Homework 3: Image Filtering with Fourier Transforms

Christopher Kantor

Department of Computer Science and Engineering

University of South Florida

1. Introduction and Description

This assignment focuses on implementing image filtering techniques using the Fourier Transform in both gray and color space images. Using the Discrete Fourier Transform (DFT) and manipulating images in the frequency space, we can implement an image blur algorithm and an edge detection algorithm using high and low pass filters. A band-stop filter was also implemented and is a combination of high and low pass filters. Additionally, we can build upon the theory of high/low pass filters to implement an edge sharpening algorithm called unsharp masking.

The report is organized as follows. In section 2, the algorithms for image filtering using Fourier transforms are described. Section 3 describes the implementation details of these algorithms. In section 4, the results of the project are displayed, and we conclude in section 5.

2. Description of Algorithms

We implemented three main image processing algorithms in this project. High/low pass filters, band-stop filters, and unsharp masking. All these approaches work on an image's frequency domain, which we can get by performing a Fourier transform on the image.

2.1. High and Low Pass Filters

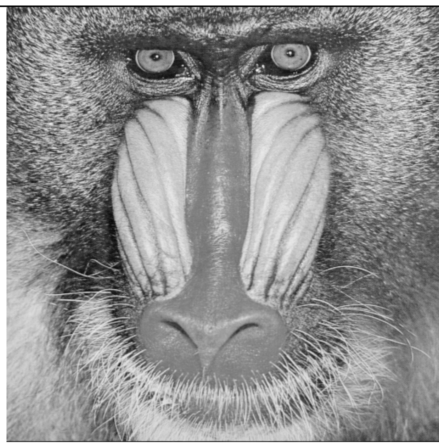
High and low pass filters work by attenuating a range of frequencies in an image's frequency domain. To do this, we first take a DFT of an image, and get two results, a frequency map, and a phase map. We then rearrange these maps so that the center of them is the lowest frequency, and the highest frequencies are towards the edges. To create the filter, we first need a radius which represents how much of the frequency domain we want to attenuate. A larger radius means more attenuation, which means a blurrier image for lowpass, and more pronounced edges in high pass filters.

The two filters work on the same concept, where we only allow a specific range of frequencies to pass through them. For low pass filters, we only allow low frequencies, or the ones around the center of the frequency map, to pass through, and vice versa for high pass filters. These filters are easy to create as we can just draw a circle on a background and set the circle to white and the background to black for low pass filters, or vice versa for high pass ones. These filters are the same size as the frequency domain.

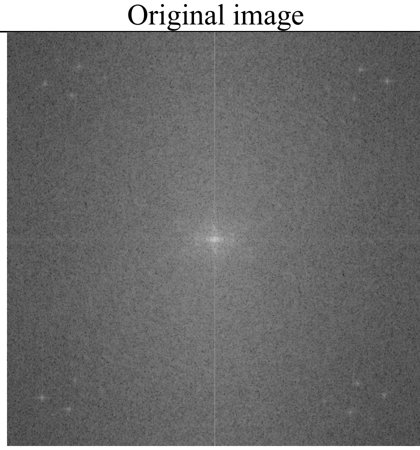
To apply the filters, we multiply each element of the frequency domain with the corresponding element of the filter. If the filter is 0 (black), we do not keep that frequency, and if it is 1 (white), we do.

After applying the filter, we take the inverse Fourier transform (IDFT) to get the final image.

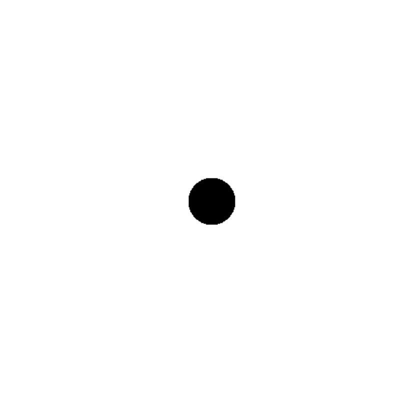
The frequency domain maps the rate of change of pixel intensities in the image. Higher rates of change correspond to the high frequencies in the Fourier domain, and low rates of change correspond to the low frequencies. High rates of change in pixel intensities over a small area intuitively describe edges, so if we only allow high frequencies in the frequency domain to pass through the filter, we are only allowing edges. Conversely, low rates of change in pixel intensities describe the insides of objects or backgrounds of an image, which tend to have the same intensity. By only allowing these low frequencies to pass through the filter, we are removing any sharp changes in intensity. This is usually edges or noise and will effectively smooth out the image.



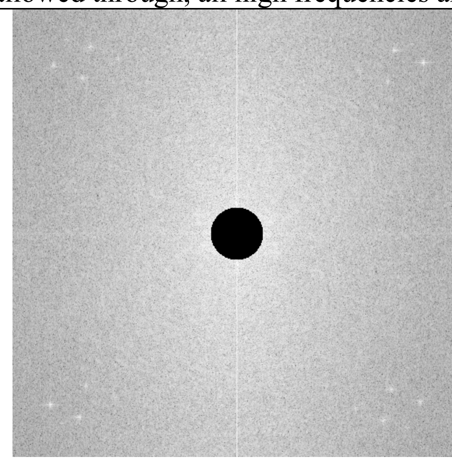
Original image



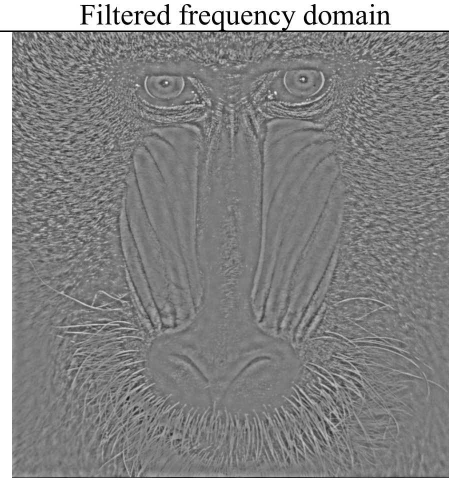
Frequency domain of image



High pass filter (low frequencies are not allowed through, all high frequencies are)

