

# Medical Image Processing 67705

*Semester A, 2023-2024*

## Homework

### *Structure segmentation in CT scans*

Prof. Leo Joskowicz

TA: Alon Olesinski, Alina Ryabtsev

**Part 1 Due:** Jan. 22, 2024.

**Part 2 Due:** Feb. 12, 2024.

### **Submission Guidelines**

- Each one of you is required to write and submit your own solution (individuals, no groups).
- Late submission policy: for each day after the submission date, 3 points will be subtracted from the total exercise grade. You may not submit the exercise if you are more than a week late.
- Submit (through Moodle) a single ZIP archive file named “[id]\_[username]\_Targil1.zip”.

The ZIP file should contain:

- Your entire code (only .py files)
- A report as a PDF document including:
  - Your name and ID number (teudat zehut)
  - A description of your solution’s design and results for each part. Add graphs, screen snapshots, and tables as needed to make your point clear.
  - Indications of which libraries – recommended and new -- you used in your code
  - A description of every function/method you wrote with a table of the following format:

Function Name with a short explanation		
Input with explanation		
Output with explanation		
Image 1	Image 2	Image 3

## **Introduction and Background**

Segmentation is an essential task in medical image processing. In this exercise, you will familiarize yourselves with medical CT scans and with software environments for CT scans. You will explore the use of thresholding and multi-seeded region growing for the segmentation of the skeleton and the liver.

In medical image processing, perhaps in contrast to other courses you may have taken before, it is often the case that we do not have a single correct answer. You should try to achieve the best segmentation possible, but keep in mind that there is always observer variability and therefore you should not expect a perfect match (Dice coefficient of 1) with the published solution.

Solutions that are correct and within the estimated variability will earn you a grade of up to 90. The remaining 10 points will be given for originality and resourcefulness.

Use the following directory to get the scans:

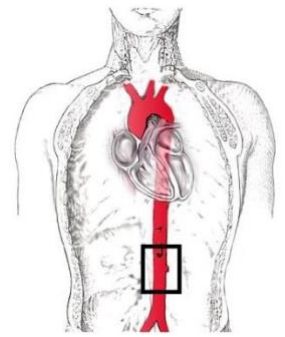
<https://drive.google.com/drive/folders/1HFNkrgKHgfYXe0sYjt9dzL1hRzSAHhS?usp=sharing>

## Scientific Background

In this exercise, we are interested in segmenting the body bones and the aorta. Here are the definitions and drawings of these structures.

Bone (Wikipedia):

*A bone is a rigid organ that constitutes part of the vertebrate skeleton in animals. Bones protect the various organs of the body, produce red and white blood cells, store minerals, provide structure and support for the body, and enable mobility. Bones come in a variety of shapes and sizes and have a complex internal and external structure. They are lightweight yet strong and hard, and serve multiple functions.*

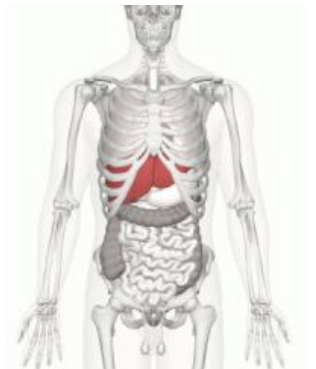


The aorta (אבי העורקים – Wikipedia):

*The aorta is the main and largest artery in the human body, originating from the left ventricle of the heart and extending down to the abdomen, where it splits into two smaller arteries (the common iliac arteries). The aorta distributes oxygenated blood to all parts of the body through the systemic circulation.*

The liver (Wikipedia):

*The liver is a major metabolic organ only found in vertebrate animals, which performs many essential biological functions such as detoxification of the organism, and the synthesis of proteins and biochemicals necessary for digestion and growth. In humans, it is located in the right upper quadrant of the abdomen, below the diaphragm and mostly shielded by the lower right rib cage. Its other metabolic roles include carbohydrate metabolism, the production of hormones, conversion, and storage of nutrients such as glucose and glycogen, and the decomposition of red blood cells.*



CT scan (Wikipedia):

*A CT scan or computed tomography scan (formerly computerized axial tomography scan or CATscan) makes use of computer-processed combinations of many X-ray measurements taken from different angles to produce cross-sectional (tomographic) images (virtual "slices") of specific areas of a scanned object, allowing the user to see inside the object without cutting.*

CT scans are stacks of 2D grayscale images. The gray values in the images correspond to Hounsfield units. Read [here](#) to learn about CT and the typical HU values of various tissues.

## Technical Background

The standard file format for medical images is DICOM. In this exercise, we will use the NIFTI file format which is simpler and better suited for image processing. NIFTI files have the suffix is .nii. Compressed NIFTI files are .nii.gz.

