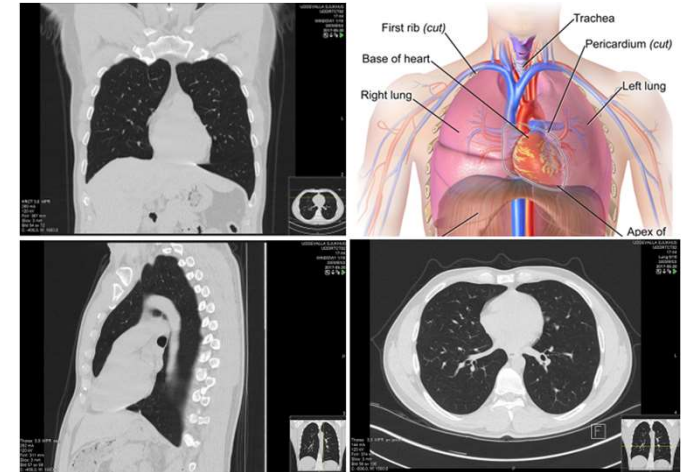


CSC621: Team Wind

Jarett (team lead), Allison, Mohammad

Introduction

- What are we working with?
 - 10+GB data set of Positive(?) COVID-19 CT Scans amongst others
 - Patients lying down in relatively similar positions
 - Slight differences due to body type and left/right lean
- So what?
 - Just get started



Logistical Set-up

- Shared progress through Github repository
- Segmentation & Registration tasks split among team members
- Tools
 - Use SITK documentation (cpp documentation was weaker than python)
 - Install Anaconda/SITK dependencies
 - Install FIJI/ITK SNAP for further analysis
 - Get comfortable with Jupyter notebooks
- **Move fast, break things and start getting things going**

Registration

- *What is it?*

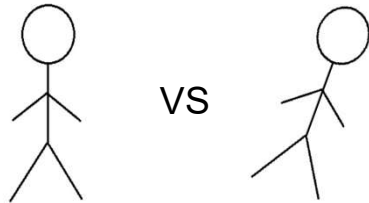
- Registration serves the purpose of bringing two or more images into spatial correspondence by determining a geometrical transformation that aligns points in one view of an object with corresponding points in another view of that object or another object
- Intensity-based rigid registration with mutual information maximization
 - Calculates transformation between images using pixel/voxel values
 - No need for landmarks or surfaces
 - Allows or multi-modal registration

- *What's the goal?*

- To make the registration beneficial in medical diagnosis or treatment the mapping that it produces must be applied in some clinically meaningful way by a system which will typically include registration as a subsystem. The larger system may combine the two registered images by producing a reoriented version of one view that can be “fused” with the other, with resampling.

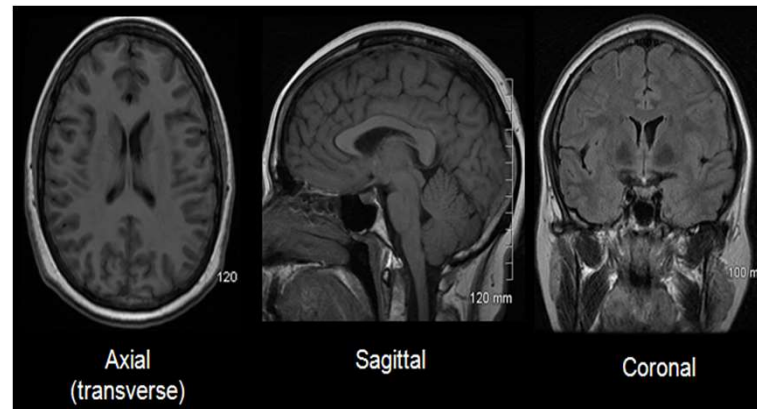
Why Registration is Necessary

- Differences in lying positions



- Added variability of weight, height, gender, BMI, torso length etc. all can affect organ positioning

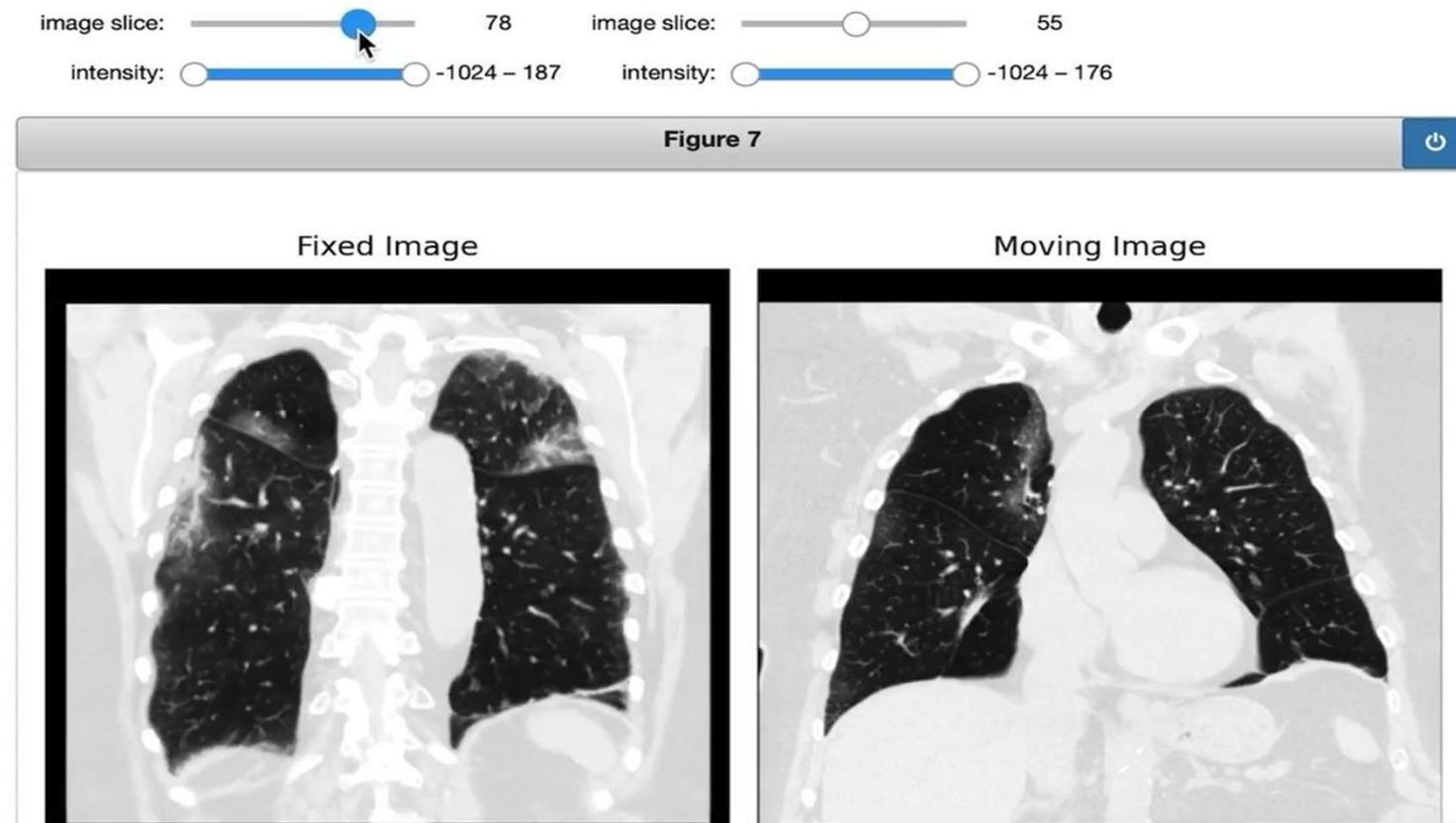
- Fuse scans from different modalities, or different patients, into one registered image (MRI-CT)



