UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Department of Electrical and Computer Engineering

ECE 417 Multimedia Signal Processing
Fall 2019

PRACTICE EXAM 2 SOLUTIONS

Tuesday, October 22, 2019

Problem 1 (30 points)

Suppose you have an RGB image $i[n_1, n_2, n_3]$ with $0 \le n_1 < N_1$ rows, $0 \le n_2 < N_2$ columns, and $0 \le n_3 < 3$ color planes. The matched filter in part (d) of this problem is of size $M_1 \times M_2$. Use big-$\mathcal{O}$ notation, in terms of the variables N_1, N_2, M_1 and M_2, to express the complexity of each of the following operations:

(a) Coverting from RGB to YPbPr color space.
Solution:

Big-O notation is defined as: an operation is $\mathcal{O}\{f(N_1, N_2, M_1, M_2)\}$ if and only if there exist some positive constants, $G, N_1^*, N_2^*, M_1^*, M_2^*$, such that the number of operations is $\le Gf(N_1, N_2, M_1, M_2)$ for all $N_1 \ge N_1^*$, $N_2 \ge N_2^*$, $M_1 \ge M_1^*$, and $M_2 \ge M_2^*$.

The conversion from RGB to YPBPr involves a 3×3 matrix multiplication per pixel:

$$\begin{bmatrix} Y \\ P_b \\ P_r \end{bmatrix} [n_1, n_2] = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} [n_1, n_2],$$

which is 9 scalar multiply-add operations per pixel. Since there are $N_1 \times N_2$ pixels, the total number of multiplications required is $9N_1N_2$, which is $\mathcal{O}\{N_1N_2\}$

(b) Computing the horizontal and vertical gradients of each color plane using a Sobel mask.
Solution:

Sobel mask is a convolution, as

$$G_x[n_1, n_2] = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} ** I[n_1, n_2], \quad G_y[n_1, n_2] = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} ** I[n_1, n_2]$$

there are 6 nonzero coefficients in each filter (12 total), so we have a total of 12 additions per output pixel. The # output pixels is $(N_1 + 2)(N_2 + 2)$, or N_1N_2, or $(N_1 - 2)(N_2 - 2)$, depending on whether we're doing full, same, or valid convolution – but in any case, all of these are $\mathcal{O}\{N_1N_2\}$. Total computation is therefore $12N_1N_2$, which is $\mathcal{O}\{N_1N_2\}$.

(c) Lowpass filtering (after zero-padding, so that the output is of the same size, $N_1 \times N_2 \times 3$, as the input) with a separable ideal anti-aliasing filter whose frequency response is

$$H(\omega_1, \omega_2) = \begin{cases} 1 & |\omega_1| < \frac{\pi}{3}, \quad |\omega_2| < \frac{\pi}{3} \\ 0 & \text{otherwise} \end{cases}$$

Solution:

The filter is separable, so we can filter each row first, then each column.

Filtering each row requires implementing the following equation once per output pixel:

$$y[n_1,n_2] = h_2[n_2] * x[n_1,n_2] = \sum_{m_2=0}^{N_2-1} x[n_1,m_2]h_2[n_2-m_2]$$

which requires N_2 multiply-add operations per output pixel. Since there are $N_1 \times N_2$ output pixels, the total computation is $N_1 N_2^2$.

Filtering each column requires implementing the following equation once per output pixel:

$$h_1[n_1] * y[n_1,n_2] = \sum_{m_1=0}^{N_1-1} y[m_1,n_2]h_1[n_1-m_1]$$

which requires N_1 multiply-add operations per output pixel. Since there are $N_1 \times N_2$ output pixels, the total computation is $N_1^2 N_2$.

So the total computation requires $N_1N_2(N_1+N_2)$ multiply-accumalate operations.

The function $N_1N_2(N_1+N_2)$ can't be simplified by stripping off any constants or any low-order terms: for example, $N_1N_2(N_1+N_2) \neq \mathcal{O}\left\{N_1^2N_2\right\}$, because, regardless of how large we choose the constant G, there will be large values of N_2 for which $N_1N_2(N_1+N_2) \not< GN_1^2N_2$. Since we can't simplify the polynomial while still keeping it as a strict upper bound on the computation, we need to keep the whole polynomial:

$$N_1N_2(N_1+N_2) = \mathcal{O}\left\{N_1N_2(N_1+N_2)\right\}$$

(d) Filtering with a matched filter of size M_1 rows, M_2 columns.
Solution:

Matched filtering is computed as

$$z[n_1,n_2] = \sum_{m_1=0}^{M_1-1}\sum_{m_1=0}^{M_2-1} h[m_1,m_2]x[n_1-m_1,n_2-m_2]$$

The double sum here can't be simplified, because the filter is not separable. Therefore, each output pixel requires computing a double-sum with M_1M_2 terms in it.

