# EE420 – Digital Image Processing

# Homework-2

Assigned: Jan 28, 2021 Due: Feb 09, 2021

Maximum Possible Grade: 70 points

# General guidelines

Please upload your response to Canvas as a ZIP file with the following filename convention

* Name of ZIP file FIRSTNAME\_LASTNAME\_Homework2.zip
* Replace FIRSTNAME with your first name
* Replace LAST NAME with your last name

**Failure to adhere to the filename convention will result in deduction of points.**

Provide a detailed response to each question, including supporting mathematical arguments. Include screenshots of image before and after image processing. Failure to do will result in deduction of points.

# Submitting your solutions

Please ensure that the ZIP file uploaded to Canvas, includes the following components:

* PDF file of your write-up including response to all questions.
* Completed MATLAB Source code for
  1. HWK2\_GrayScaleDenoising\_using\_BLFilter.m
  2. gray\_bilateral\_filter.m
  3. HWK2\_ColorImageFiltering\_using\_BLFilter.m
  4. bilateral\_filter.m
  5. HWK2\_ImageEnhancement\_using\_CrossBLFilter.m
  6. joint\_bilateral\_filter.m

**Failure to adhere to the filename convention will result in deduction of points.**

# Objective

This homework is designed to introduce you to multiple applications of bilateral filtering. As a part of the assignment you are required to implement a bilateral filter, joint/cross bilateral filter and use these in denoising grayscale and color images.

Please refer to the slide deck from Lecture-6 for an introduction to Bilateral filtering.

### *Recommended reading material*

* [*Bilateral Filtering: Theory and Applications*](https://people.csail.mit.edu/sparis/publi/2009/fntcgv/Paris_09_Bilateral_filtering.pdf), by S. Paris, P. Kornprobst, J. Tumblin, F. Durand (Attached in the zip file provided)
* [*A Gentle Introduction to Bilateral Filtering and its Applications*](http://people.csail.mit.edu/sparis/bf_course/course_notes.pdf), SIGGRAPH 2004 course notes (Attached in the zip file provided)
* [*Digital Photography with Flash and No-Flash Image Pairs*](http://hhoppe.com/flash.pdf), Petschnigg, G., Szeliski, R., Agrawala, M., Cohen, M., Hoppe, H., & Toyama, K. (2004). ACM transactions on graphics (TOG), 23(3), 664-672.

# Bilateral Filtering Best Practices

Please take care to normalize the Bilateral kernel before use. Slide-49 of Lecture-6 of the attached slide deck provides details on normalization. The position variablerepresents pixel coordinates of the spatial location under consideration, while represents pixel coordinates of a spatial location in the neighborhood of .

### *Design parameters*

Two design parameters that affect the performance of the bilateral filter are the standard deviation of the shape kernel and the standard deviation of the intensity kernel .

In so far as selecting is concerned, I’d encourage starting with a small value of and increasing in steps of . Please exercise caution when handling attempting to filter pixels near the image boundaries. Upon centering the Gaussian at a pixel you may find that multiple neighboring pixels lie outside the image domain where are the number of rows and columns in the image.

Selecting the value for is a bit more involved and is typically chosen to be greater than or equal to or in the image. A reliable method to estimate the noise variance is to identify the intensity variance of a region/patch in the image that likely has constant intensity in the absence of noise. It is strongly recommended that you read MATLAB help on bilateral filtering (using imbilatfilt) before you attempt to identify . The help page provides clues on selecting , which MATLAB refers to as the Degree of Smoothing parameter.

Two things to keep in mind when selecting

* As the range parameter increases, the bilateral filter becomes closer to Gaussian blur because the Gaussian intensity kernel is flatter (almost a constant over the neighborhood of interest). If you want to preserve edges, choose smaller value for .
* Increasing the spatial parameter smooths larger features and increases the size of the neighborhood. If you want to denoise more, choose larger values for .

A final parameter to consider in your implementation of the bilateral filter is the size of the neighborhood over which the spatial and intensity kernels will be evaluated. A conservative choice for the size of the neighborhood is . It is good practice to pick a neighborhood with odd number of pixels, so that you have an identical number of neighboring pixels, surrounding the central pixel. The following code snippet may help in your attempts to identify the correct filter size given :

filt\_size = 2\*ceil(3\*sigma\_s)+1; % filter size

# Grayscale image denoising using Bilateral filter (25 points)

The first part of this assignment is concerned with edge preserving denoising of grayscale images using the Bilateral filtering concept. The starter code HWK2\_GrayScaleDenoising\_using\_BLFilter.m supplied with the assignment loads a noisy binary image and attempts to denoise the image using your implementation of the Bilateral filter.

Please ensure that your implementation of the bilateral filter adheres to the following conventions

***Inputs:*** user supplied image, ,

***Outputs:*** result of Bilateral filtering, result of Gaussian filtering

Failure to adhere to these conventions will result in deduction of points.

