Implementation and Validation of a GPU-based X-ray Diffraction Monte Carlo Simulator for in-vivo Breast imaging applications

by

Oluwadamilola Fasina

Medical Physics Graduate Program
Duke University

Date:_____

Approved:

_____
Anuj J. Kapadia, Advisor

_____
Joel A. Greenberg

_____
Robert E. Reiman

Thesis submitted in partial fulfillment of
the requirements for the degree of Master of Science in the
Medical Physics Graduate Program in the Graduate School
of Duke University

2021

ABSTRACT

Implementation and Validation of a GPU-based X-ray Diffraction Monte Carlo Simulator for in-vivo Breast imaging applications

by

Oluwadamilola Fasina

Medical Physics Graduate Program
Duke University

Date:_____

Approved:

_____
Anuj J. Kapadia, Advisor

_____
Joel A. Greenberg

_____
Robert Reiman

Thesis submitted in partial.
fulfillment of the requirements for the degree
of Master of Science in the
Medical Physics Graduate Program in the Graduate School of
Duke University

2021

Abstract

The 5-year survival rate for metastatic breast cancer is just 28%, while the 5-year survival rate for a detected localized tumor in the breast is 99% [1]. Often, tumors originating from the breast go undetected during a mammography scan, which is a conventional transmission-based X-ray image optimized for breast imaging [2]. If these breast tumors go undetected, they can proliferate in the body and lead to a fatality. Mammography images are limited because the image resulting from a mammography scan is based on the selective interaction of X-ray photons within the structure of the human body (i.e whether a photon is transmitted or absorbed when travelling through the body). The probability of transmission or absorption is primarily dependent on the atomic number of the material and the material density. This is problematic because the density difference between cancerous tissue and surrounding tissue in the breast is on the order of ~ 0.1 g/cm^3 [3], and biological tissue in the breast have the same Z_{eff} (effective atomic number). This leads to low contrast in the image resulting from the mammography scan, making it difficult for a physician to give a proper diagnosis.

Our group's focus is to aid in the diagnosis and consequentially the prognosis of breast cancer by taking a novel approach to breast imaging: X--ray diffraction imaging. The fundamental difference between X-ray diffraction imaging and conventional transmission-based X-ray imaging is the signal being measured. X-ray diffraction

imaging measures coherently scattered photons, which give rise to chemical and structural information of the material or tissue with which the photon interacts [4]. Inclusion of chemical and structural information can aid in distinguishing healthy and cancerous tissues [4] , [5] which can lead to earlier detection of breast cancer and also negate the need for biopsy when a mammography scan is not sufficient.

In this study, we utilized graphics processing unit (GPU)-based Monte Carlo (MCGPU) [6] simulations to evaluate whether XRD imaging can be used for in-vivo breast cancer detection. Before using MCGPU for in-vivo breast imaging, we performed unit testing and validations of MCGPU against empirical data, GEANT4 (a widely used Monte Carlo simulation software toolkit), and experimental systems. Following these series of tests, an anthropomorphic 3D computational breast phantom [16] containing a breast tumor was scanned in MCGPU.

MCGPU was able to replicate simulation results from GEANT4 in a scan time 1000x faster than GEANT4 without a loss of model fidelity, due to its GPU-based architecture. Additionally, MCGPU was able to model real-world experimental systems and produce simulation results that matched with experimental results. Finally, evaluation of in-vivo breast scans performed in MCGPU demonstrate XRD imaging has the potential to differentiate cancer from healthy tissue within the breast and can be used to supplement conventional mammography scans.

Dedication

To my advisors, Anuj J. Kapadia and Joel A. Greenberg, I am forever indebted to each of you for your persistence and continual devotion to making me a better scientist, researcher, and person. Thanks to your mentorship, I have grown as an intellectual and individual in parallel and am fortunate have been pushed by both of you during my time here at Duke to develop a more mature scientific mindset and to always dig deeper when solving and analyzing problems.

To Cristiano dos santos Aveiro Ronaldo and Kobe Bean Bryant, you showed me hard work could be applied agnostically in life and that pressure is a privilege. To FASH, thank you for getting Dami elevated every weekend to make the completion of this thesis possible.

To my friends and family, your support was instrumental in helping me achieve my academic and personal goals while at Duke. Without your support, I can unequivocally state I would not be in this position; I am forever grateful for this.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to acknowledge my technical and emotional support system. This includes but is not limited to my family, my friends, Dr. Anuj J. Kapadia, Dr. Joel A. Greenberg, Dr. Robert Reiman, Dr. Josh Carpenter, Matthew Japzon, Stefan Stryker, Jordan Houri, Brian Harrawood, Dr. Andreu Badal, and Dr. Bahaa Ghammraoui

1. Introduction

The aim of this thesis is to evaluate the efficacy of using X-ray diffraction (XRD) imaging for in-vivo breast cancer detection. To investigate this problem, we used a GPU-based Monte Carlo simulation toolkit – MCGPU – to model various XRD imaging systems. This document outlines the series of validations MCGPU was put through and the subsequent application of MCGPU to evaluate XRD imaging as an imaging modality for in-vivo breast cancer detection.

In this section, the theory behind XRD imaging, the motivation for the project, conventional Monte Carlo simulations, and MCGPU are discussed to provide the reader with the appropriate context for a thorough understanding of the study undertaken.

1.1 Conventional vs XRD Imaging

Conventional radiographic images are displayed using a grayscale mapping, where black represents the highest X-ray intensity and white represent the lowest X-ray intensity [2] arriving at the detector. Conventional radiographic imaging is also known as *transmission* imaging because the image generated represents an acquisition setting where the primary beam is orthogonal to the

detector (i.e photons are directed in a straight path to the detector), and the detected image is a measure of which photons from the x-ray source (primary beam) are incident on the detector in a straight-line path (ideally; realistically, there will be some scatter). Physically, the black regions of a radiographic image can be interpreted as the region of highest photon intensity or impingement on the detector, while white regions can be interpreted as regions with the lowest photon intensity or impingement, and gray represents some intensity in-between [17]. Continuing with this reasoning, white portions of image correspond to locations where an object positioned between the primary beam and detector obstructs the path of the photons emanating from the source to the detector, thus resulting in lower photon intensity. The black regions of the image represent portions of the image where the highest number of photons impinging on the detector and therefore do not have anything obstructing the photons path.

In a radiographic (in this case, mammography) image, the probability of a photon being transmitted through the object obstructing the path from source to detector, is based on probability of a photon being absorbed via photoelectric absorption which is generally dependent on the density of the material and effective atomic number of the material, Z_{eff} for a given photon

energy [17]. Therefore, for materials with similar atomic density or effective atomic number, a radiographic scan will be limited in its ability to distinguish one material from another. For biological tissues in the breast such as fibroglandular, adipose, and cancerous tissues, the effective atomic number and atomic density is similar [3], even though they have different chemical and structural properties. Thus, an imaging modality such as X-ray diffraction imaging that can use other properties of the material could improve the ability to distinguish between healthy and cancerous tissue within the breast.

In X-ray diffraction imaging, the acquisition set-up is identical to that of a transmission imaging system; but the signal measured is different. In XRD imaging, the coherently scattered photons are measured, rather than the transmitted photons. This is fundamentally different from conventional imaging. In this situation, the signal measured is not a measure of a photon to pass through an object and interact with the detector; instead, the signal measured is due to photons that have been coherently scattered from the object and are incident upon the detector [4].

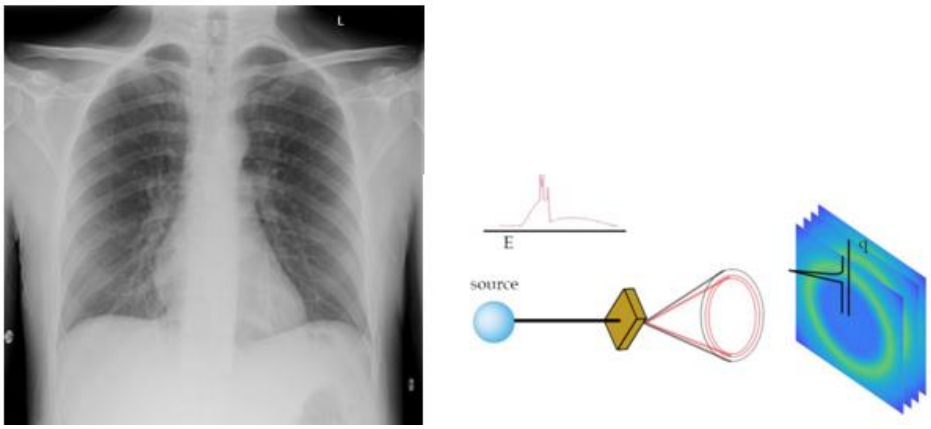


Figure 1: Conventional Transmission X-ray Image (left) vs XRD Imaging set-up where coherently scattered photons are demarked by green circular patterns

Coherent scattering is most prevalent at lower energies (up to ~60 keV) [4] and the kinetic energy of the scattered photon is an elastic scattering process in which the phase of the X-ray is preserved during the interaction with an atom [2]. In systems consisting of multiple atoms or molecules with at least some degree of order, interference between the scattered X-rays gives rise to intensity maxima that are governed by Bragg's law [7]:

$$n\lambda=2d*\sin(\frac{\theta}{2}) \quad (1)$$

where n is a positive integer, λ is the wavelength of the incoming photon, d is the inter-molecular lattice spacing in the scattering material, and theta is the resulting angle between the incident and scattered photon directions following a

coherent scatter event. To make use of the scattered photons being measured, we rewrite Bragg's law in terms of the momentum transfer, q, which can be related to the inverse of twice the interplanar distance as shown in equation 2 below:

$$q = \frac{1}{2d} * \frac{E}{hc} * \sin\left(\frac{\theta}{2}\right) \quad (2)$$

For a given atom or molecule, a scatter form factor – a measure of the photon scattering amplitude with respect to scatter angle – has a functional dependence on the momentum transferred to the scattered photon and can be used for material characterization and delineation [4]. The independent atomic approximation (IAA) form factor F_{IAA}, is a measure of the amplitude of scatter from an isolated atom, while the molecular interference function (MIF) form factor, F_{MIF}, is a measure of the scattering amplitude that incorporates the effects of molecular interference in scattering profiles [8].

The molecular interference form factor is related to the expected scatter intensity which is defined by the differential coherent scatter cross section:

$$\frac{d\sigma_C}{d\Omega} = r_e^2 \frac{(1+\cos(\theta))^2}{2} [F_{IAA}]^2 \quad (3)$$

where,

$$F_{IAA}^2 = \sum n_i * F(q_i, Z_i)^2 \quad (4)$$

and

$$F^2_{MIF}(q) = F^2_{IAA}(q)*s(q) \quad (5)$$

Where q_i is the momentum transfer for an atom, Z_i is the atomic number of the atom and n_i is the weight fraction of the atom in the entire molecule. As shown by equation 4, a weighting factor (the weight fraction of the atom relative to the molecular weight), n_i, is multiplied by the form factors of each of the atoms which the molecule is composed of, then summed to obtain a singular form factor representative of the molecule under the independent atomic approximation. The equation for the differential coherent scatter cross section has a functional dependence on the IAA form factor. Under this formalism, the differential coherent scatter cross-section is limited in how accurately it can model coherent scattering in materials, because the independent atomic approximation does not account for molecular interference in scattering profiles. Thus, the accuracy of the coherent scatter cross section can be augmented by using $F_{MIF,}$ which incorporates molecular interference, instead of $F_{IAA.}$ In MCGPU, the F_{MIF} can be obtained by multiplying the F_{IAA} by an oscillatory function, s(q),

that accounts for intermolecular and molecular effects, or it can be empirically measured [10] for both non-biological and biological material [18]. Figure 2 below [15] shows the difference between simulations that use F_{IAA} or F_{MIF} when defining the differential scatter cross section in MCGPU.

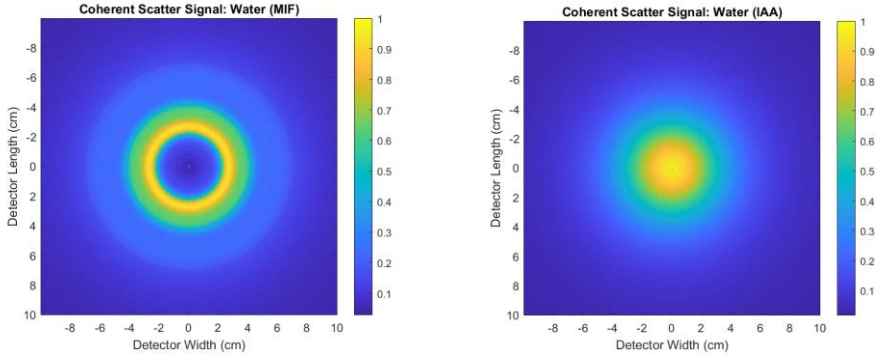


Figure 2: Effect of modeling F_{MIF} (left) and F_{IAA} (right)

1.2 Motivation

For breast cancer, the 5-year survival rate for metastatic breast cancer is 28% [1]. Essentially, if a lesion originating from the breast goes undetected and spreads to another area of the body, the odds of survival are low. Thus, it is paramount that lesions in the breast are detected early; in fact, lesions detected in the breast that are localized give the patient a 5-year survival rate of 99%.

