ENGR 151 Project 2

Thursday September 22nd 2022

This assignment is due Thursday October 13th 2022 at 11.59PM

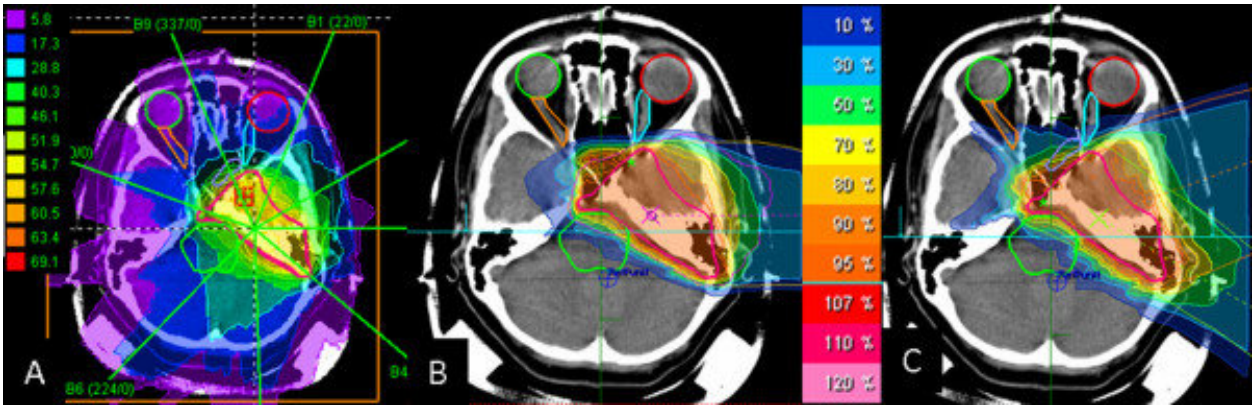

Figure 1: Dose distributions for brain tumor radiotherapy (Fig. from Kosakiet al.Radiation Oncology 2012, 7:44).

1 Overview

In the US alone, there are 1.7 million new cancer patients each year. Radiation therapy is the use of ionizing radiation to treat cancerous tumors. Radiation kills cells by causing DNA damage that leads to cell inactivation or death. Cells are typically the most sensitive to DNA damage from radiation during reproduction. Since cancer cells typically reproduce more quickly than healthy tissues, they are strongly affected by the radiation and die. The goal of radiation therapy is to kill the tumor cells while minimizing damage to healthy tissue. In External Beam Radiation Therapy, photons or electrons are delivered by a linear accelerator or Cobalt-60 photon source from multiple angles around the patient. The overlapping of these beams generates a dose pattern that is concentrated on the tumor as opposed to surrounding healthy tissue. The accuracy of dose delivery is essential for good outcomes, and this is predicted by simulations using radiation transport algorithms that must be fast and accurate, similar to those illustrated in figure 1.

2 Radiation dose

Radiation **dose** is the standard measure of radiation effects, with tumor doses prescribed in units of *Gray* (Gy), which is energy absorbed per unit mass (joules per kilogram). Radiation dose in radiotherapy is typically administered over a few days in few Gy doses up to a maximum of a few tens of Gy. As a radiation beam passes from the outside through the tissue, it deposits dose in the tissue in a way that changes with depth, as illustrated in figure 2. For photon beams (X-rays), the dose rate for a standard intensity field of 1 MeV x-rays

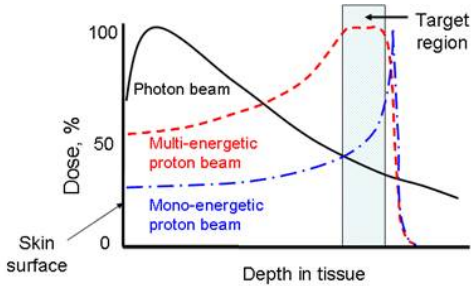

Figure 2: Dose deposited by different sources as a function of depth into tissue.

is approximately 0.01 Gy per second, but the dose rate pattern exponentially falls off and can be approximated by $D \propto \exp(-x/15\ [\text{cm}])$.

3 Design of code

This project is to write a (highly simplified) version of radiation treatment planning software. There are four parts to this project;

1. The first task is to process a series of N patient magnetic resonance imaging (MRI) scans of brains, as shown in figure 3, in files labeled `patient001mri.tif`, `patient002mri.tif` etc. These files will be assumed to be in the working directory of your code, but the total number of images is unspecified (you need to process all files in the directory. You may assume there is at least one file).

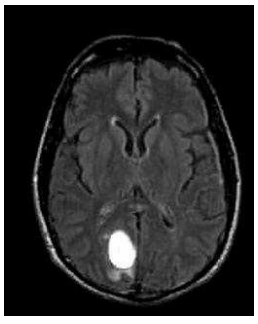

Figure 3: Scan of a brain with tumor highlighted using a contrast agent.

