Project Final Report

Image Encryption and Decryption with Radon Transform

Submitted by: Li Yan Chak; BEN AYED, Ahmed; LEE Wing Hang;
Class: Math 4336

Course Instructor: Professor Shingyu Leung

Department of Mathematics
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon Hong Kong

1. Introduction

The main use for radon transform are in the medical sector and other tomography applications e.g. MRI. This makes the data it contains highly personal and at risk of cyber-attacks. Our project will be mainly composed of two parts. First, we will develop an encryption algorithm to increase the image security by a unique key.

We designed an algorithm to achieve the encryption, the decryption, the filter and the radon transform. Second, in this report we will discuss the effect of changing different paremeters in our algorithm on the reconstructed image ex: the sampling rate.

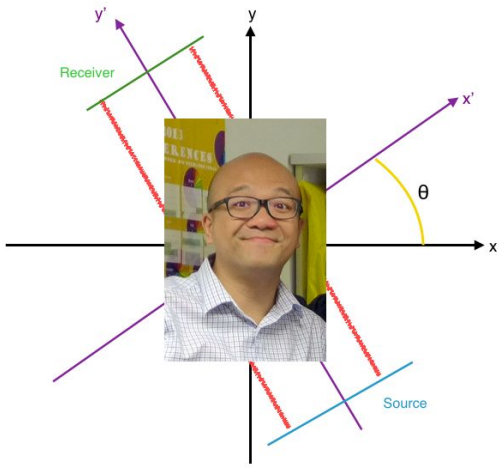
We hope this would help protect patients privacy.

2. Radon Transform

Radon transform is used to reconstruct images from their cross-sectional projection.
The radon transform of an image f at an angle θ is given by this formula:

$$R(f)(\theta, s) = \int \int f(x, y)\delta(s - xcos\theta - ysin\theta)dxdy$$

This can be illustrated as follow[1,2]:



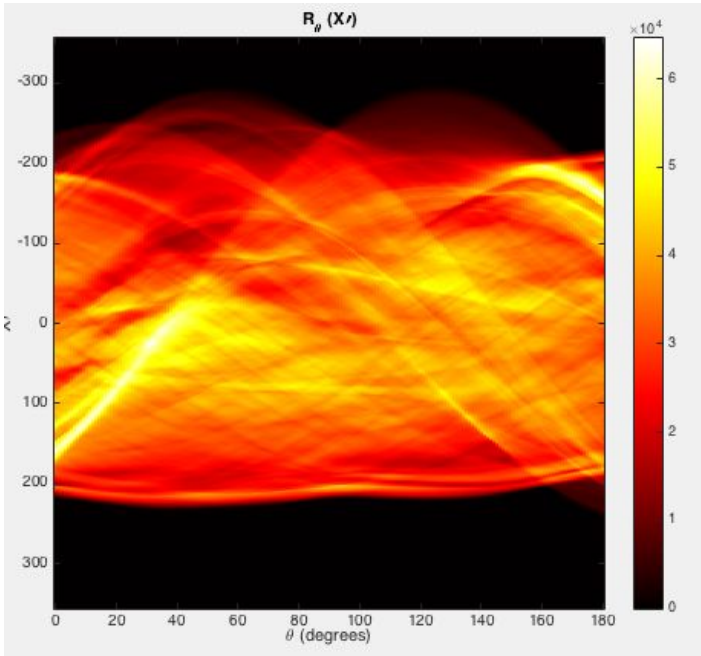
A source is rotated around an object and takes discrete projections of it. The sampling rate is controlled by the theta parameter. One, obvious, issue with the radon transform is the redundant information present in the different samples. This will result in a blurry image. MATLAB provide us with different band-pass/ high-pass filters that we used to address this issue.

MATLAB Implementation of Radon Transform and its inverse:

There has been a function 'radon' to perform radon transform for input image in R^2 and vector of sampling angle, e.g.:

```
f = imread(input);
theta = 0:1:180;
r = radon(f,theta);
```

Which will return a m-by-n sinogram, where m = length of diagonal of image, n the number of samples.



Similarly, MATLAB also provide a function "iradon" that can be used to recover the image, for more detail [3]:

```
img = iradon(f,theta, WayofInterpolation, Filter, 1, s);
```

Filter used for inverse radon transform:

As what we have mentioned before, the reconstructed image by inverse radon transform is usually very blurry. MATLAB provide us with different filters based on the Fourier slice theorem and the filtered back slice theorem. Those filters are actually really simple to design. They are based on the existance of a relation between the 2D Fourier transform of an image and its 1D Fourier transform at a fixed theta projection angle.