If radiologists and pathologists can detect lesions early when they are localized, the patient outcome and prognosis is significantly ameliorated – this generally holds for all regions of the body. However, breast cancer has proven to be difficult to detect early, due to the similarity in effective atomic number and density between normal and cancerous breast tissue. It is recommended that women have conventional screening mammography annually beginning at age 40 [11]. However, as mentioned in the above section, these conventional X-ray mammography scans are limited because the contrast is primarily dependent on the effective atomic number and the density. The problem stems from having low/negligible density differences between cancerous and normal breast tissue (cancer is 1.06 g/cm^3 while normal is ~1.05 g/cm^3) [3]. This leads to a physician being limited in their ability to diagnose a patient with breast cancer based simply on a mammogram alone. XRD imaging can circumvent this problem by providing chemical and structural information that leads to unique and distinguishable signatures form cancerous and normal tissue [5]. This phenomenon can be seen in Figure 3 below, which shows the XRD form factors for different breast tissues measured experimentally[3]. For different biological

tissues, the form factor as a function of momentum transferred to the scattered photon is distinguishable.

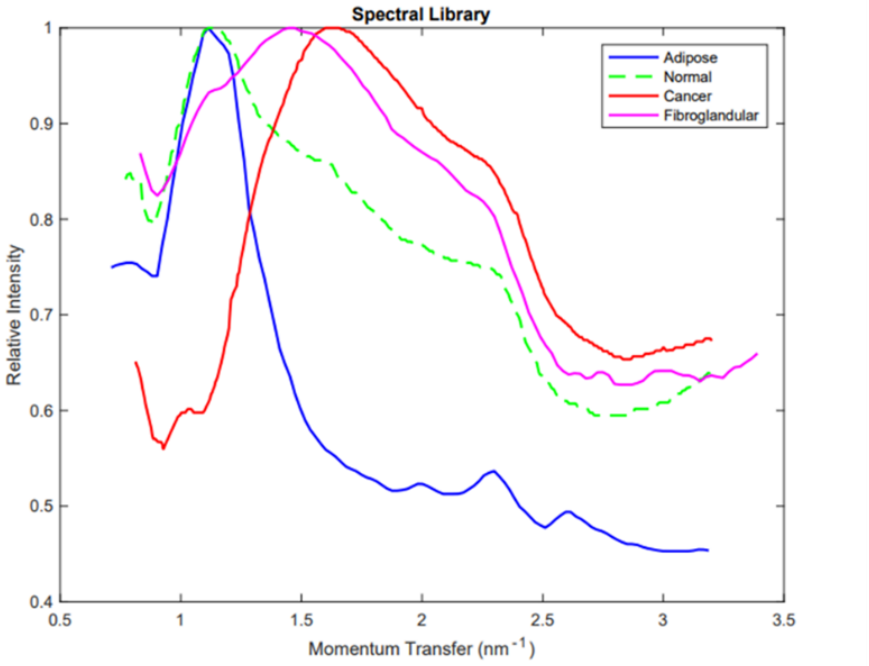


Figure 3:Tissue Delineation via XRD Imaging [3]

In this study, we aim to use MCGPU, a GPU-based Monte Carlo simulator, to determine whether XRD imaging can be used for in-vivo breast cancer detection. Monte Carlo simulations allow us to design and explore potential XRD imaging systems in a practical manner – Monte Carlo simulations can be used to optimize existing XRD system designs and model novel XRD imaging systems. In addition, the GPU computing power allows for reduced

computational expense, thus allowing for more flexibility in design capabilities. We aim to utilize MCGPU to evaluate the efficacy of XRD imaging for in-vivo breast cancer detection such that breast cancer can be detected early and subsequently treated with the proper care to optimize patient outcomes.

1.3 Monte Carlo Simulations

Monte Carlo (MC) models are distinguished from conventional analytical physics models because of their stochastic-based particle tracking nature. Every particle's fate is determined by the probability of different events occurring from particle birth to death. Monte Carlo simulations are extremely powerful because they are highly parallelizable (i.e each particle operates independently of the others), can be used to model many real-world processes, and can be extremely accurate. The accuracy of MC models stems from the algorithm's ability to track a particle at each step; however, the trade-off associated with this accuracy is computationally expensive. The GEANT4 MC toolkit is widely considered to be the gold-standard for simulations of radiation transport [12] and utilizes Penelope physics models such as Compton and coherent scattering, multiple scattering, and photoelectric absorption [13]. Although GEANT4 is generally able to produce extremely accurate results, the framework does not natively model the electromagnetic interference that arises during coherent scatter from

collections of atoms or molecules – instead, GEANT4 uses the Hubbell form factor model, which only accounts for physical interactions with individual, independent atoms. As a result, simulations using default GEANT4 physics are unable to accurately model the types of signals that would be measured in X-ray diffraction (XRD) imaging of molecular materials unless extra care is taken to represent these effects; these modifications were recently made [19] to include molecular interference. However, because of the long run-times we opted for a GPU-based Monte Carlo simulator, MCGPU, to use as our toolkit for evaluating XRD imaging as a suitable imaging modality for in-vivo breast imaging.

1.4 MCGPU

To circumvent these challenges, MCGPU, a GPU-based Monte Carlo simulator was developed. MCGPU uses a graphics processing unit (GPU) in lieu of a CPU to enhance computation speed. MCGPU uses CUDA, Compute Unified Device Architecture [14], a parallel computing platform developed by NVIDIA to accelerate particle transport. The x-ray interaction models and material properties are adapted from PENELOPE, a general-purpose Monte Carlo code for coupled electron-photon transport. Basic operation of MCGPU involves a) adapting the simulation input file, b) defining the CT trajectory, c) defining materials, and d) specification of the geometry of the voxelized object being scanned. MCGPU can simulate a single radiographic

projection or a full CT scan and launches thousands of threads in a single simulation with each thread corresponding to several X-ray tracks (typically 1E10 histories or X-rays are used). GPUs can accelerate computations times because they possess a greater number of parallel computing cores than a conventional CPU - this allows for higher number of simultaneous calculations to occur. In addition, MCGPU is inherently capable of fully modeling XRD because it incorporates the molecular interference function (MIF) in its algorithm, which accounts for molecular interference in the object during coherent scatter. Although MCGPU is capable of modelling molecular interference, the user must still provide empirical data containing molecular interference information for a given material; thus, special care must be taken to implement the effect of molecular interference for different materials.

2. Simulation Workflow

In this section, the simulation workflow including a) preprocessing, b) running an MCGPU simulation and c) post-processing are outlined.

Monte Carlo simulations use a stochastic-based approach to problem solving rather than using a conventional deterministic approach. In the context of radiation transport, particle interaction, energy, and direction of travel are determined by sampling probability distributions derived from physics and geometry. MCGPU uses an input file to describe the geometry and set-up of the simulation world. In this input file, the coordinate system geometry, spectrum, object, detector, and number of histories can all be defined. MCGPU launches thousands of threads simultaneously in the GPU to augment the number of particles that the simulation can track at an instance, which rapidly speeds up the MC simulation.

Out of the box, MCGPU models the effect of molecular interference in coherent scattering and uses its GPU architecture to accelerate simulations times without a loss of accuracy. However, it does not have a readily available database with form factors that incorporate MIF – this is necessary to model XRD of different materials. In addition, it is limited to only simulating XRD of amorphous objects because of the intrinsic coarse momentum transfer sampling. To circumvent this, we implemented empirical form factors that incorporate MIFs into the MCGPU framework and increased the form factor

sampling parameter in PENELOPE from 128 to 511 to allow for finer sampling of crystalline materials which have sharp form factor spectra. Figure 4 depicts the workflow of running an MCGPU simulation, which is discussed in detail below.

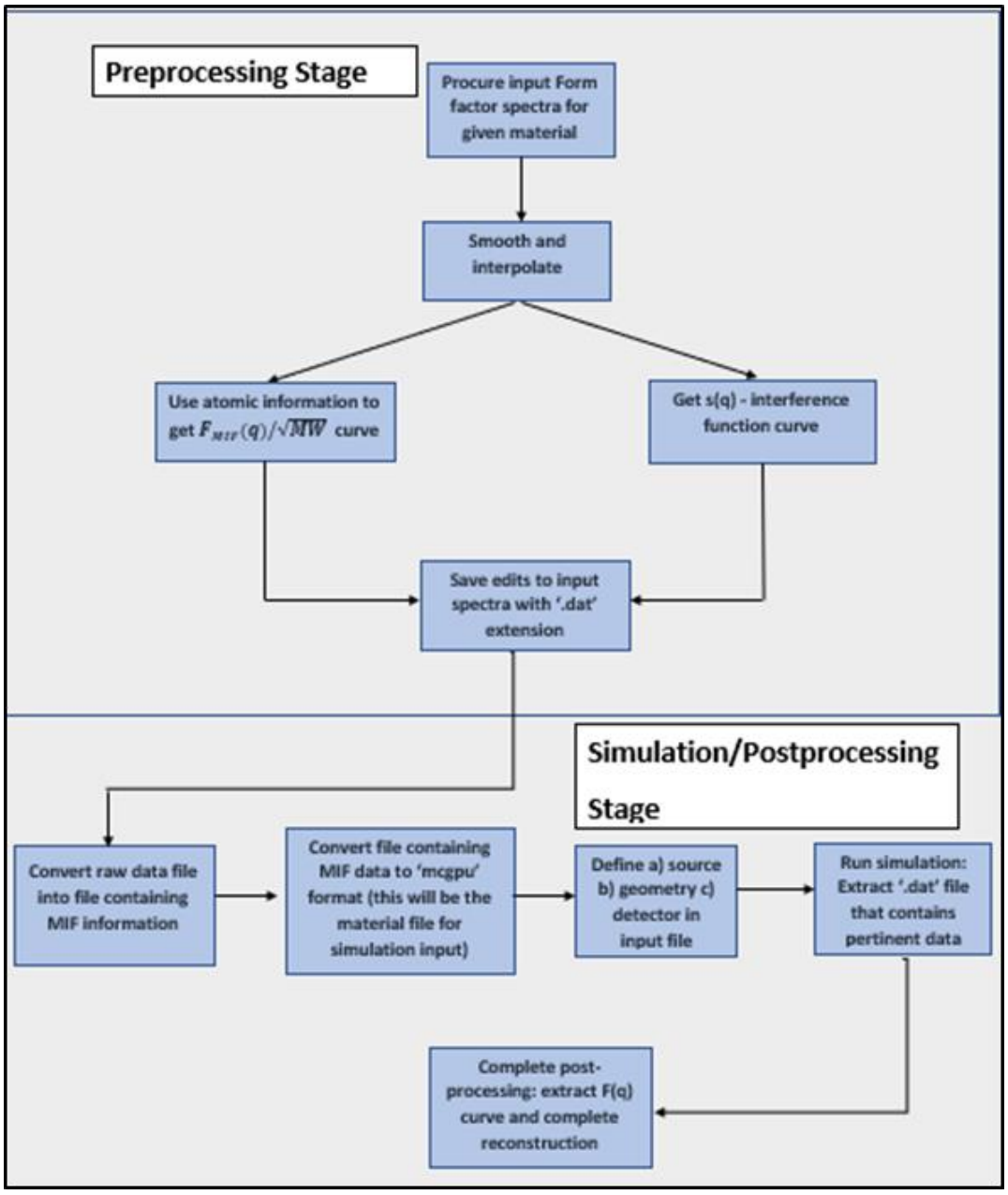


Figure 4: Simulation Workflow: Flowchart describing the steps taken to run and analyze and MCGPU simulations

Empirical form factor spectra were obtained from an in-house database comprising experimental measurements obtained using commercial XRD systems [23][24]. Depending on the data available, the raw data were edited to contain two possible input formats. The interference function, s(q), is used when molecular weight of the material is not readily available; if the molecular weight is readily available, the form factor normalized to the square root of its molecular weight is used instead, to eliminate the impact an objects' molecular weight has on the final scattering form factor. The interference function is an oscillatory structure function that accounts for molecular and intermolecular interference effects [4]. The raw data file is converted to an MCGPU material file containing the MIF using executables included in the original MCGPU package. The MCGPU material file that is created contains the mean free path for photoelectric, Compton, Rayleigh, pair production, the form factor cumulative distribution function (CDF), and the sampling data. The MCGPU material file is an input parameter for the MCGPU simulation along with source spectrum, object dimensions, detector, system geometry, and number of histories. The object dimensions are specified in a file separate from the MCGPU material file and is also specified in the input file.

The MCGPU model consists of a source code which defines the geometry, detector, and source data for the simulation and model platform for MCGPU, and a

kernel code which interfaces with the PENELOPE packages to handle the radiation transport. An MCGPU simulation outputs a file containing a matrix the size of the detector in units of $\frac{eV}{cm^2 * history}$ for each pixel on the detector. The output file had four separate detectors that contained data for the following interactions: non-scattered, Compton, coherent, and multiple scattering.

