Assignment 2: CS 663, Fall 2021

Due: 28th August before 11:55 pm

Remember the honor code while submitting this (and every other) assignment. You may discuss broad ideas with other students or ask me for any difficulties, but the code you implement and the answers you write must be your own. We will adopt a zero-tolerance policy against any violation.

Submission instructions: Follow the instructions for the submission format and the naming convention of your files from the submission guidelines file in the homework folder. Please see assignment2.zip in the homework folder. For all the questions, write your answers and scan them, or type them out in word/Latex. In eithe case, create a separate PDF file. The last two questions will also have code in addition to the PDF file. Once you have finished the solutions to all questions, prepare a single zip file and upload the file on moodle before 11:55 pm on 28th August. **Only one student per group should submit the assignment.** We will not penalize submission of the files till 10 am on 29th August. **No assignments will be accepted after this time.** Please preserve a copy of all your work until the end of the semester. **Your zip file should have the following naming convention:** RollNumber1_RollNumber2_RollNumber3.zip for three-member groups, RollNumber1_RollNumber2.zip for two-member groups and RollNumber1.zip for single-member groups.

1. Consider a 1D convolution mask given as $(w_0, w_1, ..., w_6)$. Express the convolution of the mask with a 1D image f as the multiplication of a suitable matrix with the image vector f. What are the properties of this matrix? What could be a potential application of such a matrix-based construction? [10 points]
   Answer: We know that if g is the resultant image, then $g(x) = \sum_{k=-3}^{3} f(x-k)w(k)$. Let us assume that f is appropriately zero-padded and that it originally had n pixels. After zero-padding, it will have $n+6$ pixels. Then we can express this as a matrix equation in the following form: $\boldsymbol{g} = \boldsymbol{W}\boldsymbol{f}$ where $\boldsymbol{g}$ is a vector of $n+6$ values representing the resultant image, $\boldsymbol{f}$ is a vector of $n+6$ values representing the original image and $\boldsymbol{W}$ is a $(n+6) \times (n+6)$ matrix. In more detail, we have
   $\boldsymbol{f} \triangleq \begin{bmatrix} f(-3) & f(-2) & f(-1) & f(0) & ... & f(n) & f(n+1) & f(n+2) & f(n+3) \end{bmatrix}$,
   $\boldsymbol{g} \triangleq \begin{bmatrix} g(-3) & g(-2) & g(-1) & g(0) & ... & g(n) & g(n+1) & g(n+2) & g(n+3) \end{bmatrix}$.
   Also we have

$$\boldsymbol{W} \triangleq \begin{bmatrix} w_0 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & ... & 0 & 0 \\ 0 & w_0 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & ... & 0 \\ . & . & . & . & . & . & . & . & ... & . \\ 0 & .0 & ... & w_0 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 \end{bmatrix}. \tag{1}$$

   The matrix is square and every row of the matrix is a right shift of the previous row. The diagonal contains w_0 and each subdiagonal has constant values. This matrix may or may not be invertible. A potential application is that you can obtain f from g and w using this construction provided that the matrix $\boldsymbol{W}$ is invertible.
   Marking scheme: for the construction of $\boldsymbol{f}, \boldsymbol{g}$, we have 2 points. 4 points for the matrix, 2 points for any two of its properties and 2 points for the application.

2. In bicubic interpolation, the image intensity value is expressed in the form $v(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j$ where a_{ij} are the coefficients of interpolation and (x,y) are spatial coordinates. This uses sixteen nearest neighbors of a point (x,y). Given the intensity values of these 16 neighbors, explain with the help of matrix-based equations, how one can determine the coefficients a_{ij} that determine the function $v(x,y)$? Why do you require 16 neighbors for determining the coefficients? [10 points]

