

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
NORTHEASTERN UNIVERSITY

ECE U692 INTRO TO SUBSURFACE SENSING AND IMAGING Spring 2004

Homework Set 5

Due: Friday April 9

This homework has only one problem. Please turn in a *complete* answer. In particular you should include any narrative required to understand your method and comment on your results where appropriate, label and place captions on any figures you include, and in general think of your homework as a short report. The intent is *not* to have you do a lot of writing for the sake of writing, and you can certainly assume that we know what the problem statement was, but your homework should be professionally done and we should not have to guess at how to interpret what you hand in.

Problem 1: Straight-Ray Tomography Problem

In this problem you will use the Matlab functions for doing forward Radon transforms (or, in other words, the mapping caused by taking multiple parallel-beam projections) and the inverse Radon transform (in this case, filtered backprojection) to gain some insight into what the Radon transform does as well how various practical parameters of both the sampling and the inverse transform effect the reconstructions.

Start by reading the Matlab help on the forward and inverse Radon transforms (i.e. type `help radon` and `help iradon` in Matlab). Look carefully at the examples given in the help for each of these two functions. Note that the forward transform example uses a solid square of ones in the middle of a larger zero background, while the inverse transform example uses the “Shepp-Logan phantom”, which Matlab provides in the function `phantom`. Of course you can use `imagesc` or one of its derivative to visualize the phantom.

Note: in designing this problem, I used Matlab 6.5.1 (R13.1). I also looked at the help files in Matlab 6.5 (R13) and 6.1.0 (R12.1). The help in all three versions is, I believe, the same, although R13.1 also has functions for fan-beam straight-ray tomography. I did not check Matlab5.3 so I don't know what it contains in terms of these functions.

- (a) In this part of the problem you will play with the input of the Radon transform function to gain some insight into the results.
 - (i) Start by running the example given in the help for `radon` and looking at the results. I found it more informative to use a grayscale colormap instead of the one used in

- the help example. To understand what this result means, take Radon transforms at a number of single angles (the help file describes how to do this) and plot the results. (Note that a Radon transform at one angle produces a single line, the $P_\theta(t)$ described in class.) Start by taking these projections at angles like 0, 90, 45, 30, and 60 degrees, then making some small variations from these angles.
- (ii) Now create a rectangular object similar to the square one used in the help. Make the aspect ratio significantly far from one but not so narrow as to make the object almost a line. Repeat the above exercises (taking a number of single-angle projections as well as the entire Radon transform from 0 to 180 degrees) and examine your results.
 - (iii) Explain why the single-angle projections look the way they do for both input images in the single angle case. Include as figures a small number of examples to illustrate your comments. Also explain why the “sinogram data”, that is the entire Radon transform, looks as it does in both cases. Explain both why the transform is zero where it is zero as well as the shape and intensity it has where it is non-zero.
- (b) Now we will use the forward and inverse Radon transforms on the Shepp-Logan phantom. First run the example given in the help for `iradon`.
- (i) The `iradon` function allows you to control several parameters of the filtered backprojection. Explore the difference obtained using three of the filter options and two of the interpolation options. Describe what you tried and your results. In this part, keep angles of the projections used the same as in the example in the help.
 - (ii) Suppose you want to only make half as many projections as the 180 used in the help example. One way to do this is to take projections over as wide a range of angles but space them farther apart. Another option is to keep the spacing the same but only cover half the range of angles. In other words, in one case you can take projections every two degrees from 0 to 178 degrees, and in the other case take them every degree but only from 0 to 89. Using the Matlab `radon` and `iradon` functions, try both options and describe what happens.
 - (iii) Now return to the parameters of the example given in the help, but add noise to the projections before inverting. You can use the `randn` function in Matlab to create Gaussian-distributed noise. In particular, if you want a matrix of noise values the same size as a matrix A you can use the command `randn(size(A))`. Use this command to create a noisy version of your Radon projections (that is, first use the Radon projection program and then add noise to the projections). Visually compare the noisy Radon transform to the noise-free one. Then use the noisy Radon transform as input to the `iradon` function and compare the results. Try scaling the noise to different levels by redoing the experiment but multiplying the noise matrix by a scalar before adding it to the Radon transform of the phantom. Describe at what level of noise you lose various features of the phantom in your reconstruction, as well as any other notable aspects of your results.