Main Components of Registration

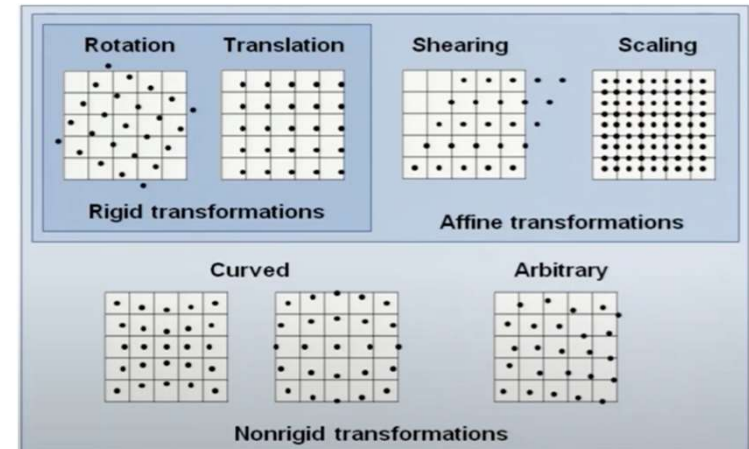
- **Target:** the object that is assumed to be static (fixed image)
- **Reference:** the object that will be transformed in order to be superimposed to the *Target* (moving image)
- **Transform:** the mapping that will convert one point from the *Reference* object space to the *Target* object space.
- **Metric:** a measure that indicates how well the *Target* object matches the *Reference* object after transformation
- **Interpolator:** the particular technique used for interpolating values when objects are resampled through the *Transform*
- **Optimizer:** the method used to find the *Transform* parameters that optimize the *Metric*

Displaying Images on SimpleITK



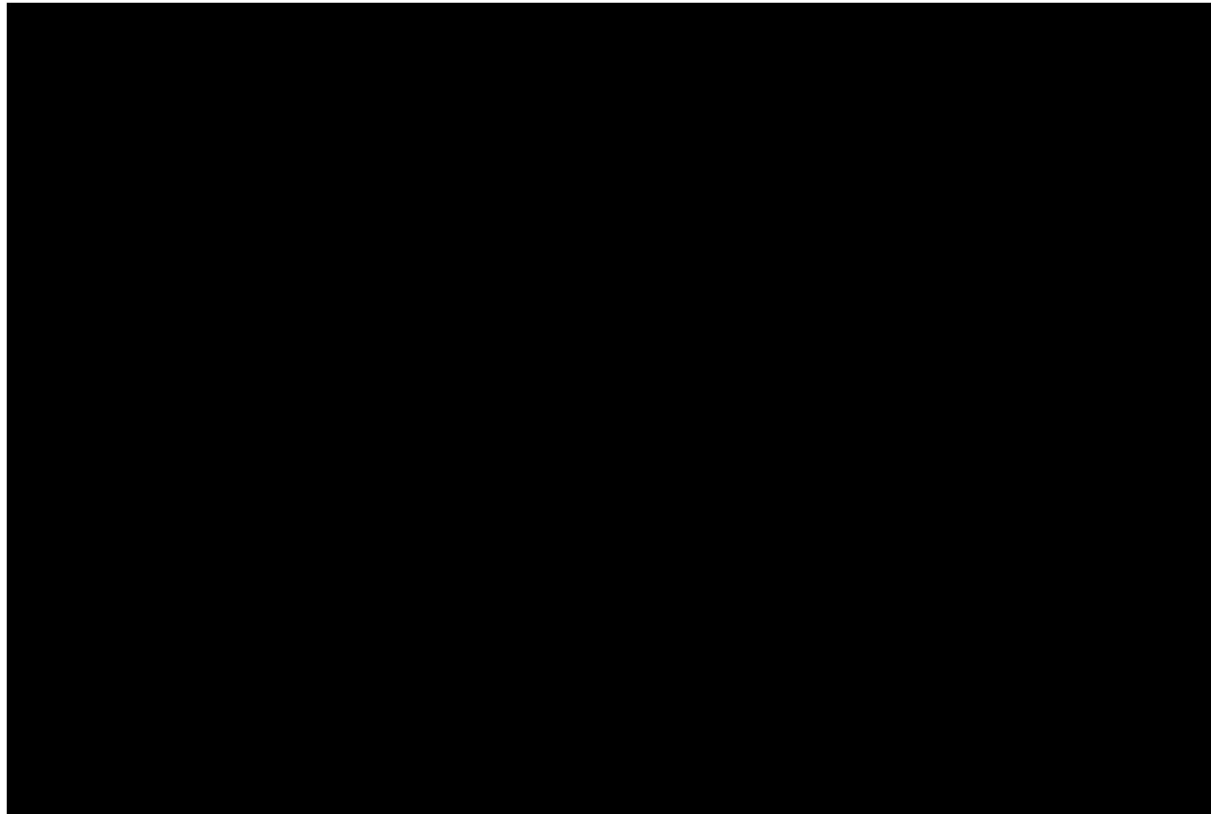
Transformation

- Used Rigid Geometric Linear Transformations since deformation is relatively simp
- The transform took place in the initialization phase of registration using the CenteredTransformInitializer(which is used for intensity-based reg.) helper class and the Euler3DTransform
 - The CenteredTransformInitializer initializes the translation and Euler3DTransform is a type of geometrical rigid transformation