There are N_1N_2 output pixels, so in total, we need $N_1N_2M_1M_2$ multiply-accumulate operations, which is $\mathcal{O}\left\{N_1N_2M_1M_2\right\}$.

(e) Calculating the integral image $ii[n_1,n_2] = \sum_{m_1=0}^{n_1}\sum_{m_2=0}^{n_2} i[m_1,m_2,0]$ for all $0 \le n_1 < N_1$ and $0 \le n_2 < N_2$.
Solution:

Each pixel of the integral image requires just four additions:

$$ii[n_1,n_2] = i[n_1,n_2] + ii[n_1-1,n_2] + ii[n_1,n_2-1] - ii[n_1-1,n_2-1]$$

There are N_1N_2 pixels in the integral image, so the total complexity is $4N_1N_2$. If we choose the constants $G = 4, N_1^* = 0, N_2^* = 0$, we find that the total computation is less than or equal to GN_1N_2 for all $N_1 \geq N_1^*$ and $N_2 \geq N_2^*$, therefore the computational complexity is $\mathcal{O}\left\{N_1N_2\right\}$.

(f) Given the integral image, find the box-summation $f[b_1, b_2, e_1, e_2]$ defined as

$$f[b_1, b_2, e_1, e_2] = \sum_{n_1=b_1}^{e_1} \sum_{n_2=b_2}^{e_2} i[n_1, n_2, 0]$$

for all values $0 \leq b_1 \leq N_1 - 1$, $0 \leq e_1 \leq N_1 - 1$, $0 \leq b_2 \leq N_2 - 1$, $0 \leq e_2 \leq N_2 - 1$.
Solution:

We can find the feature, for any given (b_1, b_2, e_1, e_2), using just four additions:

$$f[b_1, b_2, e_1, e_2] = ii[e_1, e_2] - ii[b_1, e_2] - ii[e_1, b_2] + ii[b_1, b_2]$$

There are a total of $N_1^2N_2^2$ different combinations of (b_1, b_2, e_1, e_2) to consider, so the total computation is $4N_1^2N_2^2$, which is $\mathcal{O}\left\{N_1^2N_2^2\right\}$.

Problem 2 (10 points)

Suppose you have an input image with 8-bit integer pixel values, $0 \leq i[n_1, n_2, n_3] \leq 255$, where n_1 is the row index, n_2 is the column index, and n_3 is the color plane. What are the minimum and maximum pixel values that result as the outputs of the following operations:

(a) Convert to a YPbPr color space. What are the minimum and maximum possible values of Y, P_b, and P_r?
Solution:

$$\begin{bmatrix} Y \\ P_b \\ P_r \end{bmatrix}[n_1, n_2] = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}[n_1, n_2],$$

The top row adds up to one, and all coefficients are positive, so $0 \leq Y \leq 255$. The second and third rows each have negative coefficients that total 0.5, and a positive coefficient of 0.5, so $-255/2 \leq P_b \leq 255/2$ and $-255/2 \leq P_r \leq 255/2$.

(b) Compute the horizontal and vertical gradients using a Sobel mask. What are the minimum and maximum possible values of each of the two gradient images?
Solution:

The usual definition of the Sobel mask is:

$$G_x[n_1, n_2] = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * *I[n_1, n_2], \quad G_y[n_1, n_2] = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * *I[n_1, n_2]$$

so $-4 \times 255 \leq G_x[n_1, n_2] \leq 4 \times 255$ and $-4 \times 255 \leq G_y[n_1, n_2] \leq 4 \times 255$.

Problem 3 (5 points)

Consider the infinite-sized image $i[n_1, n_2] = \delta[n_1 - 5]$, i.e.,

$$i[n_1, n_2] = \begin{cases} 1 & n_1 = 5 \\ 0 & \text{otherwise} \end{cases}$$

Use a Sobel mask to find the resulting images $G_x[n_1, n_2]$ and $G_y[n_1, n_2]$.
Solution:

To figure out this answer, it's useful to write the separable form of the Sobel mask, which would be

$$G_x[n_1, n_2] = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * ([1, 0, -1] * \delta[n_1 - 5]) = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * 0 = 0$$

Where the second equality comes from subtracting $1 - 1$ for each pixel of the fifth row, and $0 - 0$ for every other pixel in the image. On the other hand,

$$G_y[n_1, n_2] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * ([1, 2, 1] * \delta[n_1 - 5]) = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * (4\delta[n_1 - 5]) = \begin{cases} 4 & n_1 = 5 \\ -4 & n_1 = 7 \\ 0 & \text{otherwise} \end{cases}$$

Problem 4 (5 points)

