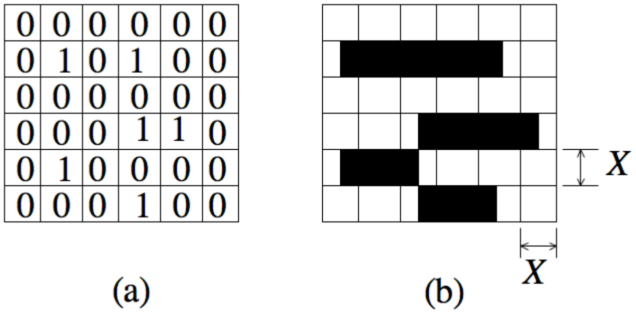
Due Tuesday, 11/23/2021 at 11:59 PM EST

ECE 638 Homework No. 6 Fall 2021

Be sure to turn in all your Matlab or Python code for the problems below.

1. Halftoning with a printer that obeys a fat-dot model

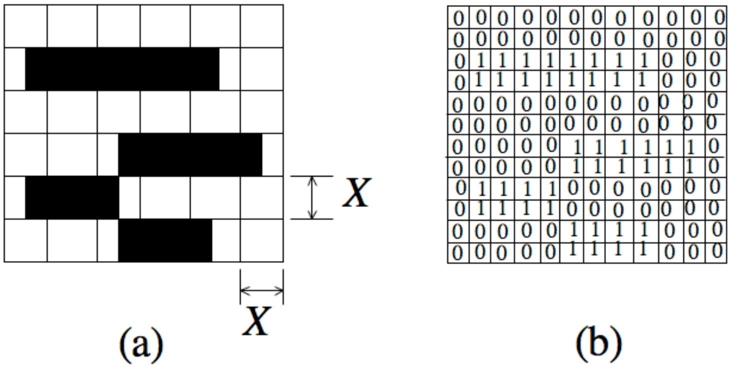
For this problem, we will use a simple *fat-dot* model. Specifically, let the unit square of the addressable printer lattice be $X \times X$. An ideal printer would generate square dots of size $X \times X$ and constant absorptance 1 within this square and absorptance 0 outside of it. Instead, we assume that each dot is in fact X units high and $2X$ units wide; so the printer dot extends from $-X/2$ to $X/2$ in the process direction and from $-X$ to X in the scan direction. We again assume that the dot has constant absorptance 1 within this rectangle and absorptance 0 outside of it. Where dots overlap, we assume that the resulting absorptance is 1, i.e. a logical "OR". The figure below shows a sample bit map and the resulting printer output.



0	0	0	0	0	0
0	1	0	1	0	0
0	0	0	0	0	0
0	0	0	1	1	0
0	1	0	0	0	0
0	0	0	1	0	0

Fig. 6.1: Fat dot printer model: (a) bit map, (b) printed image.

It seems unlikely that any printer manufacturer would want to allocate resources to actually build a printer that behaves according to the fat dot model; so to see the effect that this printer model has on your halftoning algorithm, you will need to develop a simulated printer. To do this, you will double the resolution of the halftone image using an array of 2×2 binary pixels in the simulated printer output to represent each $X \times X$ pixel at the native printer resolution. The figure below demonstrates how such a process would simulate the appearance of the printer output shown in Fig. 6.1(b).



0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	0	0	0
0	1	1	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	0
0	1	1	1	1	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	0	0	0
0	0	0	0	0	1	1	1	1	0	0	0

Fig. 6.3: Simulating a fat dot printer: (a) Fat dot printer output, (b) digital simulation of fat dot printer.

a. Write a Matlab or Python program to generate halftone images that simulate a printer with the fat-dot model. Turn in your program. Generate halftone images for the *ramp* image and the *woman_bw* image using the 8x8 periodic, clustered-dot screen that you developed for Problem 2 (b) in HW No. 5 and using the 8x8 Bayer screen from Problem 2 (d) in HW No. 5. Turn in print-outs of these four halftone images. Discuss the characteristics of the halftone images generated by these two different screens for the fat-dot printer.

b. Determine the tone-reproduction curves for these two screens when used with the fat-dot printer and your simulated printer. Based on your tone-reproduction curves, determine tone-correction curves for each of the two screens. Turn in plots of your tone-reproduction and tone-correction curves. For each screen, apply the respective tone-correction curve to the continuous-tone the *ramp* image and the *woman_bw* image, and regenerate the four halftone images. Turn in these four halftone images, and discuss the difference between your halftone images for part (a) and part (b).