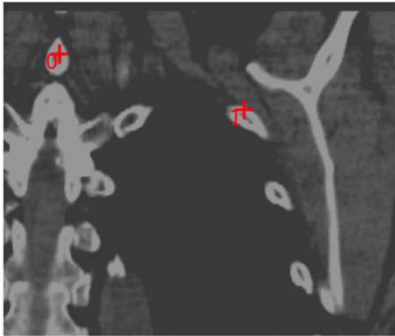
Behavior	Number of Parameters	Parameter Ordering	Restrictions
Represents a rigid rotation in 3D space. That is, a rotation followed by a 3D translation. The rotation is specified by three angles representing rotations to be applied around the X, Y and Z axis one after another. The translation part is represented by a Vector. Users can also specify the coordinates of the center of rotation.	6	The first three parameters are the rotation angles around X, Y and Z axis, and the last three parameters are the translations along each dimension.	Only defined for three-dimensional input and output spaces.

Initializing Registration

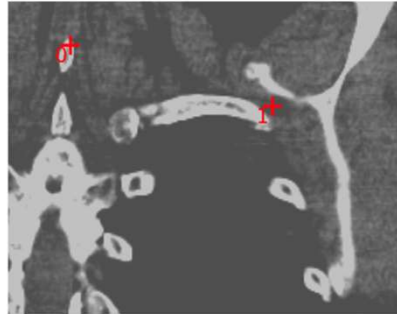


More Misalignment

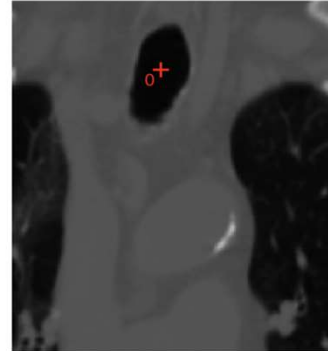
fixed image - localized 2 points



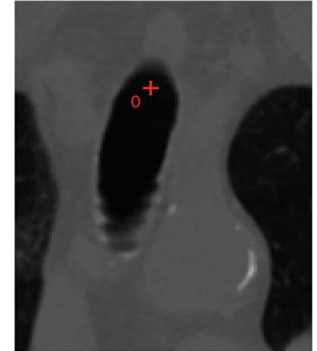
moving image - localized 2 points



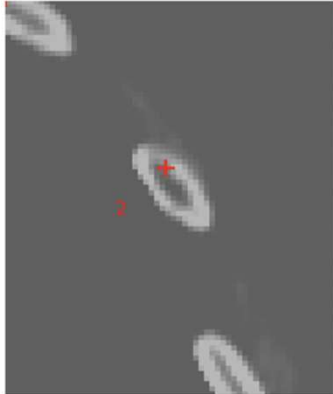
fixed image - localized 1 points



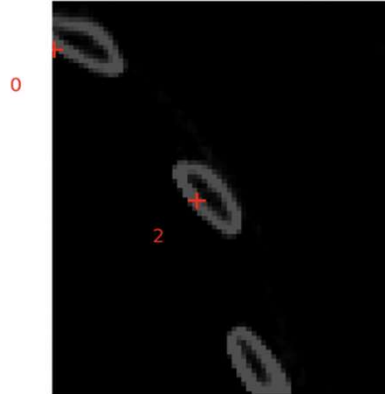
moving image - localized 1 points



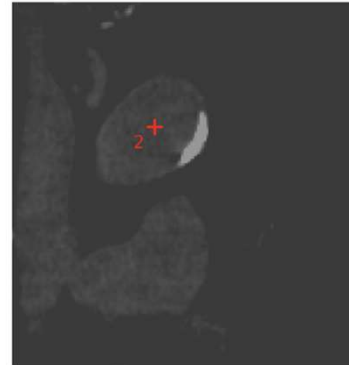
fixed image - localized 3 points



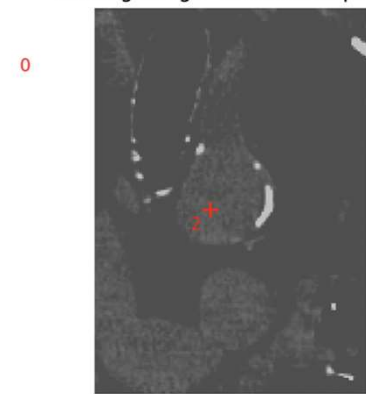
moving image - localized 3 points



fixed image - localized 3 points



moving image - localized 3 points

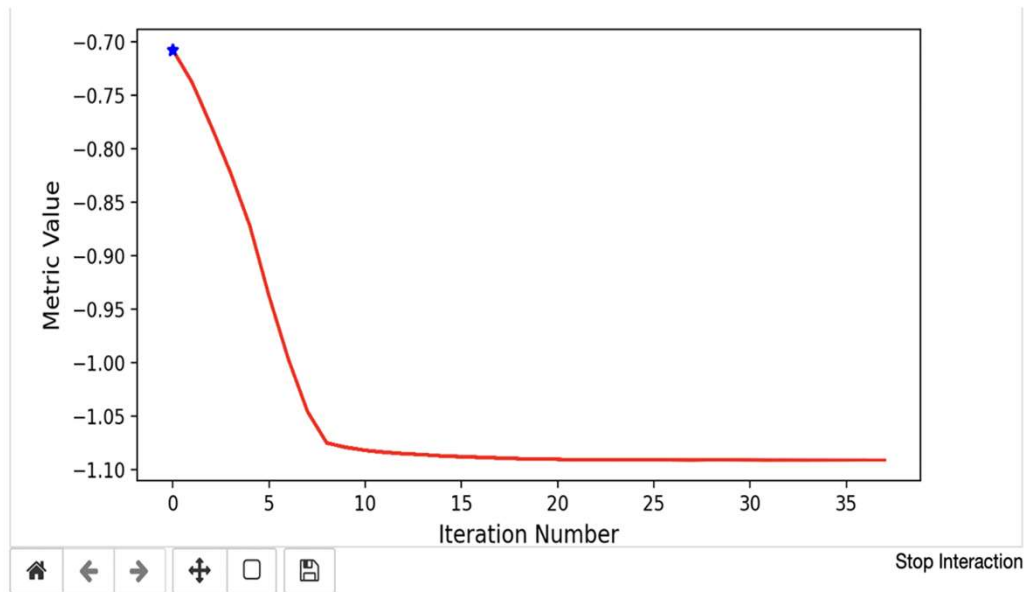


Similarity Metric: Mattes Mutual Information

- *What does it measure?*

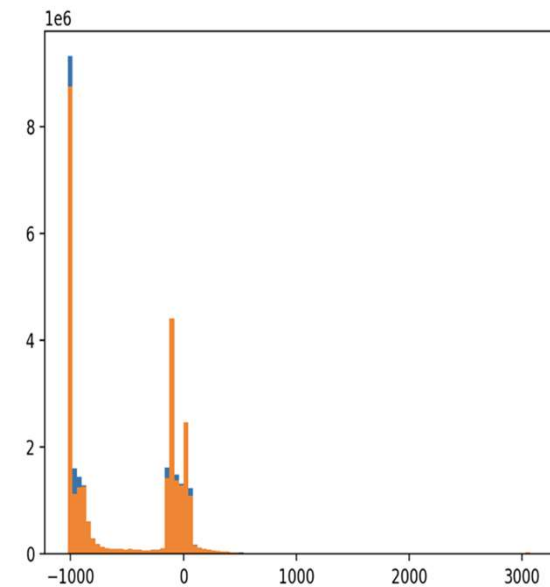
- Mutual information (MI) between two images measures the similarity of the fixed/moving images as a function of image intensity. Two images are put into register by maximizing the MI. The mutual information between two images is a function of the joint histogram between the template and target images. The MI computed from the joint histogram is based on the histogram values and not the pixel intensities.
- MI is intensity based
- MI can be used for both mono-modality and multi-modality image registration.
- The metric requires two parameters to be selected: the number of bins used to compute the entropy and the number of spatial samples used to compute the density estimates. In typical application 50 histogram bins are sufficient

Similarity Metric: Mattes Mutual Information



Final metric value: -1.0909739865851753

