Final exam: CS 663, Digital Image Processing, 15th November

Instructions: There are 180 minutes for this exam (9:30 am to 12:30 pm). Answer all 10 questions. Each question carries 5 points. This exam is worth 20% of the final grade. Some formulae are listed at the end of the paper.

✓ 1. What is meant by the inverse filter in deblurring? What are its disadvantages?

   Soln: Let a blurred image be given as $g(x,y) = f(x,y) * h(x,y)$ for blur kernel h and image f. The inverse filter estimates f given g and h as follows: $f = \mathcal{F}^{-1}(G(u,v)/H(u,v))$ where $G(u,v)$ and $H(u,v)$ are the Fourier transforms of g and h respectively. The inverse filter is accurate but it fails under noise as there is an error term given by $\mathcal{F}^{-1}(N(u,v)/H(u,v))$. Since h is a low pass filter, $H(u,v)$ will very low at high frequencies, whereas $N(u,v)$ will have value comparable to the noise variance. This produces a large error. The relative error at high frequencies is also large because $G(u,v)$ may even be smaller than $N(u,v)$ as most natural images are weak at higher frequencies.
   Marking scheme: 2 points for explaining what the inverse filter is. 2 points for stating the problem that $N(u,v)/H(u,v)$ is high. 1 point for stating that $G(u,v)$ is low and hence the relative error at this frequency is high.

✓ 2. Consider the equation $\mathbf{y} = \mathbf{C}\mathbf{x}$ where $\mathbf{y}$ and $\mathbf{x}$ are vectors with d elements each, and $\mathbf{C}$ is a circulant matrix. Describe an efficient procedure to estimate $\mathbf{x}$ given $\mathbf{y}$ and $\mathbf{C}$, assuming d is very large.

   Soln: The solution $\mathbf{x} = \mathbf{C}^{-1}\mathbf{y}$ is computationally expensive. However, we can write $\mathbf{C} = \mathbf{V\Gamma V}^*$ where $\mathbf{V}$ is the eigenvector matrix of $\mathbf{C}$ which turns out to be the discrete Fourier matrix. $\mathbf{V}$ is orthonormal. Γ stands for diagonal matrix of eigenvalues of $\mathbf{C}$. Now, we have $\mathbf{y} = \mathbf{V\Gamma V}^*\mathbf{x}$ which yields $\mathbf{V}^*\mathbf{y} = \mathbf{\Gamma V}^*\mathbf{x}$, which is equivalent to $Y(u) = \mathbf{\Gamma} X(u)$. Here $Y(u)$ and $X(u)$ stand for the Fourier transform coefficients (at frequency u) of $\mathbf{y}$ and $\mathbf{x}$ respectively. This gives $X(u) = Y(u)/\Gamma(u)$ where $\Gamma(u)$ stands for the eigenvalue corresponding to frequency (u). This is an efficient computation. Note that $\Gamma(u)$ is the Fourier coefficient of the first column of $\mathbf{C}$ (the blur kernel underlying $\mathbf{C}$).
   Marking scheme: 3 points for stating that the Fourier matrix is the eigenvector

matrix of $\mathbf{C}$, 2 points for the equation $\mathbf{y} = \mathbf{V}\boldsymbol{\Gamma}\mathbf{V}^*\mathbf{x}$ or $\mathbf{C} = \mathbf{V}\boldsymbol{\Gamma}\mathbf{V}^*$ with the meaning of Γ explained.

3. Write down an expression for the Gaussian kernel density estimate for a random variable x, given four samples x_1, x_2, x_3, x_4. Assume that the standard deviation σ of the kernel is known to you. Given such an estimate, describe the procedure to draw a single sample from it. Assume you have access to (1) a software routine that to draw a sample from a Gaussian distribution of mean 0 and standard deviation 1, and (2) a software routine to draw a number between 0 and 1 uniformly at random.