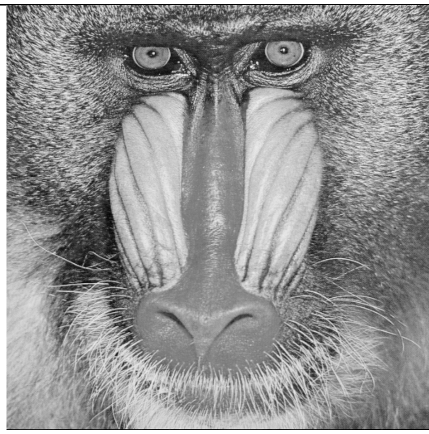


Filtered frequency domain

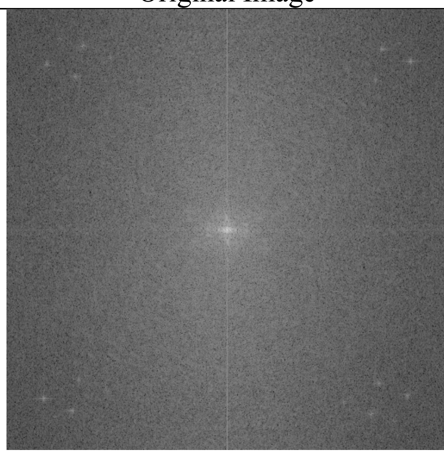


Filtered image

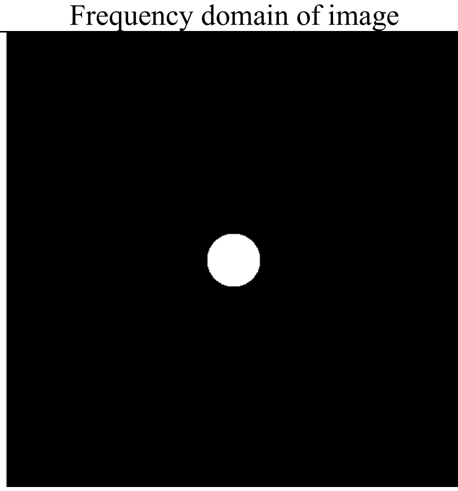
High pass filter



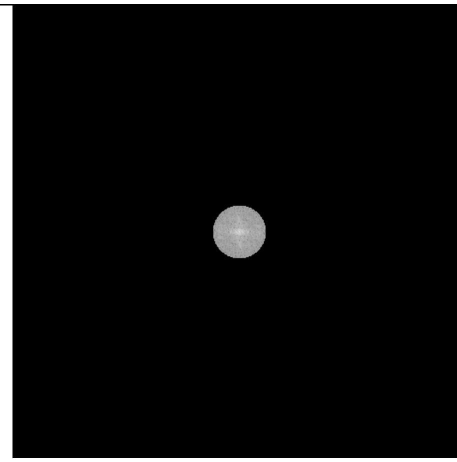
Original Image



Frequency domain of image



low pass filter (low frequencies are allowed through, and all high frequencies are not)



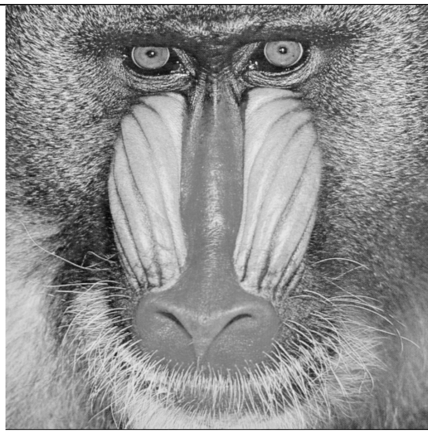
Filtered frequency domain

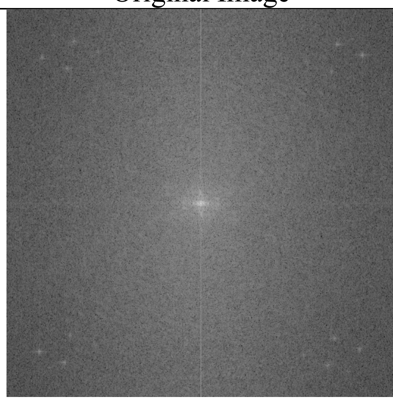
Filtered image

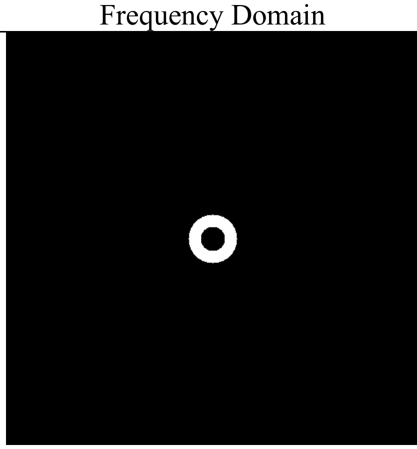
Low pass filter

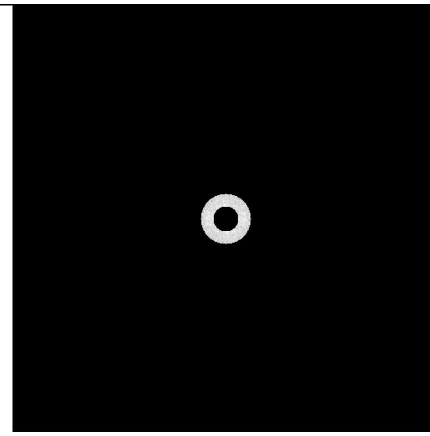
2.2. Band-stop Filter

The band-stop filter is a combination of high and low pass filters. We only allow frequencies that lie between two cutoffs that the user provides. This results in a ring-shaped filter.


Original Image


Frequency Domain


Band-stop Filter


Filtered frequency domain


Filtered Image

2.3. Unsharp Masking

Unsharp masking is a feature in multiple image processing software packages such as Adobe Photoshop and builds upon the intuition of high and low pass filters. We first find a blurry image and subtract it from the original. This can be done using a lowpass filter, which blurs the image. We can then take the high pass filter of the object, which highlights the edges of an image. by multiplying the high pass filter by some number, we can further increase the contrast between edges and the rest of the image. By adding the low and high pass frequency maps together and then converting it back into the image, we can highlight the edges while blurring out the insides of the objects, which sharpens the image.

3. Implementation

We keep the same function call structure as the previous project, where we need to supply input/output file names, number of ROIs, rotation angle, and then each ROI defined by size, function, and parameters.

These functions are built upon the previous projects and work on the ROI scheme.

High pass, low pass, and band-stop filters are available for both gray and color images. Unsharp masking is only available for gray images.

3.1. High pass, lowpass, and band-stop filters

For these functions, we first read in and save the ROI information from the image to an OpenCV Mat object. We then convert it from an 8-bit character representation to a float representation and perform a DFT on it using OpenCV's built in function.

We then call a helper function which creates a circle filter using the radius, and an inside/outside color which we can determine depending on if we are performing a high or low pass. Band stop works similarly, but we have 2 radii instead.

At this point we have a Fourier domain representation of the image and the filter we want to apply. To apply the filter, we do an element wise multiplication between the filter image and the Fourier domain image.

To get our final image, we do an inverse Fourier transform on our filtered Fourier domain image.

Color images need a little bit more overhead. We first need to convert the 3 RGB channels into the HSV color space and perform the filtering only on the channels the user specifies. After filtering, we then convert all of the channels back into the RGB color space so that we can properly display them. OpenCV's HSV color space has a different range of values than the RGB space. The H channel has a range from 0-179, while the S and V channels have a range from 0-255. Particular care was taken to ensure that we normalize the values of the filtered image with respect to the range of values for its channels.

3.2. Unsharp masking

For unsharp masking, we do all the work in the Fourier domain. We first convert the image to its Fourier domain representation using DFT. We then find a low and a high pass filter for the input image using the same radius for both. We then multiply the high pass filter by 1 + some alpha value, increasing the contrast between the high frequency portions (edges) and the rest of the image. We then add the two filters together and perform idft to get the output image.

4. Description and Analysis of Results

The results of our algorithms used are displayed here. All images have their frequency maps before and after filtering. Not all images are provided in the report, but they are generated by the program. For each ROI, we generate three images (filter, frequency domain before filtering, and frequency domain after filtering). We have three ROIs per image, and around twenty-five original images, plus images with no filters applied. This results in over three hundred total generated images.

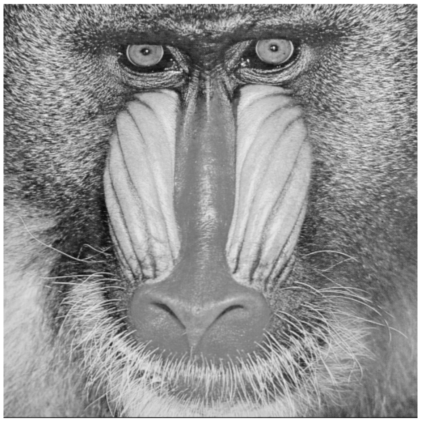
For grey images:

- 1 original file
- 3 rotated versions of the original file with 3 ROIs each. Rotations are 90, 180, and 270 degrees.
- 4 images with 3 ROIs each that have had a low pass filter applied to them. These are then rotated by 0, 90, 180, and 270 degrees.
  - For each ROI in each image, we also have additional images, the frequency maps before and after being filtered.
- 4 images with 3 ROIs each that have had a high pass filter applied to them. These are then rotated by 0, 90, 180, and 270 degrees.
  - For each ROI in each image, we also have additional images, the frequency maps before and after being filtered.
  -
- 4 images with 3 ROIs each that have been sharpened using unsharp masking. These are then rotated by 0, 90, 180, and 270 degrees.
  - For each ROI in each image, we also have additional images, the frequency maps before and after being filtered.
- 4 images with 3 ROIs each that have had a band-stop filter applied to them. These are then rotated by 0, 90, 180, and 270 degrees.
  - For each ROI in each image, we also have additional images, the frequency maps before and after being filtered.