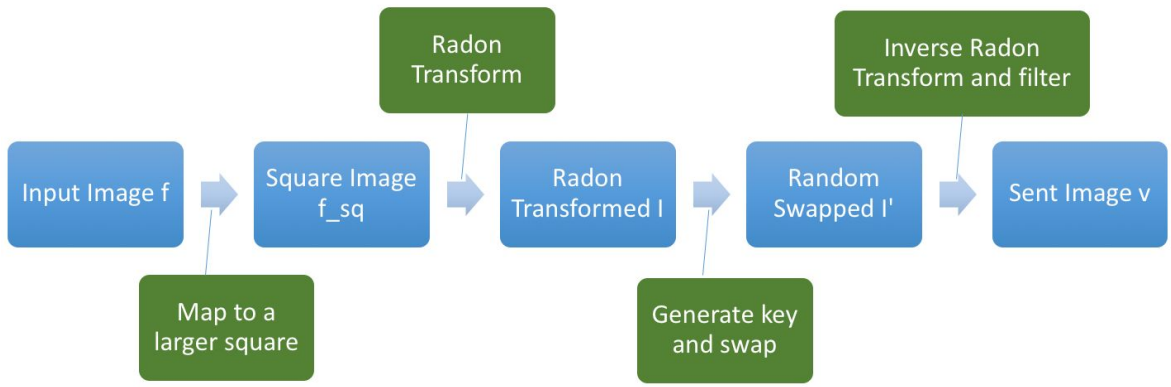
3. Encryption Algorithm

The major part of our project, as well as the most difficult part, was to design an encryption algorithm that would be compatible with the radon transform, and can be recovered using a unique key.

Our first attempt was to add some noise to the picture, this was unsuccessful because the image can be decrypted using noise filter. Our next attempt was using PLU decomposition. This approach failed because it was not compatible with radon transform.

The best result we could achieve was b swapping column vector. Swapping the rows failed because rows are the representation of a spatial domain.

Our encryption process:



We designed a function as below that will generate a random key and a corresponding matrix. For swapping:

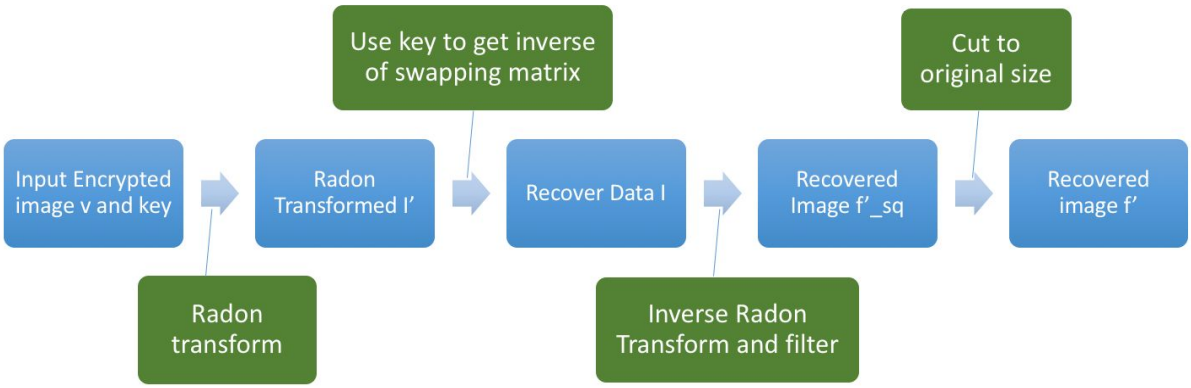
```
function [A, key] = swap_matrix(dim)

s = diag(ones(dim,1));
key = randperm(dim);
A = s(key,:);

end
```

Also, at the first time, we tried to generate image without using any back-projection filter, but we failed. Moreover, we found that mapping to a square matrix results in a better image, This will be discussed in further details in part 5.

4. Decryption Algorithm

The decryption algorithm is a bit different from encryption. First, after radon transform, we have to use the inverse/transpose (in this case the matrix is orthonormal) of swapping matrix. Also, we have to let the decrypted image convolve with a low-pass filter.