The figures presented in this study were all generated from MCGPU output files. Visualization of the data, both as 2D plots and specific line profiles, was performed in Matlab. Finally, to recover form factor spectra, the coherent scatter data was converted into a polar representation and the intensities were summed over the azimuthal scatter angle (phi). The resulting scatter intensity as a function of radius was then converted to momentum transfer, q, based on incoming photon energy and the object to detector distance. Since the XRD scatter intensity is proportional to the square of the form factor, the square-root of the detector intensity was taken and normalized to unity to produce a form factor signal for all momentum transfer values.

3. Preliminary Results/Cross Validation

3.1 Verification

Unit testing of the MCGPU software was completed by comparing the empirical input form factor spectrum to the recovered form factor spectrum generated by the MCGPU simulation. A simple geometry consisting of a 20 x20 cm^2 flat panel detector, .05 cm pixel pitch in the horizontal x-direction and .05 cm pixel pitch in the vertical z-direction, 1x1x1 mm^3 object, and a (60 keV) monochromatic pencil beam source were modeled and simulated in MCGPU. To represent MCGPU's ability to model amorphous and crystalline materials, a voxel of graphite and a voxel of water were scanned. Cross-correlation was used to compare the simulated output to the empirical input form factor scatter distributions. To determine whether MCGPU's output intensity was accurate, we compared coherent scatter cross sections between simulated and empirical data. The simulated coherent scatter cross section was derived by measuring the intensity of the coherently scattered photons arriving at the detector, I_d, and using Beer's law of attenuation -see equation 6- (object thickness, d, and initial intensity, I_o, were known) to derive the coherent scatter cross section, μ.

$$I_d = I_o * \exp(-\mu d) \qquad (6)$$

Verification Results

Figures 5a and 5b show comparisons between simulated and empirical form factors for water and graphite. The goal of this unit testing was to verify MCGPU's ability to produce output form factor spectra that matched input spectra (i.e recovered simulated data matched the empirical input data). For each subplot, the y-axis is normalized to unity and the x-axis contains units of momentum transfer. The output simulated data is denoted by the dashed blue line and was generated by simulating a monochromatic 60 keV pencil beam and a simplistic geometry (1x1x1 mm object, 20 cm x 20 cm detector, pixel pitch of .05 in both directions, and object to detector distance of 40 cm). The form factor curve was generated from the 2D detector data output from coherent scattering in the MCGPU simulation. Knowing that the square root of detected intensity is proportional to the form factor, the form factor curve was produced by extracting a radial profile of detector data from the detector center to detector edge, square-rooting, and normalizing to unity. Both scans of water and graphite demonstrated agreement between recovered simulated data and empirical data as demonstrated by the cross-correlation results and the coherent scatter cross-sections shown in figures 5a and 5b, table 1, and table 3. The noticeable spectral broadening between the graphite form factor curve is likely due to energy binning in MCGPU. These

simulation results represent MCGPU's ability to produce simulated form factor data (output) that matches empirical form factor data (input).

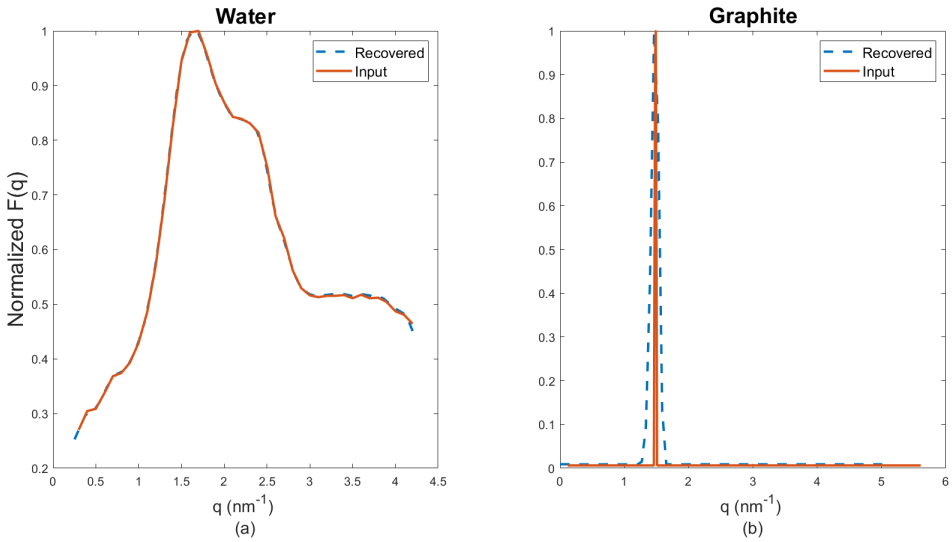


Figure 5: Recovered and Input form factor spectra for Water (left) and Graphite (right)

Table 1: Coherent Scatter cross section absolute difference for between simulation and experiment for graphite and water

Material	NIST XCOM database (cm^1)	Simulation (cm^1)	Absolute Difference
Graphite	0.02	0.021	0.001
Water	0.09	0.087	0.003

3.2 MCGPU vs GEANT4 Cross-Validation

Cross-Validation

The original GEANT4 package did not include the ability to model molecular interference during coherent scatter; however, the version of GEANT4 that

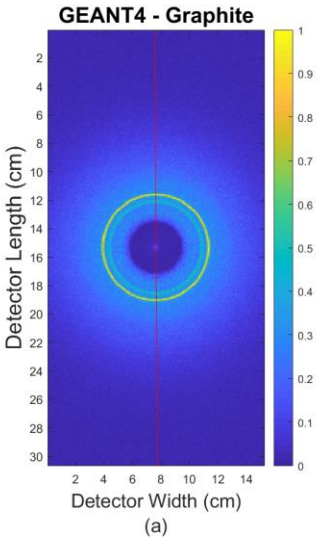
was used was modified to model coherent scattering [19] was used to cross-validate against MCGPU. The goal of the cross-validation was to evaluate MCGPU's capability to perform rapid simulations without a loss of fidelity – this was accomplished by modeling an identical system in GEANT4 (pencil-beam system described below) and comparing coherent scatter distributions and simulation time between MCGPU and GEANT4. Both MCGPU and GEANT4 simulated a scan of a 1 cm^3 object of graphite using a simplistic geometry (1x1x1 cm object, 20 cm x 20 cm detector, pixel pitch of .05 in both directions, and object to detector distance of 40 cm). Peak scatter intensity location and cross-correlation between the normalized intensity curves are reported in Table 5 while simulation time and other computer specifications are reported in Table 2 below Figure 6. Peak scatter intensity location was determined by extracting a vertical line profile across the center of the 2D scatter data in GEANT4 and MCGPU and comparing the locations where each simulation system exhibited the most intense scatter.

Cross-Validation Results

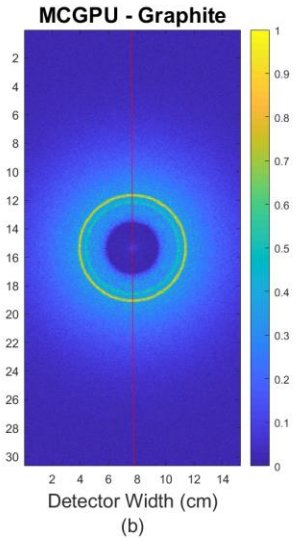
Figure 6 depicts cross-validation results of MCGPU against GEANT4. In Figure 6a and 6b, the detected 2D scatter distribution (multiple scatter, Compton, and coherent) generated from GEANT4 and MCGPU using the pencil-beam geometry to scan a cube of graphite are displayed. The x and y axes are the detector width and length, respectively, and contain the physical dimensions of the detector, and the scatter intensity is

normalized to unity for both the GEANT4 and MCGPU 2D scatter matrices. The two subplots shown in Figures 6a and 6b depict exceptional qualitative agreement – the ring location, scatter distribution, and intensity fall-off all mirror each other. To produce the plot in Figure 6c, a vertical line profile across the center of both detectors was taken and data was extracted. The vertical red line in Figure 6 denotes the location of data extraction from the detectors. The y-axis in Figure 6 represents the scatter intensity normalized to unity, and the x-axis represents the physical units of the detector width (in cm). The quantitative metric of comparison was peak location percent difference: both the peak locations at pixel values 132 and 216 on the x-axis exhibited peak location percent differences of less than 0.1% between GEANT4 and MCGPU. Of note, the simulation for GEANT4 (1 CPU core, 2.8 GHz machine) took 8400 seconds while the simulation for MCGPU (3840 GPU cores, 1.9 GHz machine) took 2.32 seconds (more computer specifications listed in Table 2) – this is a considerable speed up in computing power due to the parallel computing and number of cores MCGPU has as its disposal. Both MCGPU and GEANT simulated 1E8 histories. The relatively small number of histories run was due to the computationally expensive simulation times in GEANT4. MCGPU has 3840 cores at its disposal while GEANT4 just has one; this is the reason for significant speed-up of MCGPU relative to GEANT4. The peak-location percent

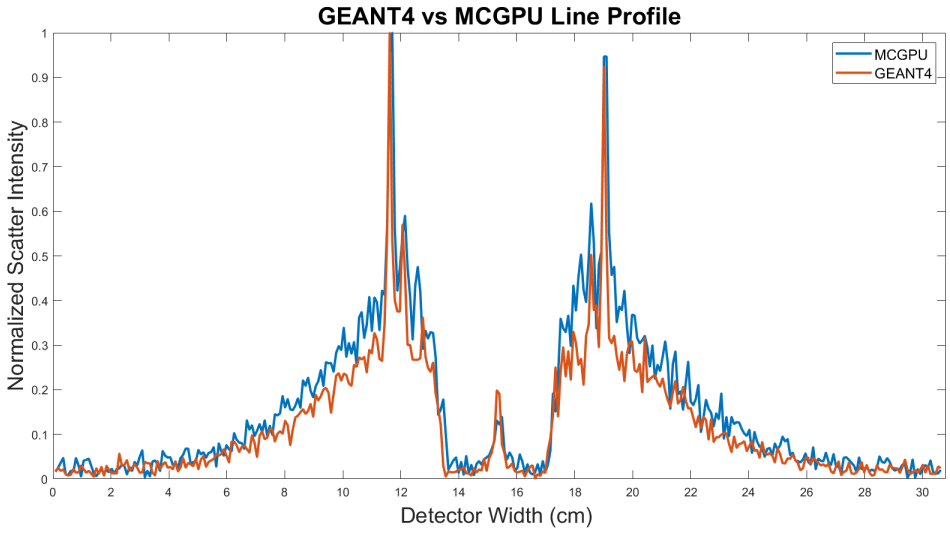
difference, cross-correlation results, and computer specifications are depicted in Figure 6, and Tables 2 and 5.



(a)



(b)



(c)

Figure 6:GEANT4 vs MCGPU 2D Scatter (top) and Line Profile Comparison (bottom) for a scan of graphite

Table 2: GEANT4 vs MCGPU Computer Specifications

	Geant4	MCGPU
Clock Rate	2.8 Ghz	1911 Mhz
RAM	12.2 Gb	15.6 Gb
Operating System	Ubuntu 18.04.3 LTS	CentOS Linux release 7.8.2003
# of Cores	1 CPU core	3840 GPU cores
# of Histories	1E8	1E8
Run Time (s)	8400	2.32

4. Model Validation against experiment

4.1 Description of Experimental Systems

Pencil Beam System

The pencil-beam system modeled in MCGPU and GEANT4 contained a 310 x 150 mm flat-panel energy-sensitive detector (out-of-the-box MCGPU is energy-integrating but was modified to only accept photons of energies 57 – 63 keV to match the energy range of the experimental data), a pixel pitch of 0.067 cm in the horizontal x-direction and 0.088 in the vertical z-direction, a 125 kVp polyenergetic X-ray source, and a 3mm thick object that was 1.3 mm in length along the axis from source to detector. The corresponding experimental pencil-beam system contained an X-ray source (Comet

MXR-225HP/11) with a tungsten anode and emitted a Bremsstrahlung spectrum at 125 kVp (shown in Figure 7) with an exposure time of 1-12 seconds per detector step and a current of 11.25 mA. The detector (Multix ME100) is an energy-sensitive linear array and moves orthogonally to the stationary pencil beam source and collects scatter signal from the stationary object. This experimental system also included a beam block to terminate the primary X-ray beam before striking the detector, which was modeled in the simulation as a virtual termination of primary X-ray photons. The schematics for the experimental pencil-beam system is shown in Figure 8.

