CS 445 HW #1 Solutions

1. **Text problem 2.2**

   Brightness adaptation

2. **Text problem 2.5**

   From the geometry of Fig. 2.3, 7mm/35mm = *z*/500mm, or *z* = 100mm. So the target size is 100mm on the side. We have a total of 1024 elements per line, so the resolution of 1 line is 1024/100 = 10 elements/mm. For line pairs we divide by 2, giving an answer of 5 lp/mm.

3. **Text problem 2.7**

   The image in question is given by

$$\begin{aligned} f(x,y) &= i(x, y)r(x, y) \\ &= 255e^{-[(x-x0)^\wedge 2+(y-y0)^\wedge 2]}(1.0) \\ &= 255e^{-[(x-x0)^\wedge 2+(y-y0)^\wedge 2]} \end{aligned}$$

   A cross section of the image is shown in Fig. P2.7(a). If the intensity is quantized using m bits, then we have the situation shown in Fig. P2.7(b), where $DG = (255 + 1) / 2^m$. Since an abrupt change of 8 gray levels is assumed to be detectable by the eye, it follows that $DG = 8 = 256 / 2^m$, or $m = 5$. In other words, 32, or fewer, gray levels will produce visible false contouring.

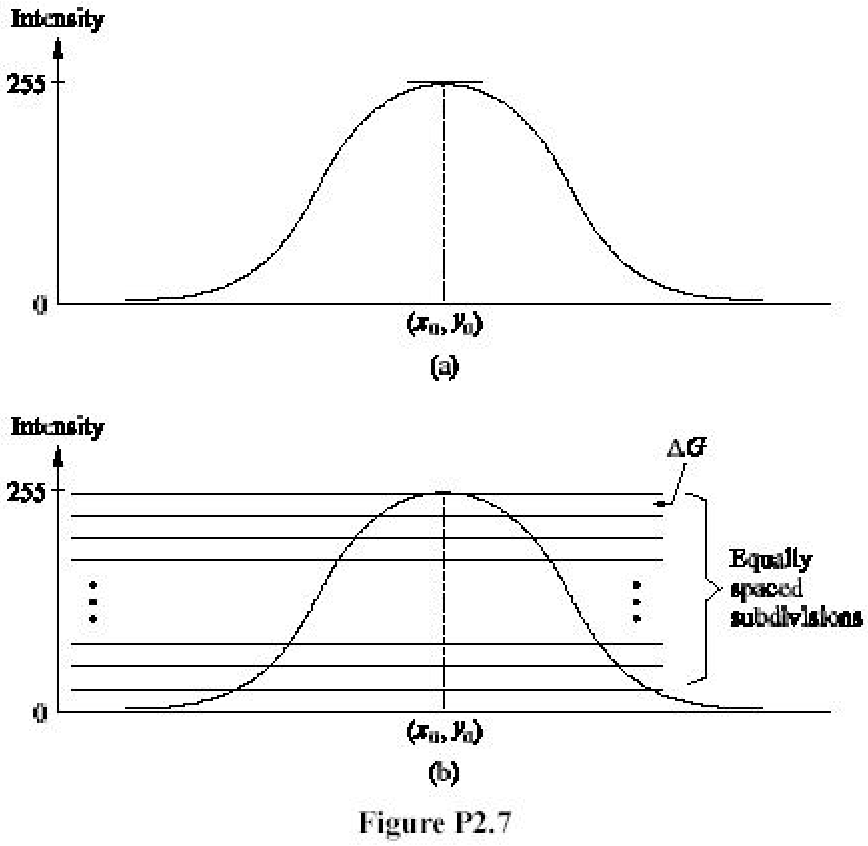

Figure P2.7

4. Text problem 2.9

(a) The total amount of data (including the start and stop bit) in an 8 – bit, *1024 x 1024* image, is $(1024)^2 \times [8 + 2]$ bits. The total time required to transmit this image over a At 56K baud link is $(1024)^2 \times [8 + 2] / 56000 = 187.25$ sec or about 3.1 min. (b) At 750K this time goes down to about 14 sec.

5. Text problem 2.10

The width-to-height ratio is 16/9 and the resolution in the vertical direction is 1125 lines (or, what is the same thing, 1125 pixels in the vertical direction). It is given that the resolution in the horizontal direction is in the 16/9 proportion, so the resolution in the vertical direction is $(1125) \times (16/9) = 2000$ pixels per line. The system "paints" a full *1125 x 2000*, 8-bit image every 1/30 sec for each of the red, green, and blue component images. There are 7200 sec in two hours, so the total digital data generated in this time interval is $(1125)(2000)(8)(30)(3)(7200) = 1.166 \times 10^{13}$ bits, or 1.458×10^{12} bytes (i.e., about 1.5 terrabytes). These figures show why image data compression (Chapter 8) is so important.

6. Text problem 2.11

Let p and q be as shown in Fig. P2.11. Then, (a) S_1 and S_2 are not 4-connected because q is not in the set $N_4(p)$; (b) S_1 and S_2 are 8-connected because q is in the set $N_8(p)$; (c) S_1 and S_2 are m-connected because (i) q is in $N_D(p)$, and (ii) the set $N_4(p) \cap N_4(q)$ is empty.