Optimizer's stopping condition, GradientDescentOptimizerV4Template: Convergence checker passed at iteration 38.



```

registration_method = sitk.ImageRegistrationMethod()

# Similarity metric settings.
registration_method.SetMetricAsMattesMutualInformation(numberOfHistogramBins=50)
registration_method.SetMetricSamplingStrategy(registration_method.RANDOM)
registration_method.SetMetricSamplingPercentage(0.01)

#Interpolator
registration_method.SetInterpolator(sitk.sitkLinear)

# Optimizer settings.
registration_method.SetOptimizerAsGradientDescent(learningRate=1.0, numberOfIterations=100, convergenceMinimumValue
#registration_method.SetOptimizerScalesFromPhysicalShift()
#uncomment line above

# Don't optimize in-place, we would possibly like to run this cell multiple times.
registration_method.SetInitialTransform(initial_transform, inplace=False)

# Connect all of the observers so that we can perform plotting during registration.
registration_method.AddCommand(sitk.sitkStartEvent, rgui.start_plot)
registration_method.AddCommand(sitk.sitkEndEvent, rgui.end_plot)
registration_method.AddCommand(sitk.sitkMultiResolutionIterationEvent, rgui.update_multires_iterations)
registration_method.AddCommand(sitk.sitkIterationEvent, lambda: rgui.plot_values(registration_method))

final_transform = registration_method.Execute(fixed_image, moving_image)

# Always check the reason optimization terminated.
print('Final metric value: {}'.format(registration_method.GetMetricValue()))
print('Optimizer\'s stopping condition, {}'.format(registration_method.GetOptimizerStopConditionDescription()))

```

- **Interpolator**
 - Linear interpolator: simplest and most popular technique

Optimization: Gradient Descent Optimizer

- *What is it?*

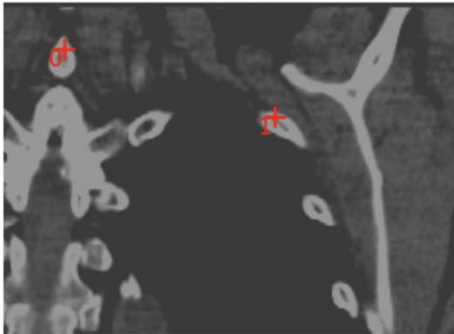
- Registration is looked at as an optimization problem alone, but the Optimization is an iterative procedure which computes the geometric and intensity transformations at which the fixed/moving images are having maximal similarity with one another
- The method used to find the *Transform* parameters that optimize the metric cost function
- the goal of the optimization is to find the set of parameters defining a transform that results in the best possible value of an image metric.

