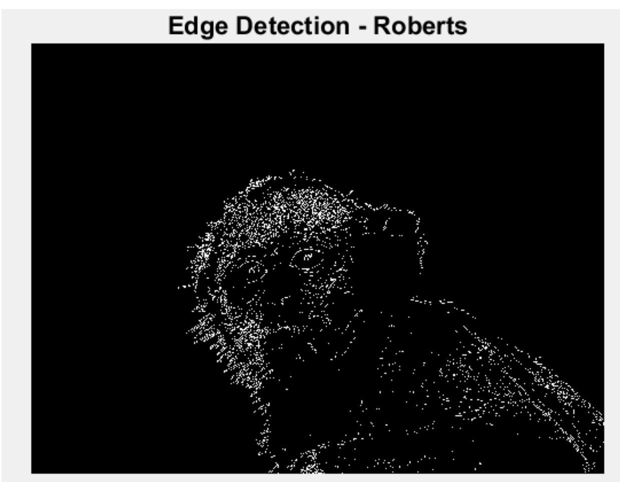
Problem 1:

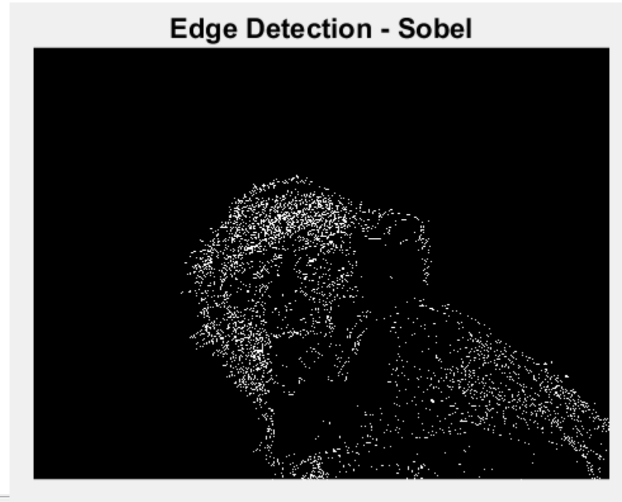
```
% Problem 1
I = imread('monkey.jpeg');
I = rgb2gray(I);
figure; imshow(I); title('Original Image');
% Roberts Edge
E1 = edge(I, 'roberts');
figure; imshow(E1); title('Edge Detection - Roberts');
% Sobel Edge
E2 = edge(I, 'sobel');
figure; imshow(E2); title('Edge Detection - Sobel');
% Prewitt Edge
E3 = edge(I, 'prewitt');
figure; imshow(E3); title('Edge Detection - Prewitt');
```

Code for Part 1 Problem 1

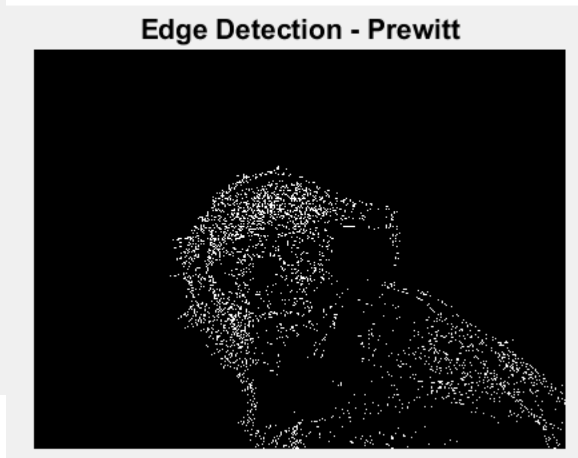
Original Picture



Edge Detection- Roberts



Edge Detection- Sobel



Edge Detection- Prewitt

The Roberts operator is the least successful of the three edge detection methods since it is unable to identify distant land near the horizon. Although the Sobel and Prewitt operators are comparable, the Sobel does appear to capture edges in more detail (for instance, it captures the edges of the several rings on the front bike tire better than the Prewitt operator).

