11763 - Medical Image Processing

Antonio Nadal Martínez

May 26, 2024

1 Introduction

This document contains the report corresponding to the final practice of the subject 11763 - Medical Image Processing, of the master's degree in Intelligent Systems of the University of the Balearic Islands. All the content presented in this report can be found in Github. Next, we will address the structure of the Github repository and proceed to resolve the proposed issues.

2 Repository Structure

In the Github repository we can find different files. Regarding dicom files, we find:

- **HCC-TACE-Seg**: contains both the CT images of the patient and the segmentations,
- **RM_Brain_3D-SPGR**: contains the image of the patient on which we have to perform the coregistration.
- **AAL3_1mm.dcm**: contains the atlas of the brain.
- **icbm_avg_152_t1_tal_nlin_symmetric_VI.dcm**: contains the phantom image that will serve to link the patient's image and the atlas, allowing us to locate the thalamus.

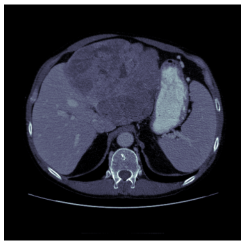
Finally, with respect to the relevant files for the resolution of the proposed problems are:

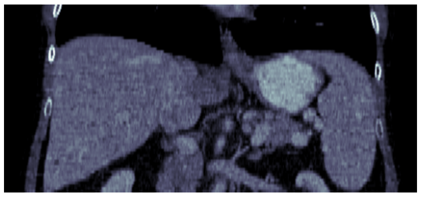
- **ej1_final.py**: contains the resolution of activity 1.
- **ej1_overlayed_animations.py**: contains the necessary code for the generation of different animations.
- **ej2_final.py**: contains the resolution of activity 2.
- **utils_final.py**: contains functions that have been used in the resolution of activities 1 and 2.

3 Activity 01

As part of activity 01, the 3D Slicer software has been used to visualize the images and the associated segmentation, as well as for the initial query of some headers, such as *ImageOrientationPatient*, which has allowed us to know that both patient and segmentation are oriented in the same way. All the development to be discussed below can be found in this file, located in the Github repository of the project.

First, I used the pydicom library to read the images in dicom format. After reading all the images, I verified the presence of a single acquisition, using the header *AcquisitionNumber*. After this check, the different slices have been ordered according to their position, given by the header *ImagePositionPatient*. For purely visual reasons, I have decided to invert the image, to visualize the body as if it were a standing person. To the obtained image I have applied a windowing, to achieve a better visualization. I have also obtained the dimensions (mm) of the voxels using the headers *SliceThickness* and *PixelSpacing*, which has allowed me to represent the images with the correct aspect ratio. The results obtained can be seen in Figure 1.


(a) Axial plane


(b) Coronal plane

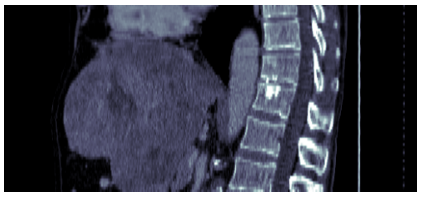
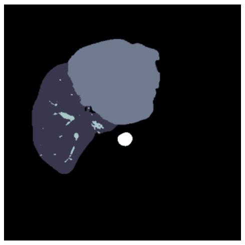
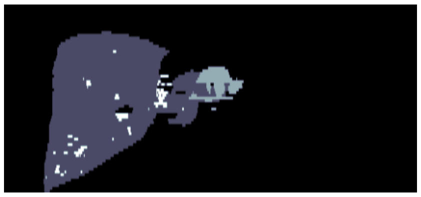

(c) Sagittal plane

Figure 1: CT image after windowing

After that, I loaded the segmentation, again using the pydicom library. Making use of the header *SegmentSequence* I have been able to detect the presence of segmentations of different objects, listed in the header *SegmentNumber* and named in the header *SegmentLabel*. All this information has been used for the correct reading, separation, sorting and representation of the four segmentations present in the file. The result of all the above is reflected in Figure 2.


