# Accelerating MIMO Acoustic Tomography on GPU

*Sarah Iftikhar, Jessie Smith*

*Research Advisor: Dr. Yuanwei Jin, Dr. Enyue Lu*

## Abstract

Graphic processing units (GPU) can be used to accelerate multiple-input multiple-output (MIMO) acoustic tomographic image reconstruction by parallelizing the iterative algorithm. Using GPUs, the computation corresponding to a source excitation is executed on a thread in the CUDA model, while the computations associated with a sensor group consisting of multiple sources are conducted by multiple threads in parallel. Our goal is to study how to utilize CUDA's dynamic parallelism ability to further accelerate imaging algorithm by parallelizing multiple sensor group excitations.

## Introduction

There are three steps taken in order to parallelize the image reconstruction process.

Step 1: Build the forward model using the 5-point difference equation with boundary conditions.

$$u_{i,j}^{t+1} = \left(\frac{\delta t}{2}\right)^2 v_{i,j}^2 \left(u_{i+1,j}^t + u_{i-1,j}^t + u_{i,j+1}^t + u_{i,j-1}^t - 4u_{i,j}^t\right) + (\delta t)^2 s_{i,j}^2 + 2u_{i,j}^t - u_{i,j}^{t-1}$$

Step 2: Calculate the field values and difference signal (the difference between the calculated acoustical potential values and the true values at the 640 sensor positions.)

Step 3: Update the reconstructed image until the difference between the actual image and the reconstructed image is less than the error margin.

## Problem

The problem we are addressing is how to implement an acoustic tomographic imaging algorithm in parallel. We are looking at how to pass the image to the next iteration. Currently, each source group gives an image reconstruction individually (Algebraic Reconstruction Technique), yet there is no way of joining all the output into one single image that we can pass on (Simultaneous Iterative Reconstruction Technique). Only the reconstructed image sections can be passed to the next iteration, which does not give much improvement because the total iteration count would be the same. We would like to think about how we can combine all the outputs in order to optimize the current algorithm and methods. Furthermore, the SIRT method would allow each sensor group to calculate values in parallel, while the ART method would not, due to dependency between sensor groups.