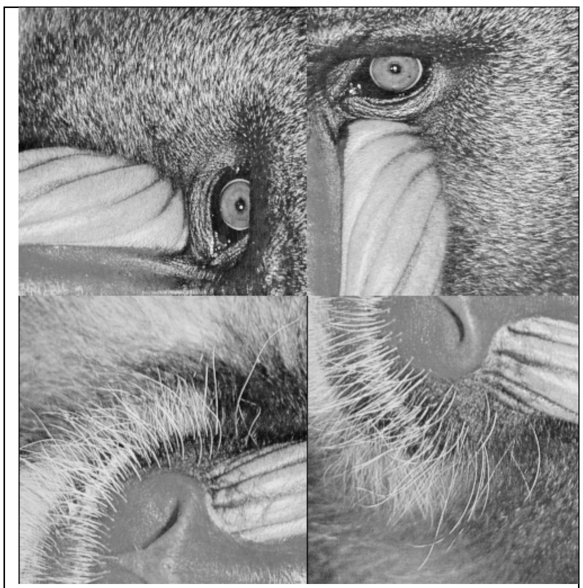
For color images:

- 1 original file
- 3 images with 3 ROIs for low pass filters. 1 image for each channel (H, S, V) and no rotations
- 3 images with 3 ROIs for high pass filters. 1 image for each channel (H, S, V) and no rotations
- 3 images with 3 ROIs for band stop filters. 1 image for each channel (H, S, V) and no rotations

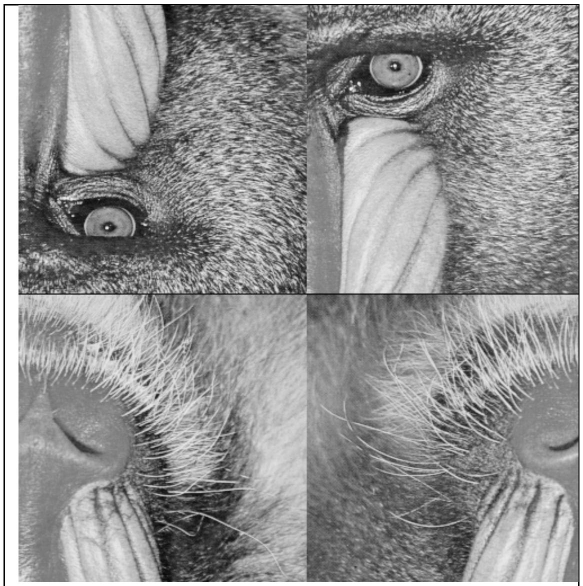
4.1. Grey Images



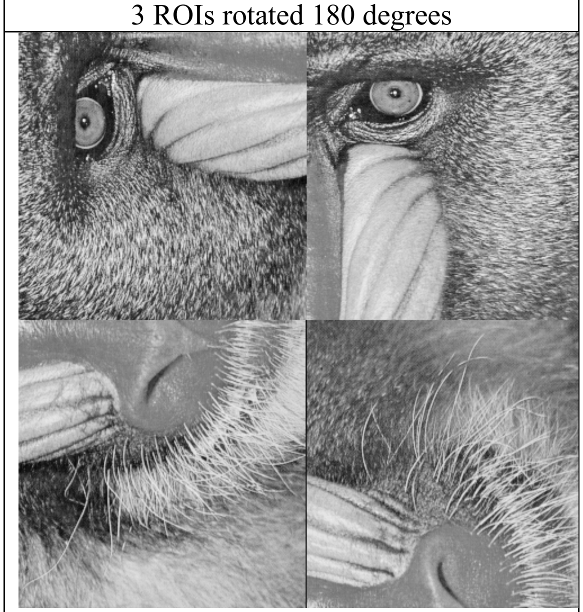
Original Image



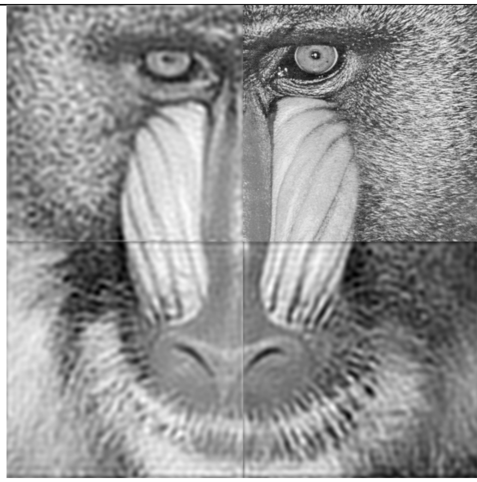
3 ROIs rotated 90 degrees



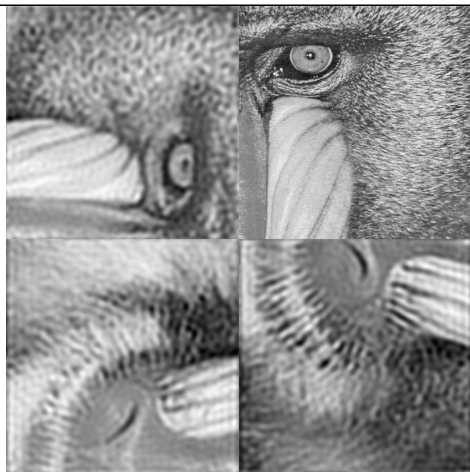
3 ROIs rotated 180 degrees



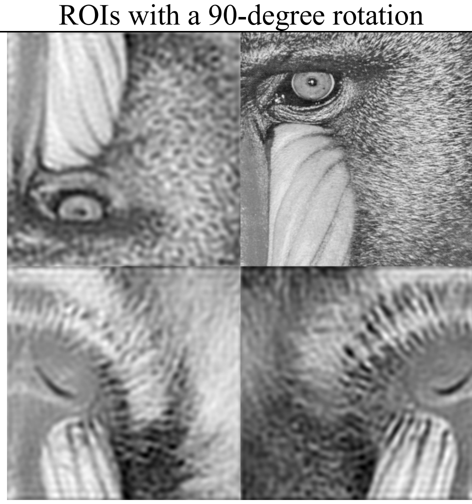
3 ROIs rotated 270 degrees



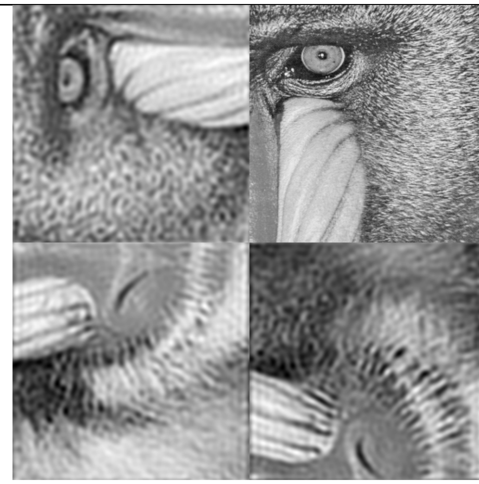
Lowpass with a radius of 30 applied to 3 ROIs with no rotation