Your task is to complete the implementation of bilateral filter (in gray\_bilateral\_filter) and compare the results for different values of and .

*Implementing Gaussian filtering using the code for Bilateral filtering:* Recall that the Bilateral filter kernel is the product of a Gaussian spatial filter kernel and a Gaussian intensity kernel . Consequently, you can identify the Gaussian filtered image by simply computing a second image wherein the intensity kernel is disregarded. Comparing the output of the Bilateral and the standard Gaussian spatial filter is useful as it gives you a sense of the effectiveness of the edge-preserving character of the Bilateral filter.

**WARNING:** You may not invoke the MATLAB function imbilatfilt in your implementation of gray\_bilateral\_filter. You may not use the MATLAB functions imfilter & fpsecial to implement Gaussian filtering.

**WARNING:** If you chose to copy the MATLAB implementation of bilateral filtering available on the internet make sure to cite the code and your reasons for using the code. Failure to do so constitutes academic dishonesty and grounds for disciplinary action (<https://www.smu.edu/simmons/AboutUs/HonorCode>).

**Deliverables & Questions**

1. Completed MATLAB code for HWK2\_GrayScaleDenoising\_using\_BLFilter.m and gray\_bilateral\_filter. (10 points)
2. Description of key steps in MATLAB implementation with references to relevant line numbers.

Be as detailed as possible in your explanation. (7 points)

1. How are pixels near the image boundary being handled? (3 points)
2. What value of did you choose and why? (1 point)
3. What value of did you choose and why? (2 points)
4. Screenshots of the original image and the denoised result using Bilateral and Gaussian filtering. Label each screenshot clearly. Failure to do so will result in deduction of points. (2 points)

# Color image processing using Bilateral filter (20 points)

The second part of this assignment is concerned with edge preserving filtering of color images using the Bilateral filtering concept. The starter code HWK2\_ColorImageFiltering\_using\_BLFilter.m supplied with the assignment loads the **coloredChips.png** image included with MATLAB and attempts to filter the image using your implementation of the Bilateral filter.

Please ensure that your implementation of the bilateral filter adheres to the following conventions

***Inputs:*** user supplied image, ,

***Outputs:*** result of Bilateral filtering, result of Gaussian filtering

Failure to adhere to these conventions will result in deduction of points.

Your task is to complete the implementation of bilateral filter (in bilateral\_filter) and compare the results with MATLAB’s built-in implementation. Recall that the Bilateral filter kernel is the product of a Gaussian spatial filter kernel and a Gaussian intensity kernel. Consequently, you can identify the Gaussian filtered version of the image by simply computing a second image wherein the intensity kernel is disregarded. Comparing the output of the Bilateral and the standard Gaussian spatial filter is useful as it gives you a sense of the effectiveness of the edge-preserving character of the Bilateral filter.

*Dealing with color:* The simplest approach to filtering color images is to process each color channel separately. There are advantages to processing color images in a color space other than RGB, such as Lab (Luminance L, Chrominance a,b space). They tend to produce results with far fewer visually perceptible artifacts. The starter code supplied with this assignment, converts the RGB image to Lab image, attempts bilateral filtering using your implementation, and converts the filtered Lab image back to RGB.

**NOTE:** Bear in mind that the spatial kernel and the intensity kernel are common to all color channels (L,a,b). However, the definition of the intensity kernel used so far must be modified to accommodate the three-color channels in a color image. The revised expression for the intensity kernel of a color image is shown below:

where is the color vector associated with the pixel of the user-supplied image, and is the intensity of the color component. The above revised expression for the intensity kernel measures the dissimilarity in color instead of dissimilarity in gray value.

In order to filter a color image using a Bilateral filter, you must first identify , and then apply these filters to each color channel separately. The code for implementing the filtering operation is identical to the grayscale case. The difference lies in the approach to choosing the weights .

*Identifying :* The starter code supplied with the assignment identifies an image patch with largely constant color and computes the variance of the colors of the patch to identify :

patch = imcrop(imLAB\_noisy,[34,71,60,55]);

patchSq = patch.^2;

edist = sqrt(sum(patchSq,3));

patchStd = std2(edist);

patchVar = patchStd.^2;

*Identifying :* The starter code supplied with the assignment uses a pixels.

*MATLAB tips:* Suppose the MATLAB variable imLAB contains the Lab space representation of a user supplied color image. The rth-row, cth-column of the ellth color channel may be accessed as imLAB(r,c,ell). The entire ellth color channel may be addressed as an image using imLAB(:,:,ell). To create storage for a color image, use the following MATALB syntax: zeros(101,201,3).

*Additional Recommendations:* It is recommended that you read MATLAB help on bilateral filtering (using imbilatfilt) before you attempt this portion of the assignment. The help page uses the same image as the one supplied in this assignment and gives you an idea of what to expect upon bilateral filtering.