## Implementation

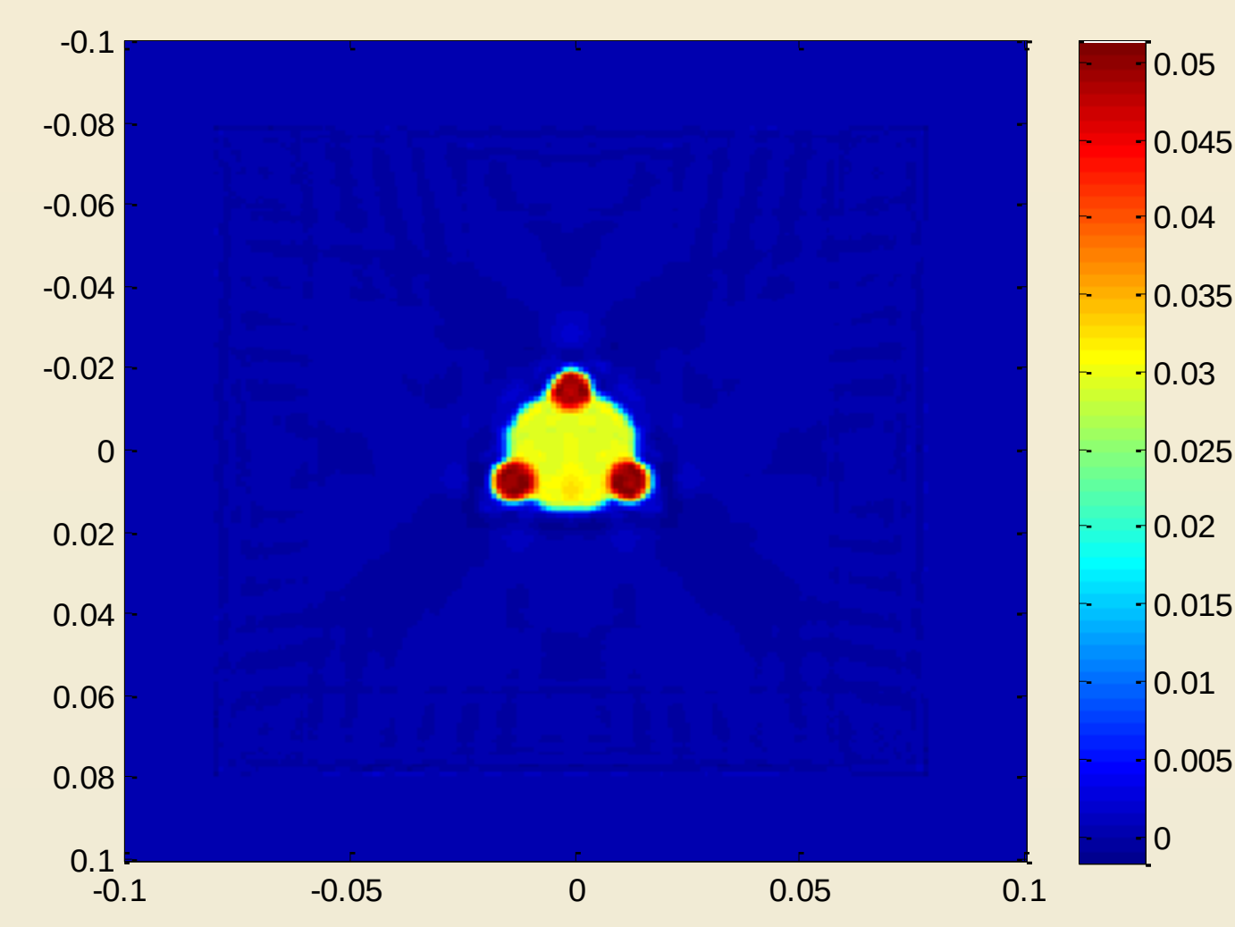
In order to pass a whole image to the next iteration, we will add together all the image pieces by implementing the SIRT algorithm. That is, for each pixel, we add together all the different sensors' calculated values for that pixel. We can implement this in parallel by having a separate thread for each sensor. This should result in a faster execution.

The sequential MIMO-PBP algorithm used currently is (ART):

