Mid-sem: CS 663, Digital Image Processing, 13th September

Instructions: There are 120 minutes for this exam. This exam is worth 20% of your final grade. Answer all 12 questions. Each question carries 5 points. Your total out of 60 will be treated as if it were out of 50 (*eg* if you got 54/60, it's treated as 54/50) and then divided by 2.5 while computing your final grade at the end of the semester.

1. While rotating an image in-plane about the origin through an angle θ, the new coordinates (u, v) of a point (x, y) are given by $u = x\cos\theta - y\sin\theta, v = x\sin\theta + y\cos\theta$. How will you modify this equation if you had to rotate the image about an arbitrary point (a, b)? Express the relationship between (u, v) and (x, y) in this case using a matrix.

   ANSWER:

   $u = (x-a)\cos\theta + (y-b)\sin\theta + a, v = (x-a)\sin\theta + (y-b)\cos\theta + b$, which can be represented as follows using matrix notation:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (1)$$

2. Suppose you scale an image $f(x, y)$ about the origin by a factor of 2 along the X axis and a factor of 3 along the Y axis, giving a scaled image $g(x, y)$. Consider a pixel location (x_g, y_g) in the image $g(x, y)$. To which location (x_f, y_f) in $f(x, y)$ does (x_g, y_g) correspond? In practice, we would want to copy the intensity value $f(x_f, y_f)$ to $g(x_g, y_g)$. What will you do if (x_f, y_f) are not integers?

   ANSWER:

   We have $x_g = 2x_f, y_g = 3y_f$. Here, we are doing reverse warping, so given a fixed value point (x_g, y_g) in the destination image, the corresponding pixel

coordinates from the source image are $(x_g/2, y_g/3)$. If these are not integers (the most likely scenario), you need to use one of the interpolation methods described in class - nearest neighbor or bilinear interpolation.

3. Suppose you randomly scramble (permute) the pixel intensities of a face image. What effect will this operation have on the intensity histogram of the image? What effect will it have on the histogram of the gradient magnitudes of the image? What effect will it have on the histogram of the gradient orientations $(\theta(x,y) = \tan^{-1} \dfrac{\partial f(x,y)/\partial y}{\partial f(x,y)/\partial x})$ of the image?

   ANSWER:

   There were two interpretation of this question, and I gave full credit for both. The first (more common) interpretation is that you merely permuted the image (so a pattern like [12234445] could become say [214235445]). In this case, the intensity histogram will not change, but the gradient magnitude histogram will spread out toward higher values (generally gradient magnitudes will tend to increase). The gradient orientation histogram will also spread out.

   The second interpreation was that only the intensity values are swapped. So for instance, the pattern like [12234445] could become say [53342221]). In this case, the intensity histogram will get shuffled depending upon which intensity values were swapped. The gradient magnitude and orientation histograms will still tend to spread out, except in some truly exceptional cases.

4. Consider the operation $g(x,y) = f(x+1,y) + f(x-1,y) - 2f(x,y)$. With what function should you convolve $f(x,y)$ to get $g(x,y)$? What is the Fourier transform of $g(x,y)$ in terms of $F(u,v)$, the Fourier transform of $f(x,y)$?

   ANSWER:

   The convolution kernel will be $[1,-2,1]$, i.e. $f(x,y) * [1,-2,1] = g(x,y)$. An alternative way of writing this kernel is $\delta(x+1,y) + \delta(x-1,y) - 2\delta(x,y)$. For the second part, $G(u,v) = F(u,v)(e^{j2\pi ux/M} + e^{-j2\pi ux/M} - 2)$.

5. State any two uses of the image Laplacian in image processing. The Laplacian is given by $\triangle f(x,y) = \dfrac{\partial^2 f(x,y)}{\partial x^2} + \dfrac{\partial^2 f(x,y)}{\partial y^2}$.