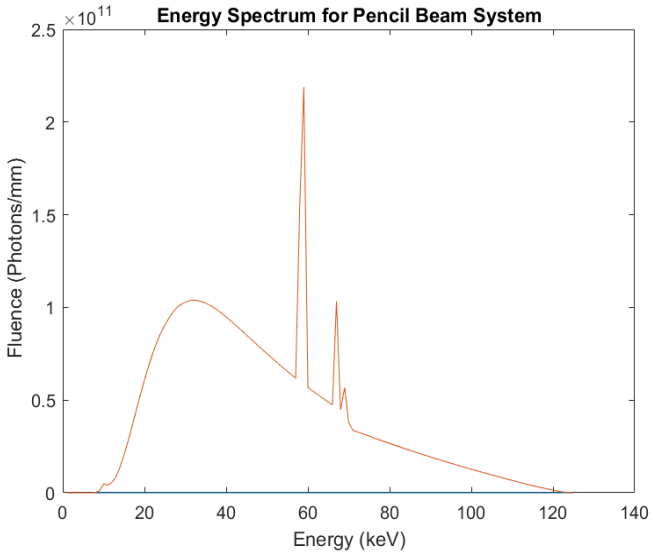


Figure 7: Pencil Beam System Energy Spectrum

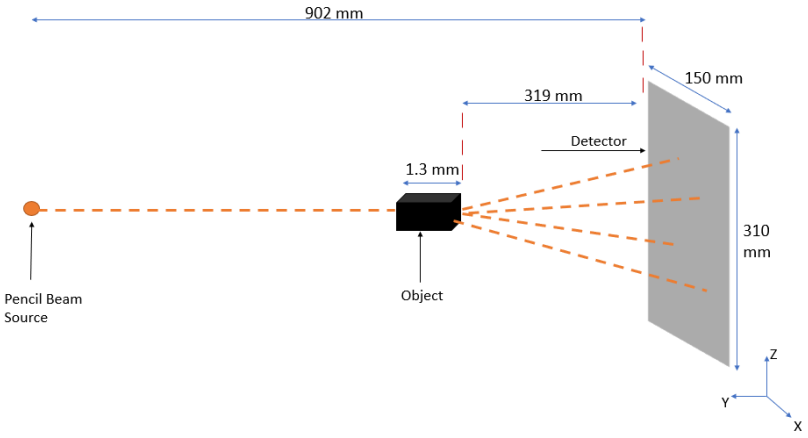


Figure 8: Pencil Beam System Configuration (Side-View)

Fan-beam system

The fan-beam system in MCGPU consists of a 286 x 100 mm flat-panel energy integrating detector, a pixel-pitch of .05 in both the horizontal and vertical directions, a 160 kVp polyenergetic X-ray source (shown in Figure 9), and an object that is 4 mm in length, height, and width. Also included in the fan-beam system in MCGPU is a 260.5 x 148 mm copper coded aperture with a pixel pitch of .125mm in both directions. The coded aperture spatially modulates the scatter signal arriving at the detector such that the signal arriving at a detector can be demultiplexed and isolated to a point of origin within the object [22]. The corresponding experimental fan-beam included a fan beam source (Spellman Monoblock XRB-X411), object, coded aperture (identical to the one modeled in MCGPU), and detector (Teledyne XINEOS 2239). The objects were ~3-5 mm in length, width, and height. The source was a tungsten anode source emitting a

bremsstrahlung spectrum at 160 kVp and 45 mAs with .05mm Hafnium filtering. A beam block was included in the experimental system to terminate the primary X-ray beam, which was modeled virtually in MCGPU.

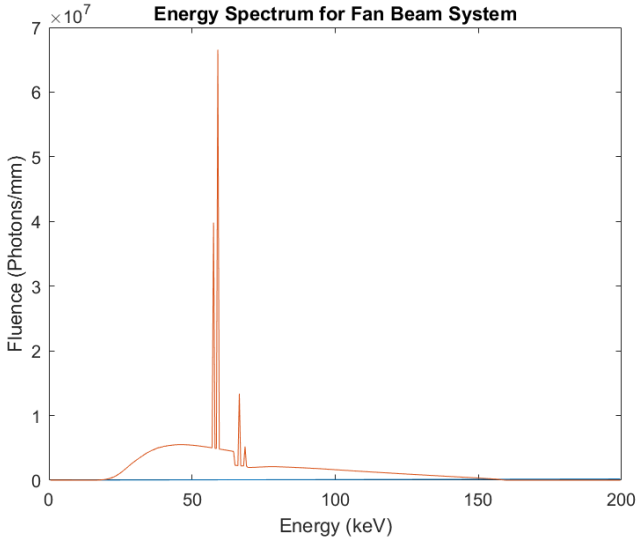


Figure 9: Energy Spectrum for the Fan-Beam System

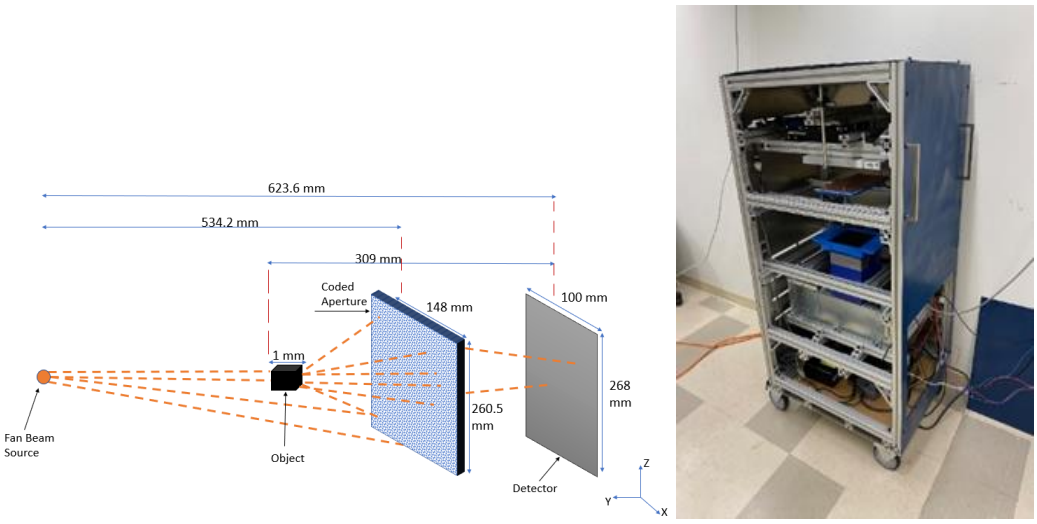


Figure 10: Experimental Fan-Beam System Schematic (Side-view) and Photograph

4.2 Reconstruction

The reconstruction model accounts for source, geometry, and energy and demultiplexes their effects from the acquired signal to recover the true form factor signal. The reconstruction model incorporated known parameters from simulation or experiment such as X-ray energy, Compton/Rayleigh scatter ratio, and geometry and used these inputs to create a forward matrix that was multiplied by the raw signal to recover the true signal (form factor curve) [4] by conducting a maximum likelihood estimation search.

4.3 Pencil Beam Results

Figures 11 a,b,c, and d, represent 2D scatter generated from MCGPU and the experimental configuration which were modeled after the schematic in Figure 10. Figures 11a and 11b are the simulated and experimental 2D scatter that is detected when scanning a vial of powdered graphite, while Figure 11a and 11b represent the simulated and experimental 2D scatter that is detected while scanning water in a plastic PLA object (both objects were 3 x 1.3 x 3 mm). Each of the figures have units of physical length on the x-axis and y-axis. The 2D scatter resulting from the simulated graphite scan differs from the 2D scatter resulting from the experimental scan, primarily because detector noise is not modeled in MCGPU. Also note, the size of the experimental 2D scatter data for the scan of water is different than the simulation because of experimental post-processing because the experimental acquisition was different than what is explicitly modeled in MCGPU (modeling 2D detector as opposed to 1D linear array that collects scatter data). This is accounted for by scaling the matrix to the appropriate aspect ratio such that the physical dimensions correspond to the pixel dimensions in MATLAB. Following this post-processing step it can be seen in Figures 11c/d, the detected 2D scatter distribution from the simulated water scan has a scatter distribution representative of what is seen in the experimental 2D scatter data for water. The form factor curves in Figure 12 were generated from the 2D scatter distributions by extracting

a radial line profile from detector center to edge and using the reconstruction model for the system configuration to reconstruct and obtain the true form factor signal at that location. Cross-correlation and peak form factor location from the from factor curves in Figure 12 are reported in Table 3 for the form factor curves. The ring locations and scatter intensity fall-off from recovered 2D scatter distributions from scans of graphite and water as well as the MSE/CC for graphite (MSE = 0.025 /CC = 0.84) and water (MSE = 0.009/CC = 0.95) illustrate MCGPU's ability to model a real-world experimental XRD system.

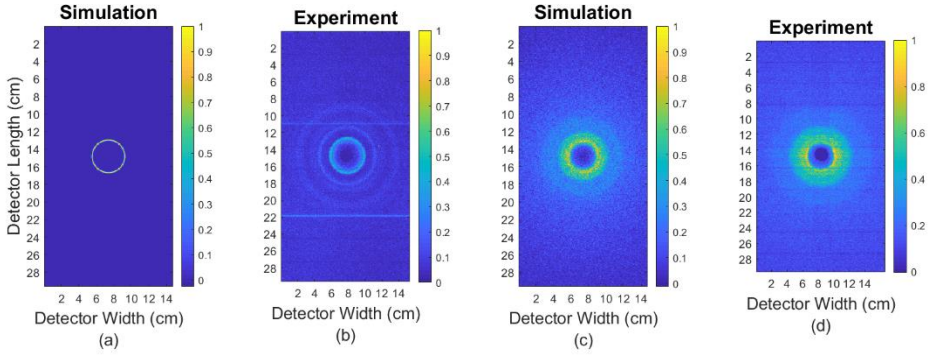


Figure 11**: Pencil-Beam System Validation of MCGPU Simulation against Experiment for graphite (Figures 11a,11b) and water (Figures 11c,11d). (recovered 2D scatter distributions)**

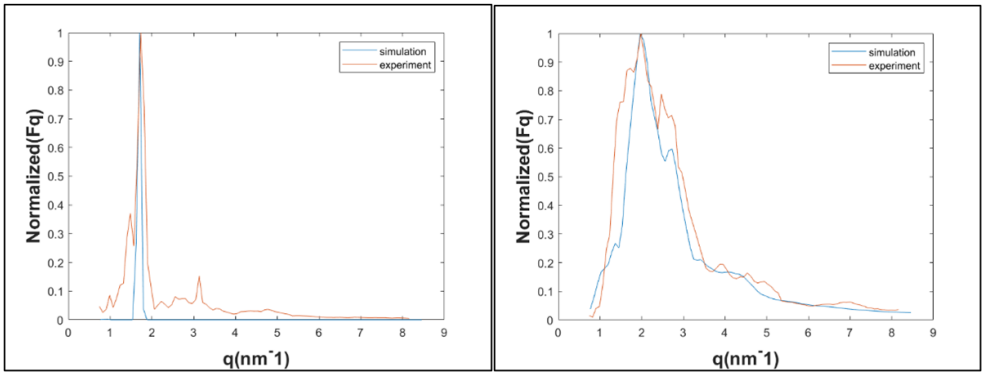


Figure 12: Pencil-Beam System Validation: Reconstructed Form factor for Graphite (left) and Water (right)

4.4 Fan Beam Results

Fan-Beam vs MCGPU

Figures 13 and 14 represent 2D scatter generated from MCGPU and the experimental configuration which were modeled after the schematic in Figure 10. Figure 13a and 13b represent the simulated and experimental 2D scatter that is detected when scanning a vial of powdered graphite, while Figure 13c and 13d represent the simulated and experimental 2D scatter that is detected while scanning water in a plastic PLA object (both objects were 1 x1 x 1 mm^3). Each of the figures have units of physical length on the x-axis and y-axis which represent the detector width and length, respectively. The 2D scatter resulting from the simulated graphite scan differs from the 2D scatter resulting

from the experimental scan, primarily because detector noise is not modeled in MCGPU, and the coherent scatter signal is more prominent in the simulation than the real world. Figures 14a and 14b are reconstructed form factor curves that were generated from the 2D scatter profiles shown in Figure 13. These form factor curves were generated using the reconstruction model for this experimental system. The reconstruction model for this system takes the entire 2D scatter matrix as input and the user can extract form factor data along the pencil beam. The recovered 2D scatter distributions from scans of graphite had ring patterns from coherently scattered photons in the same location and exhibited MSE/CC values of 0.004/0.93. The recovered 2D scatter distributions from scans of water had a similar textured pattern from the coded aperture in the background, and similar scatter fall-off intensity on the detector; the scan of water exhibited MSE/CC values of 0.037/0.88. These results demonstrate the capability of MCGPU to model a complex fan-beam coded aperture XRD system while retaining model fidelity.

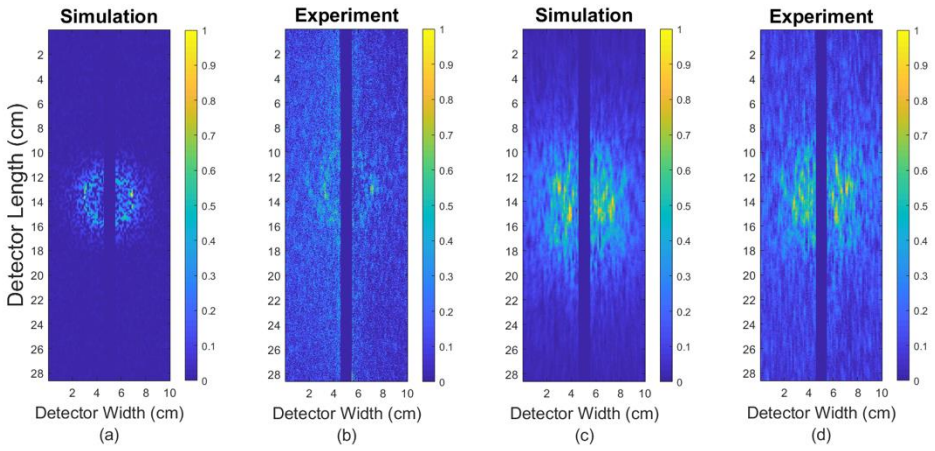


Figure 13: Fan-Beam Validation of MCGPU simulation against experiment for 2D scatter distributions for graphite (Figures 13a/13b) and water (Figures 13c/13d) (recovered 2D scatter distributions).