*Comparison:* The starter code additionally compares the output of your implementation of bilateral filtering with MATLAB’s built-in implementation imbilatfilt. Lines 43-44 of the starter code serve this purpose. Use this to verify the accuracy of your implementation.

**WARNING:** Remember to display (using imshow) results in RGB color space and not LAB color space.

**Deliverables & Questions**

1. Completed MATLAB code for HWK2\_ColorImageFiltering\_using\_BLFilter.m and bilateral\_filter. (10 points)
2. What happens to the texture of the wooden table upon bilateral filtering using the parameter values specified in the starter code? (2 points)
3. Do you notice any difference between your implementation and MATAB’s built-in implementation? If so where are the differences and speculate as to the reasons for the difference?
4. How does the appearance of the image change if you pick smaller values for and ? Try picking and for example. (5 points)
5. Screenshots of the original image, result of Bilateral filtering and Gaussian filtering. Label each screenshot clearly. Failure to do so will result in deduction of points. (3 points)

# Image enhancement by joint/cross bilateral filtering (25 points)

For this portion of the assignment you will be attempting to denoise the color image of a dimly lit scene using a second color image of the same scene acquired with the camera flash turned on. You will be implementing a simpler version of the approach described in Sections-4.1, 4.2 of the paper “[*Digital Photography with Flash and No-Flash Image Pairs*](http://hhoppe.com/flash.pdf)”. It is recommended that you read the relevant sections of the paper before you attempt this portion of the assignment. The complete for image enhancement is illustrated in Slide-55 of Lecture 6.

Your task is to complete the implementation of joint/cross Bilateral filter (in joint\_bilateral\_filter) and compare the results of image enhancement with those includes in Slide-55 of Lecture 6.

Ensure that your implementation of the joint/cross bilateral filter adheres to the following conventions

***Inputs:*** noisy image, guide image, ,

***Outputs:*** result of joint Bilateral filtering

Failure to adhere to these conventions will result in deduction of points.

Implementing the Joint/Cross Bilateral filter: To implement joint/cross bilateral filtering you need only modify your code for color bilateral filtering. The difference lies in the computation of the intensity kernel . Unlike standard bilateral filtering, depends on the color dissimilarity of the central pixel and neighboring pixels in the guide image. Please refer to Slide-50 of Lecture-6.

Identifying : The starter code supplied with the assignment identifies an image patch with largely constant color and computes the variance of the colors of the patch to identify :

patch = imcrop(imLAB\_noisy,[1,1,50,50]);

patchSq = patch.^2;

edist = sqrt(sum(patchSq,3));

patchStd = std2(edist);

Steps in Workflow

1. Joint/Cross Bilateral Filtering: Apply ***joint-bilateral filtering*** to the no-flash image imLAB\_noisy

* Choose a small value of (say 3) for this step. The starter code automatically chooses the value of .
* This should produce a denoised version of the no-flash image. The image while denoised is missing the texture (horizontal lines) evident in the flash image. The next steps in the workflow attempt to restore the texture to the denoised no-flash image.

1. Detail Identification: Apply ***bilateral filter*** to the flash image imLAB\_guide

* Choose a small value of (in excess of 11) for this step. The starter code automatically chooses the value of .
* Identify the texture/detail component of the flash image by dividing the flash image with the result of ***bilateral filtering*** from above.
* To avoid divide by zero errors it is recommended that you add a small value of to the numerator and the denominator terms as suggested in Eq.(6) of the paper [*Digital Photography with Flash and No-Flash Image Pairs*](http://hhoppe.com/flash.pdf).

1. Detail Transfer: Assemble the final enhanced image by multiplying the denoised no-flash image with the detail extracted from the flash-image. The process is analogous to Eq.(7) in the paper, but with . The resulting process mirrors the workflow described by the authors in Slide-55 of the Lecture-6.

**Deliverables & Questions**

1. Completed MATLAB code for HWK2\_ImageEnhancement\_using\_CrossBLFilter.m and joint\_bilateral\_filter. (15 points)
2. What value of did you choose for the **joint Bilateral filter** and the **standard Bilateral filter** and why? (5 points)
3. Screenshots of various steps in the workflow organized as follows: (5 points)







Please bear in mind that your results may differ slightly depending on your choice of .

Tips:

* The result of bilateral filtering of the flash image must be devoid of any texture or spatial detail. This requires a large value for to smooth over detail.
* The Detail component must largely be gray in appearance. It must contain texture missing in the denoised version of the no-flash image. It will be necessary to subtract 0.5 from the detail component before displaying the result using imshow. This is done to strictly match the appearance of the detail component with that included in the SIGGRAPH 2004 paper. Please remember that this step is strictly for display purposes and is not used in identifying the composite image. Please DO NOT email me or the grader wondering why your Detail component looks different from the one included above.