1. **Create a binary mask for the region of interest in the image, then apply low-pass filters (Gaussian and Average filters) and high-pass filters (Laplacian and Prewitt filters) in MATLAB.**

```
img = imread('/MATLAB Drive/liberty.jpeg');
gray_img = rgb2gray(img);

% Create a binary mask
threshold = 100;
binary_mask = gray_img > threshold;
roi_img = gray_img .* uint8(binary_mask);

% Low-Pass Filters (Gaussian and Average)
gaussian_filtered = imgaussfilt(roi_img, 2);
h = fspecial('average', [5 5]);
average_filtered = imfilter(roi_img, h);

% High-Pass Filters (Laplacian and Prewitt)
laplacian_filtered = imfilter(roi_img, fspecial('laplacian', 0.2));
prewitt_filtered = edge(roi_img, 'prewitt');

% Display Images
figure;
subplot(2, 3, 1);
imshow(img);
title('Image Taken');

subplot(2, 3, 2);
imshow(roi_img, 'InitialMagnification', 'fit');
title('Binary Masked Image');
subplot(2,3, 3);
imshow(gaussian_filtered, 'InitialMagnification', 'fit');
title('Gaussian Low-Pass Filter');

subplot(2, 3, 4);
imshow(average_filtered, 'InitialMagnification', 'fit');
title('Average Low-Pass Filter');

subplot(2, 3, 5);
imshow(laplacian_filtered, 'InitialMagnification', 'fit');
title('Laplacian High-Pass Filter');

subplot(2, 3, 6);
imshow(prewitt_filtered, 'InitialMagnification', 'fit');
title('Prewitt High-Pass Filter');
```