Soln: The KDE is $p(x) = \frac{1}{4}(G(x; x_1, \sigma) + G(x; x_3, \sigma) + G(x; x_3, \sigma) + G(x; x_4, \sigma))$ where $G(x; x_i, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(x - x_i)^2}{2\sigma^2})$.
Let us call the procedure to draw a sample from a Gaussian distribution of mean 0 and standard deviation σ as 'randn', and the procedure to draw a number between 0 and 1 uniformly at random as 'rand'. To draw a sample from $p(x)$, first generate $x =$rand(1). If $x < 0.25$, the final sample is given as $s = x_1 + \sigma$randn(1). If $0.25 \leq x < 0.5$, then the final sample is given as $s = x_2 + \sigma$randn(1). If $0.5 \leq x < 0.75$, then the final sample is given as $s = x_3 + \sigma$randn(1). Otherwise, the final sample is given as $s = x_4 + \sigma$randn(1).
Making scheme: 3 points for correct expression for $p(x)$ (deduct 1 point if constant factors are missing). 2 points for explaining the correct procedure for drawing a sample. If you wrote the answer as s being the average of samples drawn from the individual four Gaussians, then you get 0 out of 2.

4. In the mean shift procedure for segmentation of a gray-scale image, starting from each sample we perform a gradient ascent on a probability density. The probability density is built on what quantity? What is the step size of the gradient ascent?

Soln: The pdf is built on $(x, y, I(x, y))$. The step size of the gradient ascent is proportional to $\frac{1}{p(x, y, I(x, y))}$.
Marking scheme: 2 points for stating what the pdf is built on. No credit if you missed any of the terms. 3 points for second part.

✓ 5. Consider a matrix $\mathbf{A}$ of size $m \times n$ where m is several times larger than n. Explain how will you compute the SVD of $\mathbf{A}$ efficiently, assuming you had no access to any software function that computed the SVD, but you had access to a software routine that computed the eigenvectors and eigenvalues of a square matrix.

Soln: $\mathbf{V}$ can be computed as the eigenvectors of $\mathbf{A}^T\mathbf{A}$ (a $n \times n$ matrix). As $n \ll m$, the matrix $\mathbf{A}$ will have at most n non-zero singular values. The singular values of $\mathbf{A}$ are the positive square-roots of the eigenvalues of $\mathbf{A}^T\mathbf{A}$. The time-complexity of computing $\mathbf{A}^T\mathbf{A}$ is $O(mn^2)$ and that of computing its eigenvalues and eigenvectors is $O(n^3)$. We need to compute $\mathbf{U}$ which contains the eigenvectors of $\mathbf{A}\mathbf{A}^T$ which is too large a matrix (size $m \times m$ and its eigenvector/eigenvalue computation will be $O(m^3)$). Hence, we use the following relation:

$$\mathbf{A}^T\mathbf{A}\mathbf{v} = \lambda\mathbf{v} \tag{1}$$

$$\mathbf{A}\mathbf{A}^T(\mathbf{A}\mathbf{v}) = \lambda(\mathbf{A}\mathbf{v}) \tag{2}$$

$$\mathbf{A}\mathbf{A}^T(\mathbf{A}\mathbf{v})/(\|\mathbf{A}\mathbf{v}\|_2) = \lambda(\mathbf{A}\mathbf{v})/(\|\mathbf{A}\mathbf{v}\|_2) \tag{3}$$

Hence $\mathbf{u} = (\mathbf{A}\mathbf{v})/(\|\mathbf{A}\mathbf{v}\|_2)$ is an eigenvector of $\mathbf{A}\mathbf{A}^T$ corresponding to eigenvalue λ. We already know $\mathbf{v}$ and computing $\mathbf{u}$ from $\mathbf{v}$ is an $O(mn)$ operation. This has to be done for the n eigenvectors of $\mathbf{A}^T\mathbf{A}$ and hence the total operation is $O(mn^2)$. The net time complexity is $O(mn^2 + n^3)$.
Marking scheme: 1 point for correct method for $\mathbf{V}$. 1 point for singular values. 2 points for correct method for $\mathbf{U}$ (derivation is not needed but formula should be correct). 1 point for correct time complexity. If you said that you compute $\mathbf{U}$ by directly computing eigenvectors of $\mathbf{A}\mathbf{A}^T$, you directly lose 3 points (for the time complexity as well).

