

ECE 253a MIDTERM EXAM SOLUTIONS

1. Binary Morphology: (8 points)

Two of these are fairly easy. The circle dilated by the disk yields the donut.

$$A_1 \oplus B_1 = C_5$$

And the square outline, dilated by the square, yields the square ring:

$$A_2 \oplus B_2 = C_3$$

The other two are more difficult. The square, when dilated by the disk, will produce a squarish ring where the outer corners are rounded, but the inner corners are squared off.

$$A_2 \oplus B_1 = C_4$$

Lastly, the circle, when dilated by the square, is not any of these six shapes. One might think that the dilation is C_6 because the dilation would in fact be, roughly speaking, a ring, and it would indeed have flattened outer edges in 4 places, but it would *not* have flattened inner edges in those 4 places.

Each of these 4 answers is worth 2 points.

2. Median Filtering: (6 points)

First of all, we can rule out the sparse median filter; it will never be better than the 3x3 square median filter for this type of noise. First, they are both 9-point filters, so they have the same complexity. The sparse filter has the 9 points spread out over a 5x5 square, so those pixel values are in general less correlated with the central pixel. So, if the sparse filter and the regular filter are both deployed on non-noisy images, the sparse filter will introduce more distortion, because it is using less correlated values. The only reason then that the sparse filter sometimes offers an advantage is if the noise tends to be clustered. Consider noise that tends to come as 2x3 blobs. The regular 3x3 square median filter, when it encounters a blob, may have too many noise values within the 3x3 square, and the filtered output will be noisy. The sparse filter, because it samples over a wider area, can avoid getting too many pixels from the noise blob at any one time.

In this problem, you were told that the noise is independent from pixel to pixel. In other words, the noise is not clumped, and therefore this is not a case where the sparse filter offers any advantage.

You can get 1 point if you say that the sparse filter will always be worse than the 3x3 square for this type of noise, and you get 1 point if you give some explanation for why (e.g., by saying that this noise is independent from pixel to pixel, or if you say that sparse filters are good when the noise is clustered).

Now we consider the three levels of noise: 1%, 20% and 40%, and we consider the two median filters: cross and square. The cross is lower complexity, because you only have to put 5 numbers in order rather than 9. In the case of 1% noise, the chance of having 3 out of the 5 pixels in the cross all be noisy simultaneously is very low. So, we might as well use the cross and have the advantage of lower complexity.

For the case of 40% noise, it would be a mistake to use the 5-point cross, because the probability of having 3 out of the 5 pixels in the cross all be noisy simultaneously is not so low, and the output would be wrong. So the square is better in that case.

For the case of 20% noise, I would think it is still better to use the square, but this may be marginal, so you would get credit for this either way.

3. Interpolation: (3 points)

The slow way to do this is to use the equation I gave in class. Horizontally, pixel X is 1/4 of the way over from left to right, and vertically, it is 1/4 of the way down from top to bottom. Therefore, the value of pixel X should be

$$X = \frac{1}{4} \times \frac{1}{4} \times 15 + \frac{1}{4} \times \frac{3}{4} \times 11 + \frac{3}{4} \times \frac{1}{4} \times 7 + \frac{3}{4} \times \frac{3}{4} \times 7 = 8.25$$

The fast way to do this is by remembering that the output of bilinear interpolation is the same as the output of linear interpolation in one direction followed by linear interpolation in the other direction. If we interpolate linearly vertically, the left column has interpolated values 7,7,7,7,7. The right column has interpolated values 11,12,13,14,15. Now we interpolate across the 2nd row, and we get values 7, 8.25, 9.5, 10.75, 12. The value of pixel X is 8.25.

4. Color: (4 points)

Cyan is (1,0,0) so we convert this to RGB using
 $(R,G,B) = (1,1,1) - (C,M,Y) = (0,1,1)$

And we convert these tristimulus values to chromaticity coordinates by

$$r = \frac{0}{0 + 1 + 1} = 0$$

$$g = \frac{1}{0 + 1 + 1} = \frac{1}{2}$$

In the same way, magenta is (0,1,0) in CMY space, and is (1,0,1) in RGB space, and is (0.5,0) in rg chromaticity coordinates.

Yellow is (0,0,1) in CMY space, and is (1,1,0) in RGB space and is (0.5,0.5) in rg chromaticity coordinates.

So the CMY triangle in the rg chromaticity diagram has vertices (0,0.5), (0.5,0), (0.5,0.5).

Full credit can be had if you arrive at this triangle, even if you don't show your steps. If you don't do this all correctly, you can still get partial credit if some of your steps are correct:

1 point for being able to write down the tristimulus coordinates for cyan, magenta and yellow.
1 point for knowing the conversion from CMY to RGB is $(R,G,B) = (1,1,1) - (C,M,Y)$. 1 point for writing any correct equation showing how to convert to chromaticity coordinates, for example

$$r = R/(R + G + B)$$

5. Contrast Enhancement: (8 points)

Image (c) is the image negative. You can tell this because the brightest things in the original image are the darkest things in image (c), and vice versa.

Image (d) is the globally equalized one. There's no major strangeness in this image— it's just higher contrast, as you would expect.

Image (e) comes from remapping function (i). The abrupt transition in the remapping function right at 0.5 is what produces all these strange effects.

Image (f) comes from remapping function (h). The things which were bright in the original image are still fairly bright, but now the things which used to be really dark are fairly bright too.

Lastly, image (g) came from the windows and levels operation. Dark areas are now totally blacked out, and bright areas are completely maxed out at white. It is like a more extreme version of the histogram equalization.

Each of these was worth 1 point.

The remapping function corresponding to global histogram equalization is the cumulative distribution function that comes from the image histogram. This is worth 2 points. The remapping function looks like this:

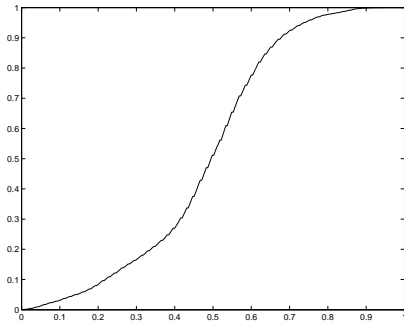


Image (e) shows false contours. This is worth 1 point.