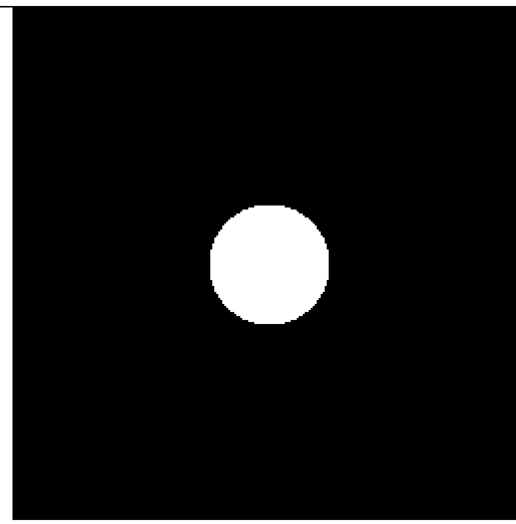
Lowpass with a radius of 30 applied to 30 ROIs with a 90-degree rotation



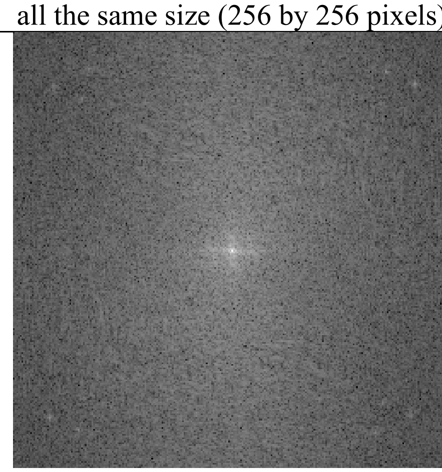
Lowpass with a radius of 30 applied to 30 ROIs with a 180-degree rotation



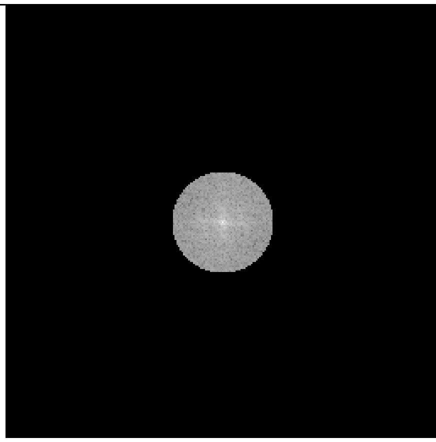
Lowpass with a radius of 30 applied to 30 ROIs with a 270-degree rotation



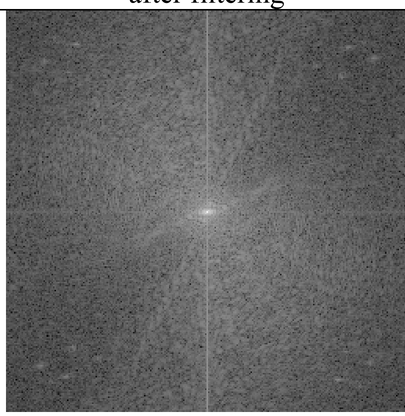
Filter for each ROI lowpass ROI as they are all the same size (256 by 256 pixels)



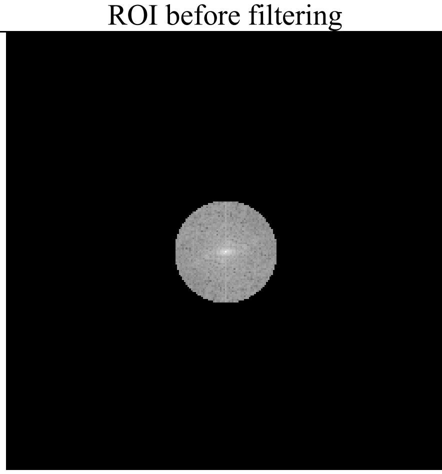
Frequency map for the 0 rotation top left ROI before filtering



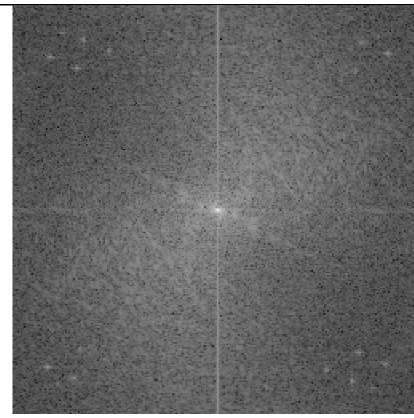
Frequency map for the 0 rotation top left ROI after filtering



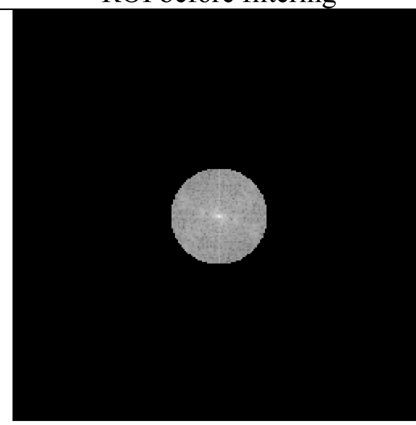
Frequency map for the 0 rotation bottom left ROI before filtering



Frequency map for the 0 rotation bottom left ROI after filtering

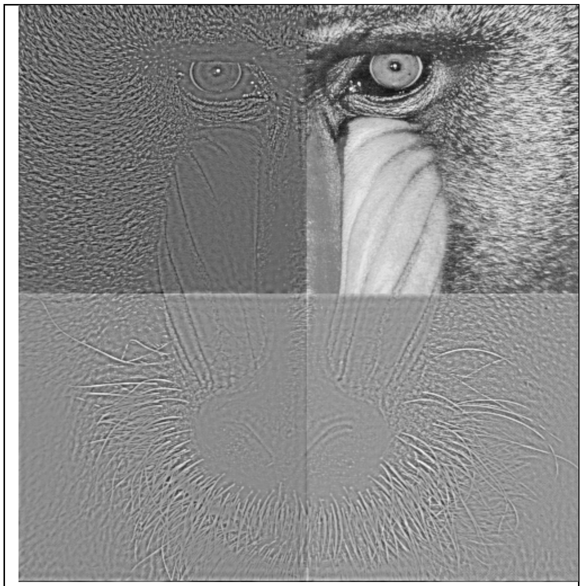


Frequency map for the 0 rotation bottom right ROI before filtering

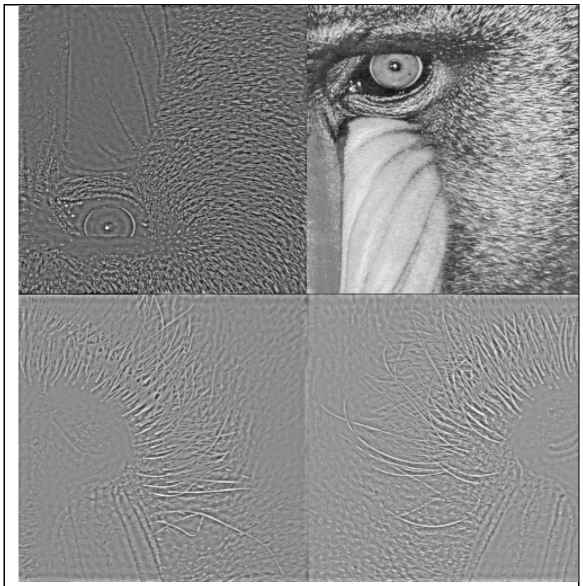


Frequency map for the 0 rotation bottom right ROI after filtering

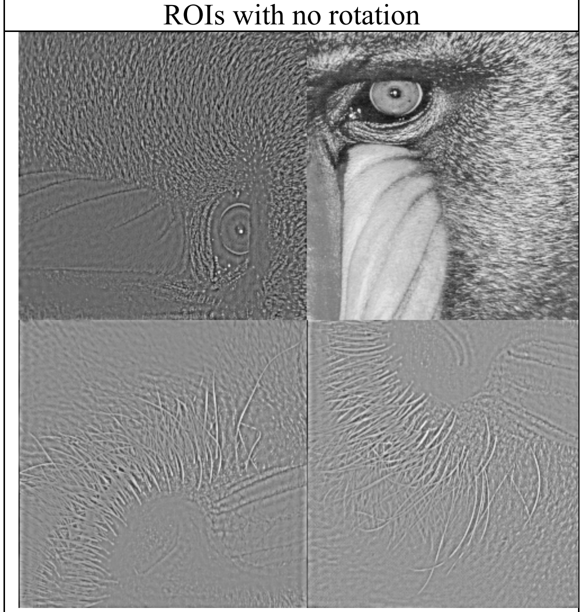
Images for the filter, frequency map before filtering, and frequency map after filtering for the 0-degree rotation lowpass image. The filter, and frequency map images for the rotated ROIs are not included in this report but are generated when the program is run.