✓ 6. Consider N vectors $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$ each having d elements. I want to find a unit-vector $\mathbf{e}$ which minimizes $Q(\mathbf{e})$ where $Q(\mathbf{e}) = \sum_{i=1}^N \|\mathbf{x}_i - (\mathbf{e}^t\mathbf{x}_i)\mathbf{e}\|^2$. How will you computationally determine $\mathbf{e}$ given the N vectors (proof not required). Is such a vector $\mathbf{e}$ always guaranteed to be unique? If yes, why? If no, give a counter-example.

Soln: The answer is that $\mathbf{e}$ is the eigenvector of $d \times d$ matrix $\mathbf{C} = \sum_{i=1}^N \mathbf{x}_i\mathbf{x}_i^t$. Note that we are not deducting the mean-vector from the vectors unlike classical PCA because of the specific manner in which the criterion has been specified. The mean vector of all vectors is a reasonable but inaccurate answer and gets much less credit. Such a vector $\mathbf{e}$ is not guaranteed to be unique if $\mathbf{C}$ has multiple maximum eigenvalues. This can happen, for example, if the data vectors are samples from a zero-mean Gaussian distribution with isotropic covariance matrix (i.e. diagonal covariance matrix with all diagonal elements equal to one another).
Marking scheme: 3 points for stating that the answer is the eigenvector of the

matrix $\mathbf{C}$ as defined above corresponding to largest eigenvalue. Deduct 1 point if the students wrote the covariance matrix, *i.e.* they deducted the mean vector from all the points first. 2 points for the counterexample. No credit for this part for mere guess-work.

✓ 7. Consider an image whose intensity values are integers in the range from 0 to 7, occurring with frequencies $0.1, 0.1, 0.2, 0.2, 0.2, 0.15, 0.025, 0.025$ respectively. Construct a Huffman tree for encoding these intensity values and find the corresponding average bit length (exact numerical values are not important, but show your steps clearly).

Soln: In this case, the average bit length $= \sum_{i=0}^{7} p_i l_i$ turns out to be 2.8, where p_i is the frequency and l_i is the bit length of alphabet i. The bit length is given by the length of the path from the root node to the leaf node. In Huffman encoding, you greedily combine the two symbols with the lowest frequencies into a single symbol whose frequency is the sum of the two frequencies. This process continues till one only symbol (the root node) remains.
Marking scheme: 4 points for correct Huffman tree (roughly 1 point for each step).1 point for correct formula of average bit length and correct substitution of the values in the formula.

8. What is the motivation for using the discrete cosine transform (DCT) instead of the discrete Fourier transform (DFT) in image compression standards such as JPEG? State any two reasons.

Soln: There are three reasons. First, DCT coefficients are real valued whereas DFT coefficients are complex-valued. Hence DCT coefficients are cheaper to store. Second, DCT has better energy compaction properties than DFT for natural image patches. This is because the DCT is the DFT of a double length signal obtained by concatenating the original signal by its reflected version. This ensures that there are no discontinuities. On the other hand, the DFT is the Discrete Fourier series of multiple periodic copies of the signal. The discontinuities between adjacent periods require several series coefficient for effective reconstruction. Third, DCT is closely related to PCA of an ensemble of small-sized image scanlines under the assumption that correlation between pixel values is of the form ρ^k where $\rho < 1$ is the correlation between a pixel and any one its immediate neighbors, and k is the distance between two pixels in a scanline. PCA is the best orthonormal matrix for compact reconstruction of a group of signals (by definition). Therefore, DCT is closely related to PCA.
Marking scheme: 2.5 points for each correct reason (any two required). Detailed explanation for the reason is not expected.