Problem 2:

1^{st} Order Derivative: $f(x+1) - f(x)$

2^{nd} Order Derivative: $f(x+1) + f(x-1) - 2f(x)$

Image: 6 6 6 6 5 4 3 2 1 1 1 1 1 1 1 6 6 6 6 6

1^{st} Derivative of the image:

0 0 -1 -1 -1 -1 -1 0 0 0 0 0 0 5 0 0 0 0

2^{nd} Derivative of the Image:

0 0 -1 0 0 0 0 1 0 0 0 0 0 5 -5 0 0 0

Problem 3:

```
% Problem 3
I = imread('monkey.jpeg');
I = rgb2gray(I);
h = fspecial('gaussian', 5, 2.5);
blurred = imfilter(I, h);
diff = I - blurred;
k = 1;
highboost_k_1 = I + k * diff;
k = 5;
highboost_k_5 = I + k * diff;

figure; imshow(highboost_k_1); title('HighBoosted Image, k = 1');
figure; imshow(highboost_k_5); title('HighBoosted Image, k = 5');
```

Code

High Boosted Image when k=1

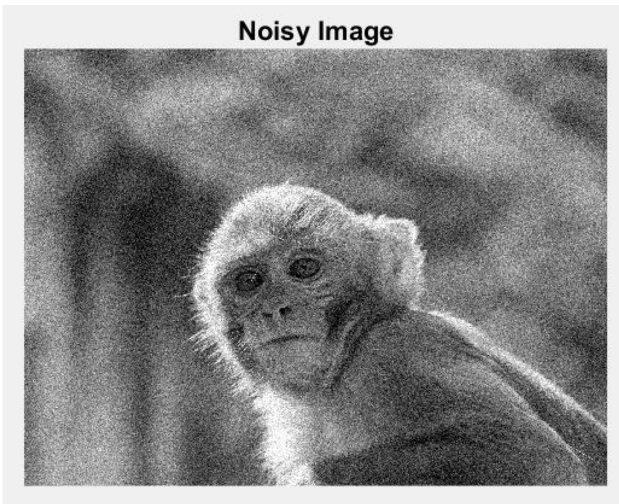
High Boosted Image when k=5

By just applying the unsharp masking to the original image when k=1, we achieve little to no effect. High boost filtering is what happens when k>1, as the second image, "High Boosted Image when k=5", demonstrates. The image has a noticeably improved level of sharpness.

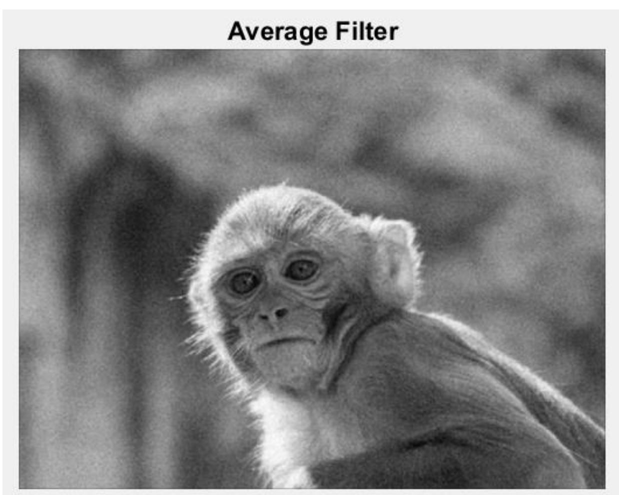
Problem 4:

```
% Problem 4
I = imread('monkey.jpeg');
I = rgb2gray(I);

blurred_noise = imnoise(I, 'gaussian');
figure; imshow(blurred_noise); title('Noisy Image');

average_filter = filter2(fspecial('average', 5), blurred_noise)/255;
figure; imshow(average_filter); title('Average Filter');

gaussian_filter = imgaussfilt(blurred_noise, 2.5);
figure; imshow(gaussian_filter); title('Gaussian Filter');
```

Code



Noisy Image



Average Filter

Gaussian Filter

The gaussian filter removes gaussian noise more effectively, as seen in the Figures. However, one negative effect is that the final denoised image is hazier.