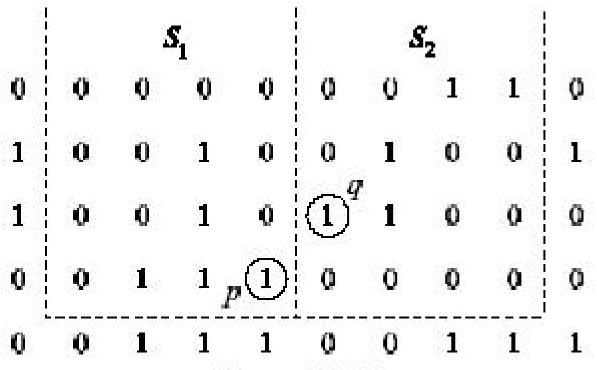

Figure P2.11

7. Text problem 2.15

(a) When $V = \{0, 1\}$, 4-path does not exist between p and q because it is impossible to get from p to q by traveling along points that are both 4-adjacent and also have values from V. Figure P2.15(a) shows this condition; it is not possible to get to q. The shortest 8-path is shown in Fig. P2.15(b); its length is 4. The length of the shortest m- path (shown dashed) is 5. Both of these shortest paths are unique in this case. (b) One possibility for the shortest 4-path when $V = \{1, 2\}$ is shown in Fig. P2.15(c); its length is 6. It is easily verified that another 4-path of the same length exists between p and q. One possibility for the shortest 8-path (it is not unique) is shown in Fig. P2.15(d); its length is 4. The length of a shortest m-path (shown dashed) is 6. This path is not unique.

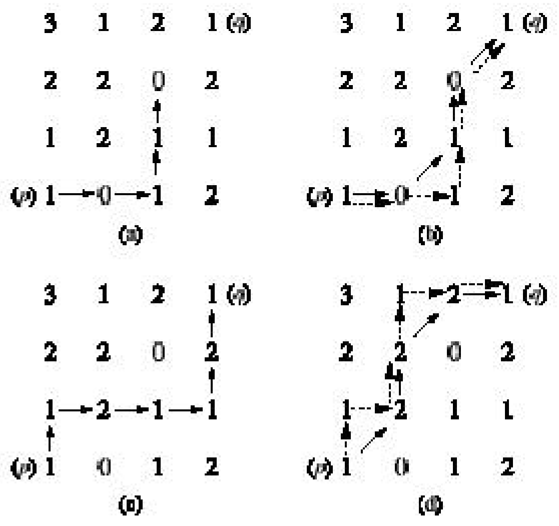

Figure P2.15

8. Text problem 2.18

With reference to Eq. (2.6-1), let *H* denote the neighborhood sum operator, let S_1 and S_2 denote two different small subimage areas of the same size, and let $S_1 + S_2$ denote the corresponding pixel-by-pixel sum of the elements in S_1 and S_2, as explained in Section 2.5.4. Note that the size of the neighborhood (i.e., number of pixels) is not changed by this pixel-by-pixel sum. The operator *H* computes the sum of pixel values is a given neighborhood. Then, $H(a\,S_1 + b\,S_2)$ means: (1) multiplying the pixels in each of the subimage areas by the constants shown, (2) adding the pixel-by-pixel values from S_1 and S_2 (which produces a single subimage area), and (3) computing the sum of the values of all the pixels in that single subimage area. Let ap_1 and bp_2 denote two arbitrary (but *corresponding*) pixels from ($a\,S_1 + b\,S_2$). Then we can write

$$\begin{aligned}
H(aS_1 + bS_2) &= \sum_{p_1 \in S_1 \text{ and } p_2 \in S_2} ap_1 + bp_2 \\
&= \sum_{p_1 \in S_1} ap_1 + \sum_{p_2 \in S_2} bp_2 \\
&= a \sum_{p_1 \in S_1} p_1 + b \sum_{p_2 \in S_2} p_2 \\
&= aH(S_1) + bH(S_2)
\end{aligned}$$

which, according to Eq. (2.6-1), indicates that *H* is a linear operator.

9. Text problem 2.19

The median, ζ, of a set of numbers is such that half the values in the set are below ζ and the other half are above it. A simple example will suffice to show that Eq. (2.6-1) is violated by the median operator. Let $S_1 = \{1,-2, 3\}$, $S_2 = \{4, 5, 6\}$, and $a = b = 1$. In this case *H* is the median operator. We then have $H(S_1 + S_2) =$ median$\{5, 3, 9\} = 5$, where it is understood that $S_1 + S_2$ is the element-by-corresponding-element sum of S_1 and S_2. Next, we compute $H(S_1) =$ median$\{1,-2, 3\} = 1$ and $H(S_2) =$ median$\{4, 5, 6\} = 5$. Then, since $H(a\,S_1 + b\,S_2) \neq$

$aH(S_1) + bH(S_2)$, it follows that Eq. (2.6-1) is violated and the median is a nonlinear operator.

<u>10.</u>

x/lambda = -X/(Z-lambda), where lambda = 4
Therefore, from the two images we have,
x1/4 = -10/(10-4) so x1=-20/3
x2/4 = -(10+5)/(10-4) so x2=-10
xd= |x1 - x2| = 10/3