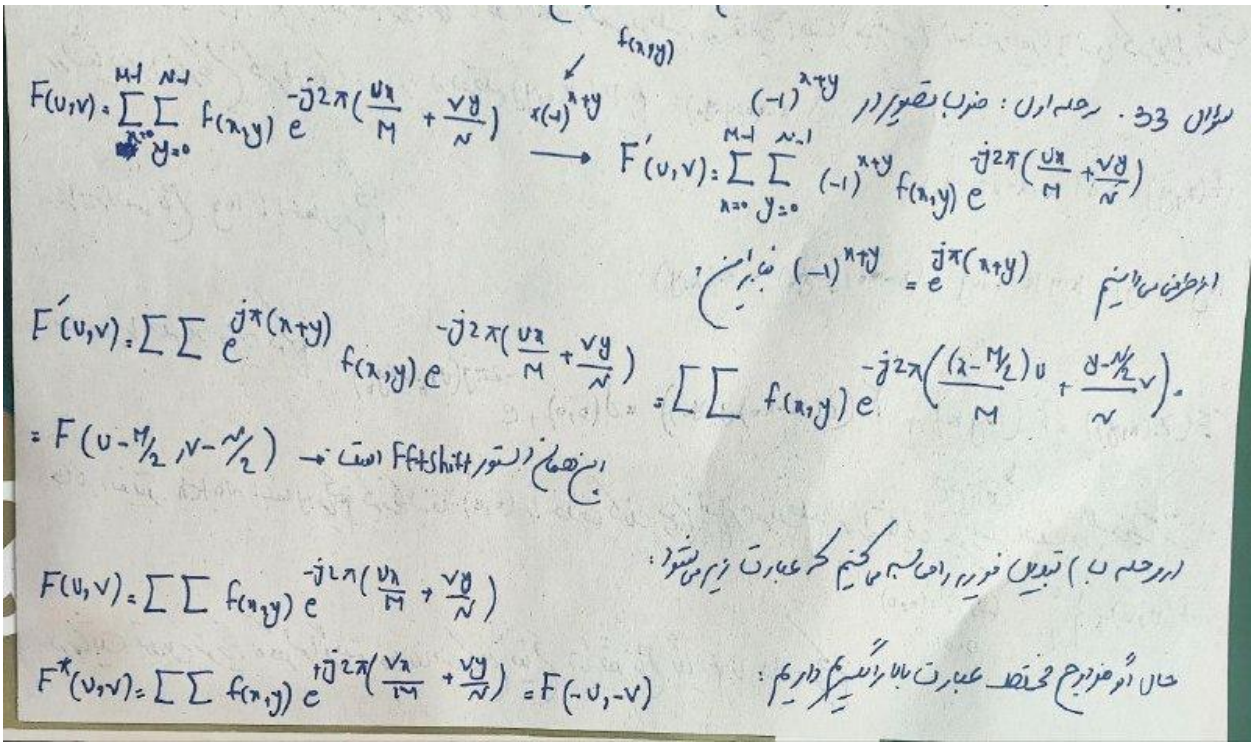
بخش اول

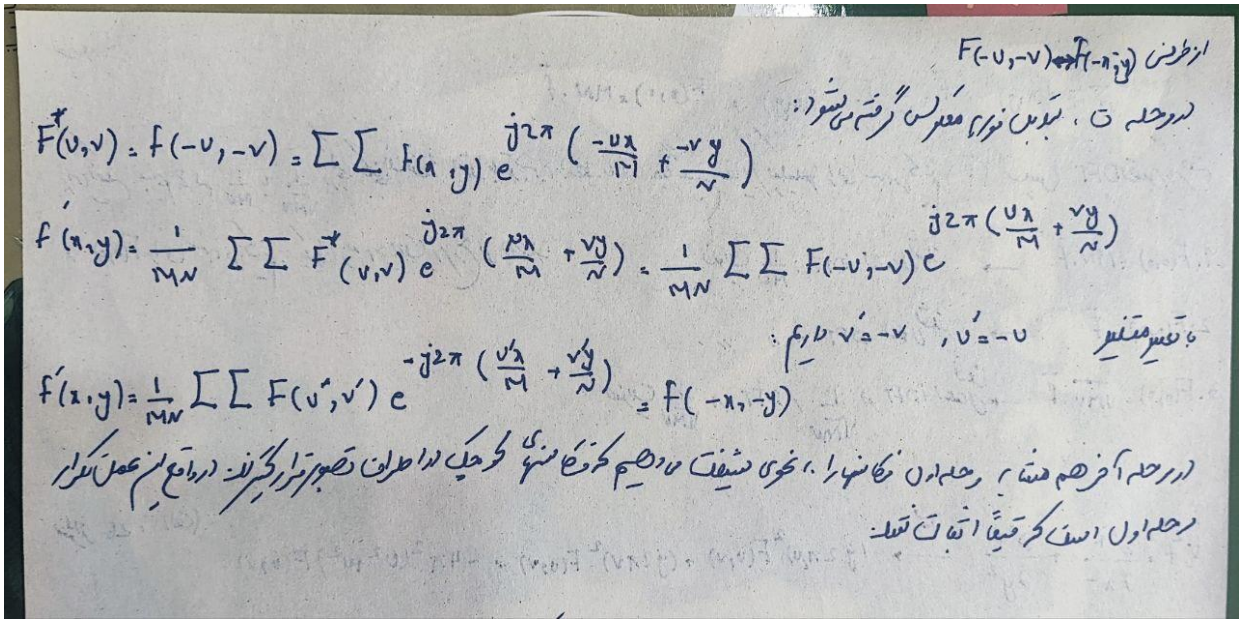