- *Gradient Descent Optimizer*

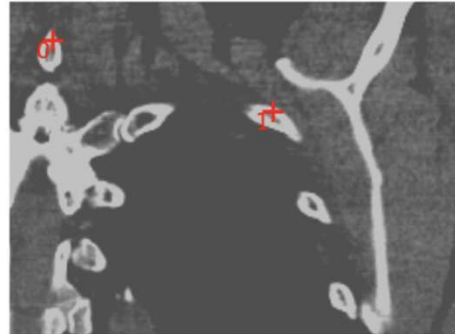
- At each iteration the Optimizer takes a step in the direction of the gradient, which we can see in the graph that plots the iterations
- The more aligned, the more minimum the optimization, the less aligned the higher

Results of Registration

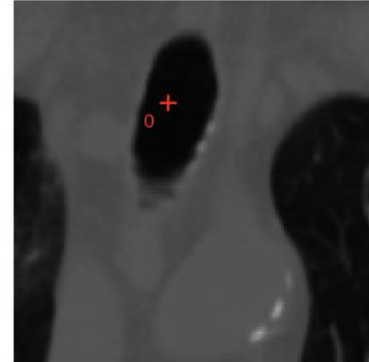
fixed image - localized 2 points



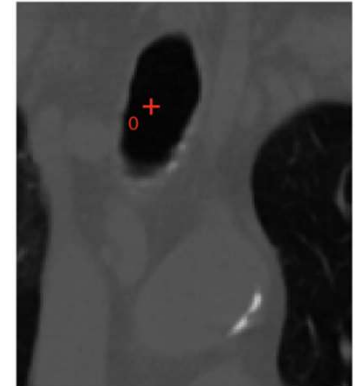
moving image - localized 2 points



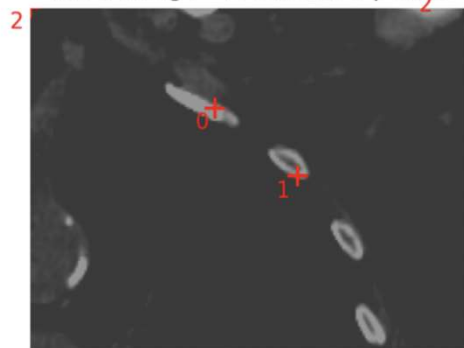
fixed image - localized 1 points



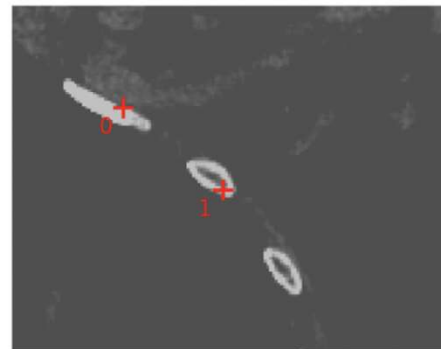
moving image - localized 1 points



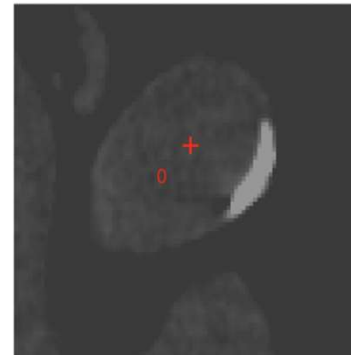
fixed image - localized 3 points



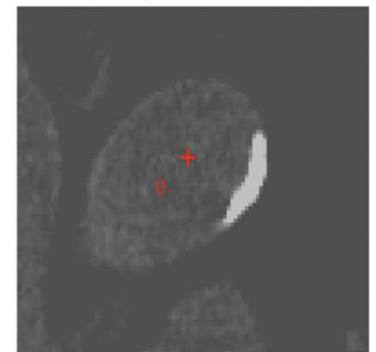
moving image - localized 3 points



fixed image - localized 1 points



moving image - localized 1 points



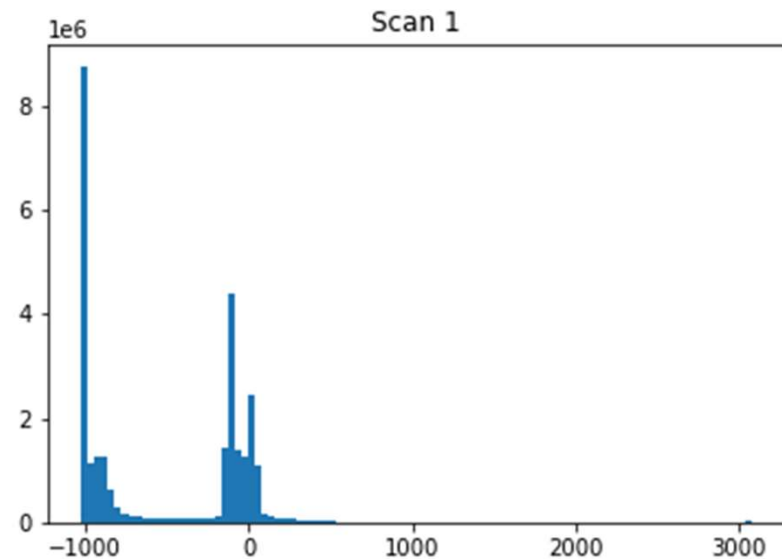
Registration Conclusion

- Challenges
 - Learning Simple ITK since we were using python and not C++, not many resources
 - Understanding if the results from the metric cost function and physical images were what was expected
 - Could have experimented with different types of optimizers, interpolators, or metric cost functions
 - Not being able to go more in depth
 - Could have found more ways to display results (gui's, image fusion)
- Results
 - Images appeared to have significantly improved alignment
 - Registration Application can be used on multi-modality problems in the future

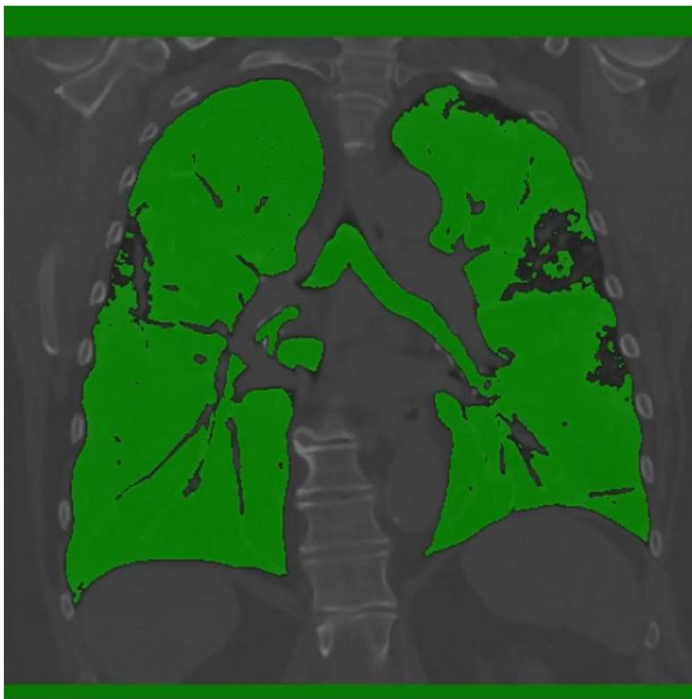
Segmentation

The simplest method to begin segmentation, from our team's experience, is by evaluating an image series' histogram and establishing a threshold visually to handle binary segmentation.

