Edge-preserving Smoothing using Bilateral Filtering.

Corrupt the image with independent and identically-distributed additive zero-mean Gaussian noise with standard deviation set to 5% of the intensity range. Note: in Matlab, randn() gives random numbers drawn independently from a Gaussian with mean 0 and standard deviation 1 .

Write code for bilateral filtering (standard “slow” algorithm is also fine) and apply it (one pass over all pixels) to all the input images. For efficiency in Matlab, the code should, ideally, have maximum 2 “for” loops to go over the rows and columns of the image. At a specific pixel “p”, the data collection with a window, weight computations, and weighted averaging can be performed without using loops.

Tune the parameters (standard-deviations for Gaussians over space and intensity) to minimize the RMSD between the filtered and the original image.

• Write a function myBilateralFiltering.m to implement this.

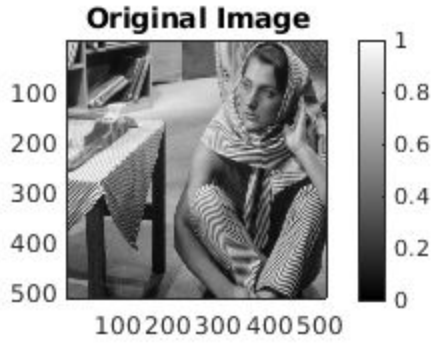
• Show the original, corrupted, and filtered versions side by side, using the same (gray) colormap.

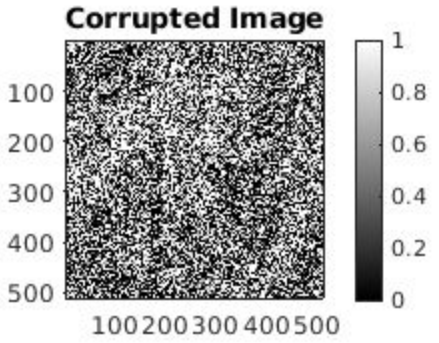
• Show the mask for the spatial Gaussian, as an image.

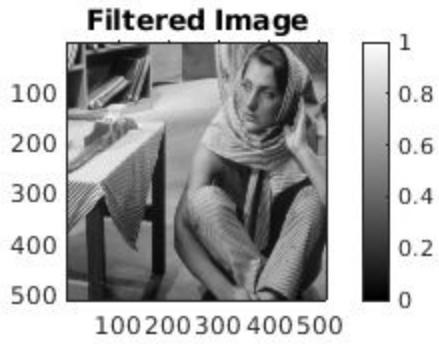
• Report the optimal parameter values found, say σ space
and σ intensity along with the optimal RMSD.

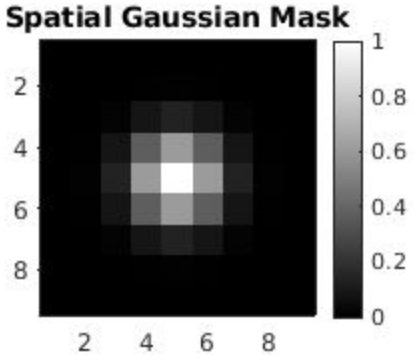
• Report RMSD values for filtered images obtained with (i) 0.9σ space
and σ intensity, (ii) 1.1σ space and σ intensity, (iii) σ space and 0.9σ intensity, and (iv) σ space and 1.1σ intensity, with all other parameter values unchanged.

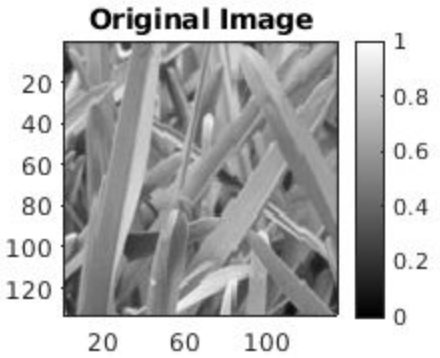
The original, corrupted , filtered versions and spatial Gaussian mask are shown below:

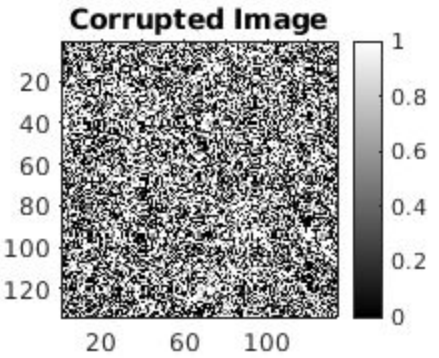

Original Image
100
200
300
400
500
100 200 300 400 500
1
0.8
0.6
0.4
0.2
0

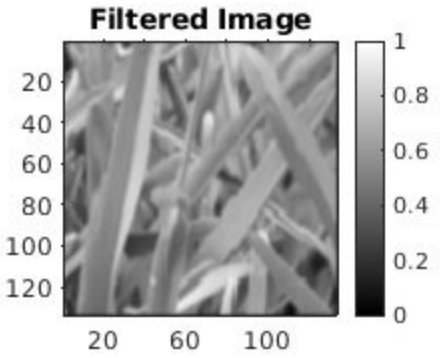

Corrupted Image
100
200
300
400
500
100 200 300 400 500
1
0.8
0.6
0.4
0.2
0

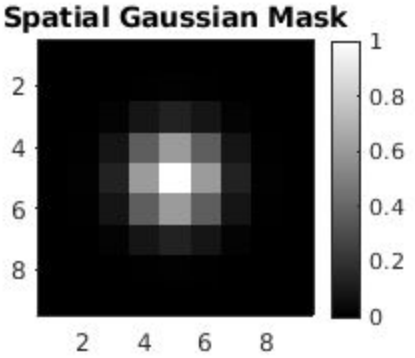

Filtered Image
100
200
300
400
500
100 200 300 400 500
1
0.8
0.6
0.4
0.2
0

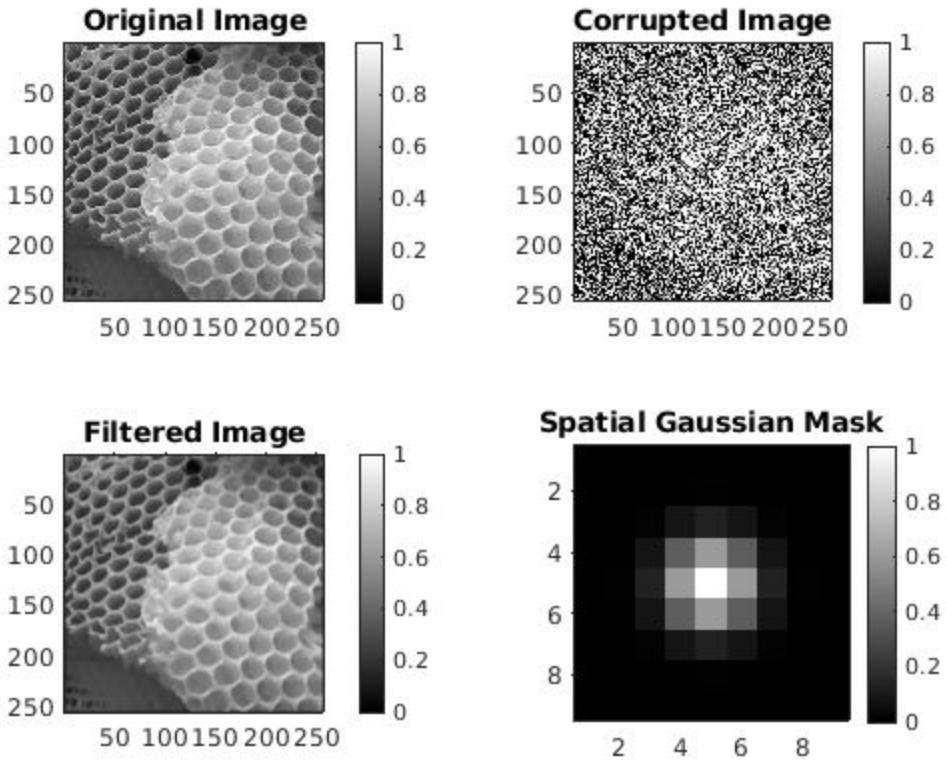

Spatial Gaussian Mask
2
4
6
8
2 4 6 8
1
0.8
0.6
0.4
0.2
0


Original Image
20
40
60
80
100
120
20 60 100
1
0.8
0.6
0.4
0.2
0


Corrupted Image
20
40
60
80
100
120
20 60 100
1
0.8
0.6
0.4
0.2
0


Filtered Image
20
40
60
80
100
120
20 60 100
1
0.8
0.6
0.4
0.2
0


Spatial Gaussian Mask
2
4
6
8
2 4 6 8
1
0.8
0.6
0.4
0.2
0



As we can see the filtered images have been smoothed out. However some of the edges are preserved.
The corrupted image has been corrupted by adding 5% of the intensity of the image thus it appears totally distorted.

The optimal parameter appears to be around sigma_spatial=1.3, sigma_intensity=5 with
RMSD=0.0599 (for barbara.mat)
RMSD=0.0491(for grass.png)
RMSD=0.0485(for honeycombReal.png)

For sigma_spatial=1.17, sigma_intensity=5
RMSD=0.0588(for barbara.mat)
RMSD=0.0494(for grass.png)
RMSD=0.0444(for honeycombReal.png)

For sigma_spatial=1.43, sigma_intensity=5
RMSD=0.0632(for barbara.mat)
RMSD=0.0533(for grass.png)
RMSD=0.0524(for honeycombReal.png)

For sigma_spatial=1.3, sigma_intensity=4.5
RMSD=0.0599(for barbara.mat)
RMSD=0.0491(for grass.png)
RMSD=0.0485(for honeycombReal.png)

For sigma_spatial=1.3, sigma_intensity=5.5
RMSD=0.0599(for barbara.mat)
RMSD=0.0491(for grass.png)
RMSD=0.0486(for honeycombReal.png)