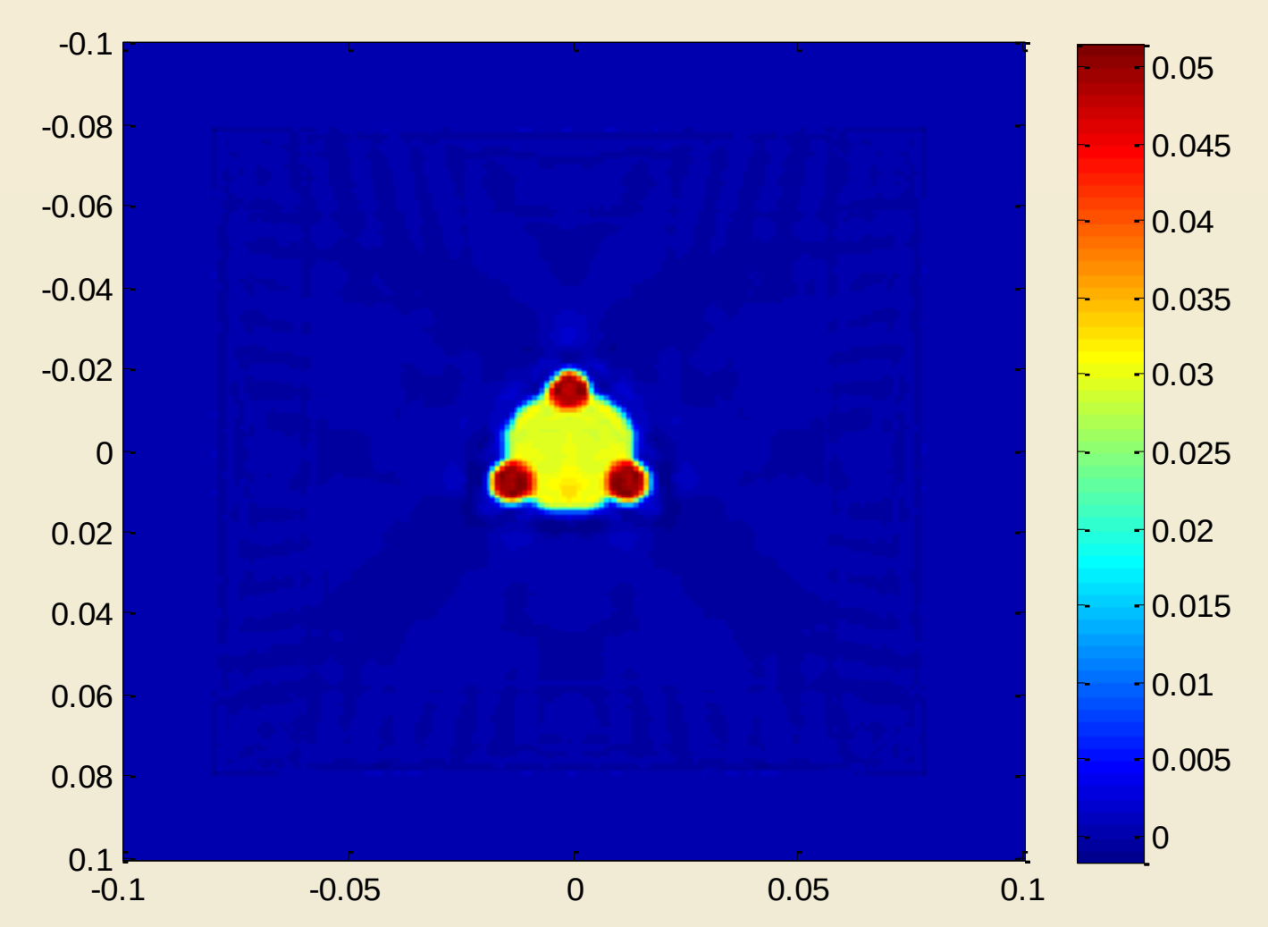
$$f^{k+1} = f^k + \omega \Delta f^k(s_m), m = 1, 2, \dots, M$$

The parallel MIMO-PBP algorithm we want to use is (SIRT):

$$f^{k+1} = f^k + \omega' \sum_{m=1}^{M} \Delta f^k(s_m)$$



**ART: 19 iterations**



**SIRT: 71 iterations**

Figure 1: With each iteration of the ART CPU-based algorithm, the image becomes clearer. This process takes about 28 minutes and 19 iterations. Our new SIRT algorithm takes about 107 minutes and 71 iterations. The image above shows the final image for both methods

## Approaches

Our first approach was to implement the SIRT algorithm sequentially, using no GPU processing. When we did this, while both methods yielded very similar images, we found that the SIRT method converged significantly slower than the ART method, as shown in Figure 1 and 2. We then attempted to use CUDA's dynamic parallelism ability to use the SIRT method to implement the sensor groups in parallel. We found that the program crashed when we used more than 100 sensors and could not gather conclusive results. However, by using only 100 sensors for both the parallel and sequential MIMO-PBP methods, we compared the convergence and saw a slower convergence of the parallel algorithm once again.