(a) Axial plane


(b) Coronal plane

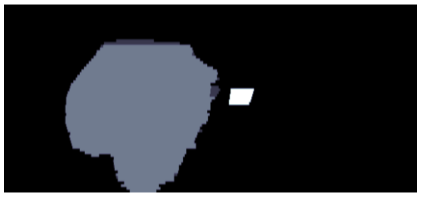
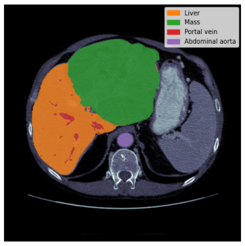
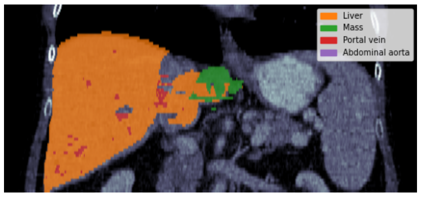

(c) Sagittal plane

Figure 2: Segmentation slices

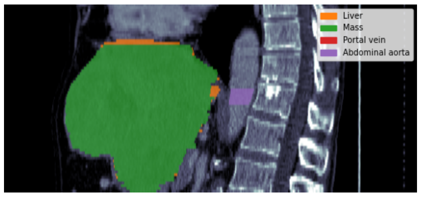
Making use of all the above information, and given that we have built the images on the same spatial reference, we can make use of alpha fusion to represent both the radiological test and the segmented regions in the same image, as can be seen in Figure 3. In addition, different animations to observe the complete volume can be found on GitHub (see axial, coronal and sagittal).



(a) Axial plane



(b) Coronal plane



(c) Sagittal plane

Figure 3: Segmentation slices

To conclude this activity, we have created two animations consisting of 256 Maximum Intensity Projections on the sagittal plane, achieved by performing rotations on the axial plane. The first one corresponds to applying a MIP on the sagittal plane to both the image and the segmentations (see basic animation). With respect to the second, on the image the MIP is applied on the sagittal plane, but on the segmentations it is not, but a realistic perspective is adopted, so that the segmentations become solid, causing some of them to cover each other. This is achieved by performing axial rotation on the three-dimensional image and locating which segmentation is in the foreground (see realistic animation).

4 Activity 02

To solve the problem posed in this activity, we first studied the headers. After that, we worked with the patient image and the reference image to make their dimensions equal, so that we could apply the coregistration. Finally, with the transformation proposed by the coregistration method applied, we applied the inverse transformation on the atlas to locate the thalamus in the original image of the patient. Each of these steps will be discussed below.

4.1 Headers of interest

With respect to the headers of interest for the resolution of this problem, we find:

- **Image Orientation Patient**: we have detected that the reference and atlas have orientation [-1 0 0 0 0 -1 0], while the patient image has the usual orientation, [1 0 0 0 0 0 0 1 0]. To work with the images, we have chosen to manipulate the reference and atlas to have the same orientation as the patient image.
- **Slice Thickness** and **Pixel Spacing**: The dimensions of the voxels in the patient image were [2.0, 0.5078, 0.5078], while the reference and atlas were [1, 1, 1]. To perform coregistration, we first resized the patient image so that each voxel measured $1mm^3$.
- **Image Position Patient**: As in activity 1, we have made use of this header to sort the slices and obtain the three-dimensional image of the patient.

In addition to the above, we have located from the atlas information that the indices of interest for thalamus segmentation are those between 121 and 150.

4.2 Matching Dimensions

In order to match the dimensions of the reference and the image of the patient, we have carried out different actions.

