Assignment 2: CS 663, Fall 2021

Due: 6th Sept. 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 6th September. **Only one student per group should submit the assignment.** We will not penalize submission of the files till 10 am on 7th September. **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 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.

2. 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

$$\boldsymbol{W} \triangleq \begin{bmatrix} \tilde{w}_0 & \tilde{w}_1 & \tilde{w}_2 & \tilde{w}_3 & \tilde{w}_4 & \tilde{w}_5 & \tilde{w}_6 & ... & 0 & 0 \\ 0 & \tilde{w}_0 & \tilde{w}_1 & \tilde{w}_2 & \tilde{w}_3 & \tilde{w}_4 & \tilde{w}_5 & \tilde{w}_6 & ... & 0 \\ . & . & . & . & . & . & . & . & ... & . \\ 0 & .0 & ... & \tilde{w}_0 & \tilde{w}_1 & \tilde{w}_2 & \tilde{w}_3 & \tilde{w}_4 & \tilde{w}_5 & \tilde{w}_6 \end{bmatrix}, \quad (1)$$

where the values $\{\tilde{w}_i\}_{i=0}^{6}$ represent the values of the mask $\{w_i\}_{i=0}^{6}$ after it has been flipped as is required for convolution. The matrix is square and every row of the matrix is a right shift of the previous row. The diagonal contains $\tilde{w}_0$ 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.

3. Prove or disprove: (a) The Laplacian mask with a -8 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 as the kernel matrix has rank 2.
(b) The Laplacian with a -4 in the center 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 with a -4 in the center can be implemented entirely using the sum of two 1D convolutions, even though it not a separable kernel. However, the Laplacian with a -8 will require additional operations of the form $f(x-1,y-1) - 2f(x,y) + f(x+1,y+1)$ and $f(x+1,y-1) - 2f(x,y) + f(x-1,y+1)$ which can implemented with 1D convolutions only after applying ± 45 degree rotations to the image. The students may write an answer that this mask cannot be implemented with 1d convolutions, or they may state that it is possible with this rotation. In either case, proper justification is required.
Marking scheme: 5 points per part with proper reasoning. No credit for that part without reasonably correct reasoning.

4. Consider an image $I(x,y)$. Argue using a 1D example that adding $\alpha\nabla^2 I(x,y)$ to $I(x,y)$ where $\alpha > 0, \alpha \approx 0$ is a small constant, causes blurring of the edges. What would happen if the operation $I(x,y) \leftarrow I(x,y) + \alpha\nabla^2 I(x,y)$ were run for a very large number of iterations? Why? On the other hand, what would happen if the operation $I(x,y) \leftarrow I(x,y) - \alpha\nabla^2 I(x,y)$ were run for a very large number of iterations? Why? [7+4+4=15 points]
Answer: You can take the same example as in slide 6 of the slides on EdgeDetection. Here, we consider a scanline $6,6,6,6,5,4,3,2,1,1,1,1,1,6,6,6,6$.

The second derivatives will be $0,0,-1,0,0,0,0,1,0,0,0,0,5,-5,0,0,0$. When you add α times this derivative, one can see that the ramp edge gets blurred. This is because the intensity at the beginning of the edge is $6-\alpha$ and the intensity at the end of the edge is $1+\alpha$. Hence the intensity difference is $5-2\alpha$ as opposed to the earlier $6-1=5$. As $\alpha > 0$, this has reduced the contrast across the edge. In somewhat more analytical terms, this operation of adding second derivatives causes blurring because the second derivatives are negative at the beginning of a downward ramp and positive at the end of it. Adding a small multiple of the second derivative causes the intensity to reduce at the beginning of the downward ramp and to increase at the end of the ramp, thus reducing contrast. This has been argued for the 1D case, but the same argument extends to 2D as the Laplacian can be implemented using consecutive 1D convolutions.
If the operation $I(x,y) \leftarrow I(x,y) + \alpha\nabla^2 I(x,y)$ were run for a very large number of iterations, the image gets blurred increasingly and moves towards becoming a constant intensity image due to successive reduction in edge contrast.
If the operation $I(x,y) \leftarrow I(x,y) - \alpha\nabla^2 I(x,y)$ were run for a very large number of iterations, then it causes successive edge sharpening. Unfortunately, the edge contrast can increase to infinity across the iterations leading to an image with very large intensity values moving towards $+\infty$ and $-\infty$. In practical implementations, this causes wraparound errors.

