# Medical Imaging Final project

Alejandro Cedillo Gámez

2025-05-04

## Summarize

1. **Setup**

* A Python virtual environment was created, and required packages installed: **pydicom**, **highdicom**, **numpy**, **matplotlib**, and **imageio**.
* **3D Slicer** was installed and used to visually inspect the RadCTTACEomics dataset, ensuring familiarity with the CT images and segmentation masks.

1. **DICOM Loading**

* **CT Series**: Loaded all axial CT slices with pydicom, sorted by ImagePositionPatient, and stacked into a 3D NumPy volume.
* **Segmentation (DICOM‑SEG)**: Loaded the segmentation file with pydicom, extracted label frames, and parsed PerFrameFunctionalGroupsSequence to obtain each frame’s spatial position.
* **Header Analysis**: Confirmed consistent AcquisitionNumber across slices and inspected key tags (InstanceNumber, PixelSpacing, SliceThickness, and segmentation labels).

1. **Mask Alignment and Visualization**

* Mapped each segmentation frame to its corresponding CT slice using Z‑position matching.
* Constructed a boolean 3D mask volume aligned to the CT data.

1. **Rotating Maximum Intensity Projection (MIP)**

* Computed sagittal MIPs by collapsing the 3D volume along the anterior–posterior axis.
* Applied a semi‑transparent red colormap to visualize the tumor region over the grayscale CT projection.
* Generated a sequence of rotated MIP frames using the class’s rotate\_on\_axial\_plane and MIP\_sagittal\_plane functions.
* Assembled frames into a high‑quality GIF animation (rotating\_MIP.gif) with correct aspect ratio.

## Pending Work

1. 3D Image Segmentation
   1. Consider the Tumor mask associated to the reference image, and extract its bounding box and centroid.
   2. Create a semi-automatic tumor segmentation algorithm that only uses the CT image, and either the bounding box or the centroid of the tumor.
   3. Visualize both the provided Tumor mask and the segmented Tumor mask on the image. Assess the correctness of the algorithm, numerically and visually.
2. 3D Rigid Coregistration
   1. Coregister the input to the reference image, implementing all steps of the image coregistration yourself (i.e. without libraries such as PyElastix).
   2. Visualize the Liver region on the input image space. Assess the correctness of the algorithm, numerically and visually.

## GitHub Repository

The code and results for this project are available at:  
[Medical Imaging Final Project](https://github.com/alexcega/Medical-Imaging-Final-Project/)

## Questions and comments

**Data:**

I’m a bit confused by the two CT folders (10\_AP\_Ax5.00mm vs. 30\_EQP\_Ax5.00mm). What’s the difference between them, and why do we need both?

I try to do the code twice with them but I still don’t see much difference

**Activity’s files:**  
Is it possible to get a small summarize of what each activity.py files does? I lost a bit of time trying to understand them, Or I just would like to know if all of them are relevant for the project.

**DICOM metadata terminology**:

Many tags (e.g. UID, SOPClassUID, TM, DA, SH) are unfamiliar. I would like to understand better what im working with, but again im not sure if that’s distracting me from the points of the project, there is also a loooot of attributes in the metadata.

With the point: “Both images are loaded with PyDicom, and their corresponding headers have been studied.” I would like to confirm the amount of headers that could be expected.

**Step 2 Segmentation:**

I believe its already highlighted the tumor in the step 1 for the gif file, what is the difference expected for the step 2?

**Step 2 and 3 Evaluation metrics:**

It is mention to “Assess the correctness of the algorithm, numerically and visually.” Does this mean to do last mention in class with accuracy, precision, sensitivity and specificity?

**Step 3**

It is mentioned to not use the library PyElastix, can I still check it out for inspiration, or do you recommend coding everything from scratch? If so any tips on where to get started would be great.