   ANSWER:

   The Laplacian is the simplest rotationally invariant double derivative operator for 2D (or higher dimensions). The zero crossings of the Laplacian occur at

image edges, so it is used for edge detection. It can be used for image sharpening, for which you need to deduct (not add!) the Laplacian from the original image. You can use the Laplacian for image smoothing, for which you add (not subtract) the Laplacian to the image - which is the isotropic heat equation (equivalent to Gaussian smoothing). Some of you wrote that convolution of an image with a Laplacian kernel smoothes the image, which is absolutely incorrect - any derivative operator is a high pass filter! Some of you wrote that the Laplacian is used in the Perona Malik equation, which is imprecise, unless you specify how ($div(g\Delta I) = g_x I_x + g I_{xx} + g_y I_y + g I_{yy}$).

6. What is the difference between the Discrete Fourier transforms of $f(t) = \sin(2\pi at)$ and $g(t) = \sin(2\pi at + \alpha)$? (assume both $f(t)$ and $g(t)$ are discrete time signals)

   ANSWER:

   $g(t) = \sin(2\pi at+\alpha) = \cos(\alpha)\sin(2\pi at)+\sin(\alpha)\cos(2\pi at)$. $F(u)$ will have peaks at frequencies a and $-a$, whereas $G(u)$ will have peaks at the same frequencies, but the peaks will be scaled by $\frac{1}{2}\sin(\alpha) + \frac{1}{2j}\cos(\alpha)$ and $\frac{1}{2}\sin(\alpha) - \frac{1}{2j}\cos(\alpha)$ respectively. Alternatively, $G(u) = F(u)e^{-j2\pi\alpha u}$.

7. Consider a blurred image $g(x, y)$ formed by the degradation of an underlying clean image $f(x, y)$ - represented by the equation $g(x, y) = h(x, y) * f(x, y)$ where $*$ denotes the convolution operator. If you knew the blur kernel $h(x, y)$, how will you estimate $f(x, y)$? Now, instead, if the image $g(x, y)$ were also noisy (a much more realistic scenario), represented by the equation $g(x, y) = h(x, y) * f(x, y) + \eta(x, y)$, would you use the same method to estimate $f(x, y)$? Why (not)?

   ANSWER:

   In the first case, we have $G(u, v) = H(u, v)F(u, v)$ and hence we estimate $F(u, v) = G(u, v)/H(u, v)$. The inverse Fourier transform of $F(u, v)$ will give you $f(x, y)$. In the latter case, due to noise, we have $G(u, v) = H(u, v)F(u, v)+ N(u, v)$. Now, if we estimate $F(u, v) = G(u, v)/H(u, v)$, we will incur an error of $N(u, v)/H(u, v)$ which will be very large at high frequencies, where $H(u, v)$ is small (as it is the Fourier transform of a low pass filter) and $N(u, v)$ is generally comparable to or larger than $F(u, v)$. So, we cannot use the inverse filter in this case.

8. Are the operations $\triangle(g(x, y)*f(x, y))$, $(\triangle g(x, y))*f(x, y)$ and $g(x, y)*\triangle f(x, y)$

equivalent? Why (not)? Note that $\triangle f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$, and $*$ denotes the convolution operator.

ANSWER:

Computing the Laplacian of an image is equivalent to convolving it with the mask $h = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$. Now convolutions are associative and commutative, and hence all three expressions are equivalent. Another way to do this is to observe that $\mathcal{F}(\triangle h(x,y))(u,v) = -4\pi(u^2+v^2)H(u,v)$. Hence the Fourier transforms of all three expressions will be $-4\pi(u^2+v^2)F(u,v)G(u,v)$.
Some students came up with an interesting 'counterexample' in the continuous case and where the signals are defined from $-\infty$ to $+\infty$. For example, $f(x,y) = x, g(x,y) = x^3$. In this case, the three expressions are not equivalent. In fact, in this case the convolutions don't even commute! What's going on? It turns out that the convolution integral is defined only for the case where the signals decay rapidly to 0 as you move towards the extremes of the real line, i.e. towards $-\infty$ or $+\infty$. I didn't mention this in class, and neither does the book. In the case of $f(x,y) = x, g(x,y) = x^3$, this decay certainly does not happen, and hence these integrals don't exist. Now the following question arises: what if these were discrete functions of finite support, i.e. non-zero only for $A \leq x \leq B, C \leq y \leq D$, for some values of A, B, C, D. In this case, you will see that all the three expressions in this question are equivalent. Try it out in MATLAB (say), with the Laplacian kernel, and $f(x,y) = x, g(x,y) = x^3$.