2. Error diffusion

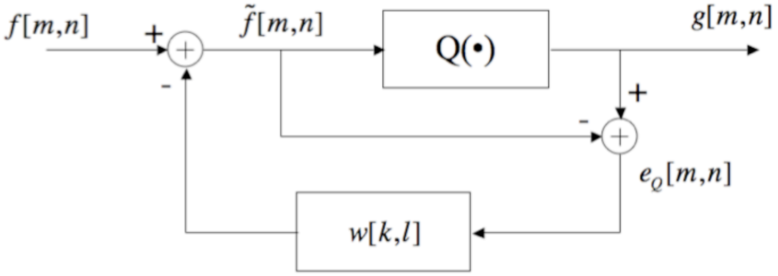


Fig. 7.1: Block diagram for standard error diffusion.

Figure 7.1 shows the block diagram for standard error diffusion. In our case, the halftone output $g[m,n]$ is binary. So $g[m,n]=0$ or 1. Thus, the quantizer $Q(\bullet)$ is a simple thresholding step with threshold value 0.5.

a. Write a program in either Matlab or Python to implement standard error diffusion using the weights originally proposed by Floyd and Steinberg. Turn in your code.

b. Use your program to generate error-diffusion halftones of both the *ramp* image and the *woman_bw* image. Turn in print-outs of all images with your homework assignment. Compare your halftone images with those generated using the screen that you designed using the void-and-cluster algorithm in Problem 4 in HW No. 5.

c. As you did in HW No. 5, compute the Fourier spectrum of the error-diffusion halftone of the *woman_bw* image. Generate an appropriately compressed version of the spectral magnitude, and turn in a print-out of the image with your homework. Compare the characteristics of the spectral magnitude with those of the other halftone images that you generated for Problem 4 in HW No. 5.

3. This problem concerns the task of detecting the banding print defect in a printed test page.

a. From the course website (Data Files), download the file "test_page_banding.tif". This particular test page was printed in landscape orientation or “long-edge-first”. Thus, the short dimension (vertical direction) is the process direction; and the long dimension (horizontal direction) is the scan direction.

b. Assume that image in the file is in the sRGB space. Transform it to linear RGB by digamma-ing it.

c. Apply a 2-D Gaussian filter to remove the halftone pattern. You will have to experiment to find the best value for sigma. Turn in zoomed-in images of a portion of the test page before and after filtering. Note that even though the image appears to have been scanned at 600 dpi, the halftone pattern is not well-resolved. It appears more as noise.

d. Compute the average linear RGB value for the part of each row that contains colorant by performing a horizontal projection. Be sure to mask out the text region that contains the bar-code and text that reports various statistics concerning the print. You do not want these to influence your average linear RGB values.

e. Transform your 1-D signal to CIE L*a*b* via CIE XYZ.

f. Compute the average L*a*b* value for the white paper background where no colorant is printed.

g. Compute DELTA L* DELTA a* DELTA b* between your 1-D projection signal and the average L*a*b* of paper white. Turn in plots of your 1-D DELTA L* DELTA a* DELTA b* signals

h. Compute DELTA E from your DELTA L* DELTA a* DELTA b* signal. Turn in a plot of your 1-D DELTA E signal.

i. Filter your DELTA E signal with some sort of smoothing filter to extract a baseline DELTA E signal. You can experiment with different smoothing filters, including a nonlinear filter, such as the median filter. Turn in a plot of your baseline DELTA E signal.

j. Subtract your baseline DELTA E signal (from part i. above) from your original DELTA E signal (from part h. above). Turn in a plot of the baseline-removed DELTA E signal.

k. Choose an appropriate threshold to detect horizontal bands where the baseline-removed DELTA E signal exceeds that threshold. Turn in a plot of the 1D baseline-removed DELTA E signal, showing where the bands occur.

l. Superimpose your band locations on the original page image, and turn in a print of this image.

m. Estimate the period of the bands. (Note: not all bands will be periodic; and some periodic bands may be missing.)