Answer: Consider a neighborhood consisting of 16 pixels with X coordinate ranging from c to $c+3$ and Y coordinate ranging from d to $d+3$, both in steps of 1, for any value of $x = c, y = d$. We are assuming the concerned coordinate (x, y) lies inside this neighborhood of 16 pixels and it will generally not be on the integer grid. We already have the formula for the bicubic interpolation. This can be represented in the form of a matrix equation $\boldsymbol{z} = \boldsymbol{X}\boldsymbol{w}$ where $\boldsymbol{z}$ is a 16-element vector containing the intensities of all 16 neighbors, $\boldsymbol{w}$ is the vector of the unknown 16 bicubic coefficients $\{\{a_{ij}\}_{i=0}^{3}\}_{j=0}^{3}$, and $\boldsymbol{X}$ is a 16×16 matrix as shown below.
We have $\boldsymbol{z} = \begin{bmatrix} v_{c,d} & v_{c+1,d} & v_{c+2,d} & v_{c+3,d} & ... & v_{c,d+3} & v_{c+1,d+3} & v_{c+2,d+3} & v_{c+3,d+3} \end{bmatrix}^t$,
$\boldsymbol{w} = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} & ... & a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix}^t$. The matrix $\boldsymbol{X}$ is given as follows:

$$\boldsymbol{X} \triangleq \begin{bmatrix} 1 & d & d^2 & d^3 & ... & c^3 & c^3 d & c^3 d^2 & c^3 d^3 \\ . & . & . & . & ... & . & . & . & . \\ 1 & d+3 & (d+3)^2 & (d+3)^3 & ... & (c+3)^3 & (c+3)^3(d+3) & (c+3)^3(d+3)^2 & (c+3)^3(d+3)^3 \end{bmatrix} \tag{2}$$

This matrix will be full rank as the coordinates are all unique, and we can obtain the coefficients as $\boldsymbol{z} = \boldsymbol{X}^{-1}\boldsymbol{w}$. We need 16 neighbors so that we have 16 known values, and can then uniquely determine the 16 unknown coefficients.
Marking scheme: 2 points for the correct construction of $\boldsymbol{z}$ and $\boldsymbol{w}$ each, 4 points for the correct construction of $\boldsymbol{X}$ and 2 points for the correct reason for needing 16 neighbors.

3. Consider a clean image $I(x, y)$ which gets corrupted by additive noise randomly and independently from a zero mean Gaussian distribution with standard deviation σ. Derive an expression for the PDF of the resulting noisy image. Assume continuous-valued intensities. [10 points]
Answer: Let J be the corresponding noisy image such that $J(x, y) = I(x, y) + N(x, y)$ where the values of N are drawn from $\mathcal{N}(0, \sigma^2)$ independently of each other as well as independently of I. We have $p_J(a) = \int_{-\infty}^{+\infty} p_{I,N}(b, a-b)db = \int_{-\infty}^{+\infty} p_I(b)p_N(a-b)db = \int_{-\infty}^{+\infty} p_I(b)\frac{e^{-(b-a)^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}}db$. This is nothing but an integral form of the convolution formula we have seen in class. The second equality above follows from the independence of I and N. The first equality follows because we are interested in those values of I, N which sum up to some a.
Marking Scheme: Deduct 2 points if the first equality is missing and if there is no mention of independence between N and I. Distribute points across the remaining steps evenly.

4. Prove or disprove: (a) The Laplacian mask with a -4 in the center (see class slides) is a separable filter. (b) The Laplacian mask with a -4 in the center (see class slides) can be implemented entirely using 1D convolutions. [5+5=10 points]
Answer: (a) The Laplacian mask is not a separable filter because it cannot be represented as the outer-product of two 3-element vectors $\boldsymbol{v}, \boldsymbol{w}$. Had it been an outperoduct, we could have expressed the elements of the mask as scalar multiples of one and the same vector $\boldsymbol{w}$. That is clearly not the case.
(b) However, the Laplacian can be represented entirely using 1D convolutions. We know that the Laplacian is given as $\nabla f = [f(x+1, y) - 2f(x, y) + f(x-1, y)] + [f(x, y+1) - 2f(x, y) + f(x, y-1)]$. Here the first term in the square bracket represents the double derivative w.r.t. x and the second term in the square bracket represents the double derivative w.r.t. y. Both these can be represented using 1D convolutions. Thus, the Laplacian can be implemented entirely using the sum of two 1D convolutions, even though it not a separable kernel.
Marking scheme: 5 points per part with proper reasoning. No credit for that part without reasonably correct reasoning.