9. Describe the procedure to compute the principal components of a given set of N images (do not derive the mathematics, only enlist the steps).

   ANSWER:

   See lecture slides.

10. This is a toy problem to test your understanding of PCA. Consider a set of N images, each containing exactly three pixels. Suppose the intensity value in the first pixel varies between 0 and 1 across the N images, the intensity value of the second pixel varies between 0 and 255 across the N images, and the intensity value of the third pixel varies between 0 and 10 across the N images. Suppose N is much larger than three. Can you make a guess about the eigenvectors and eigenvalues of the covariance matrix of these N images and justify it? I am not looking for exact answers, just an educated guess with justification.

ANSWER:

PCA seeks to find successive directions such that projections onto those directions will minimize the reconstruction error. These directions turn out to be directions of largest variance, second largest variance, third largest variance and so on. In this case, the largest variance will be in the second pixel. So the eigenvector corresponding to the highest eigenvalue will be approximately $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$. The eigenvector corresponding to the second highest eigenvalue will be approximately $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ and the eigenvector corresponding to the least eigenvalue will be approximately $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$. The eigenvalues give the variance of the projections of the data vectors along those directions. The three eigenvalues in this case will be approximately $255^2, 10^2$ and 1^2.

11. Imagine that several small regions of an image are degraded by the addition of a sinusoidal pattern $\sin 2\pi(ax + by)$ where the values of a and b are known to you. Looking at the Fourier transform of the resulting image (either phase or magnitude, real part or complex part), is it possible to locate exactly which regions of the image were degraded? If yes, explain how. If not, explain why not, and state any modifications to the Fourier transform to be able to locate the degraded regions.

ANSWER:

This question seeks to highlight an important limitation of the Fourier transform. The Fourier transform tells you which frequency components are present in a given signal or image, and what the strength (amplitude) of each component is (note: when I say frequency component, I am referring to a complex sinusoid $e^{j2\pi ux}$ of frequency u). But the Fourier transform generally does NOT tell you which **part** of the signal contained a particular frequency component. In fact, look at the formula for the Fourier Transform - the spatial coordinates x and y are integrated out!

If the sinusoid $\sin 2\pi(ax + by)$ occurred in exactly one small region of the image centered at coordinate (x_0, y_0), then you could look at the phase angle of the frequency (a, b) which can give you some idea of the location of this region which contained the sinusoid. This phase would be $e^{-j2\pi(x_0a+y_0b)}$. Even

here, you would have information only on the direction of the location vector (x_0, y_0), not its magnitude. In general, if the sinusoidal degradation occurred at multiple locations, you will have no idea of where the degradation occurred. The only way out is to divide the images into small patches, and compute the Fourier transform of each patch separately. If the Fourier transform of a particular patch contains a peak at the frequencies (a, b) and $(-a, -b)$, you can conclude that the sinusoidal pattern exists **somewhere** inside that patch. This is called as a short-time Fourier transform or a windowed Fourier transform.

12. 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. After this, prove the validity of the expression for the second directional derivative of $I(x, y)$ in the direction of its gradient vector. Note that the first directional derivative of $I(x, y)$ in a direction v is given by $\nabla I(x, y) \cdot v$.

ANSWER:

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 (though I did not take off points for it), 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.