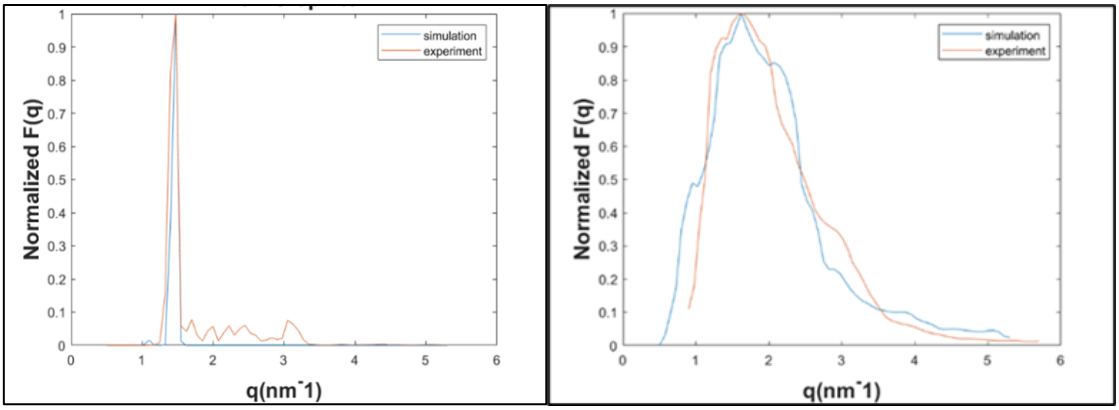


Figure 14: Fan-Beam System Validation: Reconstructed Form factor for Graphite (left) and Water (right)

Table 3 lists quantitative results for unit-testing and validation: mean squared error (MSE), and cross-correlation(CC) between the respective form factor curves are reported below.

Table 3: Unit-Testing, Cross-Validation, Validation Results

Material - test/system	MSE	CC
Water - unit-testing/simple geometry	1.76E-05	0.99
Graphite - unit-testing/simple geometry	0.015	0.99
Graphite - cross-validation/pencil-beam	1.10E-03	0.88
Water - Validation/pencil-beam	0.009	0.95
Graphite - Validation/pencil-beam	0.025	0.84
Water - Validation/fan-beam	0.037	0.88
Graphite - Validation/fan-beam	0.004	0.93

5. In-Vivo Breast Cancer Detection Application

5.1 Breast Phantom Description

Cancer Detection Applications

To demonstrate the efficacy of XRD imaging systems for in-vivo breast cancer detection, we needed to place a realistic, high-resolution breast phantom in MCGPU. We computationally constructed a breast phantom [21] and lesion [20] using models developed by Sisternes et al. and Erickson et al. All parameters of the lesion

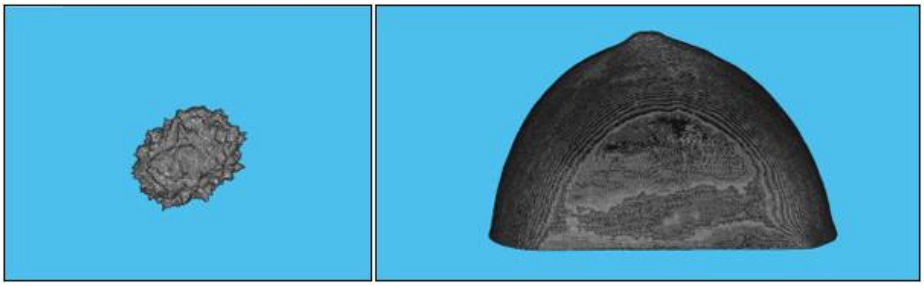
were unchanged except for the spatial resolution, which was changed to match the spatial resolution of the breast phantom into which it was computationally placed. Once the lesion was placed in the center of the breast phantom in MATLAB, the breast phantom was imported into MCGPU as a voxelized geometry. This breast phantom was scanned using two different system configurations: the fan-beam system described in the section above and a raster-scan CT configuration (described below). The most likely path to obtaining auspicious results was to start simple, then add complexity to the system; therefore, we started with the fan-beam system configuration, but only modeled a pencil-beam – then we progressed to a CT scan with a simple geometry. Details of the breast phantom used in this study as well as the pencil/fan beam and CT system scans are described below.

Description of breast phantom

Figures 15a and 15b are depictions of the mass lesion and the breast phantom, respectively. Parameters of the mass lesion such as spatial resolution, mean mass radius, and size of the mass can be readily specified in a configuration file that can be complied in Linux. The output used was a binary file that indicated whether a voxel was cancerous or background. This output file was then transferred to MATLAB and converted to a matrix data format which matched the format of the breast phantom. The

breast mass used was 368 x 368 x 368 pixels and had a total volume of 778688 mm^3. The cube was 92 x 92 x 92 mm, with a spatial resolution of 0.25 mm in each direction.

Each XCAT phantom is a 3D computational model that is based on a scan on an actual human subject. We used the provided compressed example breast phantom CTA0296, which was 716 x 297 x 416, and had a spatial resolution of .25 mm (matching spatial resolution of mass model). The breast phantom was converted from hdr format into matrix format using the MATLAB functions provided by package containing the cohorts of breast phantoms. The pixel values of the breast phantom ranged from 0 to 6. A pixel value of 0 denoted background, a pixel value of 1 denoted 100% adipose/0% fibroglandular, a pixel value of 5 denoted 100% fibroglandular/0% adipose, and a pixel value of 6 denoted normal tissue, where normal tissue was defined as a mix of adipose and fibroglandular. All intermediate classes represented a mixture of fibroglandular and adipose, but for simplicity, classes 1-3 were modified in MCGPU to contain only adipose while classes 4 and 5 were modified in MCGPU to contain only fibroglandular tissue. Class 6 was assigned normal tissue and class 0 was assigned air. When the lesion was computationally inserted into the breast phantom, the pixel values were assigned a value of 10. All the pixel values that were assigned to the breast phantom in MATLAB did not correspond to anything physically; they were used to assign density values and form factor information for the MCGPU simulation.



(a) (b)

Figure 15: Breast Lesion (Left) and Breast Phantom (right) used in MCGPU

The mass model was first compiled in Linux, converted to a matrix format, and computationally inserted in the breast phantom using MATLAB. This was done by determining the center of the breast phantom, then reading the mass lesion matrix and replacing the regular tissue with the cancerous tissue in a sequential fashion. The resulting breast phantom is depicted in Figure 16 below which is a screengrab of the sagittal, axial, and coronal slices of the center of the breast phantom following placement of the lesion in the phantom. Note that the bright white region corresponds to the cancerous region while all other regions correspond to adipose/fibroglandular tissue. This image is simply displaying the pixel values assigned to the breast phantom and does not represent the physical density of the tissues.

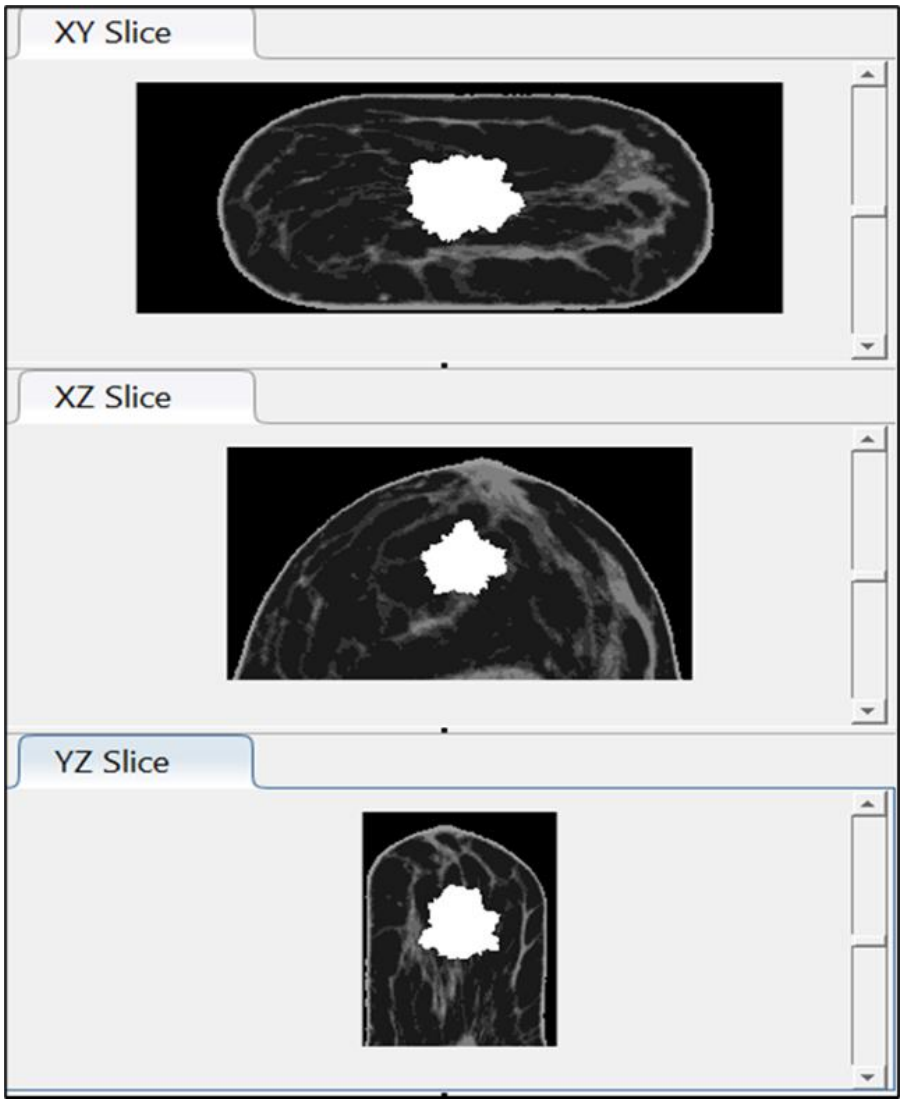


Figure 16: Depiction of Breast Lesion boundary inside voxelized Breast Phantom

5.2 Preliminary Results

5.2.1 Medical Tissue Simulations in MCGPU

Before moving on to in-vivo breast imaging applications, we first evaluated MCGPU's ability to generate form factor spectra that match empirical form factor spectra from tissues that are composed in the breast such as adipose, fibroglandular, and cancer. Figures 17 and 18 show comparisons of empirically measured and simulated form factor spectra for cancer, adipose, and fibroglandular tissue. The three plots in Figure 17 were completed using the same simplistic geometry that was used in the unit-testing stage. Figure 18b is simply a superposition of the recovered form factor curves from the simulation while Figure 18a contains the empirical biological form factor tissues from Kidane. As shown in Figure 18, the three tissue signals generated from MCGPU match the empirical form factor tissue signals. Cross-correlation and peak form factor location percent difference between the simulated and empirical data are listed in Table 4. From q values of ~1-1.25, the cancer signal relative to the fibroglandular and adipose signal is much less in both simulated and empirical spectra. From q-values of 1.5 to 3.5, the cancer signal is slightly higher than the fibroglandular signal, but they are both significantly higher than the adipose signal.