Here we can see a nice bimodal distribution, from which, we can estimate a threshold value around -500.



Segmentation

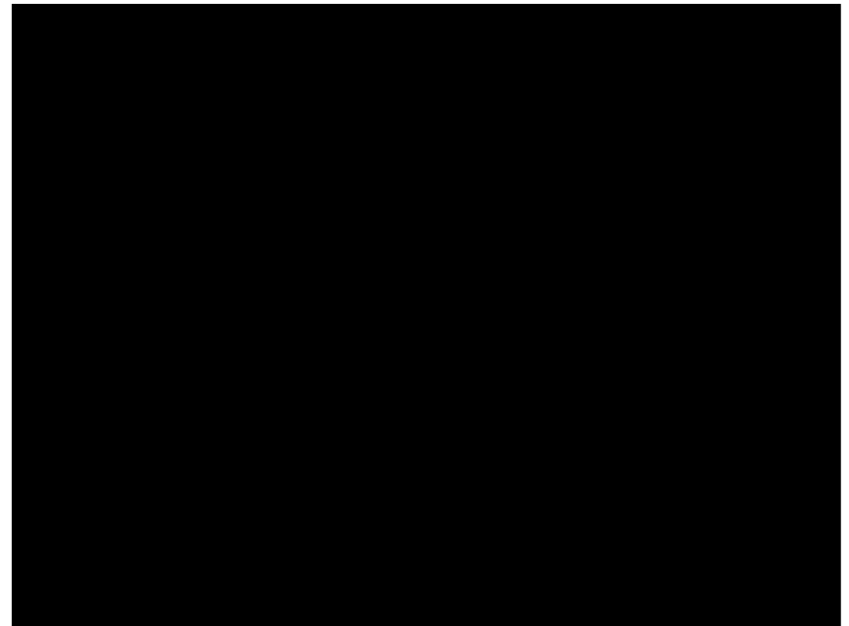


Deriving a segmentation image from that histogram threshold to overlay the original image with, we have a resulting image series that successfully highlights core areas of the lungs. But it could use some cleaning up to improve segmentation clarity and removal of non-desirable areas like the bars at top and bottom.

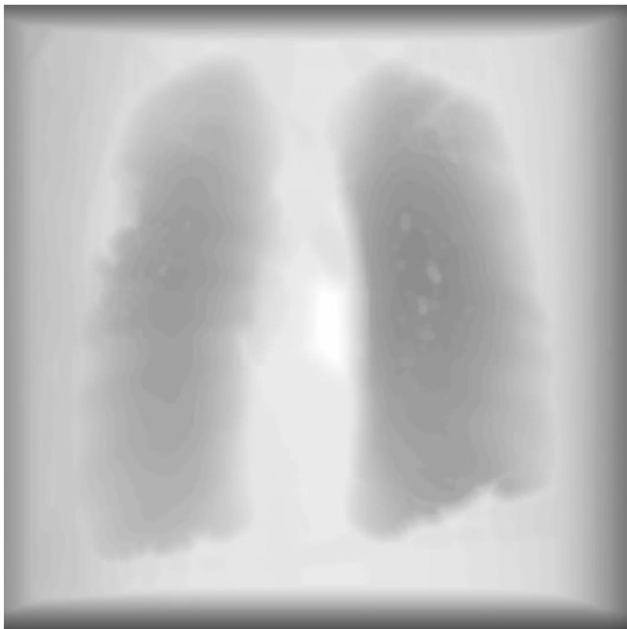
Segmentation Morphological Operations

Visual analysis of the segmentation results so far seem to indicate that the noise is causing likely connected objects to be disconnected by small margins. To rectify this, we apply two morphological operations using [10x10x10] voxel structuring elements. First opening, followed by closing.

The differences are subtle but the abscesses throughout the lung are more prominent visually now.



Preparing for Watershed Segmentation

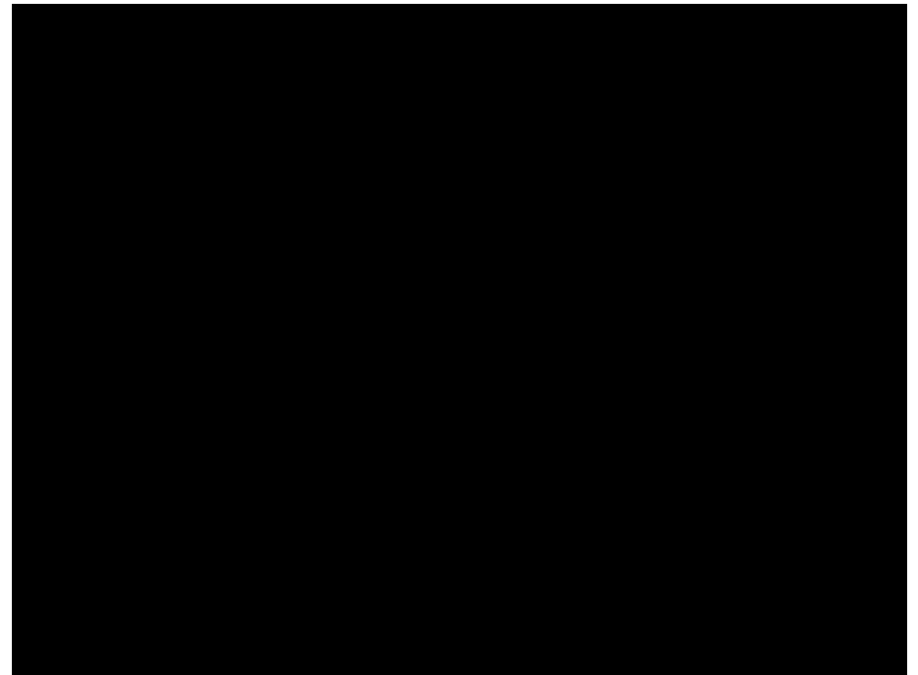


Before applying the watershed segmentation method to our improved binary threshold segmentation, we need to establish a distance map and label connected components within the segmented area.

