3.2.1 Smoothing and denoising

|  |  |
| --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_filtered.JPG | HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_filtered.JPG |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_filtered.JPG | HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_filtered.JPG |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_filtered.JPG | HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_filtered.JPG |
| Fig 1. In the left column are Gaussian-blurred images, and in the right column are median-blurred images. The filter size for each row from top to bottom is 3x3, 9x9, and 27x27. For the Gaussian blur, sigma was set to 3 for each filter. Blurring was performed on the original puppy image. | |

As the size of both of these blurring filters increases, so does the smoothing effect. In both cases, there seems to be a tradeoff between denoising and blurring. The bigger the filter, the more denoising but also more blurring. The takeaway is that the filter size must be chosen in a way that satisfies our requirements for denoising (maybe for visual purposes or edge detection) but does not impact edges too much (i.e. blurring). This tradeoff seems to be more extreme in the median case, but this may simply be due to the particular sigma chosen for Fig 1 (sigma=3). Also, the median filter has this sort of downsampling effect, where the number of color bins is decreased (i.e. the number of colors in the color map gets smaller). This is especially apparent in the low frequency areas of the 27x27 median filtered image. On the other hand, the Gaussian filter does not seem to succumb to this effect in the examples presented in Fig 1.

|  |  |
| --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-050_filtered.JPG | HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-050_filtered.JPG |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-050_filtered.JPG | HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-050_filtered.JPG |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-050_filtered.JPG | HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-050_filtered.JPG |
| Fig 2. In the left column are Gaussian-blurred images, and in the right column are median-blurred images. The filter size for each row from top to bottom is 3x3, 9x9, and 27x27. For the Gaussian blur, sigma was set to 3 for each filter. Blurring was performed on the noisy puppy image (0.50). | |

Again with the median filter, there is some reduction in the number of levels of color in the 27x27 filter case. Comparing it with Gaussian filter, it seems to have similar performance on the 3x3 and 9x9 filters but clearly is worse in the 27x27 case. In general median filters are good at removing certain kinds of noise, such as salt and/or pepper noise. Although, the puppy image does not seem to be impacted by this kind of noise.

3.2.2 Edge detection

|  |  |  |
| --- | --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_edge.jpg | HD:Users:thomas:Documents:fun:cs691:proj1:window-00-00.jpg | HD:Users:thomas:Documents:fun:cs691:proj1:window-00-00_edge.jpg |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-0.25_edge.jpg | HD:Users:thomas:Documents:fun:cs691:proj1:window-06-05.jpg | HD:Users:thomas:Documents:fun:cs691:proj1:window-06-05_edge.jpg |
| Fig 3. Canny edge detection. Left column is puppy (top original; bottom noisy). Right two columns are satellite images with respective Canny edges. High and low thresholds of 200 and 100 were used arbitrarily. | | |

Based on Fig 3 and Canny edge detection, it would seem that noise and high frequencies have significant effect on its performance. In the case of the puppy, the noise introduced fake edges in the puppy’s face as well as in the low frequency areas (top right and bottom right). In the case of the satellite images, the blurrier image (top) resulted in fewer edges than the more high frequency image (bottom). This can especially be observed in the tree areas. These examples suggest that simply blurring (removal of high frequencies) before edge detection would improve overall performance, and indeed, this is standard in practice. The questions then become how much blurring to do beforehand and how to choose Canny thresholds.

4.2 Frequency analysis

|  |  |
| --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:00_00_ft.png | HD:Users:thomas:Documents:fun:cs691:proj1:06_05_ft.png |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_ft.pngHD:Users:thomas:Documents:fun:cs691:proj1:00_00_ft.png | |
| Fig 4. Log visualizations of Fourier transforms (y=0) for two satellite images (top) and superimposed satellite images and puppy (bottom). The more intense of the two overlaid is the puppy FT. Images were converted to greyscale prior to transformation. | |

There seems to be a correlation between the amount of high frequency content in the image and the decay in the magnitude of the Fourier transform. This makes intuitive sense in the case of the puppy and satellite images, as the hair of the puppy is high frequency information while the satellite image (00\_00) in this case has relatively less high frequency information (mostly in the trees). As a result, we see a slower decay. This is actually even more pronounced, as the x axes of the two superimposed plots are not scaled correctly.