5. 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.

Marking scheme: The marking scheme is mentioned in the question itself.

6. 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 $I_{xx} + I_{yy} = I_{uu} + I_{vv}$ for any image I. Prove that the second directional derivative of an image $I(x, y)$ in the direction of its gradient vector (*i.e.* in the direction $(\frac{I_x}{\sqrt{I_x^2 + I_y^2}}, \frac{I_y}{\sqrt{I_x^2 + I_y^2}})$) is given by $\frac{I_x^2 I_{xx} + 2I_x I_y I_{xy} + I_y^2 I_{yy}}{I_x^2 + I_y^2}$. Note that $I_x = \frac{\partial I}{\partial x}, I_{xx} = \frac{\partial^2 I}{\partial x^2}$. Using this information, write down the expression for the second directional derivative of $I(x, y)$ in the direction **perpendicular** to its gradient vector and justify your answer. Note that the first directional derivative of $I(x, y)$ in a direction v is given by $\nabla I(x, y) \cdot v$. [6+6+3 = 15 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 $f_{xx} + f_{yy} = f_{uu} + f_{vv}$.

Let v be the gradient direction, i.e. $v = (\frac{I_x}{\sqrt{I_x^2 + I_y^2}}, \frac{I_y}{\sqrt{I_x^2 + I_y^2}})$. Let u be the direction perpendicular to

it, i.e. $u = (\frac{-I_y}{\sqrt{I_x^2+I_y^2}}, \frac{I_x}{\sqrt{I_x^2+I_y^2}})$. We are given I_{vv}, and we want to find the expression for I_{uu}. Recall that the Laplacian is a rotationally symmetric operator, and hence $I_{xx} + I_{yy} = I_{uu} + I_{vv}$ since u and v are perpendicular directions. From this, we can show that $I_{uu} = \frac{I_y^2 I_{xx} - 2I_x I_y I_{xy} + I_x^2 I_{yy}}{I_x^2 + I_y^2}$.

To prove the expression for I_{vv}, recall from the class notes that $I_{vv} = v^T H v$ where H is the Hessian matrix given as $H = \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix}$. Plugging in $v = (\frac{I_x}{\sqrt{I_x^2+I_y^2}}, \frac{I_y}{\sqrt{I_x^2+I_y^2}})$ into this formula gives you the required expression.

Another way to do is to first compute $I_v = \nabla I \cdot v$, and then compute $I_{vv} = \nabla I_v \cdot v = \frac{\partial}{\partial x}(\nabla I \cdot v)v_1 + \frac{\partial}{\partial y}(\nabla I \cdot v)v_2 = \frac{\partial}{\partial x}(I_x v_1 + I_y v_2)v_1 + \frac{\partial}{\partial y}(I_x v_1 + I_y v_2)v_2$. But note that in these partial derivatives here, $v = (v_1, v_2)$ is treated as a constant.

While deriving I_{uu}, some students showed that $I_u = \nabla I \cdot u = 0$ (after all u and v are perpendicular, and v is in the same direction as ∇I), and hence concluded that I_{uu} must be zero. This is incorrect, because I_{uu} is **defined** to be $I_{uu} = u^T H u$ and hence it is not zero **even though** I_u is zero. In other words $I_{xx}u_1 + I_x u_{1x} + I_{xy}u_2 + I_y u_{2x}$ will be zero, but $I_{xx}u_1 + I_{xy}u_2$ isn't.

Marking scheme: 6 points for the proof of rotational invariance: 3 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}.

7. 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.

8. 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.