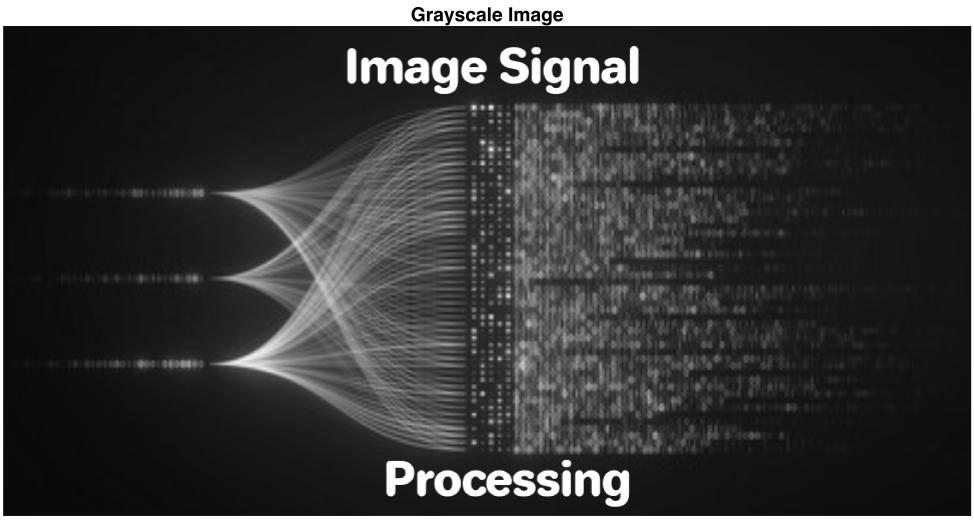
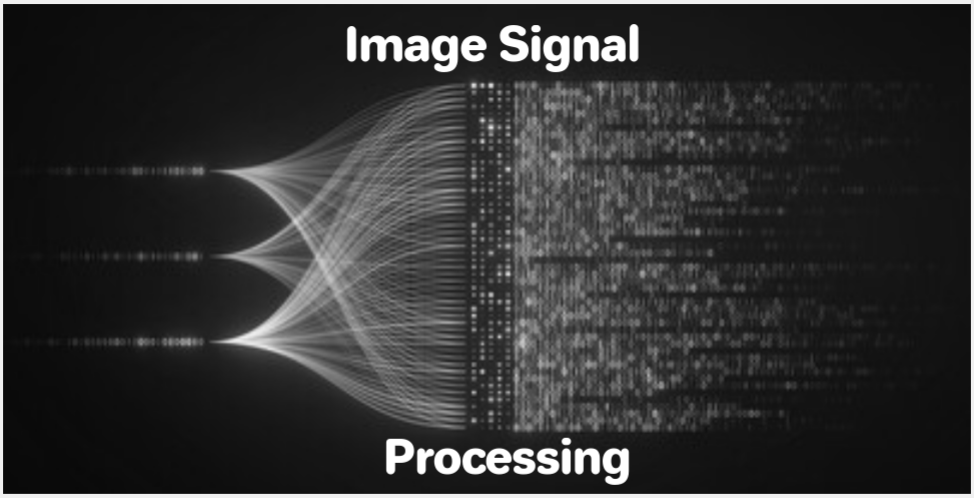
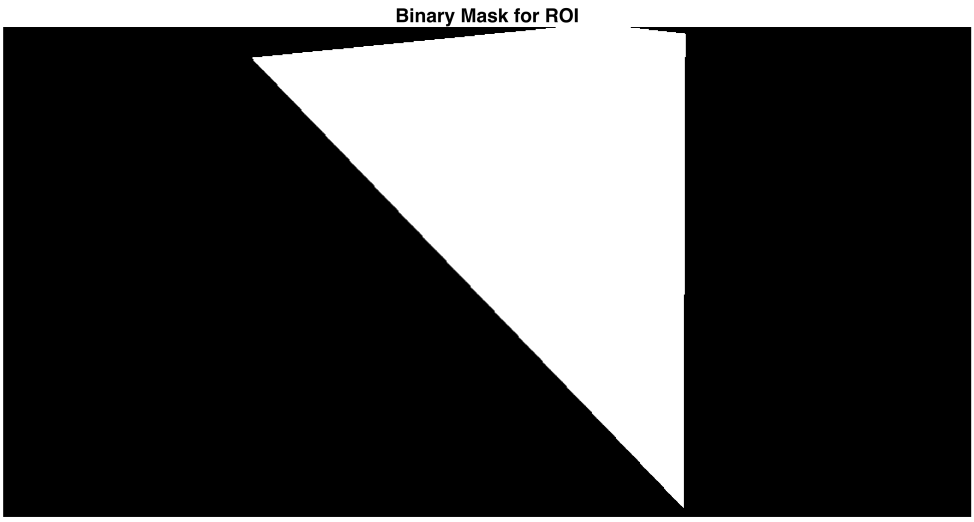
```
% Load the input image
input_img = imread('image_signal.png');  % Provide the path to your image

% Convert to grayscale if the image has multiple color channels
if size(input_img, 3) == 3
    gray_img = rgb2gray(input_img);  % Convert to grayscale if it's an RGB
image
else
    gray_img = input_img;  % If it's already grayscale, no conversion is
needed
end

% Display the grayscale image
figure, imshow(gray_img), title('Grayscale Image');
```




```
% Use roipoly to select a region of interest (ROI) and create a binary mask
roi_mask = roipoly(gray_img);  % Select the ROI interactively
```



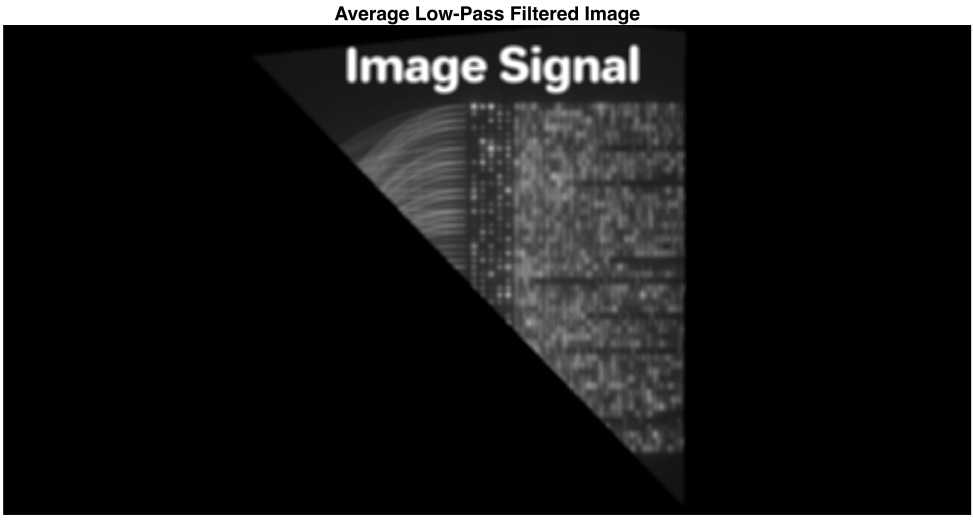

```
% Show the binary mask
figure, imshow(roi_mask), title('Binary Mask for ROI');
```



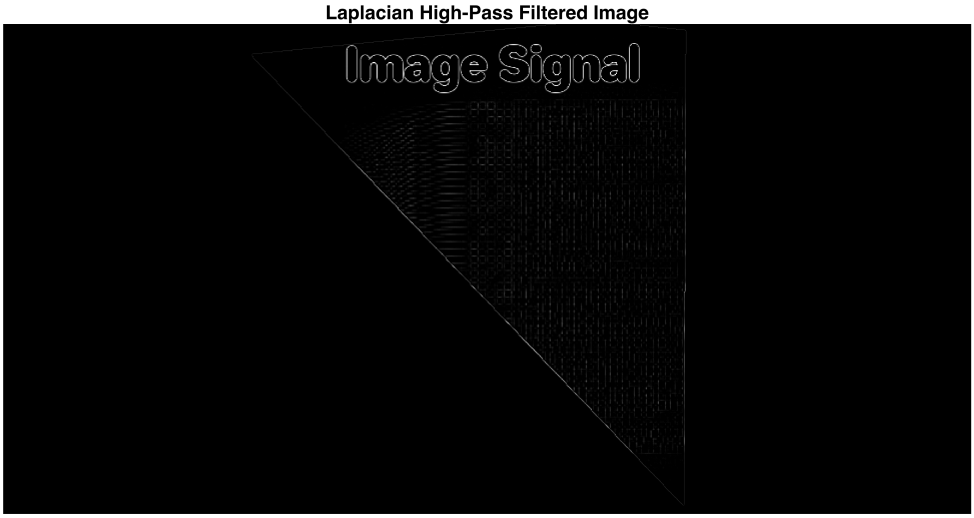

```
% Apply the binary mask to the grayscale image to isolate the region of
interest
region_of_interest = gray_img .* uint8(roi_mask);

% Show the masked region of interest
figure, imshow(region_of_interest), title('Masked Region of Interest');
```


```
% Apply a Gaussian filter with a sigma value of 2 for low-pass filtering
filtered_gaussian = imgaussfilt(region_of_interest, 2);  % Gaussian blur
figure, imshow(filtered_gaussian), title('Gaussian Low-Pass Filtered
Image');
```


```
% Apply an averaging filter (5x5 kernel) for low-pass filtering