|  |
| --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259_ft.pngHD:Users:thomas:Documents:fun:cs691:proj1:DSC_9259-0.50_ft.png |
| Fig 5. Log visualized Fourier transforms of puppy image (black=original, blue=noisy) from one band (y=0). Images were converted to greyscale prior. |

Indeed the same effect is observed in noisy images. This makes sense, as noise is high frequency. In the Fourier transform, we can see that there is a greater magnitude of higher frequencies in the noisy image (blue) than in the original image (black).

To address the question posed in the third bullet point, the reason that we see brighter values along the axes of the log plots of the Fourier coefficients is that strong vertical and horizontal edges exists in the images. The orientation of these ‘lines’ in the coeffiecient plots correspond to the direction of change of the edges (i.e. the normal vector) in the spatial domain. For example, a vertical line in the coefficient plot corresponds to (relatively) strong horizontal edge in the original image.

|  |  |  |
| --- | --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:business_fish.png | HD:Users:thomas:Documents:fun:cs691:proj1:business_fish_lowpass.png | HD:Users:thomas:Documents:fun:cs691:proj1:business_fish_highpass.png |
| Fig 6. Result of zeroing frequencies near the origin of the Fourier coefficient plot (right) and zeroing frequencies not near the origin (middle). Implemented using an ideal low pass (middle) and high pass (right) filter on business fish (left). Image converted to greyscale prior. | | |

It seems that the effect of zeroing frequencies near the origin of the Fourier coefficient plot results in an edge detection. Notice in the right most image of Fig 6 that the most intense (white) areas correspond to edges in the original image. Although, it doesn’t perform well, as there is some decay (the glowy bits) around the true edges. Though, this may have something to do with the original image that we can’t see. The effect of zeroing Fourier coefficients away from the origin results in a very blurry image. Both of these results make intuitive sense, as the former completely throws away low frequencies and the latter throws away high frequencies. Zeroing out select frequencies could be useful for filtering out periodic noise (band reject filters and Notch filters). It is shown here that zeroing out low frequencies results in approximate edge detection, and the opposite results in a decent blurring. However, in both cases, zeroing (ideal low/high pass filters) often result in ringing, which isn’t shown well in Fig 6.

5.2 Low-pass and High-pass filtering

|  |  |  |
| --- | --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:bf_LowPass.png | | HD:Users:thomas:Documents:fun:cs691:proj1:bf_HighPass.png |
| HD:Users:thomas:Documents:fun:cs691:proj1:bf_ButterLowPass.png | HD:Users:thomas:Documents:fun:cs691:proj1:bf_ButterHighPass.png | |
| Fig 7. Low-pass filters (left), high-pass filters (right), ideal (top), Butterworth (bottom). Cutoff frequency was 0.5, and order of Butterworth was 1. | | |

I chose business fish for this last example because of a unique effect it produced which I could not see in the images provided. The low frequency areas in the ideal low-pass filter have a sort of wavy feature emanating from the edges. I believe this is known as ringing and is due to the zero crossings of the corresponding spatial domain filters. It looks like a sinc function. Below is a sinc function and a column of the idea low-pass filter applied to business fish (above).

|  |  |
| --- | --- |
| HD:Users:thomas:Documents:fun:cs691:proj1:ringing.png |  |
| Fig 8. Ringing from the ideal low-pass filtered business fish. | Fig 9. sinc function – this is what the ideal low-pass filter looks like in the spatial domain. Note the zero crossings and sinusoid. |

Overall, the Butterworth filter seems to have higher performance, as it lacks the ringing observed in the low-pass ideal counterpart and the edges are crisper in high-pass counterpart. This is due to the (more) gradual attenuation of the filter in the frequency domain rather than the sharp, square-wave-like drop of the ideal low-pass filter. However, with higher orders of n, the spatial domain Butterworth filter begins to resemble the sinc function and thus produces ringing. This is why we tend to keep n small.