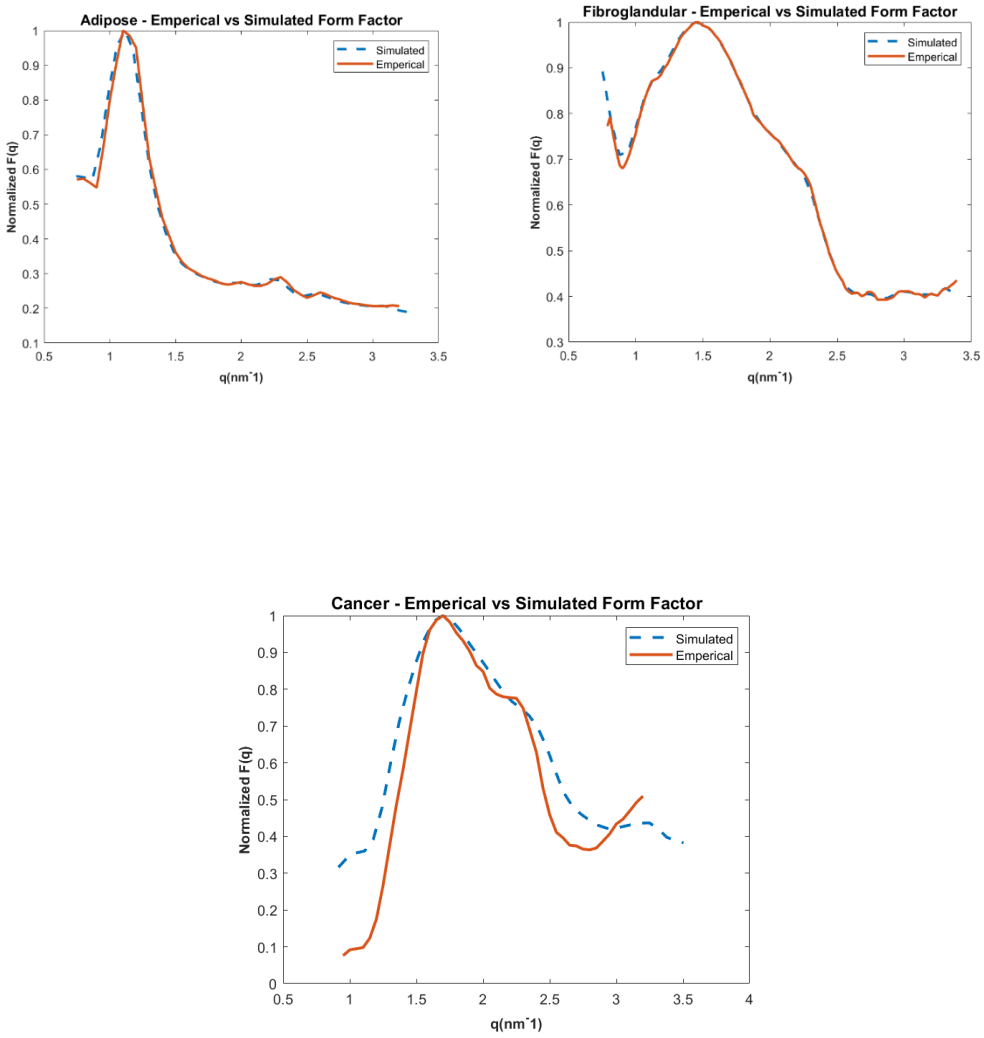


Figure 17: Adipose, Cancer, and Fibroglandular Reconstructed form factors for simple geometry unit-testing [5]

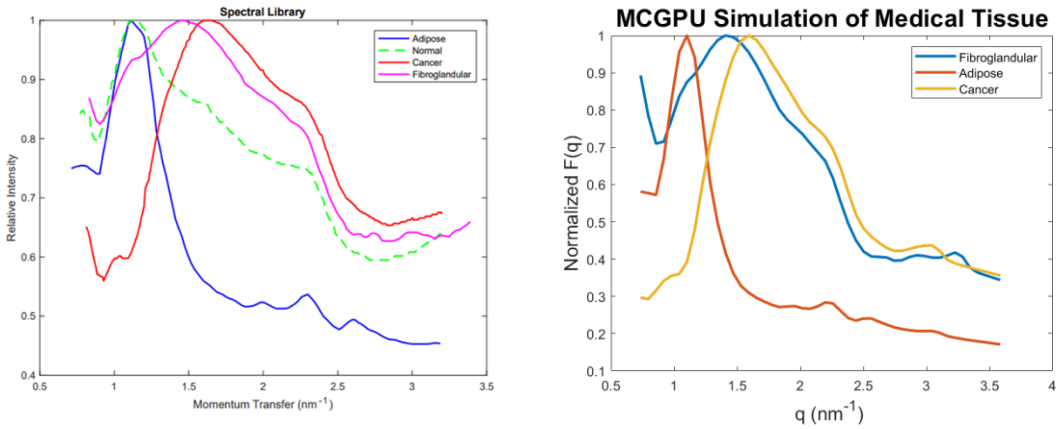


Figure 18: Empirical (left) [3] and MCGPU Biological Tissue Form Factors (right)

Table 4: Peak Location Percent Difference and Cross-Correlation

Material	%Difference	Cross-Correlation
Cancer	<0.1%	0.95
Adipose	<0.1%	0.997
Fibro	<0.1%	0.999

As mentioned before, the ability to differentiate a material or a tissue based on its relative scatter intensity as a function of momentum transfer, renders XRD a suitable imaging modality for tissue differentiation and cancer detection. Figure 19 demonstrates MCGPU can be used for XRD imaging of medical tissues: each of the tissues' signal is distinct and matches its empirical counterpart, and thus can be used for tissue delineation. The breast described in section 5.1 is composed of the three tissues in the MCGPU simulation. In MCGPU, each of the voxels were assigned a value of fibroglandular, adipose, cancer, or air. The corresponding material definition in MCGPU

was then assigned to each voxel. We used two distinct imaging acquisitions for the in-vivo breast scans. First, we used the experimental fan-beam system which was used for model validation. Next, we modified MCGPU from its basic pencil-beam CT scan capability to allow for raster-scan CT capability. The fan-beam acquisition geometry has been explained in previous sections; however, the CT acquisition and reconstruction varies from conventional CT acquisition and reconstruction and should be elucidated.

5.2.2 XRD-CT

Conventional CT scans involve complex acquisition geometries and rigorous reconstruction algorithms that involve using the measured signal obtained during the acquisition to obtain an image of the scanned object [2]. Conventional CT scans are ostensibly convoluted and difficult to understand but can be well understood by simplifying the concept to a few guiding principles. A CT scan can be thought of as taking multiple views of an object such that the observer can obtain more information about the object by leveraging these different viewpoints or perspectives. Essentially, all a CT scan is, is multiple X-ray projections at different angles. This is the fundamental difference between CT and conventional radiography [2]. In addition, radiography does not require reconstruction, since only one projection is taken during the scan.

Conventional CT-scans are typically reconstructed using filtered-back projection (FBP) (although recently algorithms such as iterative reconstruction and arithmetic

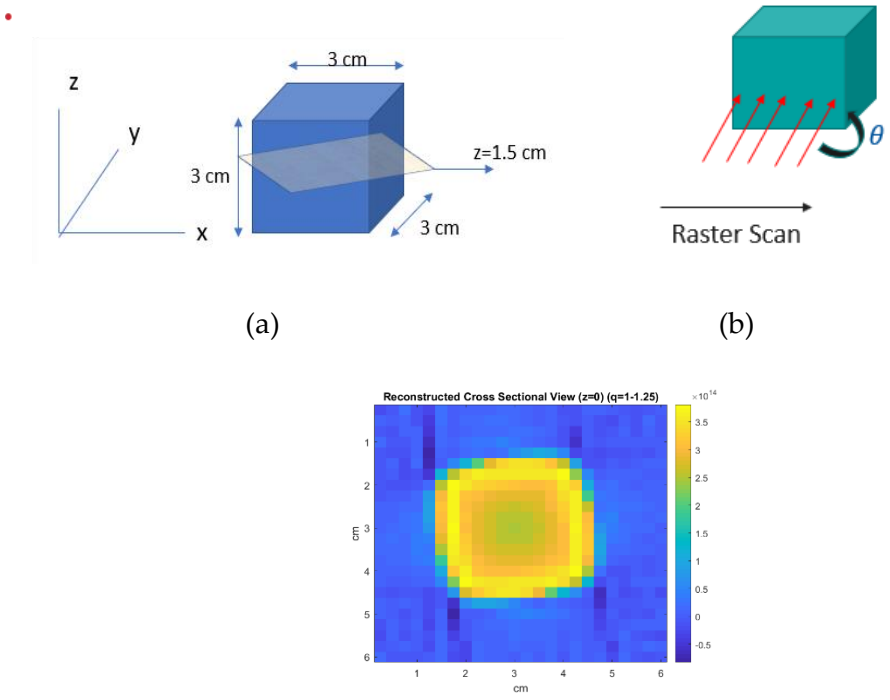
reconstruction are also utilized). Filtered back-projection is a reconstruction process that back-projects each detected transmitted signal intensity at its projection angle, to get a best estimate of the object that was scan. For FBP, a sinogram is provided to the FBP algorithm before it can reconstruct the data to get the true object or image. A sinogram is conventionally a 2D matrix that holds projection data at each angle. In our scenario, each projection slice represents the total coherently scattered photon energy generated from each position in the raster scan at a given projection angle. A useful conceptual framework is to think about a roundtable discussion: each person at the roundtable has a different/useful view, and when those perspectives are combined, one can get an enhanced view of the topic of interest. The 'filtered' portion of the name in filtered back projection corresponds to using a computational filter that filters out high-frequency data to eliminate noise from the image.

In X-Ray diffraction computed tomography (XRD-CT), the information each projection slice holds are different; instead of back-projecting the transmitted signal, the detected *coherent scatter* signal emanating from an object in the line of sight of each raster scan is back-projected to complete the reconstruction [19]. In our roundtable analogy, each persons' perspective is the same, but each person contains different information – or had a different method of obtaining their information or perspective. The detected coherent scatter signal can be energy- or angle-discriminating to determine a given q or

momentum transfer value that one is reconstructing, as seen by equation 3. For our application, we can see in Figure 19 that the q-value range where the cancer signal is markedly different from the other signals is from momentum transfer values of 1 to 1.25. This range of q-values corresponds to a minimum scatter angle of 1 degree to a maximal scatter angle of 24 degrees, and a minimum energy of 5.964 keV to a maximum energy of 71.05 keV. The source code in MCGPU was modified to only allow coherently scattered photons of this energy and angular range to be detected.

Figure 19 depicts the results of the technique described above to a simple acquisition geometry. We tested the modifications we made in MCGPU that enabled a raster-scan CT for XRD, to determine whether this approach could be used to detect a slice of a cube of cancer. The cube of cancer was 3x3x3 cm^3 and was placed in the center of the MCGPU simulation world. We used 30 projections, each 12 degrees apart for the CT scan. We used the iradon function in MATLAB with the 'Ram-Lak' filter option to reconstruct the sinogram. Figure 20 depicts the results from the sinogram on the left and the reconstructed slice corresponding to the center of the cube on the right. The bright yellow signal corresponds to the location of the midplane slice in the cube, where highest amount of coherently scattered photon energy is detected, and thus the highest amount of coherent scattered photon events occur (energy is conserved in coherent scattering, so energy is proportional to number of events). The blue or lower intensity

regions of the sinogram correspond to regions where there is little coherent scattering occurring; this matches our geometry since there was no material placed outside of the cube of cancer. These results show that the technology for an XRD-CT scan in MCGPU is working and can be applied to in-vivo breast imaging.



(a) (b)

(c)

Figure 19**: CT results - Simple Geometry: 19a and 19b show geometry, 19c is reconstructed cross-sectional**

5.3 In-vivo applications

For the in-vivo breast imaging application, we completed two XRD scans: one with the experimental fan-beam described in a previous chapter, the other was a CT XRD scan. Prior to performing both scans in MCGPU, we computationally inserted a tumor inside the 3D virtual breast phantom and subsequently placed the breast phantom inside the MCGPU simulation world. We then used the simulated fan-beam geometry (described in the validation section) to scan the breast phantom; however, we only modeled a pencil-beam in this simulation in an effort to simplify the acquisition geometry. The breast was placed such that the direction of compression was along the direction of the pencil beam for both imaging acquisitions. Figures 20 and 21 below depict slices of the breast phantom at two separate locations along the source to generator path and corresponding to the reconstructed form factors at those locations (object-to-detector distance of 280 mm and 298 mm). The plot of the form factor spectra on the right of the axial slices represents reconstructed and ground truth form factor spectra. The bright yellow region in the center of the breast phantom corresponds to cancerous tissue.

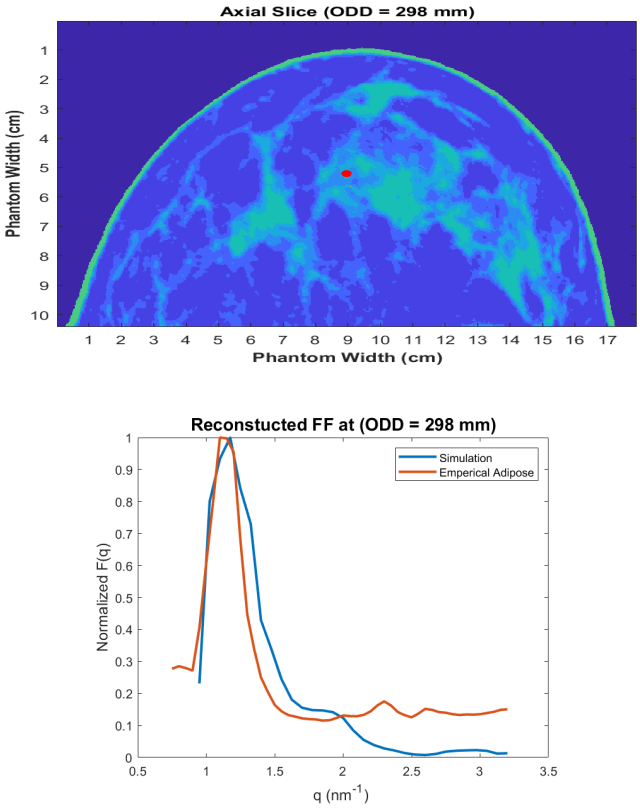
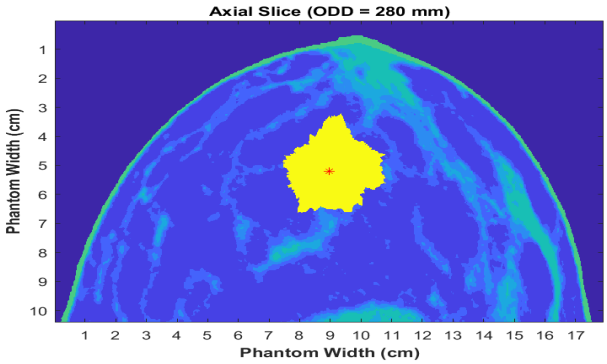


Figure 20: Axial Slice of Breast Phantom (top) and reconstructed cancer form factor (bottom)



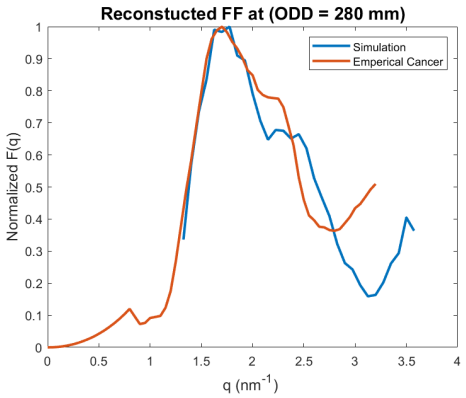


Figure 21:Axial Slice of Breast Phantom (top) and reconstructed cancer form factor (bottom)

Table 5 lists the Cross-Correlation, Mean absolute difference, and percent difference for three different comparisons. The first row of data corresponds to a comparison between the reconstructed form factor at an ODD of 298, and the expected empirical tissue (adipose). The second row corresponds to the reconstructed form factor at an ODD of 280 and the expected empirical tissue at that location (cancer).