Suppose you want to find the horizon line in a grayscale image $i[n_1, n_2]$. Suppose the horizon line is defined to be the row index n_1 that maximizes the brightness difference $BD[n_1]$, defined as

$$BD[n_1] = \sum_{m_2=0}^{N_2-1} \left(\left(\frac{1}{n_1} \sum_{m_1=0}^{n_1-1} i[m_1, m_2] \right) - \left(\frac{1}{N_1 - n_1} \sum_{m_1=n_1}^{N_1-1} i[m_1, m_2] \right) \right)$$

You are given the integral image $ii[n_1, n_2] = \sum_{m_1=0}^{n_1} \sum_{m_2=0}^{n_2} i[m_1, m_2, 0]$. Devise a formula that uses $ii[n_1, n_2]$ to compute $BD[n_1]$ with a small constant number of operations per candidate horizon line.
Solution:

Let's start out by re-arranging the order of summation, so that the m_1 and m_2 sums are in the same order as the definition of the integral image:

$$BD[n_1] = \frac{1}{n_1} \left(\sum_{m_1=0}^{n_1-1} \sum_{m_2=0}^{N_2-1} i[m_1, m_2] \right) - \frac{1}{N_1 - n_1} \left(\sum_{m_1=n_1}^{N_1-1} \sum_{m_2=0}^{N_2-1} i[m_1, m_2] \right)$$

The first term is already an integral image. The second term can be split up into two integral-image-like terms:

$$BD[n_1] = \frac{1}{n_1} \left(\sum_{m_1=0}^{n_1-1} \sum_{m_2=0}^{N_2-1} i[m_1, m_2] \right) - \frac{1}{N_1 - n_1} \left(\sum_{m_1=0}^{N_1-1} \sum_{m_2=0}^{N_2-1} i[m_1, m_2] - \sum_{m_1=0}^{n_1-1} \sum_{m_2=0}^{N_2-1} i[m_1, m_2] \right)$$

...and now, we just substitute the symbol in place of its definition:

$$BD[n_1] = \frac{1}{n_1}\left(ii[n_1-1, N_2-1]\right) - \frac{1}{N_1-n_1}\left(ii[N_1-1, N_2-1] - ii[n_1-1, N_2-1]\right)$$

Problem 5 (5 points)

Consider the problem of upsampling, by a factor of 2, the infinite-sized image

$$x[n_1, n_2] = \delta[n_1 - 5] = \begin{cases} 1 & n_1 = 5 \\ 0 & \text{otherwise} \end{cases}$$

Suppose that the image is upsampled, then filtered, as

$$y[n_1, n_2] = \begin{cases} x[n_1/2, n_2/2] & n_1/2 \text{ and } n_2/2 \text{ both integers} \\ 0 & \text{otherwise} \end{cases} \qquad z[n_1, n_2] = y[n_1, n_2] ** h[n_1, n_2]$$

Let $h[n_1, n_2]$ be the ideal anti-aliasing filter with frequency response

$$H(\omega_1, \omega_2) = \begin{cases} 1 & |\omega_1| < \frac{\pi}{2}, \quad |\omega_2| < \frac{\pi}{2} \\ 0 & \text{otherwise} \end{cases}$$

Find $z[n_1, n_2]$.

Solution:

$$h[n_1, n_2] = h_1[n_1]h_2[n_2] = \left(\frac{1}{2}\right)\text{sinc}\left(\frac{\pi n_1}{2}\right)\left(\frac{1}{2}\right)\text{sinc}\left(\frac{\pi n_2}{2}\right)$$

$$y[n_1, n_2] = \begin{cases} 1 & n_1 = 10 \text{ and } n_2 \text{ a multiple of 2} \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \left(\sum_{p=-\infty}^{\infty} \delta[n_2 - 2p]\right) & n_1 = 10 \\ 0 & \text{otherwise} \end{cases}$$

Convolving along each row gives $h_2[n_2] * y[n_1, n_2]$, which is zero, except on the $n_1 = 10$ row. On that row, $y[n_1, n_2]$ is equal to one on the even-numbered samples, and equal to zero on the odd-numbered samples. The correct answer is the obvious one: the low-pass filter computes a perfect average between 0 and 1, so each pixel winds up with a value of 1/2. If you want to do a more careful analysis, you could notice that this row is an impulse train with a period of $P = 2$, and therefore it has a DTFT which has impulses of area $2\pi/P = \pi$ at $\omega = 0$ and $\omega = \pi$. The LPF keeps only the $\omega = 0$ impulse, thus:

$$\begin{aligned}
h_2[n_2] * y[n_1, n_2] &= \begin{cases} \left(\sum_{p=-\infty}^{\infty} \delta[n_2 - 2p]\right) * \left(\frac{1}{2}\text{sinc}\left(\frac{\pi n_2}{2}\right)\right) & n_1 = 10 \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} \mathcal{F}^{-1}\left\{\left(\frac{2\pi}{2}\sum_{k=0}^{1} \delta\left(\omega - \frac{2\pi k}{2}\right)\right)\left(\begin{cases} 1 & |\omega_2| < \frac{\pi}{2} \\ 0 & \text{otherwise} \end{cases}\right)\right\} & n_1 = 10 \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} \mathcal{F}^{-1}\left\{\pi\delta(\omega)\right\} & n_1 = 10 \\ 0 & \text{otherwise} \end{cases} \\
&= \begin{cases} \frac{1}{2} & n_1 = 10 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