5. Suppose I convolve an image f with a mean-filter of size $(2a+1) \times (2a+1)$ where $a > 0$ is an integer to produce a result f_1. Suppose I convolve the resultant image f_1 with the same mean filter once again to produce an image f_2, and so on until you get image f_K in the Kth iteration. Can you express f_K as a convolution of f with some kernel. If not, why not? If yes, with what kernel? Justify. [10 points]

Solution: Convolution is an associative operation, so $f_K(x, y) = (h * h * ... * h * f)(x, y) = ([h * h *$

$...*h]*f)(x,y) = (g*f)(x,y)$ where $g \triangleq h*h*...*h$ is the convolution of h with itself $K-1$ times. That is f_K is obtained by convolving f with the kernel g. (Additional information not expected for credit: It turns out that in the limit when K tends to infinity, the resultant kernel g tends to be a Gaussian. This is an indirect result of the Central Limit Theorem from statistics. Can you figure out how?)

6. Consider a 1D ramp image of the form $I(x) = cx + d$ where c, d are scalar coefficients. Derive an expression for the image J which results when I is filtered by a zero-mean Gaussian with standard deviation σ. Derive an expression for the image that results when I is treated with a bilateral filter of parameters σ_s, σ_r. (Hint: in both cases, you get back the same image.) Ignore any border issues, i.e. assume the image had infinite extent. [10 points]
Solution: Let h be the Gaussian kernel with mean 0 and std. dev. σ. We have $g(x) = (h*I)(x) = \sum_{k=-\infty}^{+\infty} I(x-k)h(k) = \sum_{k=-\infty}^{+\infty} I(x-k)h(k) = \sum_{k=-\infty}^{+\infty}(cx-ck+d)h(k) = (cx+d)\sum_{k=-\infty}^{+\infty} h(k) - c\sum_{k=-\infty}^{+\infty} kh(k)$. A Gaussian must sum to one as it is a PDF, so we have $\sum_{k=-\infty}^{+\infty} h(k) = 1$. Also, a Gaussian is symmetric about 0, so we have must have $\sum_{k=-\infty}^{+\infty} kh(k) = 0$. Thus we have $g(x) = cx + d = I(x)$. Ideally, this should be done using integrals as this is a continuous image, but since we hadn't seen a continuous convolution in class until a few days ago, it is fine to use summations as the basic argument still holds.
For the bilateral filter, define $L = \sum_{k=-\infty}^{+\infty} \frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(I(x-k)-I(x))^2/(2\sigma_r^2)}}{\sigma_r\sqrt{2\pi}}$. Then, we will have $g_2(x) = \sum_{k=-\infty}^{+\infty} I(x-k)\frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(I(x-k)-I(x))^2/(2\sigma_r^2)}}{L\sigma_r\sqrt{2\pi}L} = \sum_{k=-\infty}^{+\infty} I(x-k)\frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(I(x-k)-I(x))^2/(2\sigma_r^2)}}{L\sigma_r\sqrt{2\pi}} = \sum_{k=-\infty}^{+\infty}(cx+d-ck)\frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(ck)^2/(2\sigma_r^2)}}{L\sigma_r\sqrt{2\pi}}$. The last equality follows because $I(x-k)-I(x) = cx-ck+d-cx-d = -ck$.
Now simplifying further,
we have $g_2(x) = \sum_{k=-\infty}^{+\infty}(cx+d)\frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(ck)^2/(2\sigma_r^2)}}{L\sigma_r\sqrt{2\pi}} - \sum_{k=-\infty}^{+\infty}(ck)\frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(ck)^2/(2\sigma_r^2)}}{L\sigma_r\sqrt{2\pi}} = (cx+d) - 0$.
The first summation term evaluates to $cx+d$ because $\sum_{k=-\infty}^{+\infty} \frac{e^{-k^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \frac{e^{-(ck)^2/(2\sigma_r^2)}}{L\sigma_r\sqrt{2\pi}} = 1$ by the definition of L. The second term is 0 because both the Gaussian terms are symmetric about 0 whereas k is anti-symmetric.