9. In class we have studied the bilateral filter, which replaces the intensity $I(x, y)$ at pixel (x, y) as follows: $I^{filtered}(x, y) = \dfrac{\sum_{(x',y') \in \mathcal{N}(x,y)} I(x', y') e^{-\beta((x-x')^2+(y-y')^2)-\gamma(I(x,y)-I(x',y'))^2}}{\sum_{(x',y') \in \mathcal{N}(x,y)} e^{-\beta((x-x')^2+(y-y')^2)-\gamma(I(x,y)-I(x',y'))^2}}$ where $\mathcal{N}(x, y)$ is a small neighborhood centered at (x, y). An inquisitive student tries the following modification to the bilateral filter. He/she decides to first filter the gradient values $I_x(x, y)$ and $I_y(x, y)$ instead of the intensity values using two different bilateral filters, and then perform an 'integration' operation (the opposite of differentiation) to obtain the intensity values from the filtered gradient values. The bilateral filters equations for filtering the gradient values look as follows:

$$I_x^{filtered}(x, y) = \frac{\sum_{(x',y') \in \mathcal{N}(x,y)} I_x(x', y') e^{-\beta((x-x')^2+(y-y')^2)-\gamma(I_x(x,y)-I_x(x',y'))^2}}{\sum_{(x',y') \in \mathcal{N}(x,y)} e^{-\beta((x-x')^2+(y-y')^2)-\gamma(I_x(x,y)-I_x(x',y'))^2}}$$

$$I_y^{filtered}(x, y) = \frac{\sum_{(x',y') \in \mathcal{N}(x,y)} I_y(x', y') e^{-\beta((x-x')^2+(y-y')^2)-\gamma(I_y(x,y)-I_y(x',y'))^2}}{\sum_{(x',y') \in \mathcal{N}(x,y)} e^{-\beta((x-x')^2+(y-y')^2)-\gamma(I_y(x,y)-I_y(x',y'))^2}}.$$

What problems if any will the student face in executing this procedure? If executed perfectly, does the student's method have any advantages over conventional bilateral filtering? If so what? You may make suitable assumptions.

Soln: The bilateral filter basically performs weighted average of pixel intensity values. It will perform very well in piecewise flat regions of the image. However, it will not be able to preserve fine shading patterns, and tends to erase them, leading to a piecewise constant image without the strong edges blurred out. This is especially true for larger filter parameters or larger number of iterations. See for instance figure 5 of `http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MANDUCHI1/Bilateral_Filtering.html`. If the student is able to execute his/her idea well, then (s)he will design a filter that will preserve shading better - especially shading in regions with constant intensity gradient. This is because the intensity gradients are being used as weights here, instead of intensity values.

The problem (s)he will face is in the integration step. He/She has filtered gradient values denoted as $I_x^{filtered}$ and $I_y^{filtered}$. To obtain I from $I_x^{filtered}$, (s)he will need to make some additional assumptions - either (s)he needs to guess the average value of the image, or else needs to assume that the gradient values on the right-hand border of the image are zero (or some other fixed value(s)). Then one can integrate across each row to obtain I from I_x. Now, let us say (s)he repeats this process using $I_y^{filtered}$. One again needs to guess the average value of the image, or else needs to assume that the gradient values on the bottom border of the image are zero (or some other fixed values(s)). Then one can integrate across each column to obtain I. The problem is that the two estimates of I are not guaranteed to be consistent. And the image can be estimated only upto an unknown additive offset. In practice, one can make an educated guess that the input image and fitlered image will have similar average intensity values and get around the latter problem. To get around the former problem, one can simply take an average of the two estimates. Or else, one can seek to minimize the following quantity w.r.t. J: $E(J) = \sum_{x,y}(J_x(x,y) - I_x^{filtered}(x,y))^2 + (J_y(x,y) - I_y^{filtered}(x,y))^2$, i.e. find an image J whose gradients are as close as possible to the provided gradients $I_x^{filtered}$ and $I_y^{filtered}$.