NIFTI files can also be used to store segmentation masks. The information stored in a NIFTI file is:

- Grayscale values: each voxel is an int16 data type, representing grayscale where black is the minimum value and white is the maximum value.
- Segmentation mask values: each voxel holds a value representing to which group it belongs. For instance: 0 – healthy tissue, 1 – old lesions, 2 – new lesions.

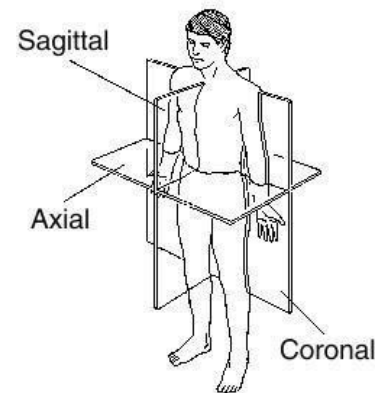
### Reading and writing NIFTI files

First, download and import the [nibabel library](#).

Then, you can read a 3D image using the *get\_fdata* method:

```
img = nib.load('Case1_CT.nii.gz')  
img_data = img.get_fdata()
```

Now, *img\_data* is a 3D numpy image, with the first axis is the Sagittal plane, the second is the Coronal, and the third is the Axial.



### Viewing 3D images

Download and install the free [itksnap program](#).

This program allows for the display and interactive segmentation of 3D images. We will use it only for display. To do so, drag the grayscale image and choose 'Load as main image' and then drag the segmentation image and choose 'load as segmentation'. Note that this is possible only if both images share the same dimensions.

## **Part 1: Bones Segmentation in Contrast CT (40%)**

In this part, you will perform threshold segmentation of the skeleton in a contrast CT.

For this purpose, you need to implement two functions – *segmentation\_by\_th*, which performs the segmentation using its input *i\_min* and *i\_max* thresholds, and *skeleton\_th\_finder* which is used to find the best-suited thresholds.

### **1. Segmentation by Thresholding (20%)**

**Implement the following function:** *segmentation\_by\_th(nifti\_file, i\_min, i\_max)*:

This function is given as inputs a grayscale NIFTI file (.nii.gz) and two integers – the minimal and maximal thresholds. The function generates a segmentation NIFTI file of the same dimensions, with a binary segmentation – 1 for voxels between *i\_min* and *i\_max*, 0 otherwise. This segmentation NIFTI file should be saved under the name

`<nifti_file>_seg_<i_min>_<i_max>.nii.gz`.

The function returns 1 if successful, and 0 otherwise. Preferably, raise descriptive errors when returning 0.

How to find *i\_min* and *i\_max*?

Recall that bones have a maximal HU value of 1000+, so 1300 will do as the *i\_max* for the segmentation. The value of *i\_min* is a bit trickier. You'll need to search for an optimal one.

### **2. Finding Threshold for Skeleton (20%)**

**Implement the following function** *skeleton\_th\_finder(nifti\_file)*:

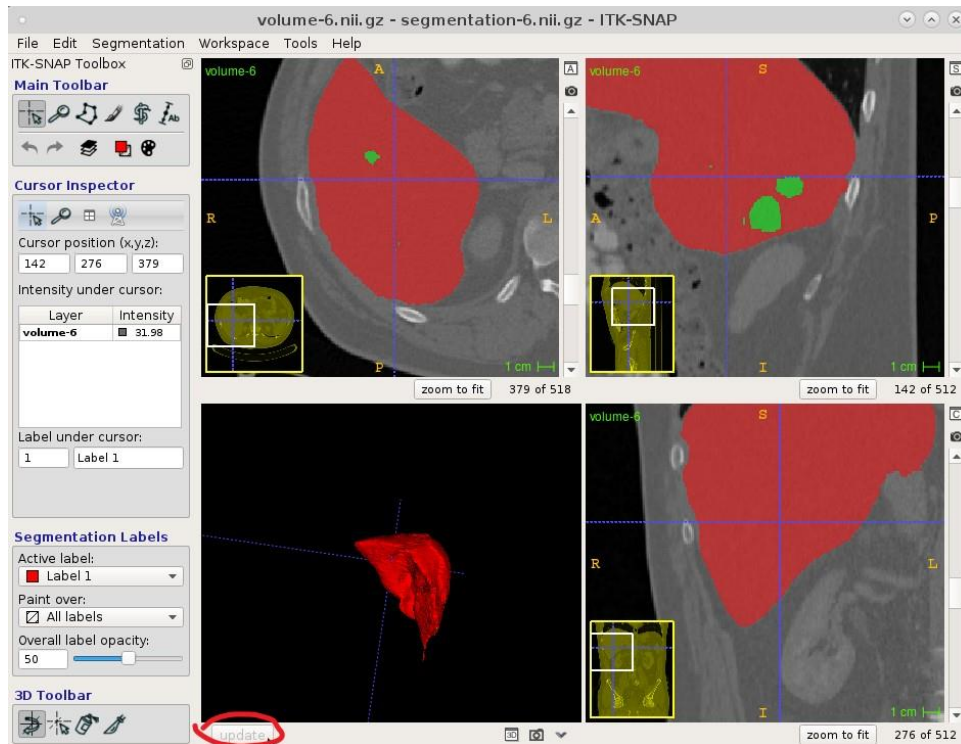
This function iterates over 25 candidate *i\_min* thresholds in the range of [150,500] (in intervals of 14). In each run, use the *segmentation\_by\_th* function and count the number of connectivity components in the resulting segmentation with the current *i\_min*. Plot your results – number of connectivity components per *i\_min*. Choose the *i\_min* which is the first or second minima in the plot. Also, make sure to include that graph in your report.