Problem 5:

1:

$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$

$$\frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\nabla^2 f = f(x+1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

2:

4	6	4
6	3	-8
1	-7	-4

Part 2:

```
% Enhance Low Light Image using Dehazing Algorithm
A = imread('low_light_1.jpg');
B = imread('low_light_2.jpg');

figure; imshow(A); title('Original Image - 1');
figure; imshow(B); title('Original Image - 2');

A_INV = imcomplement(A);
B_INV = imcomplement(B);

A_DEHAZED = imcomplement(imreducehaze(A_INV));
B_DEHAZED = imcomplement(imreducehaze(B_INV));

figure; imshow(A_DEHAZED); title('Dehazing Algorithm - 1');
figure; imshow(B_DEHAZED); title('Dehazing Algorithm - 2');
```

Code 1

```
% Improve Results Further Using imreducehaze Optional Parameters
A_DEHAZED_ENHANCED = imcomplement(imreducehaze(A_INV, 'Method', 'approx', 'ContrastEnhancement', 'boost'));
B_DEHAZED_ENHANCED = imcomplement(imreducehaze(B_INV, 'Method', 'approx', 'ContrastEnhancement', 'boost'));

figure; imshow(A_DEHAZED_ENHANCED); title('Dehazing Algorithm Enhanced - 1');
figure; imshow(B_DEHAZED_ENHANCED); title('Dehazing Algorithm Enhanced - 2');

% Another Example of Improving Poorly Lit Image
A_DEHAZED_NO_CONTRAST_ENHANCEMENT = imcomplement(imreducehaze(A_INV, 'ContrastEnhancement', 'none'));
B_DEHAZED_NO_CONTRAST_ENHANCEMENT = imcomplement(imreducehaze(B_INV, 'ContrastEnhancement', 'none'));

figure; imshow(A_DEHAZED_NO_CONTRAST_ENHANCEMENT); title('Dehazing Algorithm No Contrast Enhancement - 1');
figure; imshow(B_DEHAZED_NO_CONTRAST_ENHANCEMENT); title('Dehazing Algorithm No Contrast Enhancement - 2');
```

Code 2

```
% Reduce Color Distortion by Using Different Color Space
A_LAB = rgb2lab(A);
B_LAB = rgb2lab(B);
A_LAB_INV = imcomplement(A_LAB(:,:,1) ./ 100);
B_LAB_INV = imcomplement(B_LAB(:,:,1) ./ 100);

A_LAB_ENH = imcomplement(imreducehaze(A_LAB_INV, 'ContrastEnhancement', 'none'));
B_LAB_ENH = imcomplement(imreducehaze(B_LAB_INV, 'ContrastEnhancement', 'none'));

A_LAB_ENH(:,:,1)   = A_LAB_ENH .* 100;
A_LAB_ENH(:,:,2:3) = A_LAB(:,:,2:3) * 2;
B_LAB_ENH(:,:,1)   = B_LAB_ENH .* 100;
B_LAB_ENH(:,:,2:3) = B_LAB(:,:,2:3) * 2;

AEnh = lab2rgb(A_LAB_ENH);
BEnh = lab2rgb(B_LAB_ENH);

figure; imshow(AEnh); title('LAB Dehazing - 1');
figure; imshow(BEnh); title('LAB Dehazing - 2');
```