Our decryption process:



Result:

image recovered by radon transform and its inverse without encryption

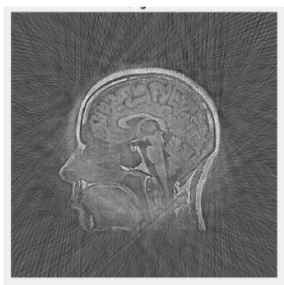


image decrypted

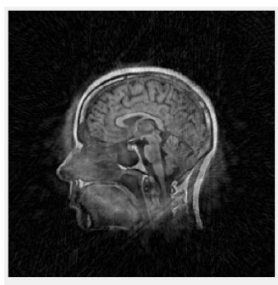


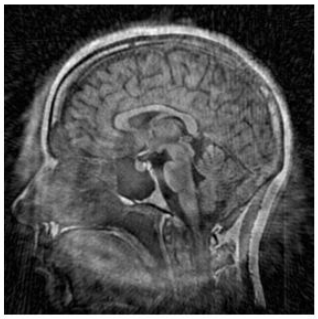
image decrypted and low-pass filtered

As the result shown, the decrypted image has some irrelevant line and we decide to convolve with a low-pass filter with certain frequency parameter.

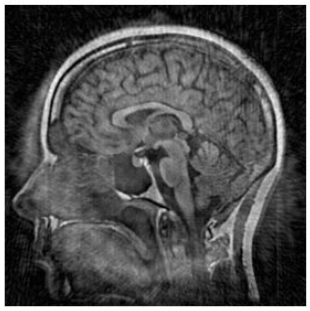
5. Result to Variation of different parameters:

a. Magnification of Size of Square mapped: L^2

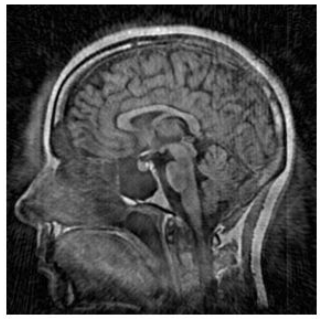
(fix θ = 1.5, const = 1, which will be introduced later)



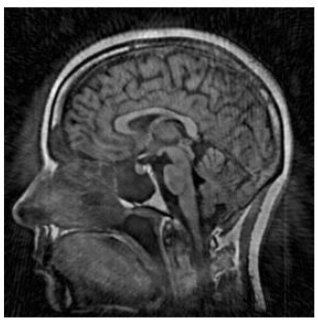
L = 1.1



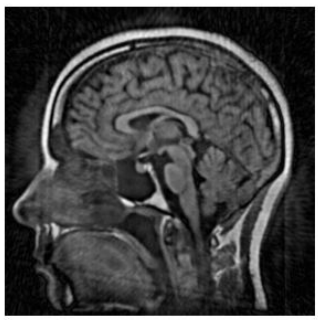
L = 1.3



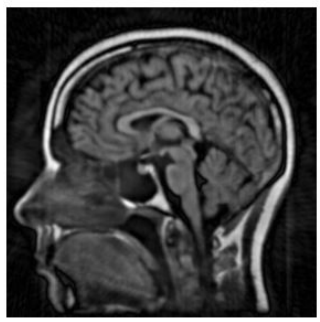
L = 1.5



L = 2.0



L = 3.0



L = 5.0

The intuition behind is that we have to map the image as a square first for fixing the size of output of inverse radon transform. And we find that for a greater magnification, the strange 'circle' effect of reconstructed image will disappear. However, the trade-off of greater magnification is that, computational complexity of the algorithm will increase, for both computational time and memory space.

b. Sampling rate θ:

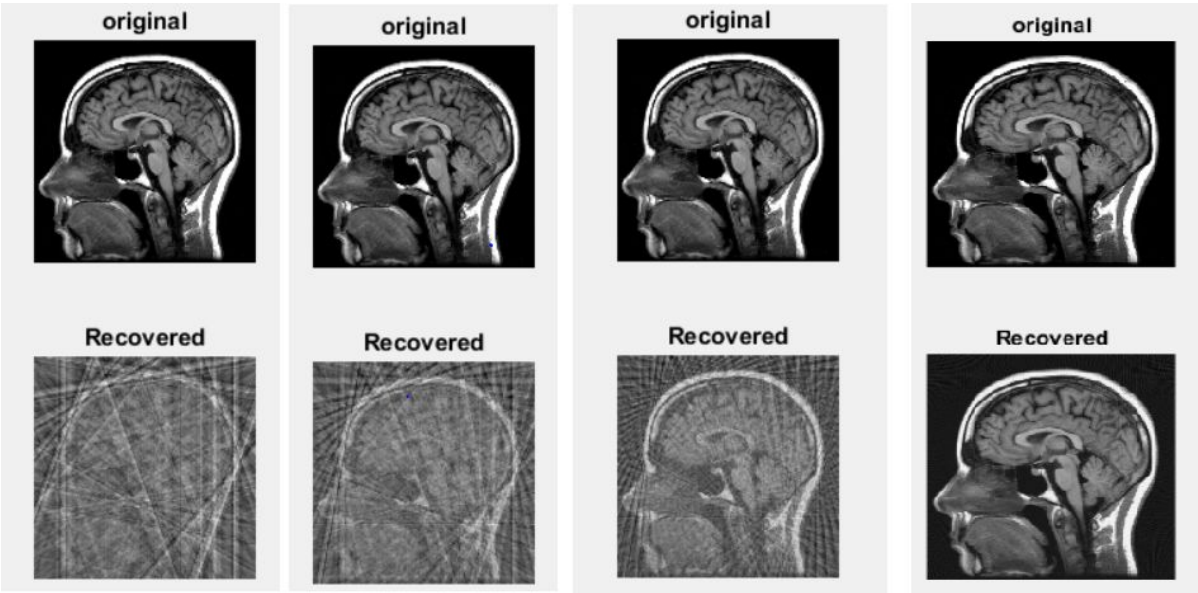
The θ parameter controls the sampling rate. The ideal θ value would be infinite, but this would result in infinite amount of information that can't be stored. In this part we will discuss the result of varying the theta parameter. The theta parameter controls the number of projections we will have. The more projections we have the better results we get. A trivial observation will conclude that if we have infinite number of samples we can reconstruct a

prefect image. This is not true in our case. The sampling source is rotated around an object and every two consecutive samples will contain some redundant information. This leads to a blurry image. On the other hand, having a small theta value will result in an incomplete reconstructed image.

To conclude, a large theta value results in a blurry reconstructed image and a small theta value can't reconstruct the initial image. To address the issue of the blurry image we can use filters based on back sliced theorem.

Those figures illustrate the effect of varying the value of theta on the reconstructed image:
(The images were reconstructed using a filter to avoid a blurry result).
# of samples:

9 18 32 180



c. Cutoff frequency of Low-pass filter: (fix L = 1.5, θ = 1.5)

We use Gaussian filter as our low-pass filter. The lower the cutoff frequency, the blur and brighter the image is. However, if we set the cutoff too high, the noise may remain in the image, the color may not be recover to what we desire. To generalize the cutoff frequency, we simply consider the dimension of the image, it is because generally, larger image can contain more sharp edge inside the image, so the cutoff frequency will be:

const * x_size * y_size / mean(x_size, y_size)

where the ***const*** can be tune to match the image, and x_size, y_size are the dimension of original image.

Unfiltered const = 0.04 const = 0.08 const = 1

Above figure show how filter darken the white noise. It seems when the ***const = 0.08*** give the best result. Although ***const = 1*** seems good, the whole image is darken and this is not we desire. Low cutoff may forbid sharp edge to pass, that's why ***const = 0.04*** result too blur.

It is hard to find a general constant to optimize every image, since each image has different color distribution. If the image is originally sharp in edge, using low cutoff may blur all the edges, but using high cutoff will also keep the edge that geneated by noise like this:

You can see unwanted white edge remains here. This is beacuse high frequency cutoff can also allow edge generated by noise to pass through it.

6. Limitation

Our algorithm can recover most of the patterns of the original image, but the intensity is hard to get fully recovered. We think the main reason behind is the ill-poseness of radon transform. Also, repeating radon inversion will amplify non-differential noise of image.[4] We have tried to use logistic function[5] for intensity transformation to make those gray area to be white, but this cannot solve 'white edge' added to the image. Moreover, we only consider handling image after radon inversion to improve recovery quanlity. However, there are ways we have not tried yet, like TV regularisation of sinogram[6] etc.

References:

[1]:http://www.math.ust.hk/~masyleung/index.html

[2]:http://www.mathworks.com/help/images/radon-transform.html

[3]:http://www.mathworks.com/help/images/ref/iradon.html

[4]:http://statweb.stanford.edu/~candes/math262/Lectures/Lecture10.pdf

[5]:https://en.wikipedia.org/wiki/Logistic_function

[6]:https://arxiv.org/pdf/1403.1272.pdf