Next, you need to perform post-processing (morphological operations – clean out single pixels, close holes, etc.) until you are left with a single connectivity component.

Finally, this function should save a segmentation NIFTI file called “<nifti\_file>\_SkeletonSegmentation.nii.gz” and return the *i\_min* used for that.

In your report, mention the final thresholds you used for each scan, and include snapshots of your result segmentations from itknap.

Note that viewing the 3D segmentation in itknap is computed by pressing the 'update' button:



An example of how your skeleton segmentation should look like (snapshot from itknap):



## **Part 2: Region Growing Segmentation of the Liver in CT scans (60%)**


In this section, you will implement MSRG (multiple-seeded region growing) using connectivity and gray-level analysis from several seed points, e.g. multiple-seeded.

Direct single-threshold segmentation for the new CT scans will result in the presence of unwanted objects, e.g. the scanner bed, and unwanted structures, e.g., other internal organs. An example of this:



### **1. Isolate Body in the Scan (5%):**

Implement a method to isolate the patient's body from the presence of unwanted objects as described above.

<i>Function name: isolate_body</i>	
Input: CT Scan Output: Body segmentation Algorithm:	
<ol style="list-style-type: none"><li>1. Perform thresholding to remove all pixels with gray level (HU) below -500 and above 2000 (keep only those between -500 and 2000)</li><li>2. Filter out noise.</li><li>3. Compute the largest connected component.</li></ol>	
<div></div> <div>Desired output:</div>	

To initialize the algorithm, we need to have an area from which we sample seeds. As we mentioned before, simple thresholds will not be sufficient here. Instead, we want to find an ROI of the liver.

## 2. Liver ROI (10%)

Implement a method for finding an ROI in the liver from which you can sample seeds for the MSRG. You are provided with a segmentation of the aorta.

<i>Function name: create_liver_roi</i>
Input: CT Scan Output: Liver ROI Algorithm:  <ol style="list-style-type: none"><li>1. Typically, in CT scans, the liver tissue appears in the intensity range of <math>[-100, 200]</math> HU</li><li>2. The liver is located to the right of the aorta, at approximately half its height. Aorta segmentation is provided in the drive folder.</li></ol> Note: You might find it useful to use the <i>isolate_body</i> function you implemented before.

## 3. Region Growing Algorithm (25%)

Implement the following functions:

<i>Function name: find_seeds</i>
Input: CT Scan and ROI Output: seeds list Algorithm: sample seeds from ROI

<i>Function name: multiple_seeds_RG</i>
Input: CT Scan and ROI Output: liver segmentation Algorithm:  <ol style="list-style-type: none"><li>1. Extract <math>N</math> seeds points inside the ROI.</li><li>2. Perform Seeded Region Growing with <math>N</math> initial points</li></ol>



Note the following:

1. The MSRG should be performed in 3D, i.e., each voxel has 26 neighbors instead of 8 in 2D.
2. Avoid using loops as much as possible. In the SRG iterations, using morphological operations.
3. Seeds are voxels inside the ROI and are part of the liver.
4. Start with ~200 seeds selected from your ROI.
5. Implement your own MSRG: do not use resources such as libraries/packages from the Internet.

#### 4. Evaluation Metrics for the Segmentation (10%)

Use the following function to evaluate your results versus the GT (ground truth) segmentations:

<i>Function name: evaluate_segmentation</i>
Input: 1) Ground truth segmentation; 2) Estimated segmentation
Output: Volume Overlap Difference and Dice Coefficient

#### 5. MSRG Liver Segmentation (10%)

Finally, implement the following fully automated function:

<i>Function name: segment_liver</i>
Input: CT filename, Aorta segmentation filename, output filename
Output: liver segmentation nifti file (to be saved under the given filename)

For this part, you can find the CT files here:

<https://drive.google.com/drive/folders/1HFNkrgKHgfYXe0sYjt9dzL1hRzSAHlhS?usp=sharing>

## **Appendix**

Morphological operations

<http://scikit-image.org/docs/dev/api/skimage.morphology.html>

Other useful functions

<http://scikit-image.org/docs/dev/api/skimage.measure.html#skimage.measure.regionprops>

<http://scikit-image.org/docs/dev/api/skimage.measure.html#skimage.measure.label>