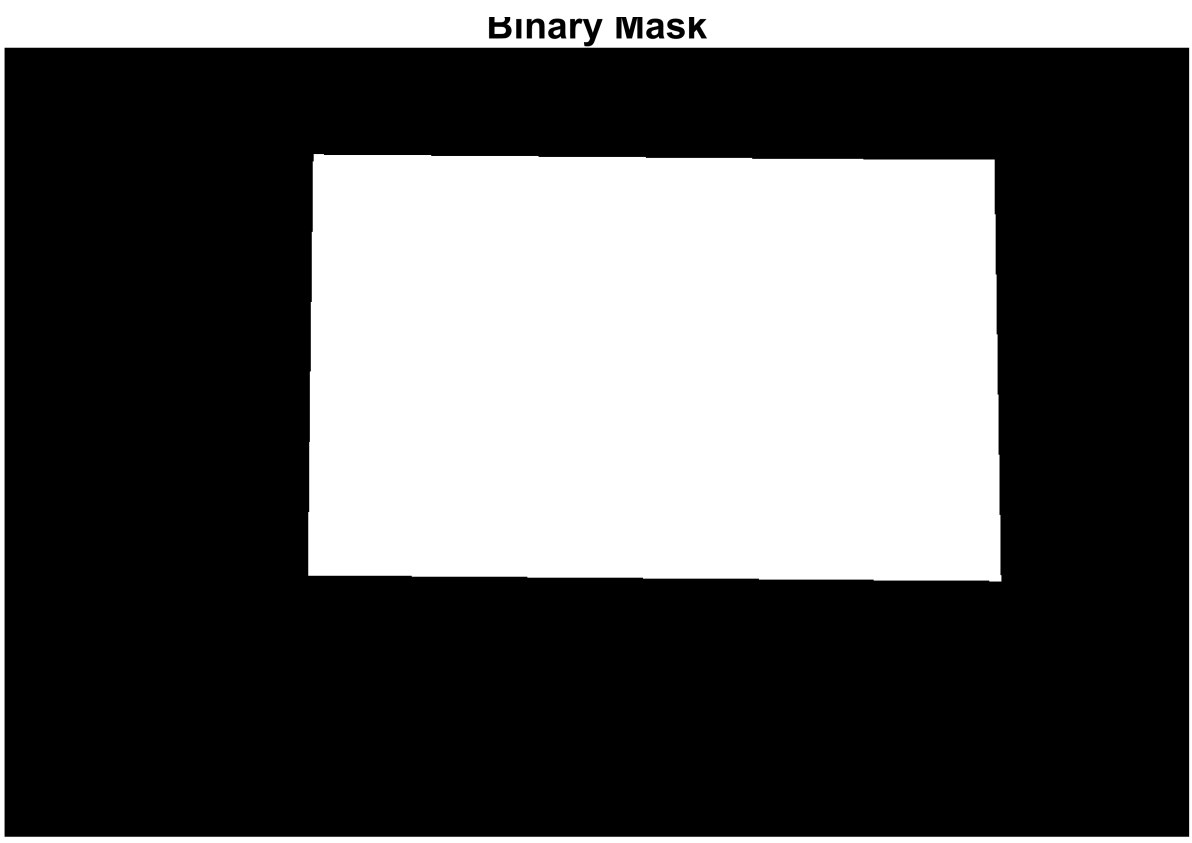
```
% Loading the image
image = imread('C:\Users\USER\Pictures\CLAG6093.JPG');
% original image
figure, imshow(image);
title('Original Image');
```


```
% Creating the binary mask using roipoly
mask = roipoly(image);
```

```
% Ensuring the mask is not empty
if isempty(mask)
    error('Mask was not properly created. Ensure that the region is selected using
roipoly.');
end

% Displaying the binary mask
figure, imshow(mask);
title('Binary Mask');
```




```
% Checking if the image is grayscale or RGB
[rows, cols, numChannels] = size(image);

% If the image is RGB, replicate the mask to match the RGB size
if numChannels == 3
    % Replicate the mask along the third dimension (RGB channels)
    maskRGB = repmat(mask, [1 1 3]);
else
    % If the image is grayscale, use the mask as is
    maskRGB = mask;
end

% Applying Low-Pass Filters

% Gaussian Filter
sigma = 2;  % Adjust sigma for the desired amount of smoothness
gaussianFiltered = imgaussfilt(image, sigma);

% Displaying Gaussian filtered image
figure, imshow(gaussianFiltered);
title('Gaussian Filtered Image');
```

Gaussian Filtered Image

```
% Average Filter
