CT Scan / Radon Project
Timothy Shepard
1519265

Objective

Investigate how to simulate CT Scanners and how to back-project 1D Radon transforms.

Project Summary

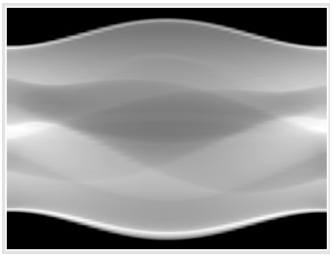
I wrote the code to do a Radon transform in python without using the built-in SciKit radon method. This code took in an angle as a parameter, rotated the image by that angle, and summed along columns of the rotated image. Because I wanted to simulate individual CT Scans for each angle, I made a button that only made one "CT Scan" at a time, incrementally. I also displayed the 1D Radon Transform for each angle stepped through in addition to the 2D Radon Transform Sinogram.

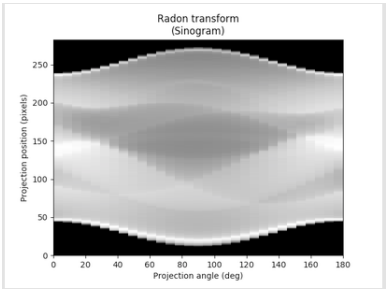
Secondly, I used the 2D Radon Transform Sinogram to back-project each 1D Radon Transform angle to try to re-construct the original image. The back-projection method took in the angles array from the Radon Transform and the filter desired. For each angle I added the "smeared" 1D Radon values to a back-projection matrix cumulatively.

To compare my implementations, I created a window in the GUI which displays SciKit Radon and SciKit inverse Radon for the same angles.

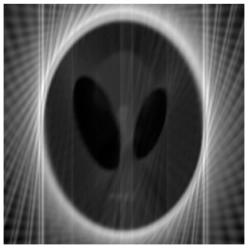
Results/Comparison

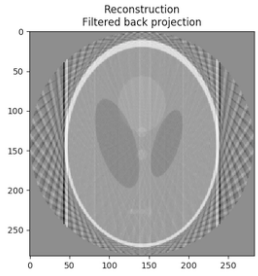
The Radon transform I made (left) looks very similar to the python SciKit radon transform (right). These Radon transforms are from the Shepp-Logan phantom with 30 angles from 0° to 180°.





To make the back-projection reconstruction, I discovered that one should filter the radon values for each angle before “smearing” across the reconstructed image. If not, the back-projection reconstruction will look blurred. Even with a simple high-pass ramp filter, the back-projection I made (left) looks worse than the python Scikit inverse radon (right) for the same array of angles. This had 30 angles between 0° and 180°. Also for a large number of angles, more than 30 or 40, the code I wrote for back-projections is very slow. In addition, for large number of angles (for example: 90 angles) the reconstructed image becomes blurred.





Conclusion

My Radon transform successfully imitated python SciKit’s built-in radon function. However, it is much slower.

In image reconstruction, I think the filter for each 1D Radon Transform back-projected is very importan. The Ram-Lak ramp filter I used improved the image reconstruction, but the python SciKit iradon function made a clearer reconstruction for small shapes in the Shepp-Logan phantom. I think that very high values on the edges of the Shepp-Logan phantom “skull” became higher in intensity with more back-projections and diminished the relative intensity of the objects on the “inside” of the Shepp-Logan phantom.

I consider this project a success, because I learned how the Radon transform works and can simulate CT Scans. Also, learned how back-projections can reconstruct an image from 1D Radon transforms. Although my back-projection method failed to imitate the python SciKit iradon function, I learned the importance of filtering before back-projecting each angle.

References

Hayden, Brendan F. *Slice Reconstruction*, 10 Feb. 2005, homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0405/HAYDEN/Slice_Reconstruction.html.

Nguyen, Minh Anh. "Biomedical Signal and Image Processing Projects Using Matlab and Labview Tools." *Matlab Code to Perform Tomographic Reconstruction of a 2-D Image Based on 1-D Projections*, 1 Jan. 1970, biomedicalsignalandimage.blogspot.com/2016/02/matlab-code-to-perform-tomographic.html.