ENGG 111 - Homework 7

Image Compression

For all the tests in the following exercises, use the image provided: CT.tif. These exercises should give you a better sense of how the steps in the JPEG compression procedure work and how effective they are. You are to answer the following questions, and give explanations. You are expected to write a matlab program that will extract the numerical answers from the file given.

a) Given what is the size of the image (pixels) and what would you expect the size of the file to be. Is there a discrepancy and if so, why?

b) How many of all the possible gray values are represented at all (you may want to take a look at the histogram). Which are most represented ?

c) How effective is run-length-encoding in this case (RLE algorithm). You should transform your image into a vector using the reshape() call and write code that does it.

d) Using DCT and the Q matrix I gave in class, and a q value of 1, how much compression for the entire image can you achieve by the combination of DCT+Quantization+entropy-encoding+RLE

e) (Optional, bonus points) How much improvement do you get for adding Huffman coding after completing all the steps in d). You may use the Matlab functions for Huffman coding to test this step, available in the stats toolbox.

3D Arbitrary Plane Intersection

Use the prebuilt stack stack2.mat for this homework. I showed in class how to produce principal plane images from a stack by rotating them and scaling them properly. In this homework assignment you will produce an arbitrary plane section of this stack.

First, study the stack and determine the relationships between indexing (r, c, slice) coordinates (x, y, z) and anatomic directions (L/R, A/P, I/S). Pixel sizes are 0.9375 mm, 0.9375 mm, with 1.5 mm slice spacing.

Think of the stack as a cube with coordinates x y and z. The upper surface of the cube is the xy plane at z=zmax. On that surface the plane of interest intersects the x axis at +100 and the y axis at +100. Units are pixels for this discussion. The bottom surface is the xy plane at z=0. On that surface the plane of interest intersects the bottom of the stack on a line that ends at x=xmax and y=ymax-100, and the point y=ymax, x=xmax-100. This oblique plane does not align itself with any of the standard planes.

I would like you to create an image of the pixels intersected by that plane. You should be able to define a destination plane with a central pixel. This central pixel will correspond to the central pixel in the oblique plane.

The next step should be to estimate a maximum extent in the destination image of the x and y ranges. This only needs to be approximate, best make sure that it will extend slightly beyond the actual image, we can trim this later.

Then, you should identify a few points (4) on the oblique plane and define their locations on the destination image. From these two sets of homologous points you should be able to compute a transformation matrix that project images on the source plane onto the destination plane.

Do not forget to take into account the different scaling in the z direction (1.5 mm between slices) and the pixel sizes in xy: 0.9375.

Method 1 - Forward Mapping, Oblique Plane to Image Plane

With the forward transformation described above, you should be able to create an oblique view of the MRI section. Explain what some of the problems may be, when using this method. Show an image and try to put in evidence some of the problems.

Submit pictures of your results for this section.

Method 2 - Reverse Mapping Method with Interpolation

A much better method exists. On the destination plane, which eventually will be your reconstructed image, define a resolution (pixel size) which will be used to display your oblique section. You could chose a 1mm grid, for example. For each pixel center in your destination image, use the reverse mapping transformation to fetch the pixel value at that point in the stack. From the source stack you may want to use interpolation (in 3D, try nearest neighbor and bilinear, extra credit for bicubic) to get the correct value. Once you have calculated the pixel value, set the

destination pixel to it.

Submit pictures of your results for this section.

Discussion

You will not be graded on programming style. You may use any code I provided for a starting point. You may use the tools provided by Matlab (maketform etc.), but I tend to think that they are much less intuitive than straight coding using affine transformations matrices. You are free to use any method that works.

Explain how you went about obtaining your results in great details in your code.