Another method: Use the fact that $I_x^{filtered}(x,y) = I^{filtered}(x+1,y) - I^{filtered}(x,y)$ and obtain $I^{filtered}(x,y)$ by a Fourier-based method, since $\mathcal{F}(I_x^{filtered})(u,v) = (e^{j2\pi u} - 1)\mathcal{F}(I^{filtered})(u,v)$. The estimate of the image is ill-posed at $u = 0$. Likewise, $I_y^{filtered}(x,y) = I^{filtered}(x,y+1) - I^{filtered}(x,y)$, and hence $\mathcal{F}(I_y^{filtered})(u,v) = (e^{j2\pi v} - 1)\mathcal{F}(I^{filtered})(u,v)$. This estimate is ill-posed at $v = 0$. One can use values from the latter estimate to handle frequencies of the form $(u = 0, v \neq 0)$, and values from the first estimate to handle frequencies of the form $(u \neq 0, v = 0)$. However, $(u = 0, v = 0)$ is still a problem. The only way to get around it is to guess that the input image and fitlered image will have similar average intensity values.

Marking scheme: 3 points for the problems faced by the student - in particular that the DC component of the image is unknown, and the two integrations are inconsistent. Some students mentioned that if the image is noisy, the gradient computations amplify the noise. This is also a valid drawback of the proposed filter. Note that I did <u>not</u> expect you to provide any potential solutions of these problems. 2 points for the advantage of this proposed method over simple bilateral filter (i.e. that is preserves regions with constant gradients or fine shading better).

✓ 10. Consider a function $f(x,y)$ defined over a bounded rectangular domain. Consider the quantity $g(\rho,\theta) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)dxdy$ where $x\cos\theta + y\sin\theta = \rho$ is the equation of a line in normal form and $\delta(z)$ is the Dirac delta function (i.e. $\delta(z) = \infty$ if $z = 0$, otherwise $\delta(z) = 0$). Consider the quantity $G(\omega,\theta)$, defined as the 1D Fourier transform of $g(\rho,\theta)$ w.r.t. ρ where ω is a frequency variable. We have $G(\omega,\theta) = \int_{-\infty}^{+\infty} g(\rho,\theta)e^{-j2\pi\omega\rho}d\rho$. Starting from this, derive an expression for $G(\omega,\theta)$ in terms of $F(u,v)$, the Fourier transform of $f(x,y)$ where (u,v) stands for the frequency variables. What is the computational advantage of such a relationship? (Hint: What does $g(\rho,\theta)$ mean geometrically?)

Soln: This question looks a lot more complicated than it really is! We have:

$$G(\omega,\theta) = \int_{-\infty}^{+\infty} g(\rho,\theta)e^{-j2\pi\omega\rho}d\rho \tag{4}$$

$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)e^{-j2\pi\omega\rho}dxdyd\rho \tag{5}$$

$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)(\int_{-\infty}^{+\infty} \delta(x\cos\theta + y\sin\theta - \rho)e^{-j2\pi\omega\rho}d\rho)dxdy \tag{6}$$

$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)e^{-j2\pi\omega(x\cos\theta + y\sin\theta)}d\rho)dxdy \text{ (by sifting property)} \tag{7}$$

Now, the 2D Fourier transform of $f(x,y)$ is given as $F(u,v) == \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)e^{-j2\pi(ux+vy)}dxdy$. Comparing this with the earlier equations, we see that $G(\omega,\theta) = [F(u,v)]_{u=\omega\cos\theta, v=\omega\sin\theta} = F(\omega\cos\theta, \omega\sin\theta)$.

Note that $x\cos\theta + y\sin\theta = \rho$ is the equation of a line (say L) in normal form. The angle θ is the angle between the X axis and the normal to L, and ρ is the perpendicular distance from the origin onto L. The quantity $g(\rho,\theta)$ means the projection of the signal $f(x,y)$ in the direction perpendicular to L. What this means geometrically is that you are sticking a needle into $f(x,y)$ in a direction perpendicular to L at a distance ρ from the origin, and 'adding' up all values through which the needle passed. By keeping θ constant and changing ρ, we get several values of $g(\rho,\theta)$. $G(\omega,\theta)$ is the Fourier transform of θ at frequency ω, and we have proved that it is equal to the Fourier transform of $f(x,y)$ at frequency $(\omega\cos\theta, \omega\sin\theta)$, i.e. a slice through the Fourier transform along the direction $(\cos\theta, \sin\theta)$ in the frequency plane. This result is called the **Fourier slice theorem** and it is a major result in the field of computed tomography, the process of determining the structure of 2D objects from a group of their 1D projections (or the structure of 3D objects from a group of their 2D projections).