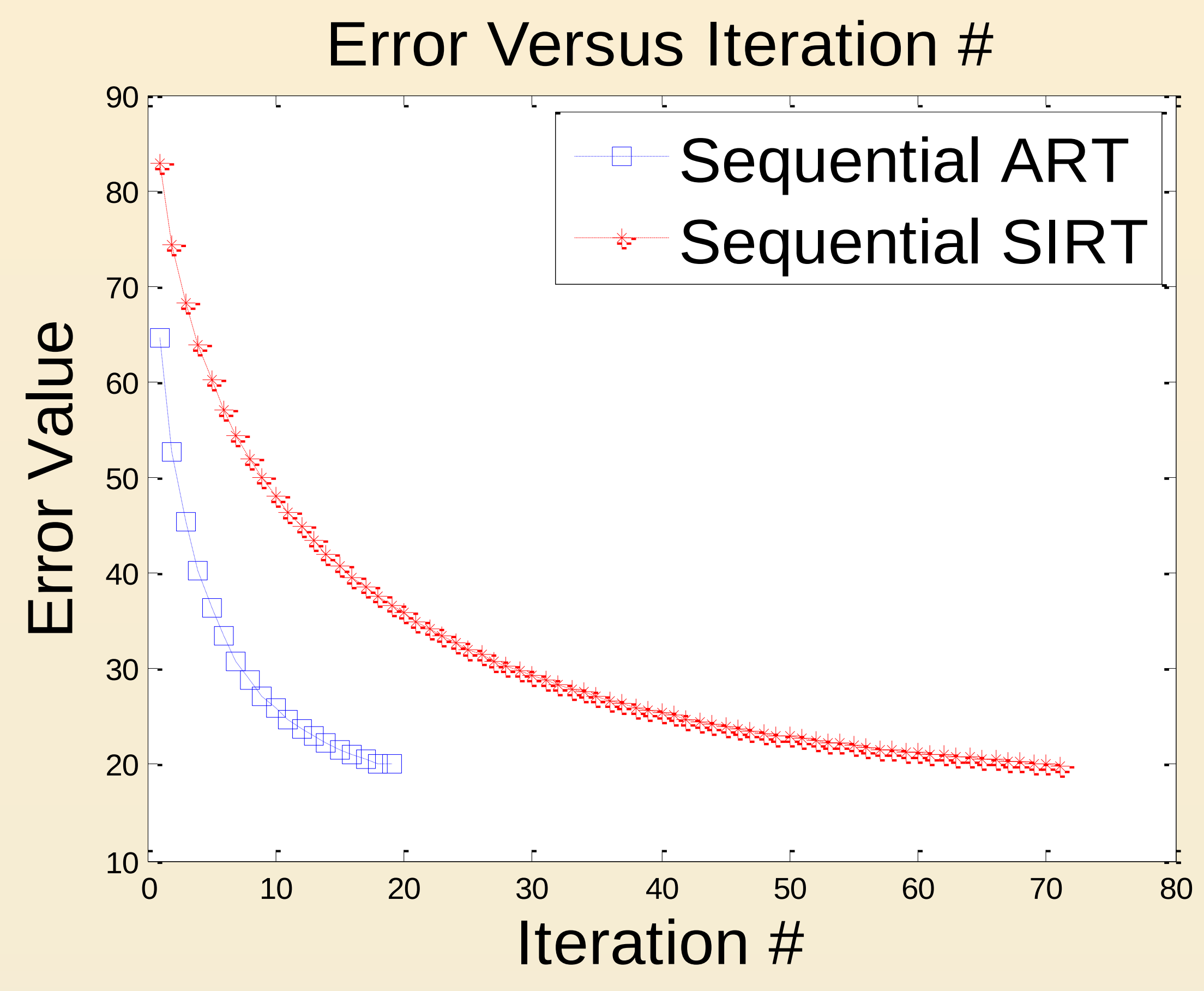


Figure 2: The sequential SIRT method we implemented takes many more iterations for the error value to converge to 20. It takes the sequential ART method only 19 iterations to converge, while the sequential SIRT takes 71 iterations.

## Conclusion and Future Work

We were able to implement the sequential SIRT algorithm and found that the successive residual errors actually converge slower than with the ART method. While each iteration takes about the same amount of time with both methods, the number of iterations required for the SIRT method is much higher, and, therefore, the total run time is higher. However, we cannot conclude that parallelizing the sensor loop would make the image reconstruction slower. In order to do this, we would need to implement the parallel MIMO-PBP algorithm using the correct number of sensors. Our next steps would be to fully implement the algorithm using 640 sensors and to understand why we may be getting a slower convergence with the SIRT method.

## Significance of Work

The development of a faster, more efficient algorithm for image reconstruction has many practical applications. In ultrasonic tomography, it would take less time to develop an image in the medical field, where time is essential. Examples ranges from ultrasounds to brain PET scans.

## Acknowledgments

Salisbury University and the National Science Foundation are gratefully acknowledged for support of the REU EXERCISE Program.