To achieve this, we used a signed distance map so that only the internal portion of the segmentation is measured, and connected component radius of only 10 pixels with a minimum object size to consider for labeling at 15 pixels

Watershed segmentation

Using the distance map and seeds from the connected components map, we can instantiate a watershed segmentation to overlay on the original image. This provides bounded volumes around contiguous areas within the patient's lungs which we can then analyze and export the data to a pandas DataFrame for quantification



Cleaning up Watershed Segmentation

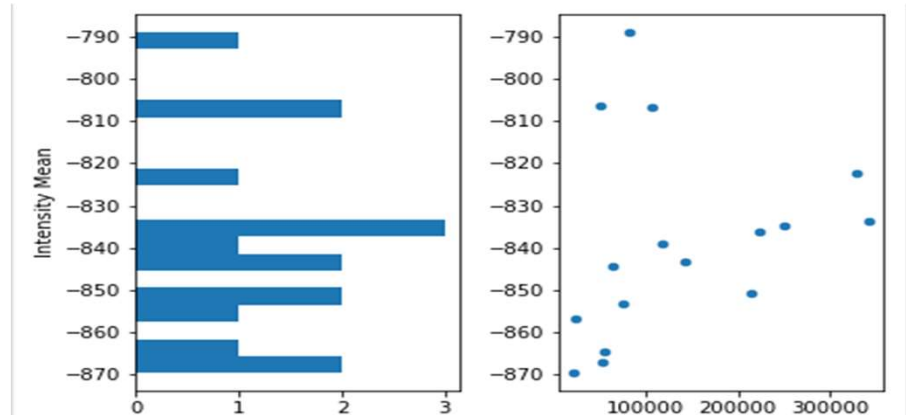
Binary Grind Peak Filter

Using the BGP filter, we can clean up all the areas that were mis-segmented by the original otsu filter. Generally this works by removing objects not connected to the boundary of the image. But if we invert the logic for the filter we can remove the borders and portions that contact the boundary, and we are left with just a clean watershed segmentation that we can now derive data and statistics from.

Statistical Quantification from Watershed

Using DataFrames from pandas, we can easily extract data from these segmentations involving a variety of metrics like volume, flatness, and intensity mean, standard deviation, and skewness. Using this data with matplotlib allows for quick visualization to be made as well.

	Volume (mm^3)	Flatness	Intensity Mean	Intensity Std Dev	Intensity Skewness
count	16.000000	16.000000	16.000000	16.000000	16.000000
mean	134406.516128	1.410422	-838.615907	91.061875	1.920755
std	105184.378412	0.259450	23.103880	8.246724	0.421457
min	21863.139305	1.005619	-869.666259	79.429796	1.017061
25%	55324.984816	1.239227	-853.937835	86.099220	1.661072
50%	94345.289869	1.338688	-841.161246	90.798929	1.841672
75%	217282.955686	1.585001	-830.976088	97.382732	2.241425
max	342172.051566	1.960234	-788.890186	110.317001	2.715024



Pros and Cons of Segmentation

Pros:

- Provides a means to quickly assess larger quantities of data and images
- Basic segmentations can be obtained with minimal manual input, which can then be used to generate empirical data to establish possible trends across datasets
- Provides a foundation for more advanced processing and input into neural networks and machine learning applications

Cons:

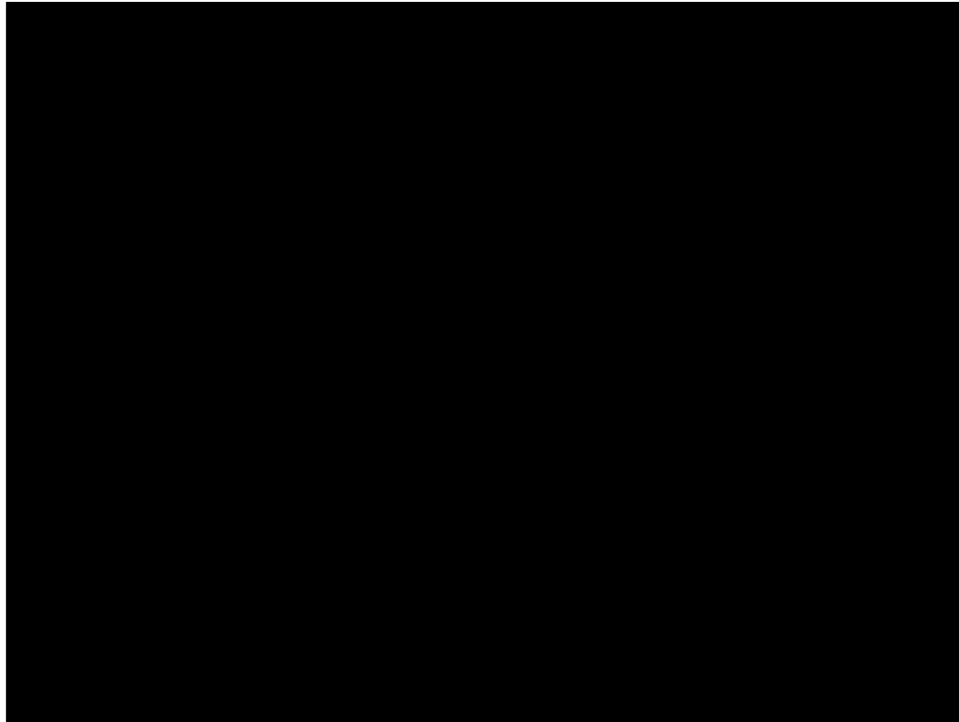
- May require pre-processing beforehand due to blurry or low contrast imaging
- Many simple methods available can over or under-segment if parameters are not set properly. Leading to a relatively useless segmentation
- Obtaining more accurate segmentation results may require manual seeding from trained personnel, and becomes very time consuming. The quality of segmentations prior to advanced processing can drastically affect the end result