Convolving along each column, then, gives

$$z[n_1,n_2] = h_1[n_1] * h_2[n_2] * y[n_1,n_2] = \left(\frac{1}{4}\right) \text{sinc}\left(\frac{\pi(n_1-10)}{2}\right)$$

Problem 6 (5 points)

The stochastic autocorrelation of a periodic signal is periodic, $R_{xx}[P] = R_{xx}[0]$. How about the signal autocorrelation? Suppose that the frame length is an integer multiple of the number of periods, $L = kP$, so that

$$x[n] = \begin{cases} \text{periodic with period } P & 0 \le n \le kP-1 \\ 0 & \text{otherwise} \end{cases}$$

Find $r_{xx}[P]$ in terms of $r_{xx}[0]$.

Solution:

$$r_{xx}[n] = \sum_{m=-\infty}^{\infty} x[m]x[m-n] = \sum_{m=0}^{L-|n|-1} x[m]x[m-n],$$

where the second equality is true because there are only $L-1$ nonzero samples in a frame. Since $x[n] = x[n+P] = x[n+2P] = \ldots$,

$$r_{xx}[0] = \sum_{m=0}^{L-1} x^2[m] = k\sum_{m=0}^{P-1} x^2[m]$$

Likewise,

$$r_{xx}[P] = \sum_{m=0}^{L-P-1} x[m]x[m-P] = \sum_{m=0}^{L-P-1} x^2[m] = (k-1)\sum_{m=0}^{P-1} x^2[m]$$

So

$$r_{xx}[P] = \left(\frac{k-1}{k}\right) r_{xx}[0]$$

Problem 7 (10 points)

Consider the signal $x[n] = \beta^n u[n]$, where $u[n]$ is the unit step function.

(a) Find the LPC coefficient, α, that minimizes ε, where

$$\varepsilon = \sum_{n=-\infty}^{\infty} e^2[n], \quad e[n] = x[n] - \alpha x[n-1]$$

Solution:

$$\varepsilon = \sum_{n=-\infty}^{\infty} (x[n] - \alpha x[n-1])^2 \tag{1}$$

$$= 1 + \sum_{n=1}^{\infty} \left(\beta^n - \alpha\beta^{n-1}\right)^2 \tag{2}$$

Differentiating w.r.t. α gives

$$\frac{\partial\varepsilon}{\partial\alpha} = -2\sum_{n=1}^{\infty} \beta\left(\beta^n - \alpha\beta^{n-1}\right)$$

which is zero iff $\alpha = \beta$.

(b) Find the signal $e[n]$ that results from your choice of α in part (a).
Solution:

$$e[n] = \beta^n u[n] - \alpha\beta^{n-1}u[n-1] = \beta^n\left(u[n] - u[n-1]\right) = \delta[n]$$

Problem 8 (10 points)

Consider the LPC synthesis filter $s[n] = e[n] + \alpha s[n-1]$.

(a) Under what condition on α is the synthesis filter stable?
Solution:

The roots of the polynomial $1 - \alpha z^{-1}$ must be inside the unit circle. That's a first-order polynomial, its only root is $z^{-1} = \alpha$, so we just need $|\alpha| < 1$.

(b) Assume that the synthesis filter is stable. Suppose that $e[n]$ is the pulse train $e[n] = \sum_{p=-\infty}^{\infty} \delta[n - pP]$. As a function of α, P, and ω, what is the DTFT $S(e^{j\omega})$? You need not simplify, but your answer should contain no integrals or infinite sums.
Solution:

The DTFT of the pulse train is a pulse train,

$$E(e^{j\omega}) = \left(\frac{2\pi}{P}\right)\sum_{k=0}^{P-1} \delta\left(\omega - \frac{2\pi k}{P}\right)$$

The DTFT of the synthesized signal is

$$S(e^{j\omega}) = H(e^{j\omega})E(e^{j\omega}) = \frac{E(e^{j\omega})}{1 - \alpha e^{-j\omega}}$$

So

$$S(e^{j\omega}) = \frac{1}{1 - \alpha e^{-j\omega}}\left(\frac{2\pi}{P}\right)\sum_{k=0}^{P-1} \delta\left(\omega - \frac{2\pi k}{P}\right)$$