بخش دوم

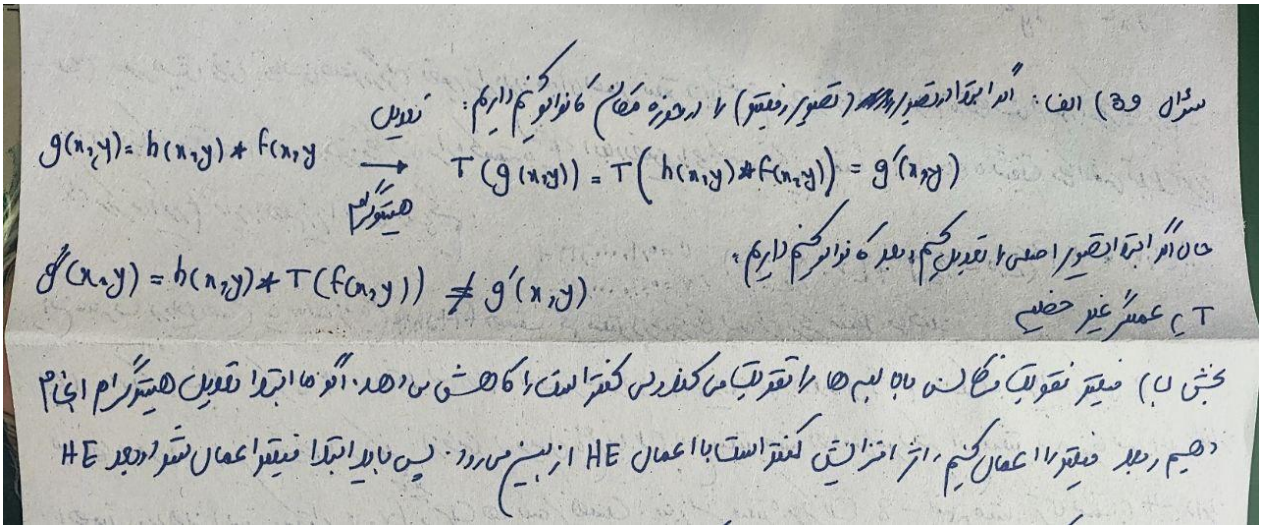


بخش سوم





بخش چهارم



بخش پنجم