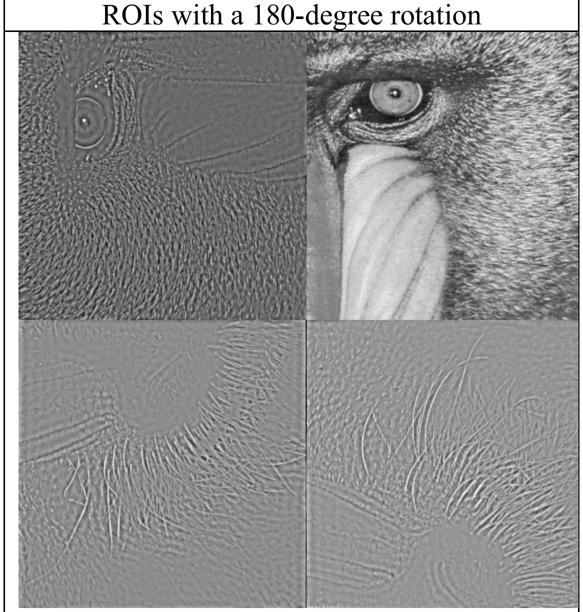
High pass with a radius of 30 applied to 3 ROIs with no rotation



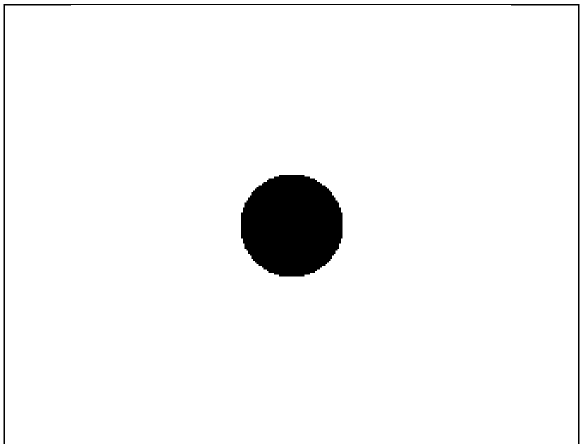
High pass with a radius of 30 applied to 30 ROIs with a 180-degree rotation



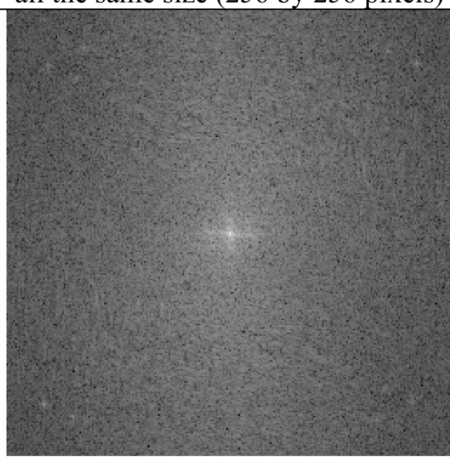
High pass with a radius of 30 applied to 30 ROIs with a 90-degree rotation



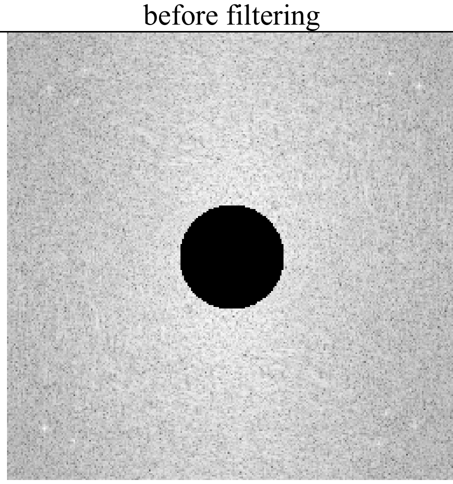
High pass with a radius of 30 applied to 30 ROIs with a 270-degree rotation



Filter for each ROI high pass ROI as they are all the same size (256 by 256 pixels)

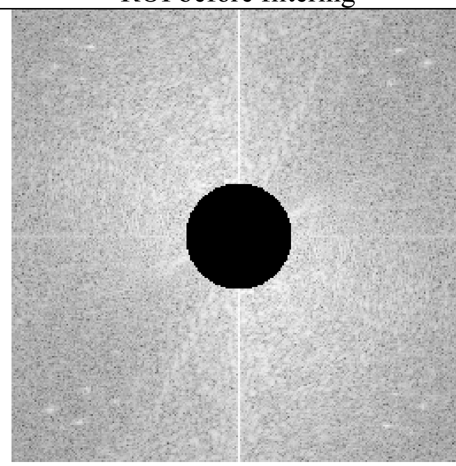


Frequency map for the 0-rotation top left ROI before filtering

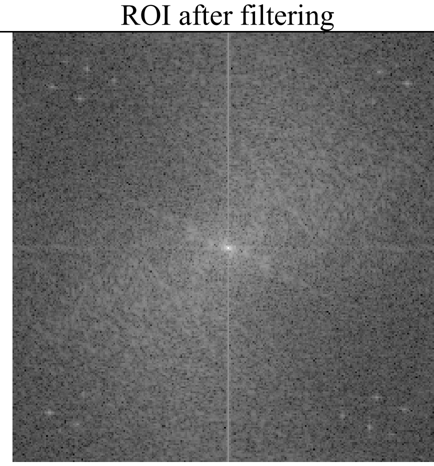


Frequency map for the 0-rotation top left ROI after filtering

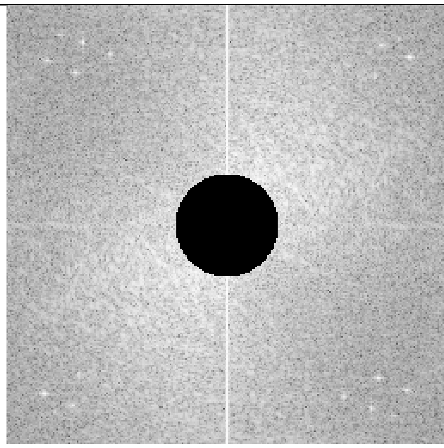
Frequency map for the 0 rotation bottom left ROI before filtering



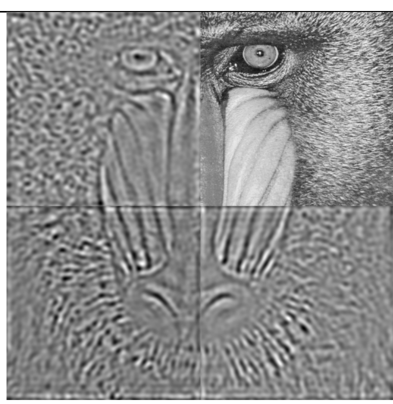
Frequency map for the 0 rotation bottom left ROI after filtering