The computational benefit of this is as follows: Let us assume a discrete image of size $N \times N$. Suppose we want to find the Fourier transform of K diferent projections of $f(x,y)$. Computing each projection is $O(N^2)$. Each projection will have N elements and computing its Fourier transform is $O(N\log N)$. For K projections, the total cost is $O(K(N^2 + N\log N))$. On the other hand, we can compute the Fourier transform of $f(x,y)$ in $O(N^2\log N^2) = O(N^2\log N)$ time, and then extract K slices in $O(KN)$ time. The total cost is $O(KN + N^2\log N)$. If K is larger than $O(\log N)$, then the latter method is more efficient.

There is a second major benefit of the Fourier slice theorem, which I did $\underline{\text{not}}$ expect you to write in a 3 hour exam. But I am putting it down here for those who are interested. We know that

$$f(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u,v)e^{j2\pi(ux+vy)}dudv \tag{8}$$

$$= \int_{0}^{2\pi}\int_{0}^{+\infty} F(\omega\cos\theta,\omega\sin\theta)e^{j2\pi\omega(x\cos\theta+y\sin\theta)}\omega d\omega d\theta. \tag{9}$$

The second step follows by converting frequency (u,v) to a polar representation with ω and θ. But we know that $F(\omega\cos\theta,\omega\sin\theta) = G(\omega,\theta)$. Hence we have $(f,xy) = \int_0^{2\pi}\int_0^{+\infty} G(\omega,\theta)e^{j2\pi\omega(x\cos\theta+y\sin\theta)}\omega d\omega d\theta$. Now $G(\omega,\theta)$ is the Fourier transform of the projections $g(\rho,\theta)$. Hence, this formula suggests to us a method for estimating $f(x,y)$ from its projections $g(\rho,\theta)$. Of course, there are problems in implementing the formula in practice, as this integral is not defined, and one has to limit the extent of ω using a low-pass filter (this is called as filtered back-projection). However, the formula lays the basic framework for tomography. You can refer to section 5.11.5 of the textbook for more details.

Marking scheme: 3 points for proving that $G(\omega,\theta) = [F(u,v)]_{u=\omega\cos\theta,v=\omega\sin\theta} = F(\omega\cos\theta,\omega\sin\theta)$. 2 points for a reasonable and sensible attempt at the computational advantage.

LIST OF FORMULAE:

1. Gaussian pdf in 1D centered at μ and having standard deviation σ: $p(x) = \frac{1}{\sqrt{2\pi}\sigma}e^{-(x-\mu)^2/(2\sigma^2)}$.
2. 1D Fourier transform and inverse Fourier transform:
   $F(u) = \int_{-\infty}^{+\infty} f(x)e^{-j2\pi ux}dx, f(x) = \int_{-\infty}^{+\infty} F(u)e^{j2\pi ux}du$
3. 2D Fourier transform and inverse Fourier transform:
   $F(u,v) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)e^{-j2\pi(ux+vy)}dxdy, f(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u,v)e^{j2\pi(ux+vy)}dudv$
4. Convolution theorem: $\mathcal{F}(f(x)*g(x))(u) = F(u)G(u); \mathcal{F}(f(x)g(x))(u) = F(u)*G(u)$
5. Fourier transform of $g(x-a)$ is $e^{-j2\pi ua}G(u)$. Fourier transform of $\frac{df^n(x)}{dx^n} = (j2\pi u)^n F(u)$ ($n > 0$ is an integer).