7. Prove that the Laplacian operator is rotationally invariant. For this consider a rotation of the coordinate system from (x, y) to $u = x\cos\theta - y\sin\theta, v = x\sin\theta + y\cos\theta$, and show that $f_{xx} + f_{yy} = f_{uu} + f_{vv}$ for any image f. [10 points]
Answer: $f_y = f_u u_y + f_v v_y = f_u(-\sin\theta) + f_v\cos\theta$. Now we have $f_{yy} = \frac{\partial}{\partial y}(f_u(-\sin\theta) + f_v\cos\theta) = f_{uy}(-\sin\theta) + f_{vy}\cos\theta$.
Now $f_{uy} = \frac{\partial}{\partial u}(-f_u\sin\theta + f_v\cos\theta) = -f_{uu}\sin\theta + f_{uv}\cos\theta$.
Also $f_{vy} = \frac{\partial}{\partial v}[-\sin\theta f_u + f_v\cos\theta] = -\sin\theta f_{uv} + f_{vv}\cos\theta$.
Hence $f_{yy} = \sin^2\theta f_{uu} - 2\sin\theta\cos\theta f_{uv} + \cos^2\theta f_{vv}$.

Along similar lines, $f_x = f_u u_x + f_v v_x = f_u\cos\theta + f_v\sin\theta$ and $f_{xx} = \frac{\partial}{\partial x}[f_u\cos\theta + f_v\sin\theta] = f_{ux}\cos\theta + f_{vx}\sin\theta$.
We have $f_{ux} = \frac{\partial}{\partial u}f_x = \frac{\partial}{\partial u}[f_u\cos\theta + f_v\sin\theta] = f_{uu}\cos\theta + f_{uv}\sin\theta$.
Also $f_{vx} = \frac{\partial}{\partial v}[f_u\cos\theta + f_v\sin\theta] = f_{uv}\cos\theta + f_{vv}\sin\theta$.
Hence $f_{xx} = \cos^2\theta f_{uu} + 2f_{uv}\sin\theta\cos\theta + f_{vv}\sin^2\theta$.
This produces $f_{xx} + f_{yy} = f_{uu} + f_{vv}$.
Marking scheme: 10 points for the proof: 5 points for the expression for f_{xx} and f_{yy} each. At least one of these must have full detail in terms of expression for f_{uy}, f_{vy} or f_{ux}, f_{vx}.

8. Consider the two images in the homework folder 'barbara256.png' and 'kodak24.png'. Add zero-mean Gaussian noise with standard deviation $\sigma = 5$ to both of them. Implement a bilateral filter and show the outputs of the bilateral filter on both images for the following parameter configurations: $(\sigma_s = 2, \sigma_r = 2); (\sigma_s =$

$0.1, \sigma_r = 0.1); (\sigma_s = 3, \sigma_r = 15)$. Comment on your results in your report. Repeat when the image is corrupted with zero-mean Gaussian noise of $\sigma = 10$ (with the same bilaterial filter parameters). Comment on your results in your report. For the bilateral filter implementation, write a MATLAB function `mybilateralfilter.m` which takes as input an image and parameters σ_r, σ_s. Implement your filter using at the most two nested for-loops for traversing the image indices. For creating the filter, use functions like meshgrid and vectorization for more efficient implementation. Include all image ouputs as well as noisy images in the report. [15 points]

Answer: See code in homework folder. As σ_s, σ_r increases, there is more and more of a smoothing effect and subtle textures start getting erased. This is true for both smoothing on the original as well as noisy image.

Marking scheme: Implementation of bilateral filter carries 10 points. 3 points to be deducted if there are more than 2 nested for loops in the code. 2 points for comments on performance and 3 points for showing all image results in the report.

9. Implement local histogram equalization of sizes $7 \times 7, 31 \times 31, 51 \times 51, 71 \times 71$ on the images 'LC1.jpg' and 'LC2.jpg' from the homework folder. Comment on your results in your report and compare it to global histogram equalization, which you can use from the image processing toolbox of MATLAB. Point out regions where the local method produces better local contrast than the global histogram equalization. [15 points]

   Answer: The local histogram equalization method enhances contrast in small regions much better than the GHE (small tree near the bottom left), though the latter may yield an image that is less noisy. LHE implementation carries 10 points. For comments and comparison with GHE, we have 5 points. The student must show example regions where LHE is better, other 3 points are to be deducted.