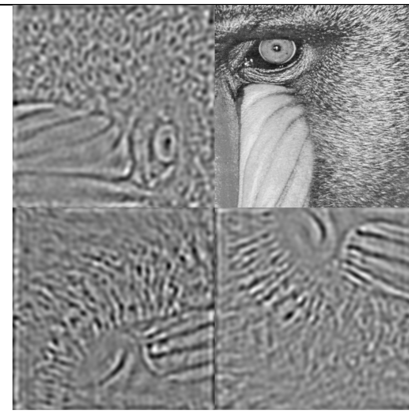


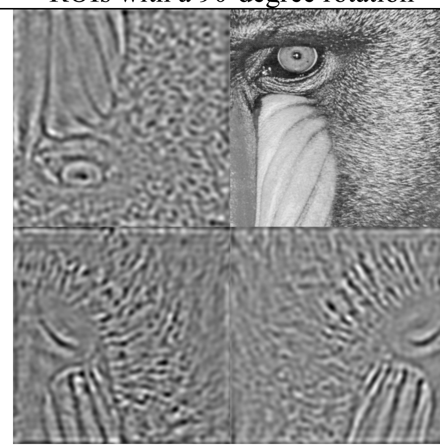
Frequency map for the 0 rotation bottom right ROI before filtering

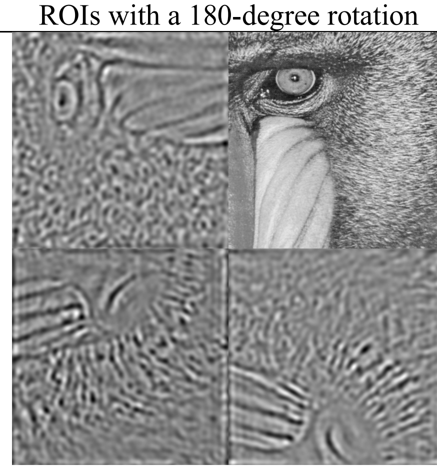

Frequency map for the 0 rotation bottom right ROI after filtering

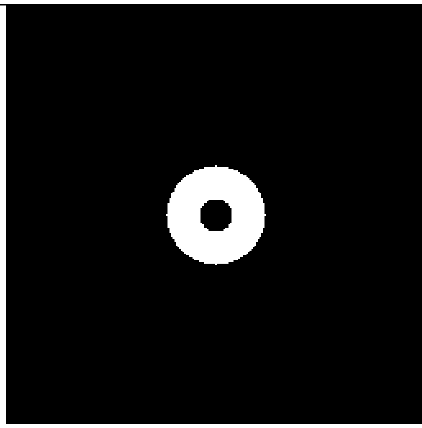
Images for the filter, frequency map before filtering, and frequency map after filtering for the 0-degree rotation high pass image. The filter and frequency map images for the rotated ROIs are not included in this report, but are generated when the program is run.

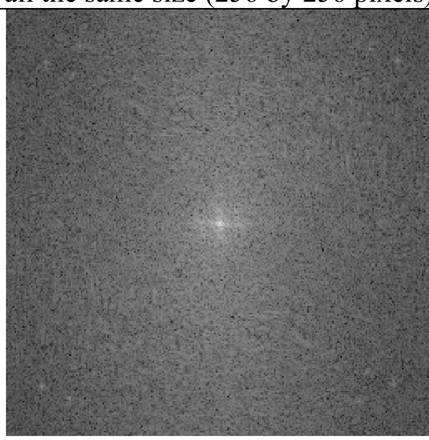

Band-stop with radii of 10 and 30 applied to 3 ROIs with no rotation

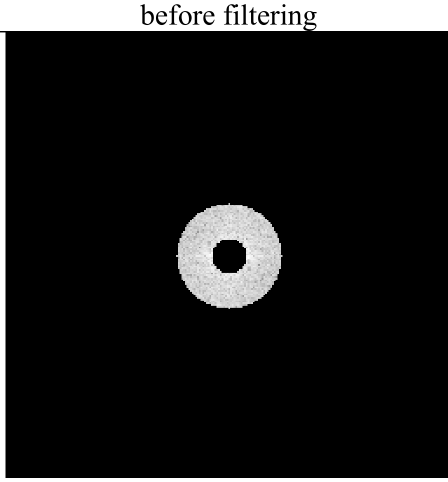

Band-stop with radii of 10 and 30 applied to 3 ROIs with a 90-degree rotation

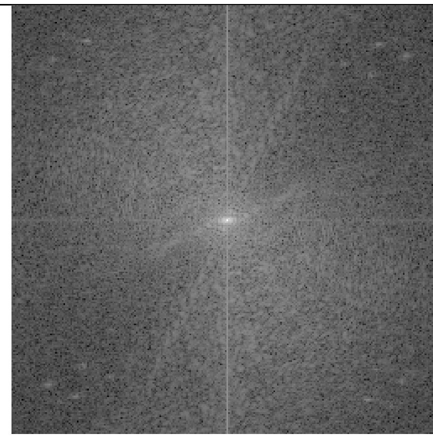

Band-stop with radii of 10 and 30 applied to 3 ROIs with a 180-degree rotation

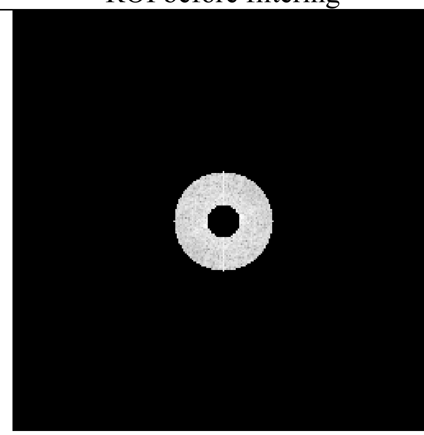

Band-stop with radii of 10 and 30 applied to 3 ROIs with a 270-degree rotation

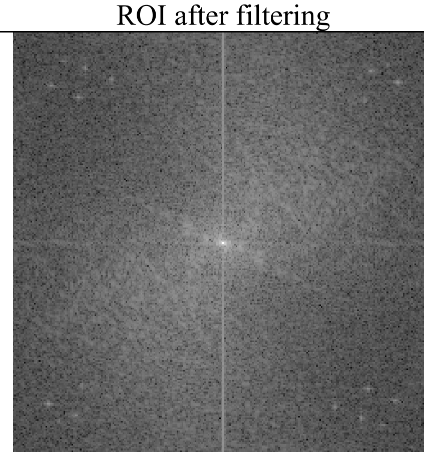

Filter for each ROI band-stop ROI as they are all the same size (256 by 256 pixels)