2. For each image file opened from task 1, the data should be a Magnetic Resonance Imaging (MRI) scan of a brain with a contrast agent that helps show the tumor location. By using a **threshold filter** to identify the pixels in the image where the tumor is, generate a list of *horizontal* and *vertical* pixel position pairs, which we can use for the next part of this project, as indicated in the figures 4a and 4b. For the threshold image, use a simple threshold value of 250 and a strictly greater than (>) threshold. The images provided will all be 0-255 grayscale. No other image processing techniques (such as erosion or noise filtering) will be needed.

3. Using the particular row and column positions corresponding to one of the tumor images identified in the previous task, generate crossed radiation beam intensity patterns on an array the same size as the image. The beams should be one pixel thick and enter from the top and right of the image, as illustrated in figure 4b.

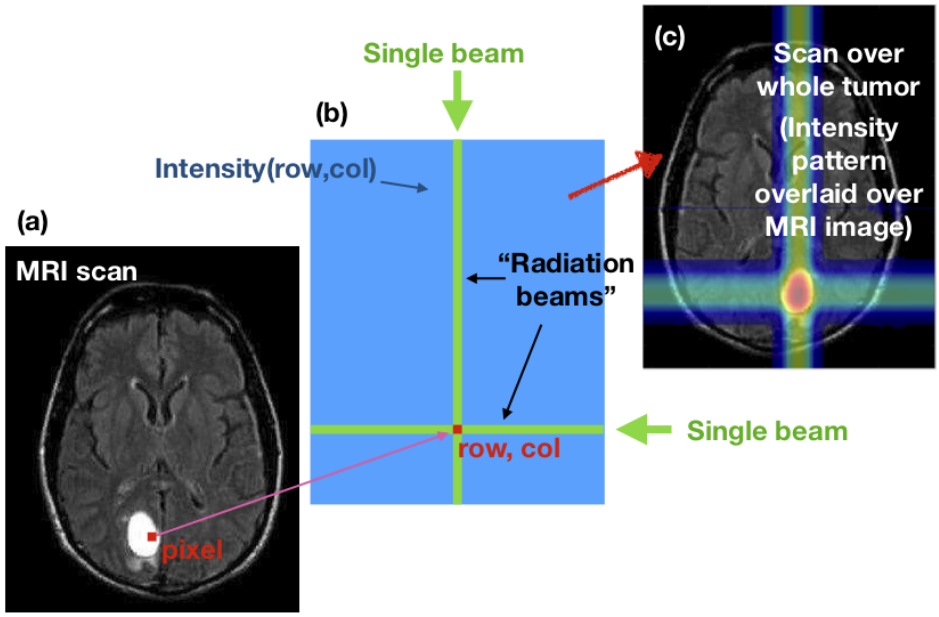

Figure 4: Steps for generating radiation dose intensity pattern. (a) Identify pixel locations of tumor using image threshold. (b) Generate single beam patterns crossed at the pixel location. (c) By adding together the contributions from the individual crossed beams build up an intensity pattern that covers the whole tumor.

4. Using the radiation beam generated from task 3, we will calculate the total radiation and the dose deposited in the tumor.

4 Specifics for project credit

1. For task 1, write a function "load_patient_data" to load these files (using imread) sequentially, with an iteration over the patient files. For each file, write the data from imread into an element of a cell array named "patient_data". Finally, output the

cell array "patient_data" and a variable named "num_files", which should contain the number of files opened. The following functions may be useful,

- `isfile(filename)` will return TRUE if a file exists and FALSE otherwise.
- `num2str(num,'%.3d')` will format a number into a string with 3 digits, e.g. `num2str(1,'%.3d')` returns the string "001'

2. For task 2, the function should be called "rowcol_pairs", taking a single input of the **cell array** "patient_data" generated in the last part (for the autograder, a correct "patient_data" cell array will be loaded, not the data from the previous part) and returning "tumor_coords", which should be an **array of structs**, with fields "rows" and "cols" which are each linear arrays consisting of the rows and column pairs where the tumor is located. For example, tumor_coords(1).rows should be an array with the rows in it.

   The following function may be useful for writing your code,

   - `[rows,cols] = find(A)` finds the rows and columns corresponding to the positions of each nonzero element in matrix A.

   To test your code, a simple test object, "test_object.tif" (figure 6), has been posted to the files folder. If you process the image by `test_object{1} = imread('test_object.tif')`, then run `test_object` through your "rowcol_pairs" function, it should generate the following row and column data:

ROWS	COLS
9	9
10	9
11	9
12	9
13	9
9	10
10	10
11	10
12	10
13	10
9	11
10	11
11	11
12	11
13	11
9	12
10	12
11	12
12	12
13	12
9	13
10	13
11	13
12	13
13	13

Figure 5: Rows and columns generated from the test object

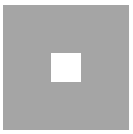

Figure 6: Test object.