h = fspecial('average', [5 5]);  % Create average filter of size 5x5
avgFiltered = imfilter(image, h);

% Display Average filtered image
figure, imshow(avgFiltered);
title('Average Filtered Image');
```

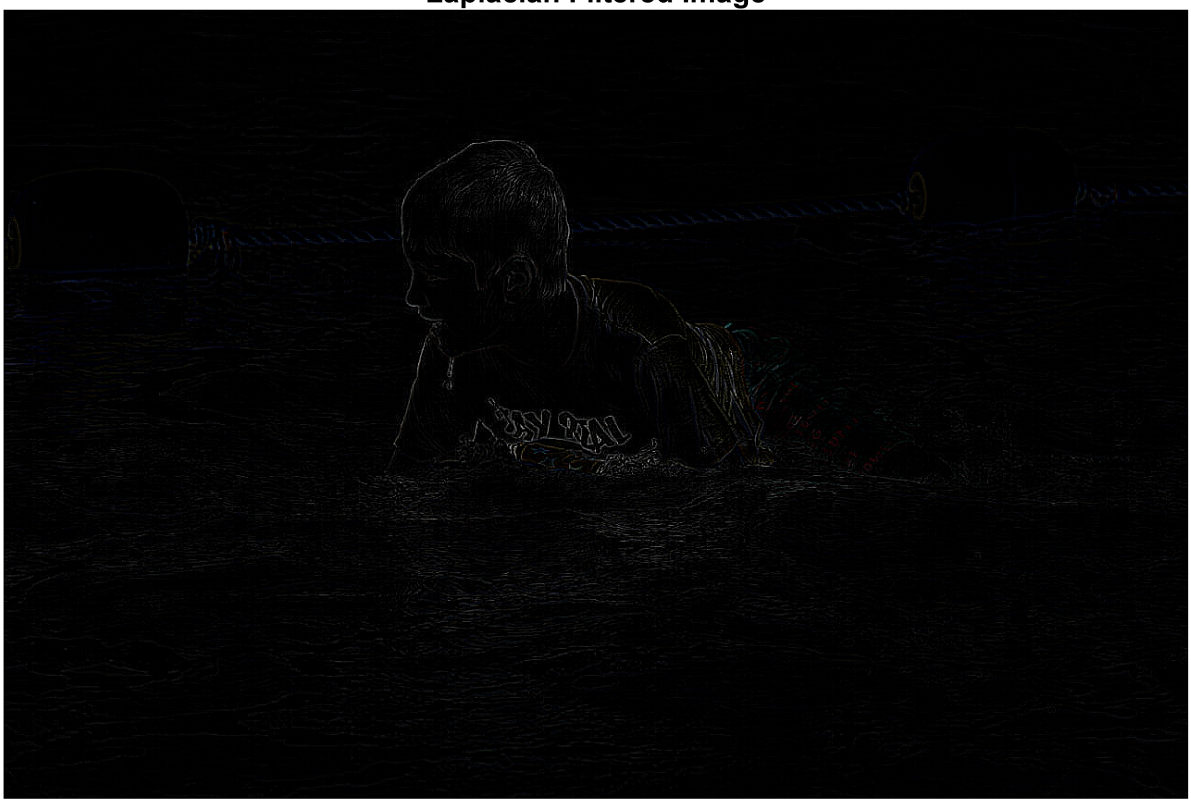
Average Filtered Image

```
% Applying High-Pass Filters

% Laplacian Filter
h = fspecial('laplacian', 0.2);  % Laplacian filter
laplacianFiltered = imfilter(image, h);

% Displaying Laplacian filtered image
figure, imshow(laplacianFiltered);
title('Laplacian Filtered Image');
```

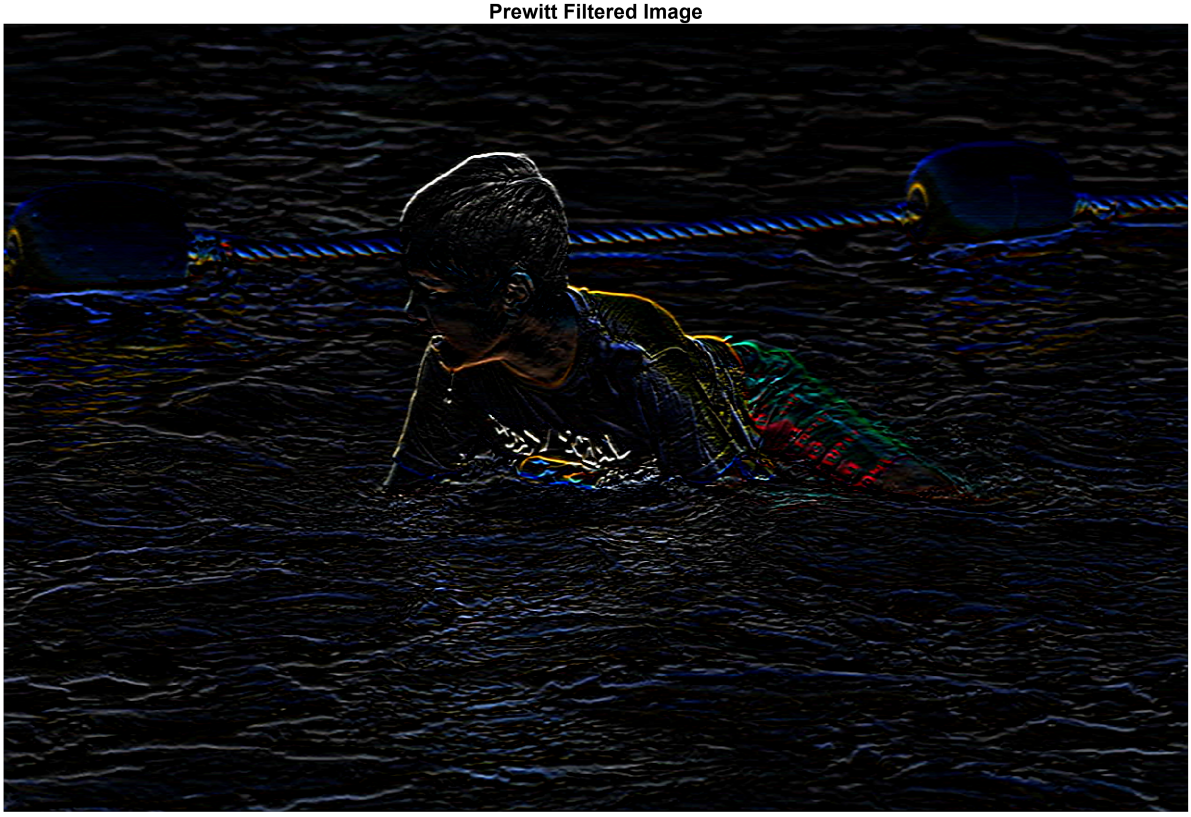
Laplacian Filtered Image



```
% Prewitt Filter
h = fspecial('prewitt');  % Prewitt filter
prewittFiltered = imfilter(image, h);

% Displaying Prewitt filtered image
figure, imshow(prewittFiltered);
title('Prewitt Filtered Image');
```

Prewitt Filtered Image



```
% Resize the Mask to Match the Filtered Image Size

[maskRows, maskCols] = size(mask);  % Get mask dimensions

% Resizing the mask if the dimensions are different from the image
if maskRows ~= rows || maskCols ~= cols
    % Resize the mask to match the image dimensions
    mask = imresize(mask, [rows, cols]);

    if numChannels == 3