avg_filter = fspecial('average', [5 5]);  % Create averaging filter
filtered_avg = imfilter(region_of_interest, avg_filter);  % Apply the
averaging filter
figure, imshow(filtered_avg), title('Average Low-Pass Filtered Image');
```




```
% Apply a Laplacian filter to detect edges (high-pass filtering)
laplacian_filter = fspecial('laplacian', 0.2);  % Create Laplacian filter
with alpha = 0.2
filtered_laplacian = imfilter(region_of_interest, laplacian_filter);  %
Apply the Laplacian filter
figure, imshow(filtered_laplacian, []), title('Laplacian High-Pass Filtered
Image');
```




```
% Apply the Prewitt filter for edge detection (high-pass filtering)
prewitt_horizontal = fspecial('prewitt');  % Horizontal Prewitt filter
prewitt_vertical = prewitt_horizontal';    % Vertical Prewitt filter
(transpose for vertical edges)
```

```
% Apply the Prewitt filters and ensure that the region of interest is
converted to double
filtered_prewitt_h = imfilter(double(region_of_interest),
prewitt_horizontal);  % Horizontal edges
filtered_prewitt_v = imfilter(double(region_of_interest),
prewitt_vertical);    % Vertical edges

% Combine the horizontal and vertical Prewitt responses
filtered_prewitt = sqrt(filtered_prewitt_h.^2 + filtered_prewitt_v.^2);  %
Combine using the square root

% Display the Prewitt filtered image
figure, imshow(filtered_prewitt, []), title('Prewitt High-Pass Filtered
Image');
```