3. For task 3, the function should be called "radiation_beam" and should take inputs "rows", "cols", "intensity" and "mri_image". "rows" and "cols" are lists of row and colume pairs, as generated in the previous part and "mri_image" is one of the matrices containing one of the MRI images. The intensity of each beam should be set to 0.5x10^{-8} Gy/s/voxel. Note a voxel is a volume element - this is a 2D image but we are imagining the tumor has some depth. You don't need to worry about this in practice, just set the amplitude of each beam to 0.5x10^{-8}. We are going to assume that the dose deposition does not depend on how far the beam travels in the tissue, so the intensity of the beams is the same everywhere (i.e. is constant within the beam). Looping over the identified pixel positions of the tumor, generate an overall intensity map for the radiation beams (formed as they scan over the tumor), as shown in figure panel (c) below. The function output, "rad_map", is the overall map of the radiation built up by scanning over row/column pairs.

   To test your code, a simple test object, "test_object.tif", has been posted to the files folder. If you run this through your function, you should get the following precise values, as indicated in the visual representation of the matrix below:

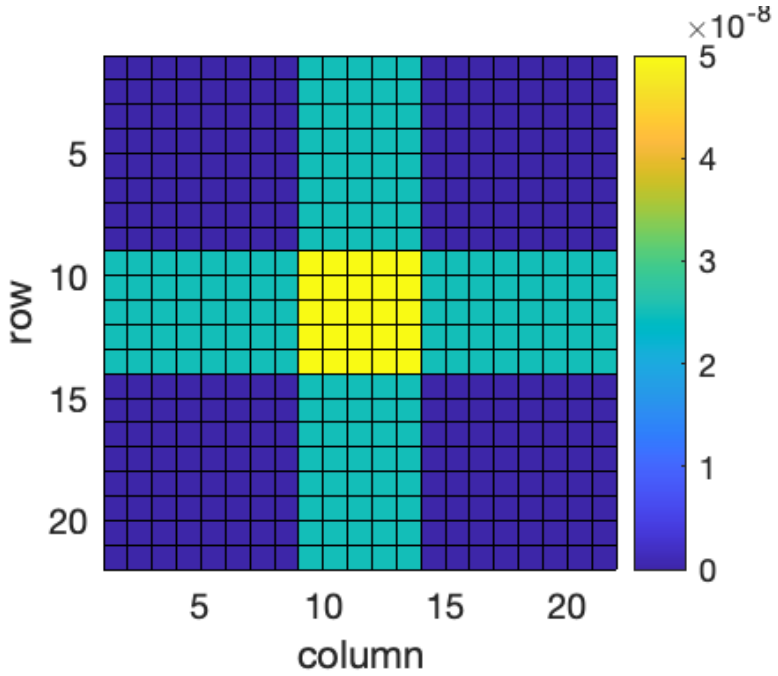

Figure 7: Radiation map of the test object.

Although it is not necessary for grading, it is important to visualize your data to see if your code is working. The following code will overlay the rad_map data over the mri_image. Note that "threshold" is an array of values between 0 and 1 that control the transparency of the image (the image shown below has the array taking the value 0.5 where the beam is and 0 where the intensity is 0).

```
ax1 = axes; %creates Cartesian axes
imagesc(mri_image);
colormap(ax1,'gray');
% alphadata controls the transparency
% threshold is an array of alpha values
ax2 = axes;
imagesc(rad_map,'alphadata',threshold);
colormap(ax2,'jet');
ax2.Color = 'none'; %switches off background color
```

4. For this last part of the project, write a function [tumor_dose,total_dose] = dose_plan (patient_data), which takes in the cell array of patient data and through a call to the function "radiation_beam" generates the radiation map rad_map, as defined in part 3 of this project, using the rows and cols calculated in part 2 of the project. (For the autograder, it will load working functions "radiation_beam" and "rowcol_pairs" so you can get this working even if parts 2 and 3 did not work for you, provided you make the right function calls). For each rad_map calculated for every element in the patient_data cell array, calculate the **total dose** in Gy by summing the rad_map intensity map (don't worry about x and y scales this time, as we have expressed everything in terms of the voxels so use a straightforward sum) for this scan. Then calculate the **dose deposited in the tumor** by only summing over the radiation that is within the tumor (using the same threshold method as in part 2). These should be returned as tumor_dose and total_dose **in units of milliGray** (i.e. x1000).

5 Hints for success

Break the problem into pieces and test each part along the way. Use modular construction, making functions that perform pieces of the task where appropriate and test each piece individually to make sure it works before moving on to the next. Remember, each function needs its own .m file. Start early so you can encounter any stumbling blocks along the way and **make use of office hours** to discuss how to implement algorithms.

6 Grading

You should submit your three files for the four tasks to Canvas under `Project 2`. The Matlab grader will evaluate the results of your submission and provide a maximum score of 100 points (each task is 25 pts).

Please take a look at the file `Style_Expectations.pdf` in the 'Files' of the canvas site for the information on the style and commenting of your code. Start to follow the good practices! For the first two projects, the code style will not be part of your credit, but it will be taken into account from Project 3.