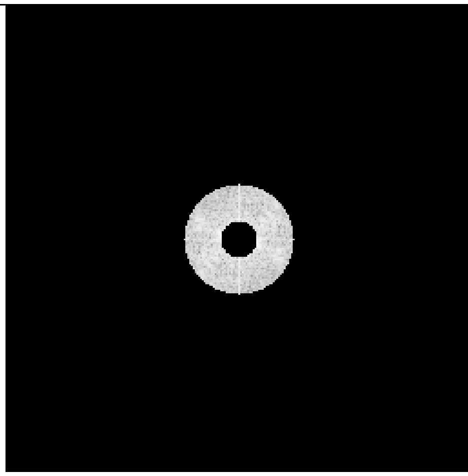

Frequency map for the 0-rotation top left ROI before filtering


Frequency map for the 0-rotation top left ROI after filtering


Frequency map for the 0-rotation bottom left ROI before filtering

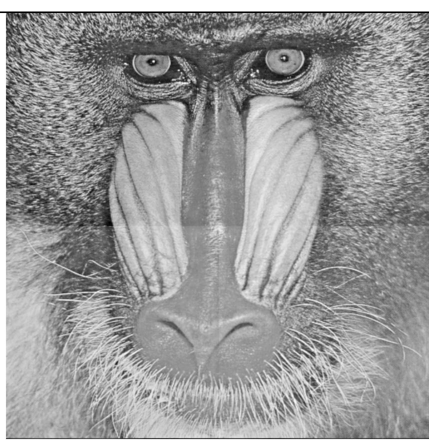

Frequency map for the 0-rotation bottom left ROI after filtering


Frequency map for the 0-rotation bottom right ROI before filtering

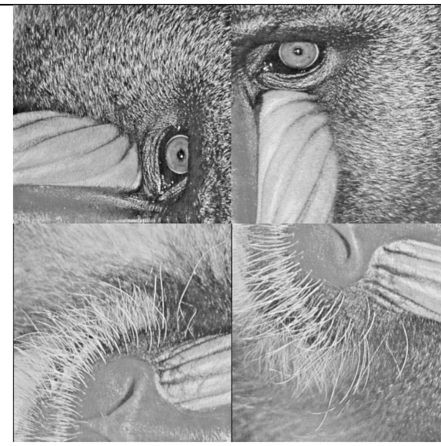


Frequency map for the 0 rotation bottom right ROI after filtering

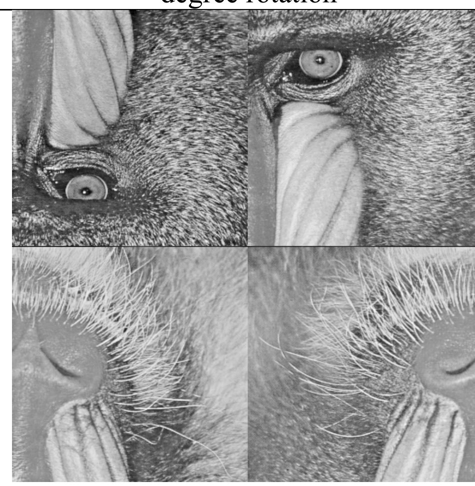
Images for the filter, frequency map before filtering, and frequency map after filtering for the 0-degree rotation band-stop image. The filter, and frequency map images for the rotated ROIs are not included in this report but are generated when the program is run.



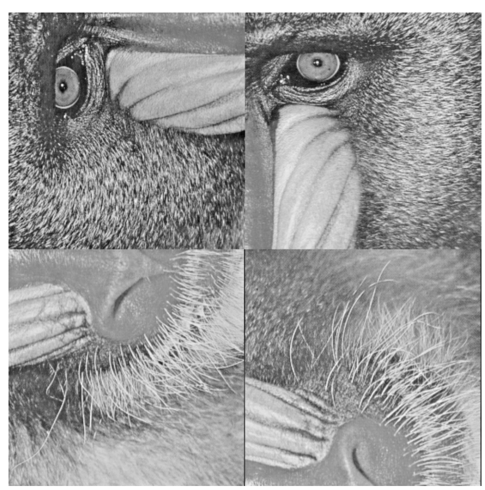
Unsharp masking with a radius of 30 and an alpha of 0.50 applied to 3 ROIs with no rotation



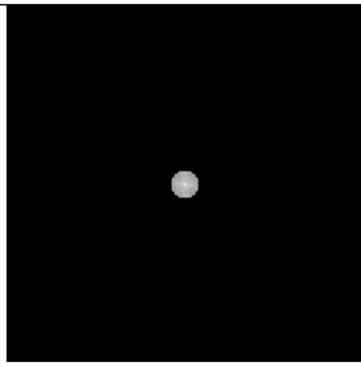
Unsharp masking with a radius of 30 and an alpha of 0.50 applied to 3 ROIs with a 90-degree rotation



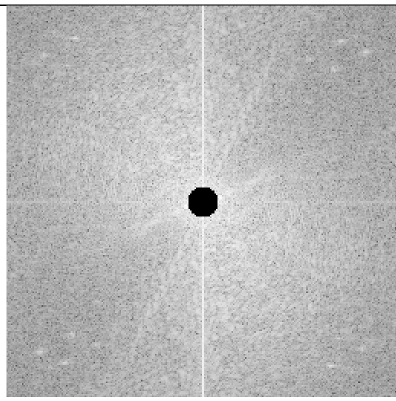
Unsharp masking with a radius of 30 and an alpha of 0.50 applied to 3 ROIs with a 180-degree rotation



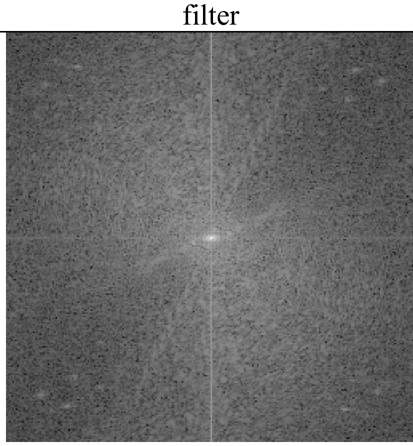
Unsharp masking with a radius of 30 and an alpha of 0.50 applied to 3 ROIs with a 270-degree rotation