Quantification (registration)

- Intensity based metrics
- Dependent on proper registration of images beforehand

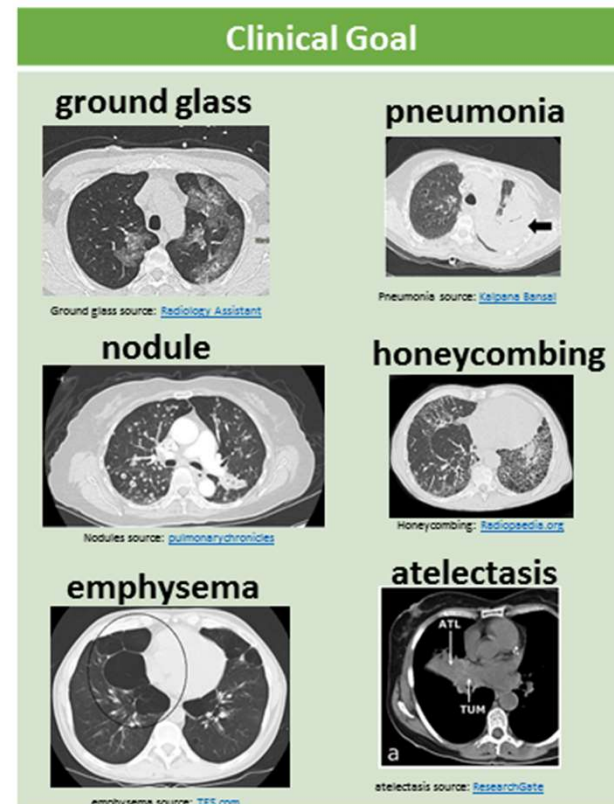


Fiji/SNAP ITK plugins

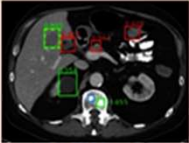
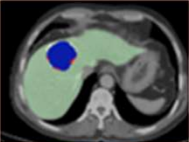


“We’re not doctors”

- Radiologists have **trained** themselves via **experience** and intentioned **practice** to identify certain diseases from scans
 - See right image for anatomical
- ...sounds like AI/Deep/Machine Learning?
 - Could we train a ML model to identify aspects of a CT scan indicating COVID-19?
 - Could we train a model to at least differentiate between other diseases vs COVID-19 (classifier)?

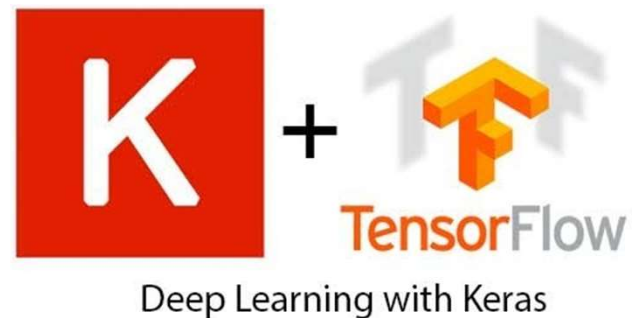


- We found various CT scan ML models before COVID based around various lung diseases (cancer, smoking related, pneumothorax etc.)
- ML Models built to do different things (see right)
- Our initial reaction
 - Build a classifier: given a scan, can it identify COVID19?
 - Accredited institution provided positive and negative data set
 - MIDRC-RICORD-1A (positive) vs MIDRC-RICORD-1B (negative)

Task	
binary classification <i>e.g. emphysema or not</i>	<div>0 or 1</div>
multi-class classification <i>for mutually exclusive categories</i>	$\begin{bmatrix} \text{IPF} & 0 \\ \text{iNSIP} & 0 \\ \text{CHP} & 1 \end{bmatrix}$
multi-label classification <i>for non-mutually-exclusive categories</i>	$\begin{bmatrix} \text{atelectasis} & 1 \\ \text{cardiomegaly} & 0 \\ \text{mass} & 1 \end{bmatrix}$
object detection <i>predict bounding boxes around findings of interest</i>	 Jen et al. 2018 DeepLusion
segmentation <i>label every pixel, i.e. trace the outlines of all findings of interest</i>	 Jiang et al. 2019

Undergrad approach

- Clean dataset and only use CT scans, making sure images are in similar orientations
- Train a ML model on our positive COVID data using one of many APIs (tensorflow/keras) using AWS/GCP computing.
- COVID19 diagnosis solved - we have a classifier that can figure out if a CT scan is positive for COVID19
- Reality is quite different



Research shows...

Mei, Xueyan, et al. "Artificial intelligence–enabled rapid diagnosis of patients with COVID-19." *Nature Medicine* (2020): 1-5.

*"While chest CT is **not as accurate as RT–PCR** in detecting the virus, it may be a useful tool for triage in the period before definitive results are obtained."*

Rachel Lea Ballantyne Draelos' analysis on Glassbox Medicine on said paper amongst many:

- PCR is the "gold standard" for diagnosis currently
- "In the context of COVID-19, machine learning on chest CTs has the potential to be helpful for: (a) **triage** in resource-constrained environments (e.g. diagnosis model), (b) **evaluating complications**, or (c) prediction of **worsening vs. improvement**."

Future Work thoughts

- Realism: lack of time/domain knowledge and a team member leaving
- Research/Inspiration:
 - Search thru white papers via Google Scholar on topic
 - SimpleITK/Fiji/SnapITK docs
 - Credit:
 - **Dr. Robert Haase's** YT channel + twitter [feed](#)
 - MD/ CS PhD candidate **Rachel Lea Ballantyne Draelos'** [Glassbox Medicine](#) blog
 - **Dr. Kazunori Okada's** foundational course on biomedical imaging without which we couldn't have understood a lot of the materials we found online above



Robert Haase

@haesleinhuepf



racheldraelos

@racheldraelos