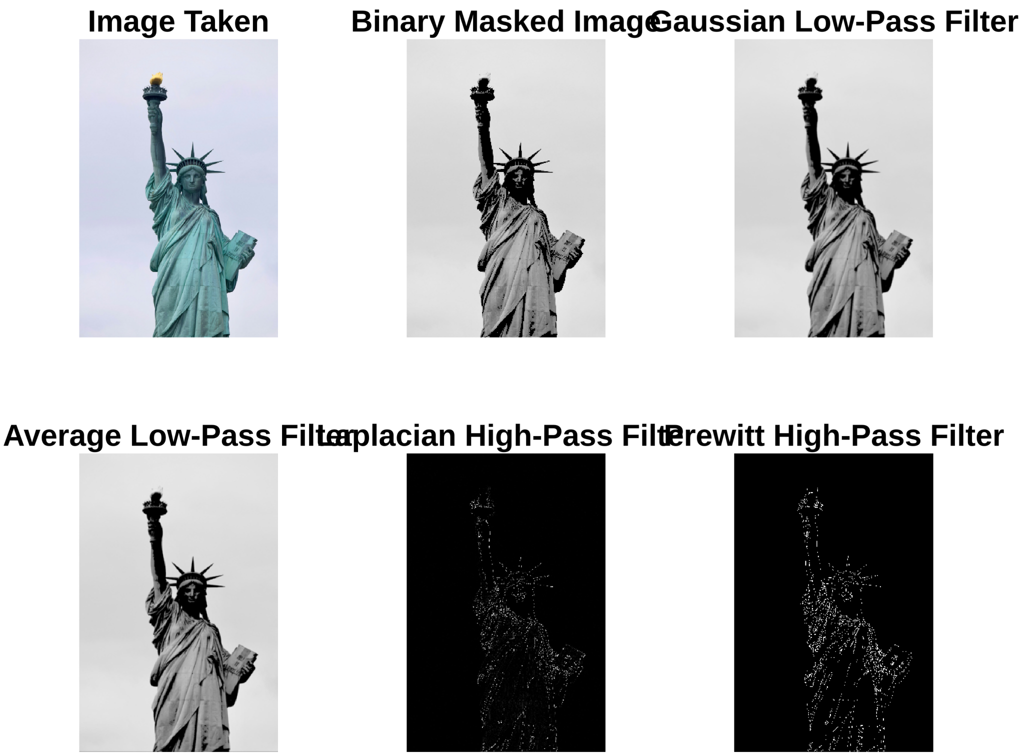


Image Taken

Binary Masked Image

Gaussian Low-Pass Filter

Average Low-Pass Filter

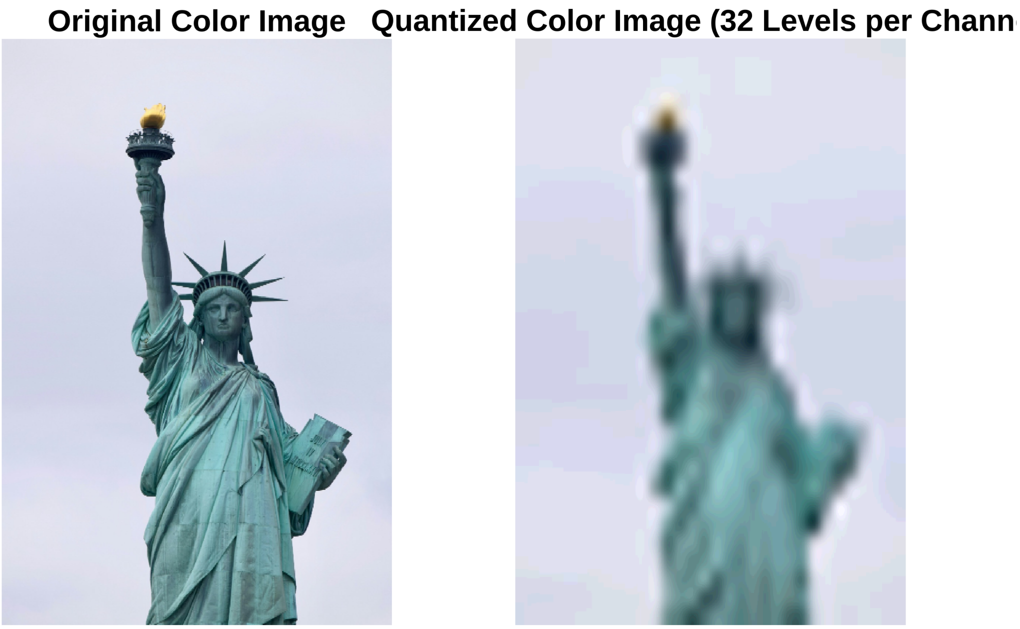
Laplacian High-Pass Filter

Prewitt High-Pass Filter

5. **Take an image and quantize it to 32 grayscale levels using only the imresize functionin MATLAB, and write the steps you followed in the process.**

```
% Load the image
img = imread('/MATLAB Drive/liberty.jpeg');

% Resize each color channel (R, G, B) to quantize to 32 levels
quantized_r = imresize(imresize(img(:,:,1), [32, 32]), size(img(:,:,1)));
quantized_g = imresize(imresize(img(:,:,2), [32, 32]), size(img(:,:,2)));
quantized_b = imresize(imresize(img(:,:,3), [32, 32]), size(img(:,:,3)));

% Pixel values to 32 discrete levels
quantized_r = round(quantized_r / 8) * 8;
quantized_g = round(quantized_g / 8) * 8;
quantized_b = round(quantized_b / 8) * 8;
quantized_img = cat(3, quantized_r, quantized_g, quantized_b);

% Show Images
figure;
subplot(1, 2, 1);
imshow(img);
title('Original Color Image');

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

```
title('Quantized Color Image (32 Levels per Channel)');
```



1. **Loading the Image**: I started by loading the color image using `imread`, which gave me the RGB channels of the image.
2. **Resizing Each Color Channel**: I separated the Red, Green, and Blue channels and used `imresize` to downsample each channel to 32x32 pixels. I then resized it back to the original dimensions to maintain the image size while reducing the number of color levels.
3. **Quantizing the Values**: To reduce the number of intensity levels to 32, I divided each channel by 8, rounded the values, and multiplied them back by 8, ensuring each pixel had only 32 possible levels.
4. **Reconstructing the Image**: After quantizing each channel, I recombined them to form the final RGB image.
5. **Displaying the Results**: Finally, I displayed the original and quantized images side by side, showing the effect of reducing color levels on the image's appearance.