Top left ROI for unsharp masking lowpass filter

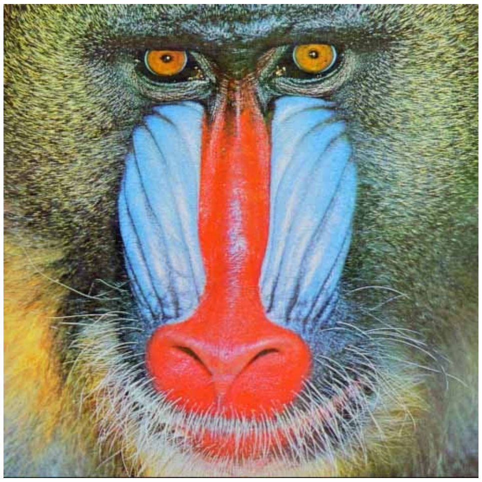


Top left ROI for unsharp masking high pass filter

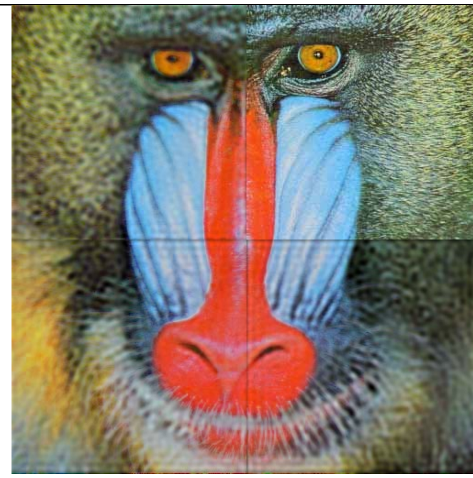


Top left ROI for unsharp masking combined filter

4.2. Color Images

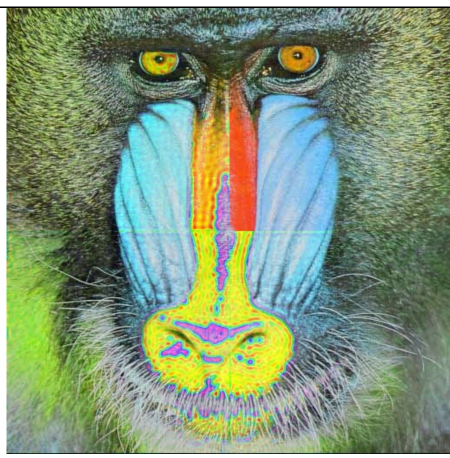


Original Color Image

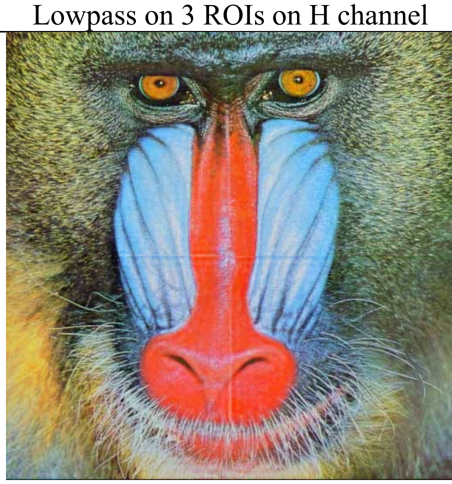


Lowpass on 3 ROIs on V channel

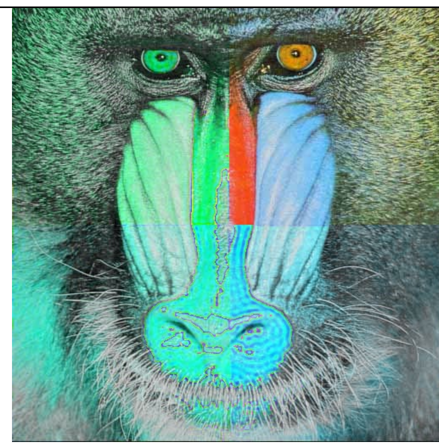
Low pass in color space on different channels. Images for filters, and before/after magnitude spectrums are generated but not included in the report. The radius for these tests was 30.



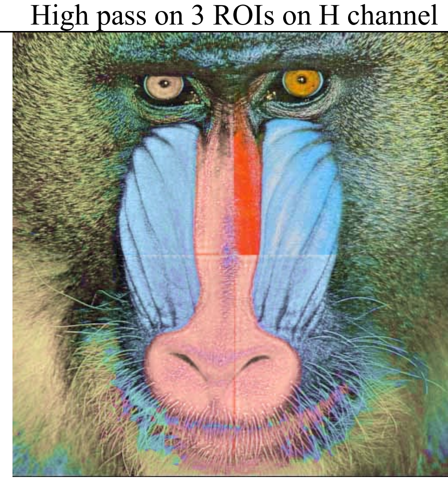
Lowpass on 3 ROIs on H channel



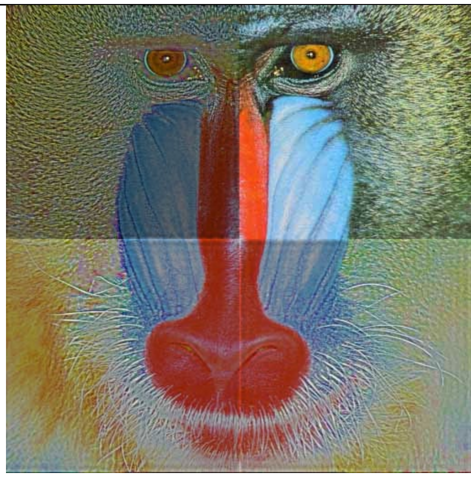
Lowpass on 3 ROIs on S channel



High pass on 3 ROIs on H channel

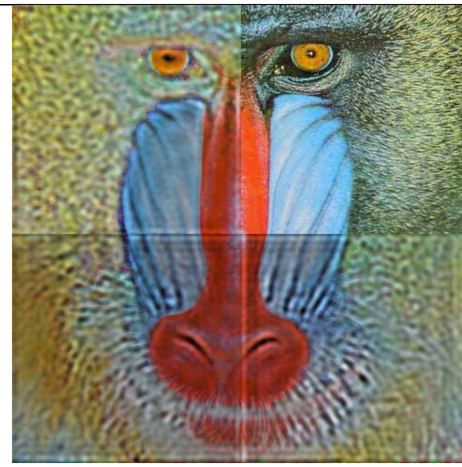


High pass on 3 ROIs on S channel



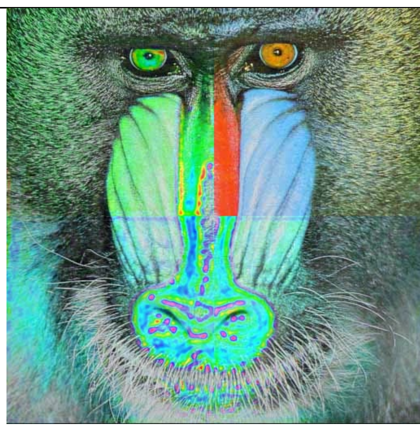
High pass on 3 ROIs on V channel

high pass in color space on different channels. Images for filters, and before/after magnitude spectrums are generated but not included in the report. The radius for these tests was 30.



Band stop filter on 3 ROIs on V channel

Band stop in color space on different channels. Images for filters, and before/after magnitude spectrums are generated but not included in the report. The radii for these tests were 10 and 30.



Band stop filter on 3 ROIs on H channel

Band stop filter on 3 ROIs on S channel

4.3. Analysis

For grey and color images, high and low pass filters work well for detecting edges and blurring the image. On grey images, the lowpass filter clearly blurs the image, and the high pass clearly shows the edges on the baboon. The radius variable determines how much to blur the image or how pronounced the edge needs to be to get picked up by the filter.

Although we can perform these operations on any number of channels on color images, the “correct” output obtained when manipulating the V channel in the HSV color space. Applying a filter on the H channel changes the color of the output image drastically, which is to be expected as we are changing the hue of the image. Filtering the saturation channel has a minor effect of the deepness of color. The value channel is the correct one to manipulate if we want to maintain the color accuracy, and we can see the blurring and edge detecting effect it has.

Band-stop works on both color spaces and produces a blurred image with some edges being more pronounced.

Unsharp masking slightly sharpens the output image and is only available on the grey images.

These filters all work on the Fourier domain space and are easy to visualize as we can just create a black/white image of the same size as the input. By comparison, other popular blurring/edge detection techniques such as Gaussian blurring or Sobel edge detection use kernels that we need to pass across the entire image. Both approaches work, with some pros and cons for each. For the Fourier domain manipulations, we can encode the FFT function in hardware to make it run extremely efficiently, or we can set up a table of the DFT of the image and multiple filters to quickly apply different high/lowpass filters to the image.

5. Conclusion

In this project, we were tasked with implementing common image filtering techniques using the Fourier transform function. We added blurring, edge detection, and image sharpening using high pass, lowpass, or band stop filters on both grey and color images to our image processing library.

Example inputs and further function documentation are available in the project’s readme file. Some example inputs are listed below:

Example 8:

infile.pgm outfile.pgm 1 0 ROI 0 0 256 256 highpass 30 //correct

infile.pgm outfile.pgm 1 0 ROI 0 0 256 256 lowpass 30 //correct

infile.pgm outfile.pgm 1 0 ROI 0 0 256 256 bandstop 10 30 //correct

infile.pgm outfile.pgm 1 0 ROI 0 0 256 256 unsharpMasking 30 0.50 //correct

infile.pgm outfile.pgm 1 0 ROI 0 0 256 256 bandstop 40 10 //incorrect

//the first radius needs to be smaller than the second

infile.pgm outfile.pgm 1 0 ROI 0 0 256 245 unsharpMasking 30 20 //incorrect

//the second argument needs to be a floating-point number

Example 9:

infile.ppm outfile.ppm 1 0 ROI 0 0 256 256 highpass v 30 //correct

infile.ppm outfile.ppm 1 0 ROI 0 0 256 256 lowpass h 30 //correct

infile.ppm outfile.ppm 1 0 ROI 0 0 256 256 bandstop hsv 10 30 //correct

infile.ppm outfile.ppm 1 0 ROI 0 0 256 256 lowpass h s V 30 //incorrect

//the color channels cannot have spaces between them