        % For RGB images, replicate the resized mask across RGB channels
        maskRGB = repmat(mask, [1 1 3]);
    else
        % For grayscale images, just use the resized mask
        maskRGB = mask;
    end
end

%  Applying the Filters to the ROI

if numChannels == 3
    % Apply mask to each color channel (RGB image)
    gaussianROI = uint8(double(gaussianFiltered) .* maskRGB);
```

```
    avgROI = uint8(double(avgFiltered) .* maskRGB);
    laplacianROI = uint8(double(laplacianFiltered) .* maskRGB);
    prewittROI = uint8(double(prewittFiltered) .* maskRGB);
else
    % Applying mask directly (Grayscale image)
    gaussianROI = uint8(double(gaussianFiltered) .* mask);
    avgROI = uint8(double(avgFiltered) .* mask);
    laplacianROI = uint8(double(laplacianFiltered) .* mask);
    prewittROI = uint8(double(prewittFiltered) .* mask);
end

% Displaying Results for Each Filter on the ROI

figure, imshow(gaussianROI);
title('Gaussian Filter Applied to ROI');
```


```
figure, imshow(avgROI);
title('Average Filter Applied to ROI');
```

```
figure, imshow(laplacianROI);
title('Laplacian Filter Applied to ROI');
```

```
figure, imshow(prewittROI);
title('Prewitt Filter Applied to ROI');
```