1. Initially, the patient image had dimensions (212, 512, 512), and after rescaling to achieve $1mm^3$ per voxel, we have obtained dimensions of (424, 260, 260) (see figure 4). These values do not match the dimensions of the reference image.
2. Also the reference dimensions do not match the atlas dimensions ((181, 217, 181)), which we have solved by padding each of the dimensions equally at start and end (12 per slice, 6 per side). An animation showing thalamus on the reference space can be found here.
3. To match the dimensions of the reference and the patient, we have padded the reference (and the atlas). In the axial axis, we have added all the padding at the beginning, while in the others we have distributed it equally at the beginning and end (see figure 5), to achieve the greatest possible similarity to the patient's image. All the above modifications have also been made on the atlas, in order to be able to carry out any necessary merging later on.

It should be noted that, it can be seen that the patient's head is larger than that of the reference image, which makes harder to achieve good coregistration results.

4.3 3D Rigid Coregistration

To carry out the coregistration, we have evaluated the use of translations followed by rotations with Euler angles or axial rotations with quaternions. After a series of tests, we have detected that the implementation using Euler angles is more time efficient, so we have discarded axial rotations with quaternions. Finally, we have implemented a coregistration method based on difference between images, making use of the MSE as the function to minimize. Given the type of function to minimize, we have at our disposal different possible methods, such as *Powell*, *BFGS* or *L-BFGS-B*. After carrying out different tests, none of them yielded good results, and we finally selected the one we considered to have performed best: *Powell*.

Before starting the minimization process, we have made a series of modifications to the images:

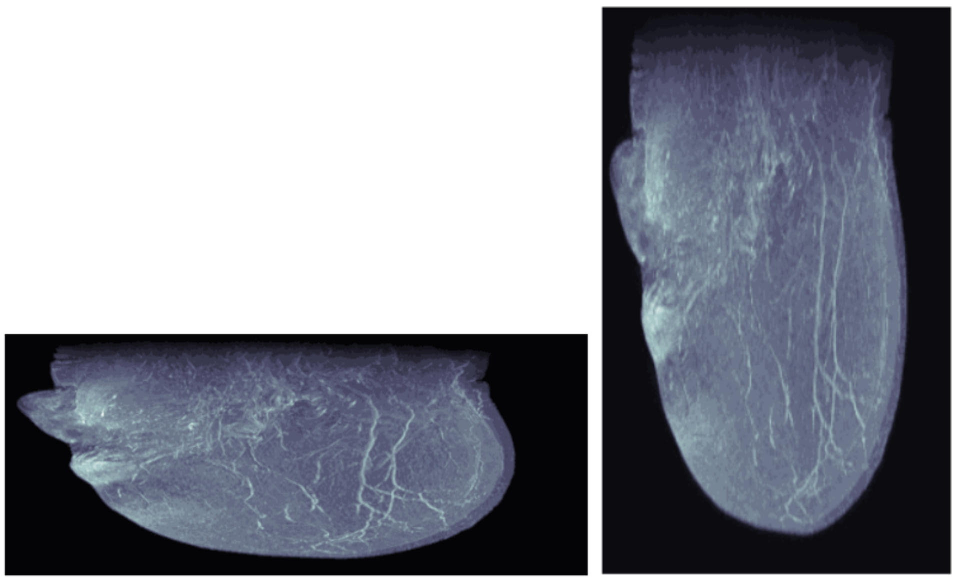

(a) Patient image before rescaling (b) Patient image after rescaling

Figure 4: Patient image before and after rescaling

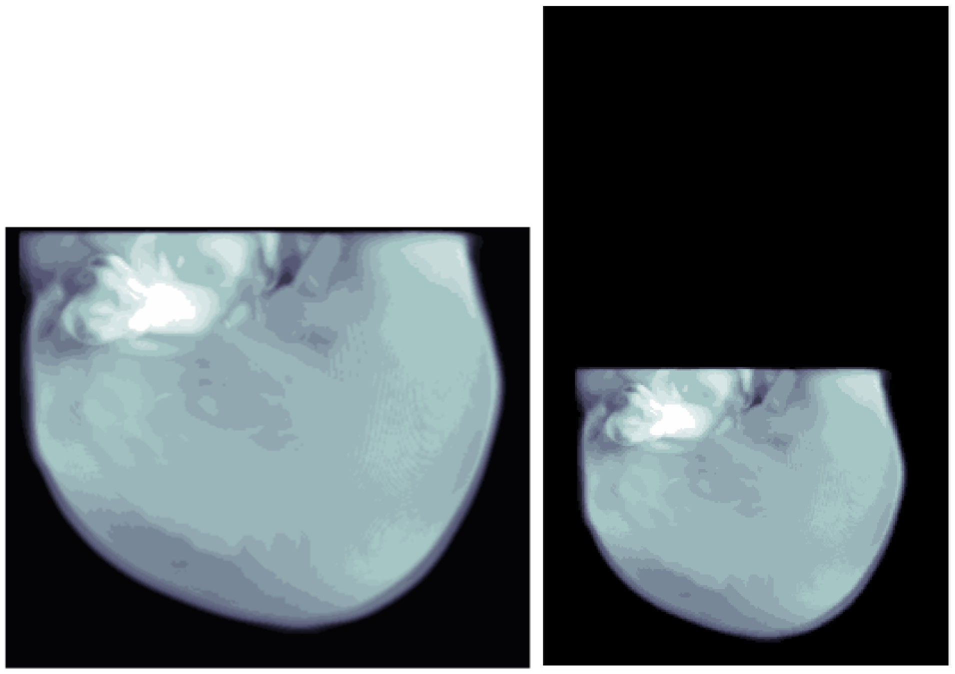
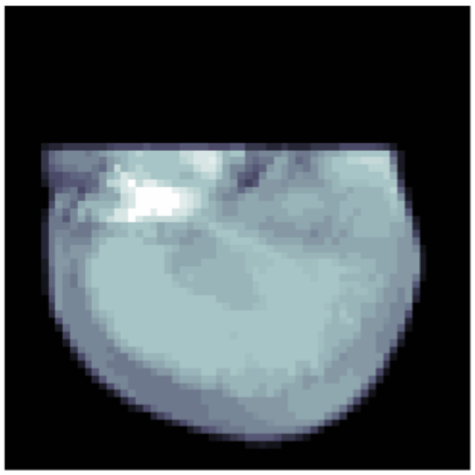

(a) Reference image before padding (b) Reference image after padding

Figure 5: Reference image before and after padding

- Crop 150 slices from both images. As can be seen in the figures 4 and 5, the upper areas do not favor the coregistration process, since the reference does not contain information about that area. By cropping, we achieve a closer resemblance between the two images.

- Reduce the dimension of both images to (64, 64, 64), making the minimization process much faster.

The result of these two modifications can be seen in Figure 6. In this figure, it can be clearly seen how perfect coregistration is impossible by performing only rigid transformations.