Table 5: Ground Truth and Reconstructed Form Factor metrics

	Correlation	Mean Absolute Difference (nm^{-1})	Max Peak Location(%diff)
Location 1- Adipose	0.87	0.22	2.15%
Location 1-Cancer	0.78	0.33	< 0.1%

6. Discussion/Conclusion

MCGPU Validation

We have implemented and validated MCGPU as an XRD simulator and demonstrated its ability to model relevant XRD imaging systems. Following implementation of empirical MIF form factors into the MCGPU framework, unit testing was completed to ensure that MCGPU could accurately model input form factor spectra by comparing the input form factor spectrum to the recovered spectra for graphite and water. Next, cross-validation of MCGPU against GEANT4 was completed by comparing detected 2D scatter and extracting a line profile for graphite and water across both simulation platforms. Finally, two experimental XRD systems were modeled in MCGPU and compared using the detected scatter (Figure 7). Completion of these tasks was motivated primarily by the desire to evaluate the speed-up offered by MCGPU over GEANT4, and more importantly to assess MCGPU's ability to obtain physically realistic results that matched experimental data.

In-vivo-breast imaging

Following multiple validations of the MCGPU model against other software (GEANT4) and other experimental systems (pencil-beam and fan-beam system), MCGPU was used to model two XRD imaging systems to determine whether MCGPU

could be used for tissue differentiation for in-vivo breast imaging applications. Although it has been shown that XRD imaging could be used for tissue delineation [5], the added complexity of an in-vivo breast scan allows for one to more accurately ascertain whether XRD imaging can be used for tissue differentiation. The first experimental system/set-up used was the fan-beam system geometry that MCGPU was previously validated against, but this time we used a pencil-beam instead of fan-beam. This system included a coded aperture, which is capable of spatially modulating the signal before it arrives at the detector such that the multiplexed signal at the detector can be demultiplexed to retrieve the location along the source to detector path from which photons are emanating.

With this in mind, we strategically reconstructed the form factor signal at two locations along the pencil beam where the tissue was known: one where the tumor existed, and another where there was just regular tissue. This was done to see if distinguishable form factors could be reconstructed at the separate axial locations where we know the signal should be different. Figures 21 and 22 show that reconstructed form factor signals align with empirical form factor of the expected tissue at each location. These results were quantified by observing the cross-correlation, mean difference, and max peak location for the a) expected and reconstructed form factor located at a object-to-detector distance of 298 mm b) expected and reconstructed form factor located at a

object-to-detector distance of 280 mm and c) the reconstructed form factor spectra at locations of 298 and 280 mm. The reconstructed form factor spectra at locations of 298 and 280 mm had a cross-correlation values of 0.87 and 0.78, which demonstrates that the two form factor spectra at the different locations along the pencil beam could be matched to their ground truth form factor curves.

Here, we only modeled a pencil-beam source but the experimental system allows for a fan-beam source: this means that one could get volumetric information about the breast by doing a single scan with a fan-beam. Figures 20, 21 and Table 5 demonstrate the proof of concept; one is able to recover form factor signals that can be differentiated along the beam – this means that for a fan beam scan, one could get depth resolution at each location in the fan beam, and with translation of either the object or source and detector, one can get lateral resolution.

The CT results also demonstrate that volumetric information can be obtained using XRD and can be subsequently applied to in-vivo breast scans for tissue differentiation. As discussed in the previous chapter, one can select an angle and energy to select a range of q values to be used to exploit the signature difference between cancer and surrounding tissues. From Figure 18, one can see that the range of q-values that are most useful in differentiating cancer from another tissue is ~1-1.25.

Conclusion

We implemented form factors that incorporate molecular interference into MCGPU's material database, validated MCGPU's ability to match simulated output relative to theoretical input, demonstrated significant speed-up in MCGPU simulations without loss of fidelity compared to GEANT4, and showcased MCGPU's ability to model more complex XRD experimental systems. This work validates MCGPU and demonstrates its utility as a viable and rapid simulation toolkit capable of modeling XRD-based imaging systems that can scan a wide class of materials.

Following these steps of validations, we then used a realistic 3D anthropomorphic breast phantom for our in-vivo breast application. A breast tumor was computationally inserted into the breast phantom and input into the MCGPU simulation world, such that in-vivo breast scans could be performed. Results demonstrated that XRD imaging can be used to differentiate biological tissues within the breast phantom. Additionally, the results provided proof-of-concept through the a) pencil-beam XRD system and b) XRD-CT system, that volumetric imaging using XRD could be utilized to differentiate between biological tissue in the breast on a **pixel-by-pixel basis**.

Although these results do provide insight into how one can use XRD imaging to differentiate between tissue on a pixel-by-pixel basis, there are still many questions that need to be answered before XRD imaging can be clinically implemented for volumetric breast imaging such as reproducibility of results, statistical uncertainty, and more

quantitative metrics to ensure a reconstructed form factor signal can be definitively differentiated from another tissue within the breast.

7. Limitations

Although MCGPU has been validated as a rapid monte Carlo simulator that can be used for material classification and delineation, some tricky aspects of MCGPU modelling should be noted. First, the outcome of the MCGPU simulation is dependent on the accuracy of the input form factor used to create the material file in MCGPU, since the input form factor is the probability distribution that is sampled to determine the scatter direction. Parameters such as the number of data points, the distance between data points (in momentum transfer), and the relative amplitude of the background radiation should be selected meticulously to procure optimal results. Second, inherent differences between experimental and simulated results will occur because of set-up uncertainty and geometry mismatches. MCGPU is not impervious to discrepancies or mismatches; however, results demonstrate when pre-processing, post-processing, and input data acquisition are performed meticulously, discrepancies and mismatches can be mitigated satisfactorily, and this validated MCGPU toolkit can be used to inform us about future XRD imaging research endeavors.

Currently, the in-vivo breast cancer detection results are lacking in quantitative data: many imaging metrics such as image quality and dose need to be assessed before ascertaining that XRD imaging holds advantage over conventional transmission-based imaging. Both the pencil-beam coded aperture results and the XRD-CT results should be improved to obtain volumetric information about the breast phantom. The pencil-beam system can be modified to a scanning fan-beam system, and the XRD-CT scanning technology established in MCGPU should be applied to a realistic breast phantom.

8. Future Work

Now that MCGPU has been validated against realistic experimental XRD systems, MCGPU can be used for a wide variety of XRD-based medical imaging applications. MCGPU can be used for system design and testing, generating data for algorithm development and training, and can be used to model more complex systems or simulate more complex scans because of its demonstrated ability to perform rapid simulations without a loss of fidelity.

In-vivo breast cancer detection is a starting point. In this study, we were able to use a fan-beam system (although we only modeled a pencil beam) containing a coded-aperture to scan an in-vivo breast phantom; this provided depth resolution from the source to detector direction without the need for tomographic rotation. However, more rigorous analysis and quantitative metrics are necessary to determine whether the

measured signal from XRD imaging can truly be used to delineate between cancerous and normal tissues.

The next steps for the MCGPU simulator are to progress to a combination of the fan-beam coded aperture system and a pencil-beam CT scan of a breast phantom. As it stands, MCGPU has been modified to model either the fan-beam coded aperture system or a pencil-beam raster scan CT. A combination of these two would provide complete volumetric information of the breast and make a real-world application more feasible.

Appendix A

A.1 MCGPU Modification: Sampling Parameter NP

To change the random sampling parameter to allow for finer form factor sampling, the parameter 'NP' should be changed in the files 'penelope.f' and 'MC-GPU_create_material_data.f' in the penelope folder. This is accomplished by changing line 5539 in 'penelope.f' under 'SUBROUTINE GRAaI' to

"PARAMETER (NP = 511)"

and changing line 300 in 'MC-GPU_create_material_data.f' under "SUBROUTINE GRAaI_linear_energy to

"PARAMETER (NP = 511)".

The 'MC_GPU_create_material_data.f'and 'penelope.f' files can be compiled using the command line: 'gfortran -O MC-GPU_create_material_data.f penelope.f -o MC-GPU_create_material_data.x'

Note that the default number of samples in the grid was 128 and the maximum number of samples is set at 511 in the MCGPU source codes. Modifying this maximum number has caused issues in the past, so one should proceed with caution if trying to supersede this number.

A.2 MCGPU Modification: Pencil Beam System: 2D Energy Discriminating Detector

Out-of-the-box, MCGPU does not have energy-discriminating capabilities. Any time a particle strikes a detector, the total energy of that particle is counted, regardless of the energy range. However, during the validation stage, comparisons were made against an energy-sensitive detector; therefore, the following modification was made in the MCGPU kernel to only allow photons of certain energies to strike the detector. In line 539 and 583 of 'MC-GPU_kernel_v1.3.cu'

"if ((pixel_coord_z>-1))&&((pixel_coord_z<detector_data_SHARED->num_pixels.y)&&(*energy>Lbound)&&(*energy<=Ubound))"

Where Lbound and Ubound are energy units in eV and are specified for the energy range you want to consider. This code was compiled with the following command:

"nvcc -g -DUSING_CUDA MC-GPU_v1.3.cu -o MC-GPU_v1.3.x -O3 -use_fast_math -L/usr/lib -I. -I/usr/local/cuda/include -I/usr/local/cuda/samples/common/inc -I/usr/local/cuda/samples/shared/inc/ -lz --ptxas-options=-v".

A.3 MCGPU Modification: Inclusion of Coded Aperture

To match the experimental fan-beam system that included a coded aperture, numerous edits to the source, kernel, and header files were added. First, a structure for

the coded aperture that would contain the aperture data was defined on a new line – line 447:

"struct aperture_struct aperture_data;"

Next on new lines (466, 467, and 470), variables were defined as:

"unsigned long long int **aperture_ptr;" on line 466 and

"unsigned char *aperture_final;" on line 467, and

"int aperture_bytes; on line 470;"

Next, on line 620, 625, and 626 we open the file containing the aperture data, allocate memory, and copy the data to MCGPU:

"myFile = fopen("Aperture.bin","r");" on line 620,

"aperture_final = (unsigned char*) malloc(2342416*sizeof(unsigned char));" on line 621 and

"fread(aperture_final, sizeof(unsigned char), 2342416, myFile);"

To create a pointer to the device global memory so that the GPU can access the aperture data, we create a device structure that will be used to copy the aperture data to the GPU during the simulation. This is done by creating two new lines on 777 and 778 with the other pointers to device global memory:

"struct aperture_struct *aperture_data_device = NULL;"

"unsigned char *aperture_values_device = NULL;"

Next, these devices are passed by reference in the function “init_CUDA_device” as &aperture_values_device and &aperture_data_device. Within the function “init_CUDA_device” the local data is copied to the device data using GPU functions. First, within the function modify line 2865 to include the size of the aperture:

“double total_mem = sizeof(struct voxel_struct) + sizeof(struct source_struct) + sizeof(detector_struct) + sizeof(struct linear_interp) + sizeof(struct aperture_struct) + 6*sizeof(short int)”

Next, make new lines on 2889 and 2090 to use the cudaMalloc and checkCudaErrors functions to allocate the correct amount of memory for the coded aperture like so:

“checkCudaErrors(cudaMalloc((void**) aperture_data_device, sizeof(struct aperture_struct)));”

And on the following line

“checkCudaErrors(cudaMalloc((void**) aperture_values_device, 2342416)));”

Next, modify line 2900 in a similar fashion as line 2865 to include “sizeof(struct aperture_struct”. Then, on line 2900 in the if statement, include. “*aperture_data_device ==NULL.” Now, on line 2920 use the CudaMemcpy functions to copy the aperture data to the device data:

“checkCudaErrors(cudaMemcpy(*aperture_data_device, aperture_data, sizeof(struct aperture_struct), cudaMemcpyHostToDevice));”

And on line 2921,

"checkCudaErrors(cudaMemcpy(*aperture_values_device, aperture_final, 2342416, ,cudaMemcpyHostToDevice));"

Now, all the aperture data will be copied to the global device variables which are accessible by the GPU during a given simulation; we just need to include these variables in the "track_particles" function that calls the computing kernel. This can be done by adding the variables "aperture_data_device" and "aperture_values_device" to the function call.