Code 3

```
% Improve Results Using Denoising
A_GUIDED_FILTER = imguidedfilter(A_DEHAZED);
B_GUIDED_FILTER = imguidedfilter(B_DEHAZED);

figure; imshow(A_GUIDED_FILTER); title('Guided Filter - 1');
figure; imshow(B_GUIDED_FILTER); title('Guided Filter - 2');

% Estimate Illumination Map
[A_ENH_INV, A_ILLUM_INV] = imreducehaze(A_INV, 'Method', 'approxdcp', 'ContrastEnhancement', 'none');
[B_ENH_INV, B_ILLUM_INV] = imreducehaze(B_INV, 'Method', 'approxdcp', 'ContrastEnhancement', 'none');
A_ILLUM = imcomplement(A_ILLUM_INV);
B_ILLUM = imcomplement(B_ILLUM_INV);

figure; imshow(A_ILLUM); colormap(hot); title('Illumination Map - 1');
figure; imshow(B_ILLUM); colormap(hot); title('Illumination Map - 2');
```

Code 4

1:

Original Image- 1

Original Image- 2

Dehazing Algorithm 2

1:

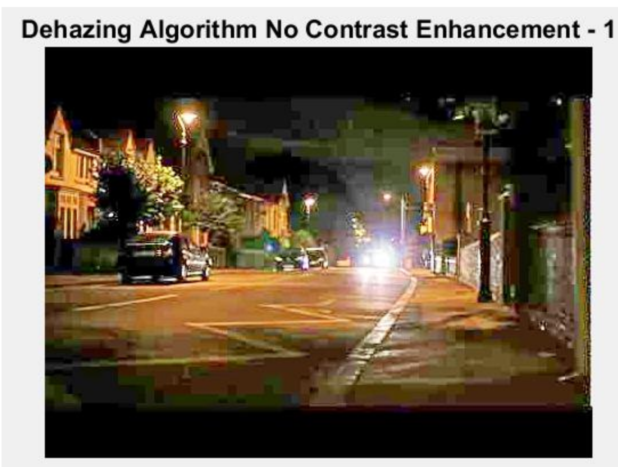
Dehazing Algorithm 1

2:

Dehazing Algorithm Enhanced 1

Dehazing Algorithm Enhanced 2

3:



Dehazing Algorithm No Contrast Enhancement 1

Dehazing Algorithm No Contrast Enhancement 2

4:

LAB Dehazing 1

LAB Dehazing 2

5:

Dehazing Algorithm 1

Dehazing Algorithm 2

Guided Filter 1

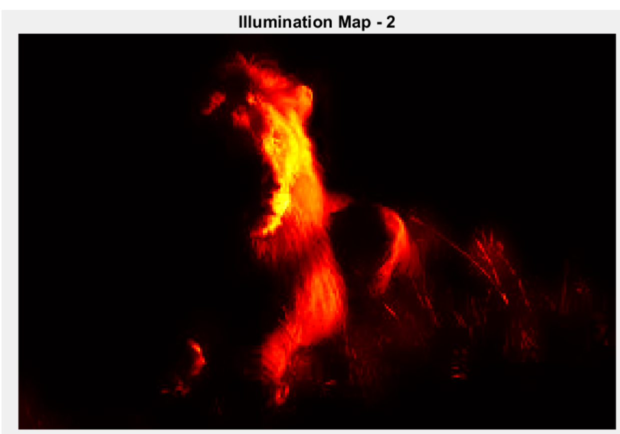
Guided Filter 2

6:

Original Image 1

Illumination Map 1

Original Image 2



Illumination Map 2

An effective approach for improving a low light image is the image enhancement algorithm. It functions in two key steps. The low light image is first inverted, and then it is subjected to an optimised use of an image dehazing method.

The general dehazing algorithm is initially applied to the low light photos. By including certain optional options in the second stage, such as raising the contrast enhancement, we further strengthen this technique. The identical process is used in step three, but without any contrast augmentation, producing a superior overall low-light image. The dehazing method is applied to our image in the fourth step, and then we convert it back to RGB, which regrettably produces subpar results. In the fifth stage, we apply a denoising filter after dehazing because low light photographs typically have a lot of noise because cameras with higher focal lengths do (lower focal length cameras create more clear and less noisy low light images). To see where the majority of the light is coming from, we construct an illumination map from the original photos in the sixth phase, which is shown by brighter colours in the heat map.

Part 3:

The idea I have for the project is Object Tracking Techniques revolving around self driving cars.