(a) Reference image after modifications and before minimization

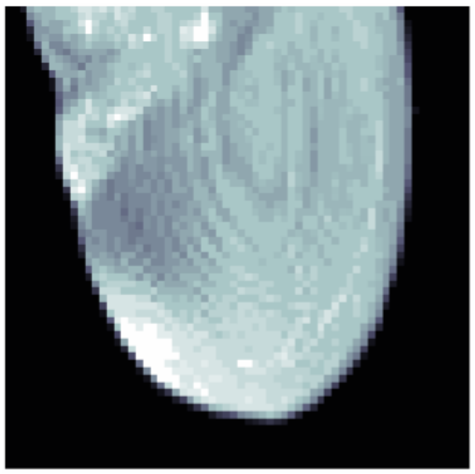

(b) Patient image after modifications and before minimization

Figure 6: Reference image before and after padding

The code implemented to carry out the minimization process is as follows:

```python
def coregister_images(ref_image, input_image, initial_parameters=False):
""" Coregister two sets of landmarks using a rigid transformation. """
if not initial_parameters:
    initial_parameters = (
        5, 0, 0, # t1, t2, t3 (translation)
        10, 0, 0 # angle_0, angle_1, angle_2 (rotations)
    )

def function_to_minimize(parameters):
    """ Transform input landmarks, then compare with reference landmarks."""
    print(parameters)
    moved_image = translation_then_axialrotation(input_image, parameters=
                                                  parameters)

    return mse_3d_array(ref_image, moved_image)

result = minimize(
    function_to_minimize,
    x0=initial_parameters,
    method="Powell"
    )
return result

results = coregister_images(reference_img_tomin, input_img_tomin,
                                              initial_parameters=(5, 0, 0, 0, 0, 0))
```

It should be noted that, since we have corrected the orientation of the images, the required rotation is minimal and it is not necessary to initialize any rotation angle. Regarding the translation, we initialize it with vector (5, 0, 0) since the patient's image is displaced in that direction.

After carrying out the minimization process, we obtain as a result the translation vector (18.99999773, -0.63118217, -0.49264742) followed by the Euler rotation of angles (-3.45796167, 0.94065069, 1.02547539).

It should be noted that the above applies to images of dimensions (64, 64, 64)... which in the case of angles remains invariant for higher dimension, unlike translation. After scaling the translation vector, the resulting transformation to transform the patient space to the reference space consists of a translation of vector (81.34374028, -2.56417755, -2.00138013) followed by an Euler rotation of angles (-3.45796167, 0.94065069, 1.02547539). The result of the coregistration of the patient image over the reference image can be seen in Figure 7.

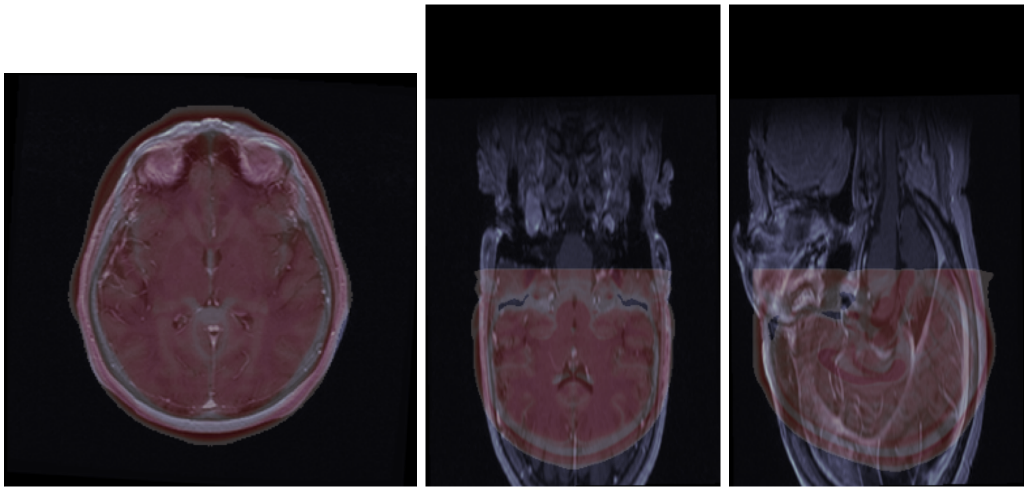

(a) Axial plane (b) Coronal plane (c) Sagittal plane

Figure 7: Patient image in the reference space

With respect to the inverse transformation, in order to transform the reference space to the patient space and to be able to place the thalamus on it, we must first perform an Euler rotation with the inverse angles, i.e. ((3.45796167, -0.94065069, -1.02547539), and a translation with the inverse vector, i.e. (-81.34374028, 2.56417755, 2.00138013). The representation of the thalamus on the patient space can be seen in an animation of the axial plane, on a rotation of the 3D segmentation mask and on a rotation of the 3D image.