Kernel Modifications

In the global function "void track_particles", define "aperture_struct* aperture_data_array" and "unsigned char* aperture_matrix". Next, on line 184 define a structure variable to contain the SHARED data:

"aperture_struct aperture_data_SHARED"

On line 157 define a track state variable as a float3 for the position of the particle relative to the aperture as:

"float3 position_a"

Then on line 194, the global data can be copied to the shared data as

"aperture_data_SHARED = aperture_data_array[num_p]; "

Next, shared aperture data needs to be passed to the "source" function on line 223 like so:

"source(&position, &direction, &energy, &seed, &absvox,
&source_data_SHARED, &detector_data_SHARED,
&aperture_data_SHARED);"

as well as the "tally_image" function on line 389:

tally_image(&energy, &position, &position_a, &direction, &scatter_state, image,
&source_data_SHARED, &detector_data_SHARED, &aperture_data_SHARED,
aperture_matrix);"

In the "void tally_image" function definition, one should ensure that the aperture data previously added is defined properly as a function input variable. On lines 493 and 494, at the start of the "void tally_image" function, a float variable "dist_aperture" and unsigned char variable "aperture_pixel" are defined for later use. The distance from the particle to the aperture plane can be defined on line 527 as if initial source direction is not in the +Y direction (0,1,0):

"dist_aperture = (source_data_SHARED->direction.x *
(aperture_data_SHARED->A_center.x - position->x) +(source_data_SHARED-
>direction.y * (aperture_data_SHARED->A_center.y - position->y) +

(source_data_SHARED->direction.z * (aperture_data_SHARED->A_center.z - position->z)))) / cos_angle; :

Next, the particle is translated to the aperture plane in a similar fashion as its translation from the particle to the detector plane on line 542:

"position_a->x = position->x + dist_aperture * direction->x;

position_a->y = position->y + dist_aperture * direction->y;

position_a->z = position->z + dist_aperture * direction->z;"

If the initial source direction is in the +Y position the distance from the particle to the aperture can readily be computed as:

"dist_aperture = (aperture_data_SHARED->A_center.y - position->y)/(direction->y);"

Next, on line 607 the x-position the particle strikes on the aperture can be determined by the following:

"nt pixel_coord_xa = __float2int_rd(((position->x + dist_aperture*direction->x - aperture_data_SHARED>A_corner_min_rotated_to_Y.x)*aperture_data_SHARED>A_pixel_size_x)-2.96);"

Where 2.96 accounts for a geometric discrepancy in the experimental system. Likewise, on line 611, the z-position where the particle strikes the aperture can be determined by:

```
"int pixel_coord_za = __float2int_rd(((position->z + dist_aperture*direction->z -
aperture_data_SHARED-
>A_corner_min_rotated_to_Y.z)*aperture_data_SHARED-
>A_pixel_size_z)+1.6);"
```

Where again the 1.6 accounts for a geometric discrepancy in the experimental system. To determine the pixel value of where the photon strikes the detector, the following lines of code can be deployed:

```
"if ((pixel_coord_xa >-1) && (pixel_coord_za>-1) &&
(pixel_coord_xa<aperture_data_SHARED->A_num_pixels.x) &&
(pixel_coord_za<aperture_data_SHARED->A_num_pixels.y)){
        aperture_pixel = aperture_matrix[pixel_coord_za
pixel_coord_xa*(aperture_data_SHARED->A_num_pixels.y)];}
else{
        aperture_pixel = 1;
}"
```

Finally, the photon will be attenuated or transmitted based on the binary value of the pixel of the aperture: on line 630, the photon is transmitted if it strikes a pixel on the coded aperture with value of 1:

"if ((pixel_coord_z>-1)&&(pixel_coord_z<detector_data_SHARED->num_pixels.y) && (aperture_pixel == 1))"

And is attenuated if it strikes a pixel value on the coded aperture with a value of 0 by modifying lines 636-641:

```
"else if((pixel_coord_z>-1)&&(pixel_coord_z<detector_data_SHARED->num_pixels.y) && (aperture_pixel == 0))
    atomicAdd( ( image +
            (int)(*scatter_state) * detector_data_SHARED->total_num_pixels +
            (pixel_coord_x + pixel_coord_z*(detector_data_SHARED->num_pixels.x)) ),
        __float2ull_rn((*energy)*SCALE_eV*.0139) );
}"
```

Header File

The main edit in the header file is to define a structure defining the variables for the coded aperture:

"aperture_struct{float sad, float3 A_corner_min_rotated_to_Y, A_center, float A_rot_inv[9], width_A, height_A, A_pixel_size_x, A_pixel_size_z, int2 A_num_pixels, int A_total_num_pixels, A_rotation_flag};"

NOTE: All modifications to functions in the source and kernel codes as well as global variables must be reflected in the header file.

A.4 MCGPU Modification: CT raster-scan

Out-of-the box, MCGPU can complete a pencil, fan, or cone beam CT-scan; however, for our application a pencil-beam raster scan was most appropriate. To accomplish this, the source code needed to be modified. First, the initial source position is initialized on line 488 as:

"src_pos = 'starting position'"

Where 'starting position' is the physical location of the source. On line 489, the 'src_pos' variable is passed by reference as a function argument. In the 'read_input' function on line 1287 a corresponding variable 'src_loc' is defined as a parameter. On line 1382 the x-position of the source is modified to be updated by the 'src_loc' variable that is passed by reference through the function:

"source_data[0].position.x = src_loc"

Next, on lines 664 and 665, the file to store the Rayleigh scattered data is opened and the variable for coherent scatter energy is defined as:

"FILE *fp = fopen("RAYLEIGH.dat","w"); on line 664 and

"float terayleigh" on line 665.

The raster scan is accomplished by doing a CT scan at each source position: this means there is a nested for loop where the inner for loop is for the CT scan and the outer for loop is for the source position. This is accomplished by the following: on line 671, the for loop for the raster scan is defined and on line 674 the spatial resolution of the CT-scan can be defined by the increment in units of cm:

"for (int i = 1; i < num_positions_in_raster_scan; i++)" on line 671 and

"src_pos = src_pos + .25" on line 674

Next, a block of code from likes 683-689 within the CT for loop is defined to accomplish the following tasks a) read the new source position b) define a new source to rotation axis distance based on new source position c) set the new CT coordinates d) use the GPU CUDA functions to allocate memory to copy the source data to the global memory such that it can be accessed in the global memory by the GPU:

```
"int pixels_per_image = detector_data[0].num_pixels.x *
detector_data[0].num_pixels.y;
read_input(argc, argv, myID, &total_histories, &seed_input, &gpu_id,
&num_threads_per_block, &histories_per_thread, detector_data, &image, &$
 float x = abs(src_pos – 1.5);// constant corresponds to center of object
 float nSROT = sqrt((SRotAxisD*SRotAxisD) + (x*x));
```

set_CT_trajectory(myID, num_projections, D_angle, angularROI_0,

angularROI_1, nSROT, source_data, detector_data, vertical_translation_per_pr$

checkCudaErrors(cudaMalloc((void**) &source_data_device,

num_projections*sizeof(struct source_struct)));

checkCudaErrors(cudaMemcpy(source_data_device, source_data,

num_projections*sizeof(struct source_struct), cudaMemcpyHostToDevice));”

References

[1] "Survival Rates for Breast Cancer," *American Cancer Society*. [Online]. Available: https://www.cancer.org/cancer/breast-cancer/understanding-a-breast-cancer-diagnosis/breast-cancer-survival-rates.html. [Accessed: 22-Mar-2021].

[2] J. T. Bushberg, J. A. Seibert, E. M. Leidholdt, J. M. Boone, and C. K. Abbey, *The essential physics of medical imaging*. Philadelphia: Wolters Kluwer, 2021.

[3] ICRP, 2002. Basic Anatomical and Physiological Data for Use in Radiological Protection Reference Values. ICRP Publication 89. Ann. ICRP 32 (3-4).

[4] J.Greenberg and K.Iniewski, [X-Ray Diffraction Imaging: Technology and Applications Devices, Circuits, and Systems, Taylor & Francis Group (2018).

[5] G. Kidane, R. Speller, G. Hanby and A. Hanby, "X-ray scatter signatures for normal and neoplastic breast tissues," Phys. Med. Biol, vol. 44, p. 1791, 1999.

[6] Badal, Andreu, and Aldo Badano. "Accelerating Monte Carlo Simulations of Photon Transport in a Voxelized Geometry Using a Massively Parallel Graphics Processing Unit." *Medical Physics*, vol. 36, no. 11, 2009, pp. 4878–4880., doi:10.1118/1.3231824.

[7] W. W. Bragg and Bragg, "The Reflection of X-rays by Crystals," Proc. R. Soc. London A, vol. 88, no. 605, p. 428–438, 1913.

[8] J. Hubbell, "Atomic form factors, incoherent scattering functions, and photon scattering cross sections," Journal of Physical and Chemical Reference Data, vol. 6, no. 615, 1977.

[9] D. E. Verghese and K. Peplow, "Measured Molecular Coherent Scattering Form Factors of Animal Tissues, Plastics and Human Breast Tissue," Physics in Medicine & Biology, vol. 43, no. 9, pp. 2431-2452, 1998.

[10] B. Ghammraoui and A. Badal, "Monte Carlo simulation of novel breast imaging modalities based on coherent x-ray scattering," Phys. Med. Biol., vol. 59, p. 3501–3516, 2014.

[11] "What Is Breast Cancer Screening?," *Centers for Disease Control and Prevention*, 14-Sep-2020. [Online]. Available: https://www.cdc.gov/cancer/breast/basic_info/screening.htm#:~:text=For%20many%20women%2C%20mammograms%20are,breast%20cancer%20for%20most%20women. [Accessed: 22-Mar-2021].

[12] S. Agostinelli, J. Allison, K., Geant4—a simulation toolkit, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Volume 506, Issue 3, 2003, Pages 250-303, ISSN 0168-9002,

[13] NEA (2019), PENELOPE 2018: A code system for Monte Carlo simulation of electron and photon transport: Workshop Proceedings, Barcelona, Spain, 28 January – 1 February 2019, OECD Publishing, Paris, https://doi.org/10.1787/32da5043-en.

[14] J. Sanders, E. Kandrot, and J. J. Dongarra, *CUDA by example: an introduction to general-purpose GPU* programming. Upper Saddle River etc.: Addison-Wesley/Pearson Education, 2015.

[15] M. Japzon, Optimization of X-Ray Diffraction Imaging of Medical Specimens by Monte Carlo Methods. M.S. thesis, Duke University (2019)

[16] Luis de Sisternes, Jovan G. Brankov,et al., "A computational model to generate simulated three-dimensional breastmasses," Medical Physics2015 42(2):1098-1118https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4320152/

[17] J. L. Prince and J. M. Links, *Medical imaging signals and systems*. Boston etc: Pearson Education, 2015.

[18] "Comprehensive data set to include interference effects in Monte Carlo models of x-ray coherent scattering inside biological tissues." *Physics in Medicine & Biology*

[19] M. N. Lakshmanan, B. P. Harrawood, E. Samei, and A. J. Kapadia, "Volumetric x-ray coherent scatter imaging of cancer in resected breast tissue: a Monte Carlo study using virtual anthropomorphic phantoms," *Physics in Medicine and Biology*, vol. 60, no. 16, pp. 6355–6370, 2015.

[20] L. de Sisternes, J. G. Brankov, A. M. Zysk, R. A. Schmidt, R. M. Nishikawa, and M. N. Wernick, "A computational model to generate simulated three-dimensional breast masses," *Medical Physics*, vol. 42, no. 2, pp. 1098–1118, 2015.

[21] D. W. Erickson, J. R. Wells, G. M. Sturgeon, E. Samei, J. T. Dobbins, W. P. Segars, and J. Y. Lo, "Population of 224 realistic human subject-based computational breast phantoms," *Medical Physics*, vol. 43, no. 1, pp. 23–32, 2015.

[22] J. A. Greenberg, "Coded apertures for faster x-ray scatter imaging," SPIE Newsroom, pp. 10.1117/2.1201608.006646 1-3, 16 08 2016.

[23] Carpenter, Joshua, et al. "Motivations and Methods for the Analysis of Multi-Modality x-Ray Systems for Explosives Detection." *Anomaly Detection and Imaging with X-Rays (ADIX) IV*, 2019, doi:10.1117/12.2518781.

[24] Coccarelli, David, et al. "Modeling Real World System Geometry and Detector Response within a High-Throughput x-Ray Simulation Framework." *Anomaly Detection and Imaging with X-Rays (ADIX) IV*, 2019, doi:10.1117/12.2518870.